



Software Development Life Cycle

GAME 2338

Recap

- What is Quality?
- What is difference between QA and QC?
- Name 5 of the 11 Quality Factors
- What are the 5 Perspectives of Quality?



What do you think?

- What is a project?
- Project Management?
- What is a software development lifecycle?



Project Management Process Groups


- Initiating
- Planning
- Executing
- Monitoring & Controlling
- Closing



Project Management Knowledge Areas

- Integration
- Scope
- Time
- Cost
- Quality
- Human Resource
- Communications
- Risk
- Procurement





Software Development Life Cycle (SDLC) is a software development process structure that is used to develop a software product.

SDLC Models

- Waterfall
- Prototype
- Spiral/Phased
- Agile/Scrum



Waterfall

User Need

Define Requirements

Design

Code/ Development

Testing

Maintenance



Waterfall

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.



Waterfall

- Strengths

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

- Weakness

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements. Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang" at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

Prototyping

Provides “look and feel” of system early in the development process

Prototype is a working model of software with some limited functionality. The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.

Basic Requirement Identification

Developing the initial Prototype

Review of the Prototype

Revise and Enhance the Prototype



Prototyping

- Strengths

- Increased user involvement in the product even before its implementation.
- Since a working model of the system is displayed, the users get a better understanding of the system being developed.
- Reduces time and cost as the defects can be detected much earlier.
- Quicker user feedback is available leading to better solutions.
- Missing functionality can be identified easily.
- Confusing or difficult functions can be identified.

- Weaknesses

- Risk of insufficient requirement analysis owing to too much dependency on the prototype.
- Users may get confused in the prototypes and actual systems.
- Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
- Developers may try to reuse the existing prototypes to build the actual system, even when it is not technically feasible.
- The effort invested in building prototypes may be too much if it is not monitored properly.

Spiral/Phased

A combination of prototype and waterfall model, usually used on large complex projects

1. Identification	2. Design
4. Evaluation and Risk Analysis	3. Construct or Build



Spiral/Phased

- Strengths

- Changing requirements can be accommodated.
- Allows extensive use of prototypes.
- Requirements can be captured more accurately.
- Users see the system early.
- Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

- Weaknesses

- Management is more complex.
- End of the project may not be known early.
- Not suitable for small or low risk projects and could be expensive for small projects.
- Process is complex
- Spiral may go on indefinitely.
- Large number of intermediate stages requires excessive documentation.

Agile

Sped up development process that can bypass more of phase in a lifecycle, less formal, reduced scope

- Individuals and interactions over processes and tools
 - Working software over comprehensive documentation
 - Customer collaboration over contract negotiation
 - Responding to change over following a plan
- <http://agilemanifesto.org/>
 - <http://agilemanifesto.org/principles.html>



Agile

- Strengths

- Is a very realistic approach to software development.
- Promotes teamwork and cross training.
- Functionality can be developed rapidly and demonstrated.
- Resource requirements are minimum.
- Delivers early partial working solutions.
- Good model for environments that change steadily.
- Minimal rules, documentation easily employed.
- Enables concurrent development and delivery within an overall planned context.
- Little or no planning required. Easy to manage.
- Gives flexibility to developers.

- Weakness

- Not suitable for handling complex dependencies.
- More risk of sustainability, maintainability and extensibility.
- An overall plan, an agile leader and agile PM practice is a must without which it will not work.
- Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
- Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
- There is a very high individual dependency, since there is minimum documentation generated.
- Transfer of technology to new team members may be quite challenging due to lack of documentation.

Scrum

Scrum is an Agile framework for completing complex projects. Scrum originally was formalized for software development projects, but it works well for any complex, innovative scope of work. The possibilities are endless. The Scrum framework is deceptively simple.

- <https://www.scrum.org/about>
- https://www.collab.net/sites/default/files/videos/Intro_to_scrum/Intro_to_scrum.htm



Scrum

- Backlog
- Sprint Planning
- Daily Scrum
- Sprint Review
- Sprint Retrospective



When does testing happen?

- Waterfall
 - All testing occurs during define period
- Prototype
 - Testing occurs prior to each prototype being released or sometimes just before final release
- Spiral/Phased
 - Testing occurs at the end of each spiral or phase (monthly, quarterly)
- Agile/Scrum
 - Testing occurs often (daily or weekly)



Game Development Lifecycles

Games are usually built using a combination of the prototype (with some aspects of agile) and phased models

A typical game lifecycle consists of:

- Design
- Prototype
- Alpha
- Beta
- Gold



Game Lifecycle Phases

- **Prototype**
 - Rough draft, used to get a publishing deal
- **Alpha**
 - First major milestone Includes a full playthrough (Beginning to End)
- **Beta**
 - Game is virtually finished
 - Closed /Open Beta testing
- **Gold**
 - Ready for the market





Testing Process

- Define Test Objective
- Develop Test Plan
- Develop Test Cases
- Execute Tests
- Report Test Results



Define Test Objective

- Organizing a test team
- Determining what needs to be tested or what requirements need to be met (Quality)
 - Testing Strategy



Develop Test Plan

- A document that lays out the plan for testing the system, includes the what, when, location, and how
- Test assumptions or constraints
- Breakdown of testing functions



Develop Test Cases

A test case also known as a test procedure is how a test will be conducted.

Test Cases include

- a test condition - what is being validated
- an expect result - the desired outcome
- test data, if needed



Execute Tests

Running each test, recording the actual results of the test (Pass/Fail)



Report Formal Test Results

- A formal record of the results of each executed test and a list of defects found during testing
- Test report sometimes includes a recommendation from the test team on whether to move forward



The Test Team

- QA Manager
 - Interface with project leaders
 - Make all major testing decisions
- Test Lead
 - Divide up work load among testers
 - Multiple Test Leads may report to QA Manager
- Tester
 - Test case writing and execution



Test Team Skillsets

- Written and Oral Communication
- Listening
- Eye for Detail
- Interviewing
- Negotiation and Complaint resolution
- Good Project Relationships



Sources

- <https://www.tutorialspoint.com/sdlc/>

