DrillBit

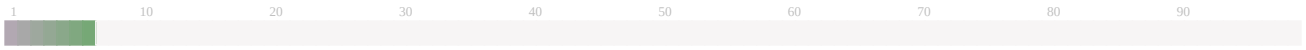
The Report is Generated by DrillBit Plagiarism Detection Software

Submission Information

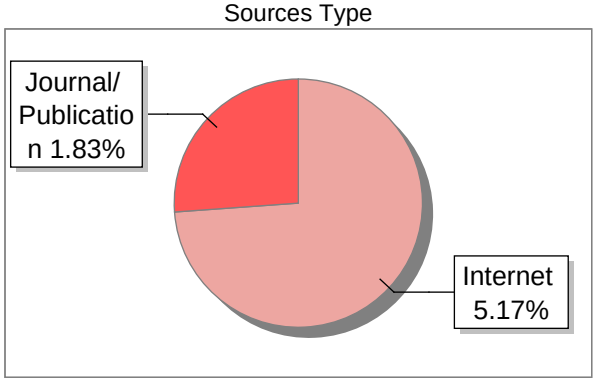
Author Name	Akash g
Title	Task
Paper/Submission ID	3335293
Submitted by	nnm23is030@nmamit.in
Submission Date	2025-02-16 08:20:56
Total Pages, Total Words	15, 2460
Document type	Project Work

Result Information

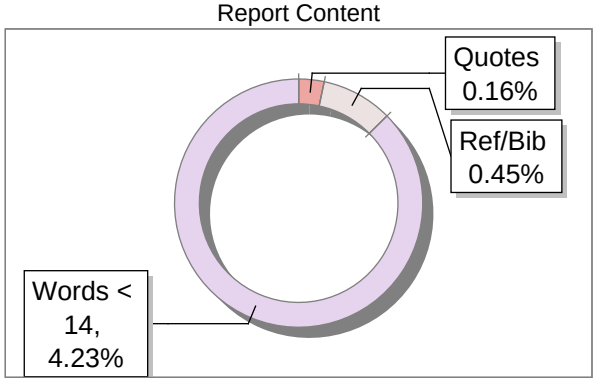
Similarity **7 %**



Sources Type



Report Content



Exclude Information

Quotes	Not Excluded
References/Bibliography	Not Excluded
Source: Excluded < 14 Words	Not Excluded
Excluded Source	**0 %**
Excluded Phrases	Not Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	Yes

A Unique QR Code use to View/Download/Share Pdf File

DrillBit

DrillBit Similarity Report

7	12	A
SIMILARITY %	MATCHED SOURCES	GRADE

A-Satisfactory (0-10%)
B-Upgrade (11-40%)
C-Poor (41-60%)
D-Unacceptable (61-100%)

LOCATION	MATCHED DOMAIN	%	SOURCE TYPE
1	dl.lib.uom.lk	1	Internet Data
2	frontiersin.org	1	Internet Data
3	dochero.tips	1	Internet Data
4	ncgg.org.in	1	Publication
5	www.zendesk.co.uk	1	Internet Data
6	Design of a service-oriented architecture for AAL by Peruzzini-2016	<1	Publication
7	docshare.tips	<1	Internet Data
8	www.diva-portal.org	<1	Publication
9	www.mobilelive.ca	<1	Internet Data
10	moam.info	<1	Internet Data
11	www.dx.doi.org	<1	Publication
12	www.ncbi.nlm.nih.gov	<1	Internet Data

Software Process Models and Process Engineering

SDLC for Ebay

Problem Statement:
Analyse the software development lifecycle (SDLC) of a real-world system by conducting a comparative study of process models and their impact on requirements management.

By:
AKASH G

NNM23IS008

NMAM Institute of Technology

Abstract

Design and maintenance of an online market cannot be done without a systematic and cogent approach towards the software development of their information system. This report will describe the different SDLC models that eBay has so far used to improve user experiences, to ensure transactions, and to modify upgrades according to the technologies. We outline the incremental, spiral, and waterfall approaches as possible methods eBay uses to stay ahead in the online competition.

An analysis of eBay's approaches to development contributes a revealing understanding of how a successful online marketplace devises, develops, and maintains its platform for confronting challenges and responding to changes in the industry.

Table of Contents

3.3 How eBay Validates Its Requirements

- Challenges in Software Development at eBay
- Future Trends in eBay’s SDLC Approach
- Conclusion
- References

1. Introduction

Software development is not merely about writing code; more significantly, it's about creating a secure, stable, and highly scalable platform meeting business and customer requirements. For an e-commerce giant such as eBay, which processes millions of transactions a day, software development needs to be planned carefully and executed in the most efficient manner.

With new features and security updates constantly being introduced, eBay requires an SDLC strategy that balances innovation with reliability. This report discusses the various SDLC approaches used by eBay and how they contribute to its long-term success.

eBay’s SDLC process is designed to support a global user base, ensuring that software updates are deployed seamlessly across different regions while considering local regulatory requirements.

Models of the System Development Life Cycle (SDLC) and Their Relevance in eBay's Development.

1.1 What is SDLC:

The Software Development Life Cycle (SDLC) is a structured approach to planning, developing, testing, and deploying software systems. Choosing the right SDLC model is crucial for the success of large-scale platforms like Ebay, a leading e-commerce solution. The goal of SDLC is to ensure the delivery of high-quality, scalable, and user-friendly software that meets business and customer needs. By following a well-defined SDLC model, development teams can efficiently manage each

stage of the process, reducing costs and ensuring timely delivery while maintaining software reliability and performance.

1.2 Why is SDLC important :

The Software Development Life Cycle (SDLC) is fundamental to successful software project delivery for several critical reasons:

- **Keeps things organized:** Without a plan, software projects can get out of hand fast. SDLC provides that much-needed structure, so everyone's on the same page and knows what they're doing.
- **Catches problems early:** Nobody wants to discover a major bug right before launch. SDLC helps catch those issues early on, when they're easier and cheaper to fix. It's like having a good editor for your work.
- **Improves quality:** Going through the different SDLC stages helps ensure you're building a solid product. Think of it as quality control at every step, not just at the end.
- **Makes communication easier:** When everyone's following the same process, it's way easier to talk about the project. No more misunderstandings or crossed wires – hopefully!
- **Saves time and money:** A well-defined SDLC can actually save you time and money in the long run. It reduces wasted effort and prevents costly mistakes down the line. It's an investment that pays off.
- **Reduces risk:** Software development is inherently risky. SDLC helps minimize those risks by providing a roadmap and a framework for managing the project.

1.3 How does SDLC work:

The Six Stages of the Software Development Life Cycle

The Software Development Life Cycle consists of six distinct stages, each playing a crucial role in creating successful software solutions. Understanding these stages helps teams deliver high-quality products that meet client expectations.

Planning and Requirements Analysis

The journey begins with assembling a skilled team of engineers who can lay a strong foundation for the project. During this initial phase, the team conducts thorough preliminary analysis to understand client objectives and challenges. They then develop multiple solution proposals, each with detailed budget estimates, allowing clients to choose the most suitable approach for their needs.

Defining Requirements

Once a solution direction is chosen, the team delves deeper into the specific requirements. This involves careful analysis of project documentation and evaluation of the client's existing systems. While some consider this stage an extension of the planning phase, it serves as a crucial bridge between initial planning and actual development work.

Product Architecture Design

With a clear understanding of requirements, the development team creates several potential product architectures. These designs are presented to the client for review and selection. The chosen architecture is then documented in detail through a Design Document Specification (DDS). This document undergoes thorough evaluation for potential risks, operational reliability, flexibility, and cost-effectiveness, serving as the blueprint for all subsequent development work.

Product Development

The development stage typically consumes the largest portion of project time and resources. During this critical phase, developers transform the approved design into working code, regularly demonstrating progress to the client. It's worth noting that projects often extend beyond initial timelines due to scope changes or additional client requirements, particularly in flexible SDLC models that accommodate evolving product specifications.

Product Testing

Quality assurance takes center stage in this phase, as testing engineers meticulously search for potential issues and bugs. While testing activities may occur throughout the development process, this dedicated testing phase provides a comprehensive assessment of the product's functionality and reliability. The team creates detailed reports of any identified issues, ensuring they can be addressed before release.

Deployment and Maintenance

The final stage involves launching the completed software solution and ensuring its continued effectiveness. After deployment, ongoing maintenance becomes essential for addressing any emerging issues and ensuring optimal performance. This support phase helps protect the client's investment and maintains the software's value over time.

1.4 What are SDCL models:

SDLC Models and Their Implementation

The Software Development Life Cycle can be implemented through various models, each offering distinct advantages for different project needs. Organizations typically choose specific models based on their project requirements, team structure, and business objectives.

Common SDLC Models

Several established models have proven effective in different contexts:

- The Waterfall Model follows a sequential, linear approach
- Incremental Development breaks projects into manageable chunks
- Iterative Development focuses on continuous refinement
- The Spiral Model emphasizes risk management and prototyping
- Agile methodologies prioritize flexibility and customer collaboration

Shopify's SDLC Implementation

Shopify employs a hybrid approach combining Agile and Iterative models in their development process. This combination proves particularly effective for their e-commerce platform, allowing them to:

- Maintain rapid deployment cycles for new features
- Respond quickly to merchant feedback and market changes
- Manage complex integrations with third-party services
- Balance innovation with platform stability

The Agile component enables Shopify to adapt swiftly to evolving e-commerce trends and merchant needs, while the Iterative model ensures robust testing and refinement of features before full deployment. This hybrid approach has proven especially valuable in managing Shopify's extensive ecosystem of apps and plugins while maintaining the platform's reliability for millions of merchants worldwide.

2.1 Incremental Development Model

This model weighs on releasing new features in the smaller stage releases to ensure constant growth and easy adaptability based on the feedback given by the users.

Functionality

eBay updates its marketplace with new features on a regular basis.

New features such as AI-driven recommendations are introduced incrementally.

Payment system improvements and fraud detection are introduced step by step.

Non-Functionality

Architectural inconsistencies may exist in multiple increments.

Features can render the previous ones obsolete as new ones are introduced.

Risk & Change Management

Less risk since eBay can fix issues in the next increment.

Changes can be introduced efficiently following customer feedback.

Description:

The features are built and tested in phases while not all at once.

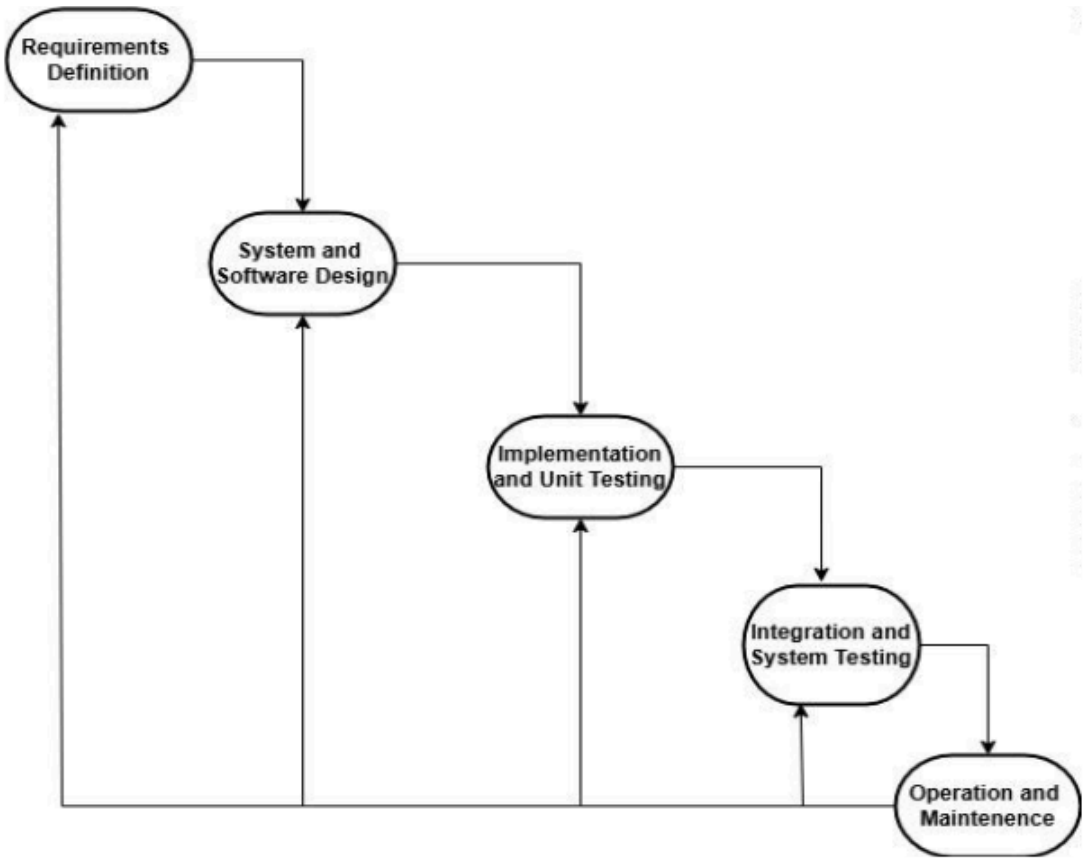
Each phase builds onto the last one to enhance the system at each stage of the development.

The next improvements can shape directly from the suggestions given by the users.

The eBay Case:

eBay could first introduce a simple product-search feature and then sophisticated AI-based recommendations based on observed user behaviour.

Another specialty of this model on eBay is that different development teams can work on different components in parallel; hence, it helps speed up the deployment cycle.



2.2 Spiral Model

Some mix of this and that, so-called iterative modelling and top-down model management, is particularly handily applied for projects where the requirements are expected to change during the course of the project and for complicated projects.

Functionality

Suitable for large projects such as eBay's AI-based personalization.

Helps evaluate security threats during transactions.

Regular improvement and evolution.

Non-Functionality

Can be expensive due to reiterative risk analysis.

Requires experienced risk management professionals.

Risk & Change Management

Effective at handling security risks like fraud and cyber attacks.

Changes are put in place after every iteration through real-time intelligence.

Time & Cost Constraints

High cost because of reiterative iterations and risk analysis.

Development cycles become longer but promise reliability and safety.

Reasons why eBay Uses This:

Useful in predicting and eliminating security and performance-based risks at the project outset.

Permits constant reiteration when coupled with the concept of phased testing.

Hence very useful in practical implementations like a new fraud detection system.

eBay example.

Before rolling out a new AI-based fraud detection tool, eBay would carry out periodic reviews, allowing iterative improvements to minimize failure by the time the tool goes live.

eBay uses this initiative to comply with various data protection laws like the CCPA and GDPR that call for constant vulnerability assessments that could affect proper data security.

2.3 Waterfall model

The waterfall model is a linear model that is applicable when working on projects whose requirements are clear and do not change very often.

Functionality

Used in fixed processes like payment gateway integration.

Easy to document and organized development process.

Non-Functionality

Is not appropriate for rapidly changing market conditions.

Errors discovered late may be costly.

Risk & Change Management

Higher risk if initial requirements are not well-defined.

It is difficult to make changes once a phase is complete.

Time & Cost Constraints

Lower cost for small, well-defined projects.

Time required is larger due to sequential execution.

No early delivery of working components.

How eBay Uses It:

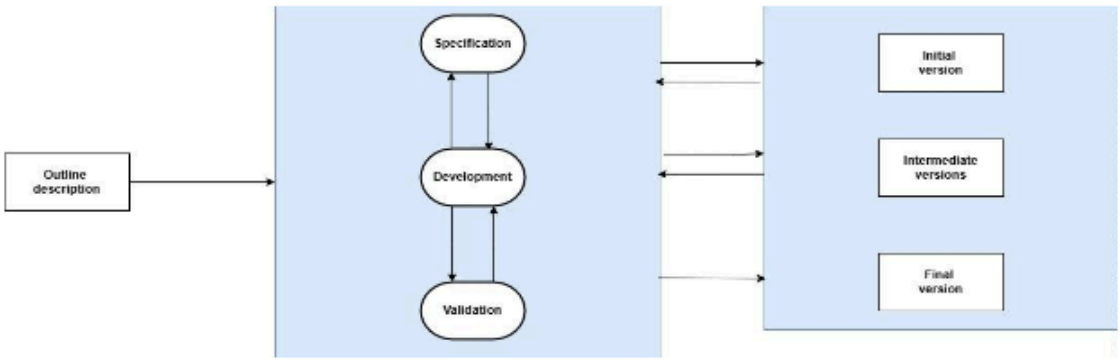
Updating large-scale infrastructure where changes will have to be meticulously mapped out beforehand.

It will typically be used to be compliant with financial and legal regulations.

An example at eBay:

This structured approach might be followed when upgrading payment systems in order to comply with new international tax laws.

This type model will be useful particularly so eBay may partner with third-party financial institutions, as this guarantees clear documentation and sequential development phases.



3. How eBay Handles Software Requirements

3.1 Functional Requirements

- User Authentication: secure sign on, password recovery, and account security.
- Item Listings via an easy, efficient product management process for sellers.
- Cart & Checkout: Smooth transactions with multiple options for selling payment systems.
- Order Tracking and Customer Support-oh, tracking is clear and support for buyers is made accessible.

3.2 Non-functional Requirements

- Performance: Fast-loading pages, even during high-traffic periods.
- Security-a must-have that ensures the safety of all users while preventing fraudulent behaviour.
- Scalability-must have the capacity to come to terms with an influx of numerous users or transactions.

- User Experience-simple and intuitive interface across multiple devices for buyers and sellers.

3.3 How eBay vets its requirements

Software must align with business goals.

Extensive testing before a larger rollout.

Any changes have a management procedure, so no inconvenience is experienced while delivering services.

- eBay employs machine learning in monitoring systems performance and autonomously adjusts server workloads to alleviate predicted spikes in traffic.

- Scalability-a problem, even for a small part; millions of transactions daily, and one slight slip up on the computation side can create total chaos.

- Online extortion – Ebay is a common target of fraud and other cyber-security threats from virtually anywhere in the world.

- Regulatory compliance and adherence-All updates take effect in line with the local law of other countries' jurisdictions.

4. Challenges in Software Development at eBay

eBay, as a huge online market, encounters various software development problems that arise from immense numbers of its users, ongoing business requirements, and other complexities in its structural framework. Among other concerns, others include:

Scalability and Performance

eBay specializes in millions of transactions a day, so a very scalable system with high performance is required. In grounded reality, it is a challenge to maintain a good performance under a lot of pressure, especially during holidays when shopping seasons normally rise. The developers will need to optimize the databases, use distributed computing, and apply caching mechanisms to prevent the system from going slow.

Search and Recommendation Algorithms
The ultimate eBay success factor is the eBay search engine and recommendation systems. Meanwhile, in the writers' attempts to change the algorithms in cases of preventing fraud, providing personalized suggestions, and ensuring accuracy in the search engine, they would also incorporate maximizing relevance and speed as measures of optimization.

Security and Frauds
The huge scale of both buyers and sellers that have to be handled by the site makes it susceptible to getting infected with cybercrime, fraud, account hijacking, and payment fraud. The setting up of a secure environment would require AI-driven fraud protection, 2-step verification, and data encryption: all cumbersome when on the maintainer's desk.

Legal System Modernization
Since eBay has been there since the late 1990s, it means a big part of its operations is supported by legacy systems. Change is not easy without stopping operations, which complicates the migration plan and API integration.

5. Future Trends in eBay's SDLC Approach

Greater Use of AI & Automation-Development must move faster with all of AI's automated coding assistants and test automation frameworks.

- Microservices Architecture: Breaking big systems into smaller, manageable services.

- Blockchain for Security: An exploration of blockchain to improve transaction security and trust.

6. Conclusion

Software development at eBay goes well beyond the realm of coding for a powerhouse e-commerce firm such as it ensures a pleasant, secure, and reliable experience for each of its many million users. By mixing and matching all sorts of SDLC models, achieving a proper balance between innovation every now and then and platform stability at the other end remains eBay's recipe for success. Whether it's in the guise of rolling out deep learning-based features or the further strengthening of fraud detection, they have the quality of approach that long-term growth and success embody.

7. References

Sommerville, Ian. "Software Engineering (9th Edition)."

Developer Documentation, eBay.