1. Explore the waterfall and agile approaches to software development, focusing on the implications of developing secure software using each.

In the modern business environment, efficient project management is crucial for satisfying client expectations and producing high-quality deliverables (Lockhart, 2023).

The two primary approaches for software development are waterfall and agile. Agile is an iterative, incremental technique, whereas waterfall is linear and sequential (Saafan, 2023). For instance, Lockhart (2023) says that agile and waterfall project management methodologies have advantages and disadvantages. Choosing the best strategy depends on each team's requirements and the project's specifics, which is the project manager's responsibility.

1.1 Waterfall Approach:

A conventional and sequential way of software development is the waterfall approach. Each phase must be finished before going on to the next in the process (Sandhu, 2023). Furthermore, Lewis & Lutkevich (2019) claimed that software engineering and product development often use the waterfall model, a linear, sequential approach to the software development lifecycle (SDLC).

However, Zulqadar (2022) says the phases typically include gathering requirements, designing the system, putting it into action, testing it, deploying it, and maintaining the software product.

According to Laoyan (2022), the waterfall approach to building secure software are the requirements collecting, design, development, testing, and deployment phases are only a few of the phases into which the waterfall technique divides the software development process. Before the next step may start, the previous phase must finish. This method works best for projects with precise requirements and a foreseeable schedule. However, if conditions change, the altering direction might be rigid and challenging. If security is not considered early in the development process, this could result in security vulnerabilities.

1.2 Agile Approach:

Agile is a flexible software development approach that focuses on efficiently delivering individual pieces of software. It improves team collaboration and efficiency and allows for testing throughout the process, resulting in a high-quality product. Agile has replaced waterfall and is currently popular but faces competition from DevOps (Brush & Silverthorne, 2022).

According to Hamilton (2023), the software development process is divided into brief sprints that last around two weeks according to the agile methodology. Delivering a functional product increment is the primary goal of each sprint. The team meets frequently to review progress, pinpoint dangers, and modify the strategy. This method works effectively in projects with dynamic requirements and a hurried setting. Additionally, it enables security to be considered at every stage of development, reducing the chance of vulnerabilities.

The implications of these different approaches for developing secure software are significant.

The waterfall is more structured and risk-averse, making identifying and mitigating security risks easier. However, it can also be more time-consuming and expensive, and it can be challenging to adapt to changes in requirements (Nash, 2021).

Agile is more flexible and responsive to change, which can benefit security as it allows security risks to be identified and addressed more quickly. However, agile can also be more chaotic and less organised, making it more difficult to ensure security is implemented correctly (Nash, 2021).

In general, the waterfall is a better approach for developing secure software when the requirements are well-defined and there is little risk of change. Agile is a better approach for developing secure software when the conditions are not well-defined or there is a high risk of instability.

2. Become acquainted with the Unified Modelling Language and how it can be used to support software development.

OMG (2017) states that the Unified Modeling Language (UML) is a modeling language for software and business processes. It provides tools for analysis, design, and implementation. UML has improved syntax and modularity for large-scale systems. It specifies human-readable notation elements for different diagram types. UML models consist of classifiers, events, and behaviours that make statements about different kinds of individuals within the system. UML's framework deals with discrete behaviours but can simulate continuous behaviours with small event intervals. Supplementary modeling constructs include use cases, deployments, and information flows.

Visual Paradigm (2019) states that UML is a modeling language developers use to specify, visualise, construct, and document software systems. It is a collection of best practices essential to developing object-oriented software. The UML uses graphical notations to express software project designs, which helps project teams communicate, explore potential designs, and validate the architectural design of the software.

According to Nishadha (2022), UML diagrams generally fall into one of two categories:

2.1 Structure Diagrams:

- **Class Diagram:** It displays a system's classes, along with each class's properties, operations, and relationships to other classes.

- **Component Diagram:** A component diagram shows how structurally connects a software system's components.
- **Deployment Diagram:** A deployment diagram displays the system's hardware and the software running on it.
- **Object Diagram:** Object diagrams show relationships between objects, similar to class diagrams.
- **Package Diagram:** The interdependencies between several packages in a system are displayed in a package diagram.
- **Profile Diagram:** A new type of diagram, the profile diagram, was introduced in UML 2. This style of diagram is incredibly infrequently used in any specification.
- **Composite Structure Diagram:** Composite structure diagrams display a class's internal organisation.

2.2 Behavioural Diagrams:

- **Use Case Diagram:** Use case diagrams show user and system interactions.
- **Activity Diagram:** Activity diagrams show a system's control flow.
- **State Machine Diagram:** State machine diagrams show the different states that an object can be in and the transitions between states.
- **Sequence Diagram:** Sequence diagrams show the interactions between objects in a system over time.
- **Communication Diagram:** Communication diagrams resemble sequence diagrams for object message passing. The same information can be displayed in various objects.
- **Interaction Overview Diagram:** Unlike activity diagrams, interaction overview diagrams show a series of interaction diagrams rather than processes.
- **Timing Diagram:** Like sequence diagrams, timing diagrams depict object behaviour across a predetermined time frame. The diagram is straightforward if there is just one object involved.

Several phases of the software development lifecycle are supported by UML, including:

Requirements of the system: Use case diagrams assist in identifying and describing the system's many functionalities and how users or external systems interact with it (IBM, 2021).

Design: Class diagrams, object diagrams, and component diagrams are valuable tools for planning the organisation of components and their connections (IBM, 2021).

Implementation: UML diagrams can act as a road map for developers, guiding them as they construct and implement the code (IBM, 2021).

Testing and Validation: UML diagrams precisely depict the system's anticipated behaviour, making it easier to create test cases and validate system behaviour (Geekboots, 2018).

Documentation: UML represents best practices for developing and describing many software elements, and business system modelling is represented by UML (Geekboots, 2018).

Collaboration and Communication: By offering a common visual language, UML diagrams enable effective communication between developers, architects, stakeholders, and other team members (Cybermedian, 2023).

3. Gain familiarity with the standards used by the industry to develop secure software.

Various standards, guidelines, and certifications are available for software security to ensure that software development projects comply with applicable regulations. These best practices include Common Criteria, TOGAF, SAMM, BSIMM, ASVS, OWASP, and SAFECode, as well as national and international standards groups such as PCI, NIST, and ISO/IEC. Implementing these measures can help improve the security of software development projects (Ramirez et al. 2020).

Furthermore, Kirvan & Granneman (2021) state that IT security management encompasses several topics, including perimeter defence, encryption, application security, and disaster recovery. It becomes even more difficult due to compliance rules like HIPAA, PCI DSS, Sarbanes-Oxley, and GDPR. Frameworks and standards for IT security help comprehend and abide by these laws. They serve as a recipe by outlining the requirements for compliance. These guidelines must be followed by an IT organisation that is efficiently managed.

The following are some significant standards that the sector uses to provide secure software:

ISO/IEC 27001: This standard, ISO/IEC 27001, offers a framework for managing information security. From risk analysis to incident response, it includes every facet of information security (ISO, 2022).

OWASP Top 10: Software development teams can learn from OWASP, a worldwide nonprofit organisation, how to design, buy, use, and maintain safe apps. The OWASP Top 10 is a yearly study that lists the ten most significant web application and API security concerns (Foster, 2020).

CWE Top 25: This ranking of the top 25 software security issues. It is a helpful resource for developers to discover typical security flaws (Foster, 2020).

CERT Secure Coding Standards: These CERT Secure Coding Standards offer a risk assessment for each advice or rule in the secure coding standard to help determine the potential repercussions of breaking that particular recommendation or law (Foster, 2020).

IEC 62443: IEC 62443 is a collection of security guidelines that protect industrial networks from online dangers. The security standards provide a comprehensive and organised list of cybersecurity guidelines (Foster, 2020).

Hence, as Snyk (2022) states, following these standards is essential to implement secure development practices throughout the software development lifecycle (SDLC). These practices include:

- Performing security risk analyses
- Utilising secure coding techniques
- Carrying out penetration tests
- Putting secure development tools in place

- Teaching security best practices to developers

4. Appreciate the importance of developing a risk-aware culture within an organisation.

Being aware of potential risks is crucial for businesses. It means identifying all possible threats, not just the most common ones (Gerard, 2022). However, risk awareness alone is not enough to prevent harm. To respond effectively to emergencies, improving risk identification and understanding is essential.

Everyone in the organisation must be aware of the value of managing risks and actively participate in their identification, evaluation, and mitigation to be considered to have a risk-conscious culture. Any organisation that wishes to succeed in today's unpredictable and uncertain world must have this kind of culture.

Here are some of the benefits of developing a risk-aware culture, according to Langer (2022):

- It can help organisations avoid or mitigate losses. By identifying and assessing risks early, organisations can prevent them from happening or reduce their impact.
- It can help organisations improve their decision-making. Organisations can make better decisions about allocating resources and managing their operations when everyone knows the risks.
- It can help organisations build trust and credibility. When employees see that the organisation is taking risks seriously, they are more likely to trust management and be willing to go the extra mile.
- It can help organisations comply with regulations. Many regulations require organisations to have a risk management framework in place. By developing a risk-aware culture, organisations can demonstrate that they are taking steps to comply with these regulations.

According to The Institute of Risk Management, Risk culture Under the Microscope Guidance for Boards (N.D.), to develop a risk-aware culture, organisations can create a risk-aware culture that requires commitment, communication, training, incentives, rewards, and accountability. Senior leaders must prioritise risk management and ensure everyone knows the risks and how to manage them. Employees should be rewarded for identifying and mitigating risks and held accountable for managing risks within their areas of responsibility.

Developing a risk-aware culture is an ongoing process. It takes time and effort to embed risk management into the fabric of an organisation. However, the benefits of a risk-aware culture are significant and can help organisations achieve their goals and objectives.

References:

Lockhart, L. (2023). *Agile vs. Waterfall: 10 Key Differences Between the Two Methods*. [online] www.float.com. Available at: https://www.float.com/resources/agile-vs-waterfall/.

Saafan, A. (2023). *Agile vs Waterfall: Comparing Software Development Life Cycle Methods*. [online] Available at: https://www.linkedin.com/pulse/agile-vs-waterfall-comparing-software-development-life-amr-saafan#:~:text=Approach%3A%20Agile%20is%20an%20iterative [Accessed 16 Aug. 2023].

Sandhu, A. (2023). *Advantage Of The Waterfall Methodology [2022]» BestOutcome*. [online] Available at: https://bestoutcome.com/knowledge-centre/waterfall-methodology-advantage/.

Zulqadar, A. (2012). *SDLC Waterfall Model: The 6 phases you need to know...* [online] Rezaid. Available at: https://rezaid.co.uk/sdlc-waterfall-model/.

Lewis, S. & Lutkevich, B. (2019). *What is waterfall model? - Definition from WhatIs.com*. [online] SearchSoftwareQuality. Available at: https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model.

Laoyan, S. (2022). *Everything You Need to Know About Waterfall Project Management • Asana*. [online] Asana. Available at: https://asana.com/resources/waterfall-project-management-methodology.

Brush, K. & Silverthorne, V. (2022). *What is Agile Software Development (Agile Methodologies)?* [online] SearchSoftwareQuality. Available at:

https://www.techtarget.com/searchsoftwarequality/definition/agile-software-development.

Hamilton, T. (2023). *Waterfall Vs. Agile: Must Know Differences*. [online] Guru99.com. Available at: https://www.guru99.com/waterfall-vs-agile.html.

Nash, B. (2021). *What is the Difference Between Agile vs Waterfall? | TrustRadius*. [online] TrustRadius Blog. Available at: https://www.trustradius.com/buyer-blog/difference-between-agile-vs-waterfall.

OMG UML (2017). *An OMG ® Unified Modeling Language ® Publication OMG ® Unified Modeling Language*. Available at: https://www.omg.org/spec/UML/2.5.1/PDF.

Visual Paradigm (2019). *What is Unified Modeling Language (UML)?* [online] Visual-paradigm.com. Available at: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/.

Nishadha (2022). *UML Diagram Types | Learn About All 14 Types of UML Diagrams*. [online] Creately Blog. Available at: https://creately.com/blog/diagrams/uml-diagram-types-examples/.

IBM (2021). *Use-case diagrams*. [online] www.ibm.com. Available at: https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case.

IBM (2021). *Class diagrams*. [online] www.ibm.com. Available at: https://www.ibm.com/docs/en/rsm/7.5.0?topic=structure-class-diagrams.

Geekboots (2018). *How UML help in software development? | Geekboots*. [online] www.geekboots.com. Available at: https://www.geekboots.com/story/how-uml-help-in-software-development [Accessed 16 Aug. 2023].

Cybermedian (2023). *A Comprehensive Guide to Understanding and Implementing Unified Modeling Language in Software Development*. [online] Cybermedian. Available at: https://www.cybermedian.com/a-comprehensive-guide-to-understanding-and-implementing-unified-modeling-language-in-software-development/.

Ramirez, A., Aiello, A. & Lincke, S.J. (2020). *A Survey and Comparison of Secure Software Development Standards*. [online] IEEE Xplore. doi:https://doi.org/10.1109/CMI51275.2020.9322704.

Kirvan, P. & Granneman, J. (2021). *Top 7 IT security frameworks and standards explained*. [online] SearchSecurity. Available at: https://www.techtarget.com/searchsecurity/tip/IT-security-frameworks-and-standards-Choosing-the-right-one.

ISO (2022). *ISO/IEC 27001 standard – information security management systems*. [online] ISO. Available at: https://www.iso.org/standard/27001.

Foster, S. (2020). *Secure Coding Practices: What Are Secure Coding Standards? | Perforce*. [online] www.perforce.com. Available at: https://www.perforce.com/blog/qac/secure-coding-standards#cert [Accessed 17 Aug. 2023].

Gerard, S. (2022). *How to Improve Risk Awareness in the Workplace With Practice and Communication*. [online] Available at: https://www.alertmedia.com/blog/risk-awareness/#:~:text=Risk%20awareness%20is%20the%20ability.

The Institute of Risk Management Risk Culture Under the Microscope Guidance for Boards. (n.d.). Available at:

https://www.theirm.org/media/8447/risk_culture_a5_web15_oct_2012-executive-summary.pdf.