ĐẠI HỌC TÔN ĐỨC THẮNG
TON DUC THANG UNIVERSITY

502045
Software Engineering

Chapter 03
Lesson 03: Agile Process

Topics covered

- ✧ Agile methods
- ✧ Plan-driven and agile development
- ✧ Extreme programming
- ✧ Agile project management
- ✧ Scaling agile methods

Rapid software development

- Rapid development and delivery is now often the most important requirement for software systems
  - Businesses operate in a fast –changing requirement and it is practically impossible to produce a set of stable software requirements
  - Software has to evolve quickly to reflect changing business needs.
- Rapid software development
  - Specification, design and implementation are inter-leaved
  - System is developed as a series of versions with stakeholders involved in version evaluation
  - User interfaces are often developed using an IDE and graphical toolset.

Rapid software development

- Phát triển và giao hàng nhanh chóng hiện nay thường là yêu cầu quan trọng nhất đối với các hệ thống phần mềm.
  - Các doanh nghiệp hoạt động trong một môi trường yêu cầu thay đổi nhanh chóng và thực tế là không thể tạo ra một bộ yêu cầu phần mềm ổn định.
  - Phần mềm phải phát triển nhanh chóng để phản ánh nhanh chóng các nhu cầu kinh doanh thay đổi.
- Phát triển phần mềm nhanh chóng:

  - **Đặc tả, thiết kế và triển khai được xen kẽ lẫn nhau.**

  - Hệ thống được phát triển dưới dạng một loạt các phiên bản với các bên liên quan tham gia vào việc đánh giá phiên bản.

  - Giao diện người dùng thường được phát triển bằng cách sử dụng một **môi trường phát triển tích hợp (IDE)** và **bộ công cụ đồ họa.**

Agile methods

- Dissatisfaction with the overheads involved in software design methods of the 1980s and 1990s led to the creation of agile methods. These methods:
  - Focus on the code rather than the design
  - Are based on an iterative approach to software development
  - Are intended to deliver working software quickly and evolve this quickly to meet changing requirements.
- The aim of agile methods is to reduce overheads in the software process (e.g. by limiting documentation) and to be able to respond quickly to changing requirements without excessive rework.

Agile methods

Sự không hài lòng với các chi phí liên quan đến các phương pháp thiết kế phần mềm của những năm 1980 và 1990 đã dẫn đến việc tạo ra các phương pháp linh hoạt (agile). Các phương pháp này:

- Tập trung vào mã nguồn thay vì thiết kế.
- Dựa trên một phương pháp lặp lại trong quá trình phát triển phần mềm.
- Được thiết kế để cung cấp phần mềm hoạt động nhanh chóng và tiến hóa nhanh chóng để đáp ứng các yêu cầu thay đổi.

Mục tiêu của các phương pháp linh hoạt là giảm thiểu chi phí liên quan đến quy trình phần mềm (ví dụ: bằng cách giới hạn tài liệu) và có khả năng phản ứng nhanh chóng với các yêu cầu thay đổi mà không cần phải thực hiện quá nhiều công việc lại.

Agile manifesto

- *We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:*
  - *Individuals and interactions over processes and tools*
  - *Working software over comprehensive documentation*
  - *Customer collaboration over contract negotiation*
  - *Responding to change over following a plan*

Agile manifesto

✧ *Chúng tôi đang khám phá cách phát triển phần mềm tốt hơn bằng cách thực hiện nó và giúp đỡ người khác thực hiện nó. Qua công việc này, chúng tôi đã đánh giá cao:*

- *Cá nhân và tương tác hơn là quy trình và công cụ*
- *Phần mềm hoạt động hơn là tài liệu chi tiết*
- *Sự hợp tác với khách hàng hơn là đàm phán hợp đồng*
- *Phản ứng với sự thay đổi hơn là tuân theo kế hoạch*

The principles of agile methods

Principle	Description
Customer involvement	Customers should be closely involved throughout the development process. Their role is provide and prioritize new system requirements and to evaluate the iterations of the system.
Sự tham gia của khách hàng	Khách hàng nên tham gia chặt chẽ trong suốt quá trình phát triển. Vai trò của họ là cung cấp và ưu tiên các yêu cầu hệ thống mới và đánh giá các phiên bản của hệ thống.

The principles of agile methods

Principle	Description
Giao hàng tăng dần	Giao hàng tăng dần Phần mềm được phát triển theo từng phần với khách hàng chỉ định các yêu cầu cần được bao gồm trong mỗi phần tăng dần.
Incremental delivery	The software is developed in increments with the customer specifying the requirements to be included in each increment.

The principles of agile methods

Principle	Description
People not process	The skills of the development team should be recognized and exploited. Team members should be left to develop their own ways of working without prescriptive processes.
Embrace change	Expect the system requirements to change and so design the system to accommodate these changes.
Maintain simplicity	Focus on simplicity in both the software being developed and in the development process. Wherever possible, actively work to eliminate complexity from the system.

The principles of agile methods

Principle	Description
Con người quan trọng hơn quy trình	Con người quan trọng hơn quy trình Các kỹ năng của nhóm phát triển nên được công nhận và tận dụng. Các thành viên trong nhóm nên được để tự phát triển cách làm việc của họ mà không cần có các quy trình chi tiết.
Chấp nhận sự thay đổi	Mong đợi các yêu cầu hệ thống thay đổi và thiết kế hệ thống để chứa đựng những thay đổi này.
Giữ đơn giản	Tập trung vào sự đơn giản cả trong phần mềm đang được phát triển và trong quy trình phát triển. Bất cứ khi nào có thể, hãy tích cực làm việc để loại bỏ sự phức tạp khỏi hệ thống.

Agile method applicability

- Product development where a software company is developing a small or medium-sized product for sale.
- Custom system development within an organization, where there is a clear commitment from the customer to become involved in the development process and where there are not a lot of external rules and regulations that affect the software.
- Because of their focus on small, tightly-integrated teams, there are problems in scaling agile methods to large systems.

Agile method applicability

- Phát triển sản phẩm: Đây là quá trình phát triển sản phẩm phần mềm của một công ty với quy mô nhỏ hoặc trung bình để bán ra thị trường.
- Phát triển hệ thống tùy chỉnh trong một tổ chức: Đây là quá trình phát triển hệ thống theo yêu cầu cụ thể của một tổ chức, trong đó có sự cam kết rõ ràng từ phía khách hàng tham gia vào quá trình phát triển và không có quá nhiều quy tắc và quy định bên ngoài ảnh hưởng đến phần mềm.
- Vì tập trung vào các nhóm nhỏ, tích hợp chặt chẽ, nên có những vấn đề khi mở rộng các phương pháp linh hoạt cho các hệ thống lớn.

Problems with agile methods

- It can be difficult to keep the interest of customers who are involved in the process.
- Team members may be unsuited to the intense involvement that characterises agile methods.
- Prioritising changes can be difficult where there are multiple stakeholders.
- Maintaining simplicity requires extra work.
- Contracts may be a problem as with other approaches to iterative development.

Problems with agile methods

- Có thể khó khăn trong việc giữ sự quan tâm của khách hàng tham gia vào quá trình.
- Các thành viên trong nhóm có thể không phù hợp với sự tham gia mạnh mẽ mà đặc trưng của các phương pháp linh hoạt.
- Ưu tiên thay đổi có thể khó khăn khi có nhiều bên liên quan.
- Việc duy trì sự đơn giản yêu cầu công việc phụ trợ.
- Hợp đồng có thể là vấn đề như với các phương pháp phát triển theo phương pháp lặp lại khác.

Agile methods and software maintenance

- Most organizations spend more on maintaining existing software than they do on new software development. So, if agile methods are to be successful, they have to support maintenance as well as original development.
- Key issues:
  - Are systems that are developed using an agile approach maintainable, given the emphasis in the development process of minimizing formal documentation?
- Problems may arise if original development team cannot be maintained.

Agile methods and software maintenance

- Hầu hết các tổ chức chi tiêu nhiều hơn cho việc duy trì phần mềm hiện tại hơn là cho việc phát triển phần mềm mới. Vì vậy, nếu các phương pháp linh hoạt muốn thành công, chúng phải hỗ trợ việc duy trì cũng như phát triển gốc.
- Các vấn đề chính:
  - - Liệu hệ thống được phát triển bằng cách sử dụng phương pháp linh hoạt có thể được duy trì không, khi trong quá trình phát triển có sự tập trung vào việc giảm thiểu tài liệu chính thức?
- Có thể xảy ra vấn đề nếu nhóm phát triển gốc không thể được duy trì.

Plan-driven and agile development

✧ Plan-driven development

- A plan-driven approach to software engineering is based around separate development stages with the outputs to be produced at each of these stages planned in advance.

✧ Agile development

- Specification, design, implementation and testing are inter-leaved and the outputs from the development process are decided through a process of negotiation during the software development process.

Plan-driven and agile development

ĐẠI HỌC TÔN ĐỨC THẮNG
TON DUC THANG UNIVERSITY

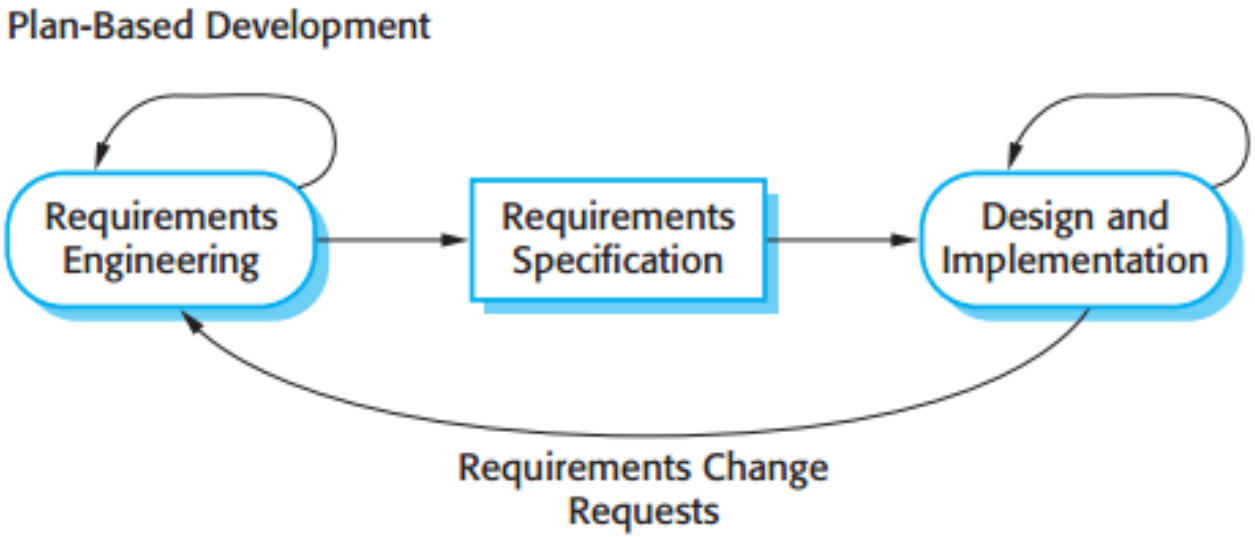
✧ Phát triển dựa trên kế hoạch

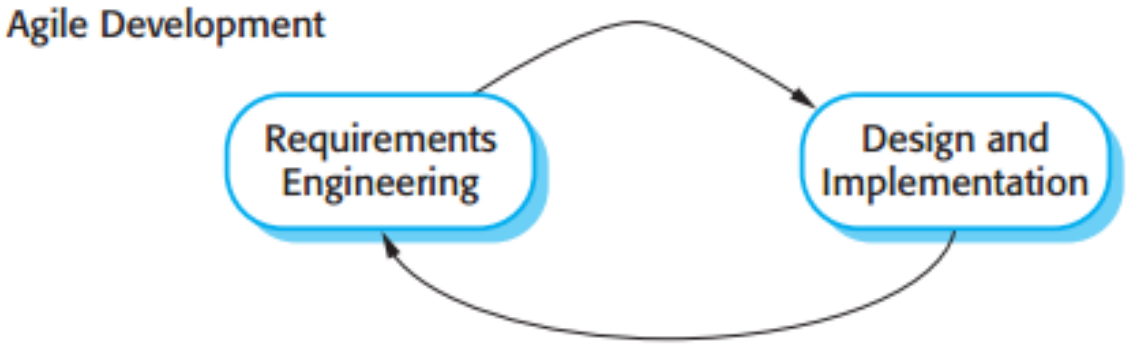
- Một phương pháp phát triển phần mềm dựa trên kế hoạch được xây dựng xung quanh các giai đoạn **phát triển riêng biệt** với các kết quả cần được sản xuất tại mỗi giai đoạn này được lập kế hoạch trước.

✧ Phát triển linh hoạt

- Quá trình phát triển phần mềm gồm các bước: **đặc tả, thiết kế, triển khai và kiểm thử được xen kẽ** và các kết quả từ quá trình phát triển phần mềm được quyết định thông qua quá trình đàm phán trong **quá trình phát triển phần mềm.**

Plan-driven and agile specification





Technical, human, organizational issues

- Most projects include elements of plan-driven and agile processes. Deciding on the balance depends on:
  - Is it important to have a very detailed specification and design before moving to implementation? If so, you probably need to use a plan-driven approach.
  - Is an incremental delivery strategy, where you deliver the software to customers and get rapid feedback from them, realistic? If so, consider using agile methods.
  - How large is the system that is being developed? Agile methods are most effective when the system can be developed with a small co-located team who can communicate informally. This may not be possible for large systems that require larger development teams so a plan-driven approach may have to be used.

Technical, human, organizational issues

- Hầu hết các dự án bao gồm các yếu tố của quá trình dựa trên kế hoạch và linh hoạt. Việc quyết định sự cân bằng phụ thuộc vào:

  1. Có quan trọng để có một đặc tả và thiết kế rất chi tiết trước khi di chuyển sang triển khai không? Nếu có, có thể cần sử dụng một phương pháp dựa trên kế hoạch.

  2. Chiến lược giao hàng tăng dần, trong đó bạn giao phần mềm cho khách hàng và nhận phản hồi nhanh chóng từ họ, có thực tế không? Nếu có, hãy xem xét việc sử dụng các phương pháp linh hoạt.

  3. Hệ thống đang được phát triển có lớn không? Phương pháp linh hoạt hiệu quả nhất khi hệ thống có thể được phát triển với một nhóm nhỏ cùng địa điểm làm việc có thể giao tiếp một cách không chính thức. Điều này có thể không thể thực hiện được đối với các hệ thống lớn cần đội ngũ phát triển lớn hơn nên có thể phải sử dụng phương pháp dựa trên kế hoạch.

Technical, human, organizational issues

- What type of system is being developed?
  - Plan-driven approaches may be required for systems that require a lot of analysis before implementation (e.g. real-time system with complex timing requirements).
- What is the expected system lifetime?
  - Long-lifetime systems may require more design documentation to communicate the original intentions of the system developers to the support team.
- How is the development team organized?
  - If the development team is distributed or if part of the development is being outsourced, then you may need to develop design documents to communicate across the development teams.

Technical, human, organizational issues

- ✧ Loại hệ thống đang được phát triển là gì?
  - Các phương pháp dựa trên kế hoạch có thể được yêu cầu cho các hệ thống đòi hỏi nhiều phân tích trước khi triển khai (ví dụ: hệ thống thời gian thực với yêu cầu thời gian phức tạp).
- ✧ Tuổi thọ dự kiến của hệ thống là bao lâu?
  - Các hệ thống có tuổi thọ lâu có thể yêu cầu nhiều tài liệu thiết kế hơn để truyền đạt ý định ban đầu của các nhà phát triển hệ thống đến nhóm hỗ trợ.
- ✧ Nhóm phát triển được tổ chức như thế nào?
  - Nếu nhóm phát triển phân tán hoặc nếu một phần của quá trình phát triển được giao cho bên thứ ba, thì bạn có thể cần phải phát triển các tài liệu thiết kế để truyền đạt thông tin qua các nhóm phát triển.

Technical, human, organizational issues

- Are there cultural or organizational issues that may affect the system development?
  - Traditional engineering organizations have a culture of plan-based development, as this is the norm in engineering.
- How good are the designers and programmers in the development team?
  - It is sometimes argued that agile methods require higher skill levels than plan-based approaches in which programmers simply translate a detailed design into code
- Is the system subject to external regulation?
  - If a system has to be approved by an external regulator (e.g. the FAA approve software that is critical to the operation of an aircraft) then you will probably be required to produce detailed documentation as part of the system safety case.

Technical, human, organizational issues

1.Có những vấn đề văn hóa hoặc tổ chức nào có thể ảnh hưởng đến việc phát triển hệ thống?

Các tổ chức kỹ thuật truyền thống có văn hóa phát triển dựa trên kế hoạch, vì đây là tiêu chuẩn trong ngành kỹ thuật.

2. Khả năng của các nhà thiết kế và lập trình viên trong nhóm phát triển là tốt như thế nào?

Đôi khi được cho là các phương pháp linh hoạt yêu cầu trình độ kỹ năng cao hơn so với các phương pháp dựa trên kế hoạch, trong đó các lập trình viên chỉ đơn giản là dịch một thiết kế chi tiết thành mã code.

3. Hệ thống có chịu sự quy định từ bên ngoài không?

Nếu một hệ thống phải được phê duyệt bởi một cơ quan quy định bên ngoài (ví dụ: FAA phê duyệt phần mềm quan trọng đối với hoạt động của một chiếc máy bay) thì bạn có thể sẽ phải tạo ra tài liệu chi tiết như một phần của bảo đảm an toàn của hệ thống.

Extreme programming

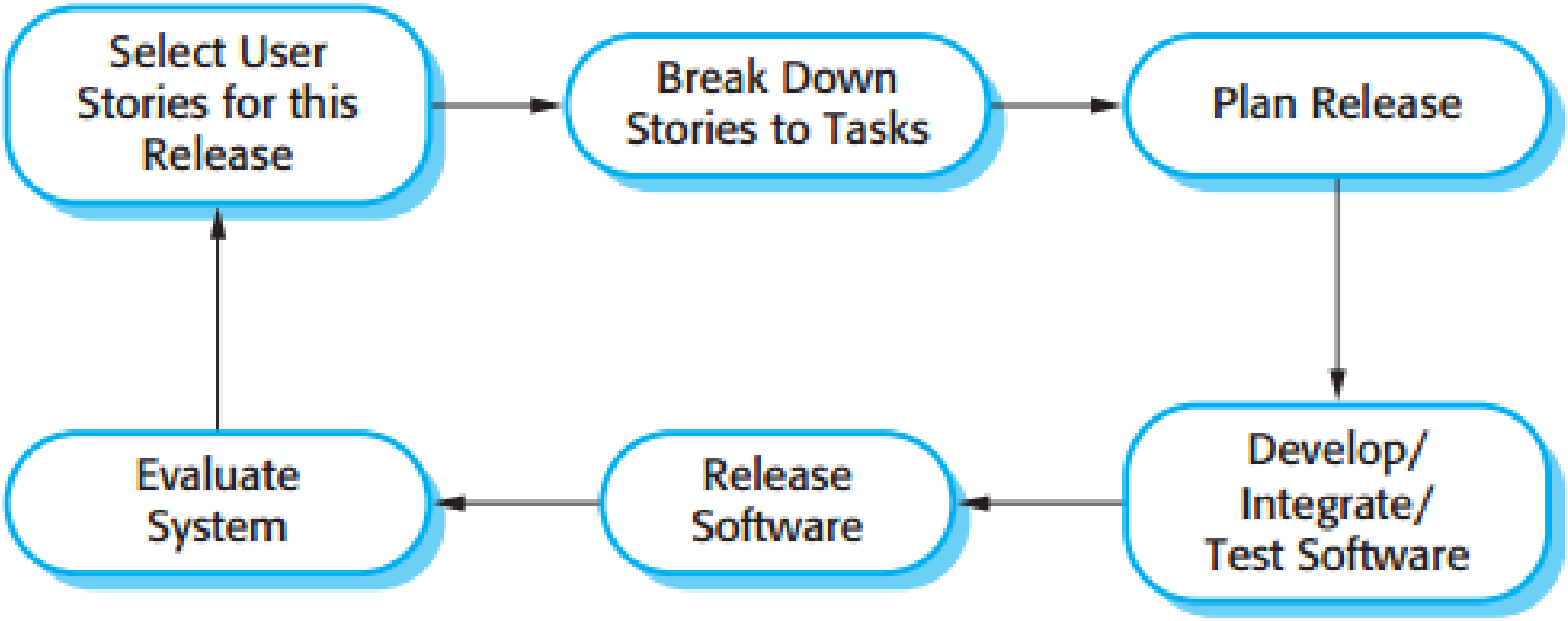
- Perhaps the best-known and most widely used agile method.
- Extreme Programming (XP) takes an 'extreme' approach to iterative development.
  - New versions may be built several times per day;
  - Increments are delivered to customers every 2 weeks;
  - All tests must be run for every build and the build is only accepted if tests run successfully.

Extreme programming

- Extreme Programming (XP) là một trong những phương pháp linh hoạt nổi tiếng nhất và phổ biến nhất.
- Extreme Programming (XP) thực hiện một cách tiếp cận "cực đoan" đối với phát triển lặp lại.

  - Các phiên bản mới có thể được xây dựng nhiều lần trong một ngày;
  - Các phần tăng được giao cho khách hàng mỗi 2 tuần;
  - Tất cả các bài kiểm tra phải được chạy cho mỗi phiên bản và phiên bản chỉ được chấp nhận nếu các bài kiểm tra chạy thành công.

The extreme programming release cycle



Requirements scenarios

- In XP, a customer or user is part of the XP team and is responsible for making decisions on requirements.
- User requirements are expressed as scenarios or user stories.
- These are written on cards and the development team break them down into implementation tasks. These tasks are the basis of schedule and cost estimates.
- The customer chooses the stories for inclusion in the next release based on their priorities and the schedule estimates.

Requirements scenarios

- Trong Extreme Programming (XP), một khách hàng hoặc người dùng là một phần của nhóm XP và chịu trách nhiệm quyết định về yêu cầu.
- Yêu cầu của người dùng được biểu diễn dưới dạng kịch bản hoặc câu chuyện của người dùng.
- Những câu chuyện này được viết trên thẻ và nhóm phát triển chia chúng thành các nhiệm vụ triển khai. Những nhiệm vụ này là cơ sở của việc ước lượng lịch trình và chi phí.
- Khách hàng chọn các câu chuyện để bao gồm trong bản phát hành tiếp theo dựa trên ưu tiên của họ và các ước lượng về lịch trình.

Extreme programming practices (a) (Self study)

Principle or practice	Description
Incremental planning	Requirements are recorded on story cards and the stories to be included in a release are determined by the time available and their relative priority. The developers break these stories into development 'Tasks'. See Figures 3.5 and 3.6.
Small releases	The minimal useful set of functionality that provides business value is developed first. Releases of the system are frequent and incrementally add functionality to the first release.

Extreme programming practices (a)

Principle or practice	Description
Simple design	Enough design is carried out to meet the current requirements and no more.
Test-first development	An automated unit test framework is used to write tests for a new piece of functionality before that functionality itself is implemented.
Refactoring[improve]	All developers are expected to refactor the code continuously as soon as possible code improvements are found. This keeps the code simple and maintainable.

Extreme programming practices (b)

Pair programming	Developers work in pairs, checking each other's work and providing the support to always do a good job.
Collective ownership	The pairs of developers work on all areas of the system, so that no islands of expertise develop and all the developers take responsibility for all of the code. Anyone can change anything.
Continuous integration	As soon as the work on a task is complete, it is integrated into the whole system. After any such integration, all the unit tests in the system must pass.

Extreme programming practices (b)

Sustainable pace	Large amounts of overtime are not considered acceptable
On-site customer	A representative of the end-user of the system (the customer) should be available full time for the use of the XP team. In an extreme programming process, the customer is a member of the development team and is responsible for bringing system requirements to the team for implementation.

XP and change

- Conventional wisdom in software engineering is to design for change. It is worth spending time and effort anticipating changes as this reduces costs later in the life cycle.
- XP, however, maintains that this is not worthwhile as changes cannot be reliably anticipated.
- Rather, it proposes constant code improvement (refactoring) to make changes easier when they have to be implemented.

XP and change

- Truyền thống trong kỹ thuật phần mềm là thiết kế cho sự thay đổi. Đáng giá khi bỏ thời gian và công sức để dự đoán các thay đổi vì điều này giảm chi phí sau này trong vòng đời của sản phẩm.
- Tuy nhiên, XP khẳng định rằng việc này không đáng giá vì các thay đổi không thể được dự đoán một cách đáng tin cậy.
- Thay vào đó, nó đề xuất cải tiến mã code liên tục (refactoring) để làm cho việc thực hiện các thay đổi dễ dàng hơn khi cần thiết.

Refactoring

- Programming team look for possible software improvements and make these improvements even where there is no immediate need for them.
- This improves the understandability of the software and so reduces the need for documentation.
- Changes are easier to make because the code is well-structured and clear.
- However, some changes requires architecture refactoring and this is much more expensive.

Refactoring

- Nhóm lập trình tìm kiếm các cải tiến phần mềm có thể có và thực hiện những cải tiến này ngay cả khi không có nhu cầu cụ thể cho chúng.
- Điều này cải thiện khả năng hiểu được của phần mềm và giảm nhu cầu về tài liệu.
- Các thay đổi trở nên dễ dàng hơn vì mã code được cấu trúc tốt và rõ ràng.
- Tuy nhiên, một số thay đổi yêu cầu cải tiến kiến trúc và điều này tốn kém hơn nhiều.

Examples of refactoring

- Re-organization of a class hierarchy to remove duplicate code.
- Tidying up and renaming attributes and methods to make them easier to understand.
- The replacement of inline code with calls to methods that have been included in a program library.
- For Example (5'):
  - Suggestion: sort array in C#, ...

Examples of refactoring

- Re-organization of a class hierarchy to remove duplicate code.
- Tidying up and renaming attributes and methods to make them easier to understand.
- The replacement of inline code with calls to methods that have been included in a program library.
- For Example (5'):
  - Suggestion: sort array in C#, ...

Key points

- Agile methods are incremental development methods that focus on rapid development, frequent releases of the software, reducing process overheads and producing high-quality code. They involve the customer directly in the development process.
- The decision on whether to use an agile or a plan-driven approach to development should depend on the type of software being developed, the capabilities of the development team and the culture of the company developing the system.
- Extreme programming is a well-known agile method that integrates a range of good programming practices such as frequent releases of the software, continuous software improvement and customer participation in the development team.

Scrum

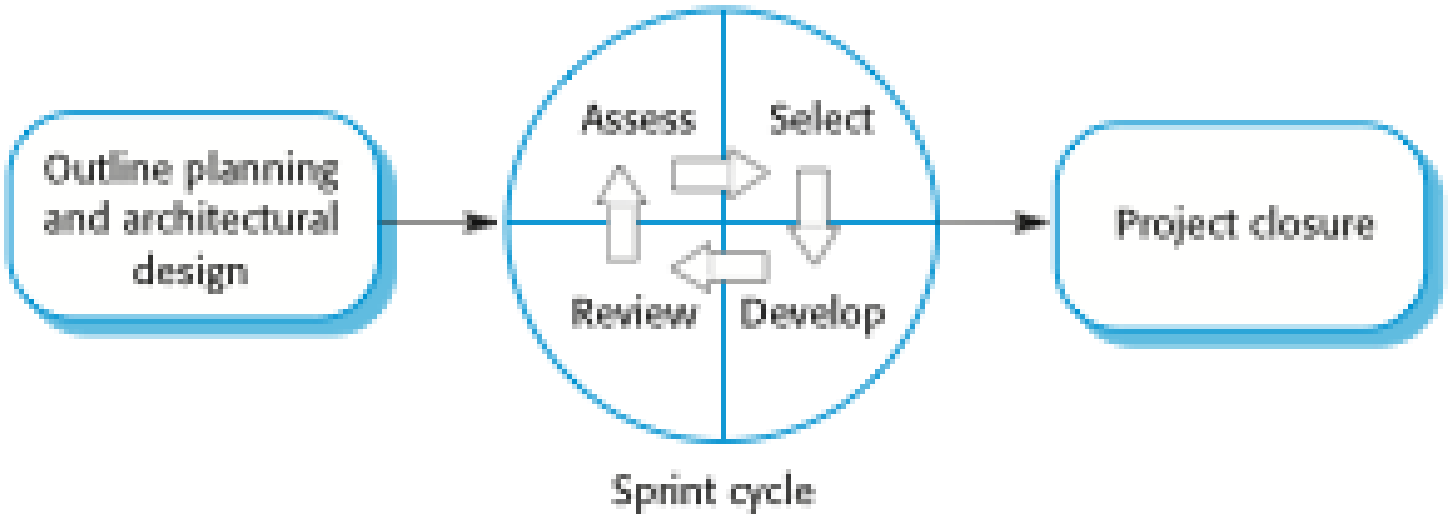
- The Scrum approach is a general agile method but its focus is on managing iterative development rather than specific agile practices.
- There are three phases in Scrum.
  - The initial phase is an outline planning phase where you establish the general objectives for the project and design the software architecture.
  - This is followed by a series of sprint cycles, where each cycle develops an increment of the system.
  - The project closure phase wraps up the project, completes required documentation such as system help frames and user manuals and assesses the lessons learned from the project.

Scrum

- Phương pháp Scrum là một phương pháp linh hoạt chung nhưng trọng tâm của nó là quản lý phát triển lặp lại thay vì các thực hành cụ thể của phương pháp linh hoạt.
- Scrum có ba giai đoạn:

1. Giai đoạn lập kế hoạch đại bộ, trong đó bạn thiết lập các mục tiêu chung cho dự án và thiết kế kiến trúc phần mềm.
2. Tiếp theo là một loạt các chu kỳ sprint, trong đó mỗi chu kỳ phát triển một phần tăng của hệ thống.
3. Giai đoạn đóng dự án kết thúc dự án, hoàn thành tài liệu cần thiết như hệ thống trợ giúp và hướng dẫn sử dụng và đánh giá những bài học rút ra từ dự án.

The Scrum process



The Sprint cycle

- Sprints are fixed length, normally 2–4 weeks. They correspond to the development of a release of the system in XP.
- The starting point for planning is the product backlog, which is the list of work to be done on the project.
- The selection phase involves all of the project team who work with the customer to select the features and functionality to be developed during the sprint.

The Sprint cycle

- Các Sprint có độ dài cố định, thường là từ 2 đến 4 tuần. Chúng tương ứng với việc phát triển một phiên bản của hệ thống trong XP.
- Điểm khởi đầu cho việc lập kế hoạch là Product Backlog, đó là danh sách công việc cần được thực hiện trong dự án.
- Giai đoạn lựa chọn liên quan đến toàn bộ nhóm dự án làm việc với khách hàng để chọn lọc các tính năng và chức năng cần được phát triển trong Sprint.

The Sprint cycle

- Once these are agreed, the team organize themselves to develop the software. During this stage the team is isolated from the customer and the organization, with all communications channelled through the so-called 'Scrum master'.
- The role of the Scrum master is to protect the development team from external distractions.
- At the end of the sprint, the work done is reviewed and presented to stakeholders. The next sprint cycle then begins.

The Sprint cycle

- Sau khi các tính năng được thống nhất, nhóm tự tổ chức để phát triển phần mềm. Trong giai đoạn này, nhóm được cô lập khỏi khách hàng và tổ chức, với tất cả thông tin được truyền qua người gọi là 'Scrum master'.
- Vai trò của Scrum master là bảo vệ nhóm phát triển khỏi những sự làm phiền từ bên ngoài.
- Ở cuối chu kỳ sprint, công việc được xem xét và trình bày trước các bên liên quan. Sau đó, chu kỳ sprint tiếp theo bắt đầu.

Teamwork in Scrum

- The whole team attends short daily meetings where all team members share information, describe their progress since the last meeting, problems that have arisen and what is planned for the following day.
  - This means that everyone on the team knows what is going on and, if problems arise, can re-plan short-term work to cope with them.

Scrum benefits

- The product is broken down into a set of manageable and understandable chunks.
- Unstable requirements do not hold up progress.
- The whole team have visibility of everything and consequently team communication is improved.
- Customers see on-time delivery of increments and gain feedback on how the product works.
- Trust between customers and developers is established and a positive culture is created in which everyone expects the project to succeed.

Scrum benefits

- Sản phẩm được chia thành một tập hợp các phần quản lý và dễ hiểu.
- Yêu cầu không ổn định không làm chậm tiến độ.
- Toàn bộ nhóm có khả năng nhìn thấy mọi thứ và do đó, giao tiếp trong nhóm được cải thiện.
- Khách hàng thấy việc giao các phần tăng đúng hẹn và nhận được phản hồi về cách hoạt động của sản phẩm.
- Sự tin cậy giữa khách hàng và nhà phát triển được xây dựng và một văn hóa tích cực được tạo ra trong đó mọi người đều mong đợi dự án thành công.

XP vs Scrum

- Scrum is an agile project management methodology. It does not address the practices required to create "goods" of any kind, but instead gives us the *process* that will take us from the inception of a vision to the final product, regardless of the actual development process. The Scrum process doesn't tell you *how* to create quality. It shows you what the quality is, where your problems are, and challenges you to fix them.

XP vs Scrum

- Scrum là một **phương pháp quản lý dự án linh hoạt**. Nó không giải quyết các thực hành cần thiết để tạo ra "hàng hóa" bất kỳ loại nào, mà thay vào đó cung cấp cho chúng ta quy trình sẽ dẫn chúng ta **từ ý tưởng ban đầu đến sản phẩm cuối cùng**, bất kể quy trình phát triển thực sự là gì. Quy trình Scrum không nói cho bạn biết cách tạo ra chất lượng. Nó chỉ cho bạn biết chất lượng là gì, vấn đề của bạn đang ở đâu, và thách thức bạn để khắc phục chúng.

XP vs Scrum

- Extreme programing is an agile software development methodology. It gives us a process with which to create software in an agile and productive way. It deals with, though doesn't specialize in the *management* of the development process, and focuses mostly on the *engineering practices* required to deliver software, with quality.

XP vs Scrum

- Extreme Programming (XP) là một phương pháp phát triển phần mềm linh hoạt. Nó cung cấp cho chúng ta một quy trình để tạo ra phần mềm một cách linh hoạt và hiệu quả. Nó xử lý, mặc dù không chuyên sâu vào việc quản lý quy trình phát triển, và tập trung chủ yếu vào các thực hành kỹ thuật cần thiết để cung cấp phần mềm, với chất lượng cao.