Công nghệ Phần mềm
(Software Engineering)

PHƯƠNG PHÁP AGILE

NỘI DUNG

1. Khái niệm
2. Các nguyên lý cơ bản của phương pháp Agile
3. Ưu, nhược điểm của phương pháp Agile
4. Extreme Programming
5. Scrum
6. Các phương pháp Agile khác

MỤC TIÊU CỦA BÀI HỌC

- Biết phương pháp phát triển phần mềm nhanh (Agile)
- Hiểu các nguyên lý cơ bản của phương pháp agile
- Biết một số phương pháp phát triển phần mềm nhanh

1. KHÁI NIỆM

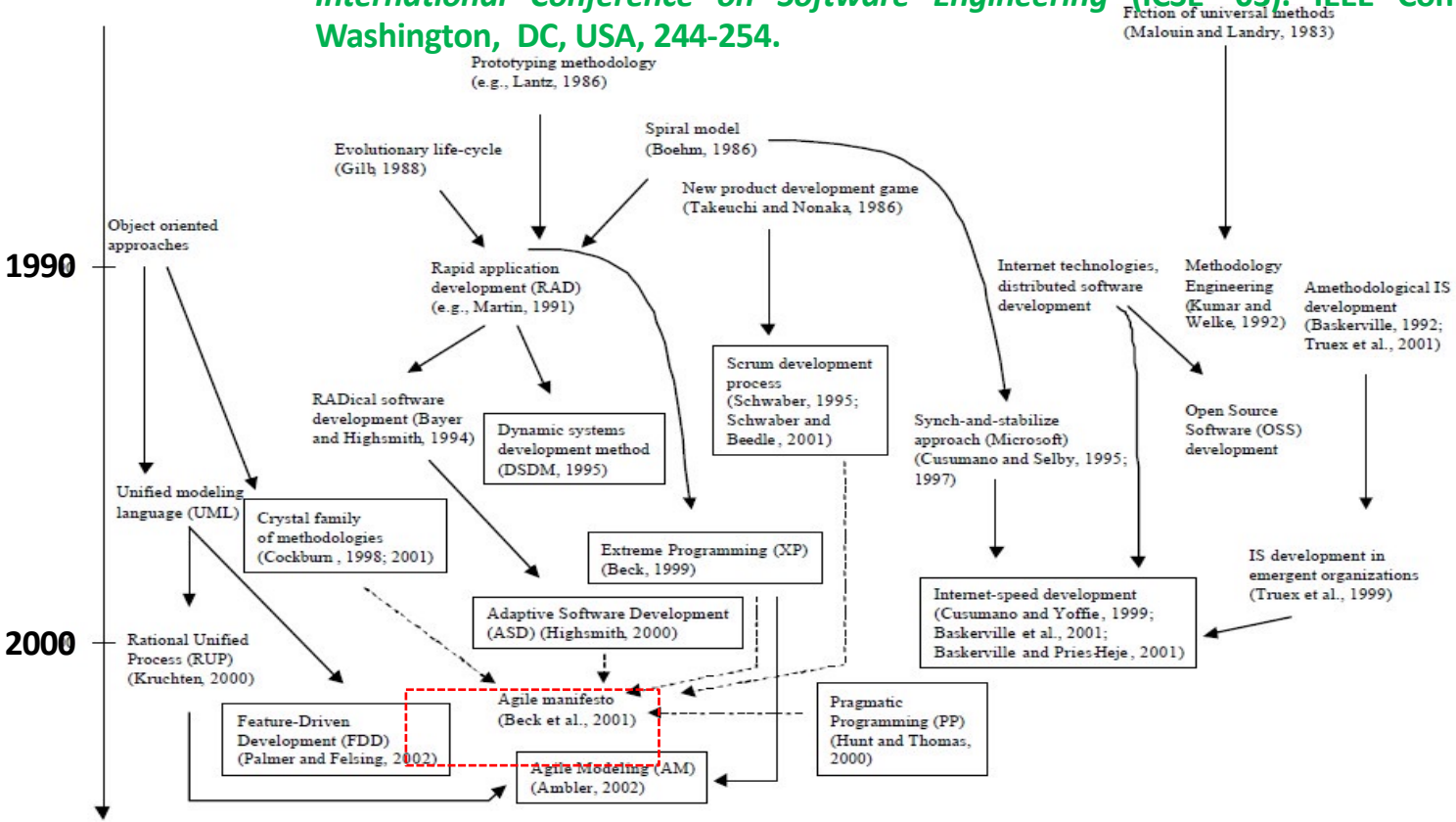
- Phương pháp Agile:
  - phản ứng hiệu quả (nhanh chóng và thích ứng) với sự thay đổi
  - kết hợp quá trình tạo mẫu và dựa trên thử nghiệm
  - có sự phát triển liên tục của lập trình
  - liên tục xác nhận các yêu cầu của khách hàng
  - mục tiêu: nhanh chóng phân phối phần mềm

LỊCH SỬ CỦA AGILE

- Agile không phải là một bộ công cụ hay một phương pháp duy nhất, mà là một triết lý được đưa ra vào năm 2001.
- Agile thay đổi đáng kể từ cách tiếp cận phát triển phần mềm nặng theo hướng tài liệu (như mô hình thác nước).

LỊCH SỬ CỦA AGILE

Pekka Abrahamsson, Juhani Warsta, Mikko T. Siponen, and Jussi Ronkainen. 2003. New directions on agile methods: a comparative analysis. In *Proceedings of the 25th International Conference on Software Engineering* (ICSE '03). IEEE Computer Society, Washington, DC, USA, 244-254.



NỘI DUNG

1. Khái niệm
2. Các nguyên lý cơ bản của phương pháp Agile
3. Ưu, nhược điểm của phương pháp Agile
4. Extreme Programming
5. Scrum
6. Các phương pháp Agile khác

2. CÁC NGUYÊN LÝ CƠ BẢN CỦA PHƯƠNG PHÁP AGILE

- Cách tốt hơn để phát triển phần mềm là làm và giúp người khác làm. Từ đó, các giá trị sau sẽ được nhận ra:
  - **Các cá nhân và tương tác** qua **các quy trình và công cụ**
  - **Phần mềm làm việc** dựa trên **tài liệu toàn diện**
  - **Sự hợp tác của khách hàng** trong quá trình **đàm phán**
  - **Đáp ứng sự thay đổi** so với việc **tuân theo kế hoạch**
- Các mục **bên trái** được coi trọng hơn bên phải.

http://agilemanifesto.org/

2. AGILE: 12 NGUYÊN LÝ

1. Ưu tiên cao nhất là làm hài lòng khách hàng thông qua việc phân phối sớm và liên tục các phần mềm có giá trị.
2. Hoan nghênh các yêu cầu thay đổi, ngay cả trong giai đoạn muộn của việc phát triển. Các quy trình nhanh khai thác sự thay đổi vì lợi thế cạnh tranh của khách hàng.
3. Cung cấp sản phẩm phần mềm thường xuyên, từ vài tuần đến vài tháng, ưu tiên khoảng thời gian ngắn hơn.
4. Người kinh doanh và nhà phát triển phải làm việc cùng nhau hàng ngày trong suốt dự án.
5. Xây dựng các dự án xung quanh những cá nhân có động lực. Cung cấp cho họ môi trường và sự hỗ trợ mà họ cần, và tin tưởng để họ hoàn thành công việc.
6. Phương pháp hiệu quả nhất để truyền tải thông tin đến và trong nhóm phát triển là trò chuyện trực tiếp.

2. AGILE: 12 NGUYÊN LÝ

7. Phần mềm làm việc là thước đo chính của sự tiến bộ.
8. Các quy trình Agile thúc đẩy sự phát triển bền vững. Các nhà tài trợ, nhà phát triển và người dùng sẽ có thể duy trì một tốc độ phát triển liên tục.
9. Sự quan tâm liên tục, sự xuất sắc về kỹ thuật và thiết kế tốt giúp tăng cường sự nhanh nhẹn.
10. Sự đơn giản - nghệ thuật tối đa hóa khối lượng công việc chưa hoàn thành - là điều cần thiết.
11. Các kiến trúc, yêu cầu và thiết kế tốt nhất xuất hiện từ các nhóm dự án.
12. Theo định kỳ, các nhóm phản ánh về cách trở nên hiệu quả hơn, sau đó điều chỉnh hoạt động của mình sao cho cho phù hợp.

NỘI DUNG

1. Khái niệm
2. Các nguyên lý cơ bản của phương pháp Agile
3. Ưu, nhược điểm của phương pháp Agile
4. Extreme Programming
5. Scrum
6. Các phương pháp Agile khác

ƯU, NHƯỢC ĐIỂM CỦA PHƯƠNG PHÁP AGILE

- Phù hợp với những dự án nhỏ thường có những yêu cầu không được xác định rõ ràng (có thể thay đổi thường xuyên).
- Khách hàng có thể được xem trước từng phần dự án trong quá trình phát triển, luôn sẵn sàng cho bất kỳ thay đổi từ phía khách hàng.
- Agile chia dự án thành những phần nhỏ và giao cho nhóm phát triển, hàng ngày tất cả mọi người phải họp trong khoảng thời gian ngắn để thảo luận về tiến độ và giải quyết những vấn đề nảy sinh nếu có nhằm đảm bảo đúng quy trình phát triển dự án.
- Tỉ lệ thành công của các dự án sử dụng Agile thường cao hơn các quy trình khác.

ƯU, NHƯỢC ĐIỂM CỦA PHƯƠNG PHÁP AGILE

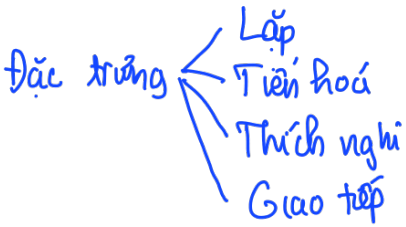
- Khó xác định về loại dự án phần mềm nào phù hợp nhất cho cách tiếp cận Agile.
- Nhiều tổ chức lớn gặp khó khăn trong việc chuyển từ phương pháp truyền thống sang một phương pháp Agile (linh hoạt).
- Khi Agile có rủi ro?:
  - Phát triển quy mô lớn (> 20 nhà phát triển)
  - Phát triển phân tán (các nhóm không nằm chung)
  - Khách hàng hoặc người liên hệ không đáng tin cậy
  - Bắt buộc quy trình nhanh trong nhóm phát triển
  - Nhà phát triển thiếu kinh nghiệm

Duy trì sự quan tâm, tham gia của khách hàng khó

Tạo ra áp lực, cản trở nỗ lực cao của nhân sự

Nội dung

1. Khái niệm
2. Các nguyên lý cơ bản của phương pháp Agile
3. Ưu, nhược điểm của phương pháp Agile
4. Extreme Programming
5. Scrum
6. Các phương pháp Agile khác



Extreme Programming (XP)

XP is not this *extreme*!

Extreme Programming (XP)

- Không thể lập trình cho đến khi biết mình đang làm gì.
- Cần khắc phục ở giai đoạn đầu tiên khi hệ thống đi vào sản xuất càng nhanh càng tốt. Tuy nhiên, mọi dự án đều phải bắt đầu từ điểm nào đó.
- Kết hợp phân tích tổng thể dưới dạng các vấn đề. Mỗi vấn đề phải theo định hướng kinh doanh, có thể kiểm tra và ước tính được.
- Một tháng là một khoảng thời gian dài để đưa ra những vấn đề cho một dự án.

Embracing Change with Extreme Programming, Beck, 1999

Extreme Programming (XP)

❖ Xác định thời điểm lập trình?

- Khách hàng chọn bản phát hành tiếp theo tính năng có giá trị nhất (được gọi là các vấn đề trong XP) trong số tất cả các vấn đề có thể có, xác định bởi chi phí của các vấn đề và tốc độ của nhóm trong việc cài đặt chúng.
- Khách hàng xác định vấn đề của lần lặp tiếp theo bằng cách chọn những vấn đề có giá trị nhất còn lại trong bản phát hành, được thông báo lại bằng chi phí của và tốc độ của nhóm.

Embracing Change with Extreme Programming, Beck, 1999

Extreme Programming (XP)

- Xác định thời điểm lập trình?
  - Các lập trình viên nhận trách nhiệm cho các nhiệm vụ chi tiết.
  - Xác định các trường hợp kiểm thử để chứng minh rằng nhiệm vụ đã hoàn thành.
  - Làm việc với đối tác để chạy các trường hợp kiểm thử, đồng thời cải tiến để duy trì một thiết kế đơn giản nhất có thể cho toàn bộ hệ thống.

Lập trình cặp đôi
lập trình đi kèm vs kiểm thử

Embracing Change with Extreme Programming, Beck, 1999

Extreme Programming (XP)

- Để XP thành công, cần có kiến thức chuyên môn.
  - Xây dựng, mô tả các vấn đề cần giải quyết của hệ thống đang được xây dựng.
  - Không mô tả một giải pháp, sử dụng ngôn ngữ kỹ thuật hoặc chứa các yêu cầu truyền thống.
  - Chuyển vấn đề thành mã.

Extreme Programming (XP)

TDD test driven development

- Biến thành mã thử nghiệm
- Kiểm thử đơn vị là trọng tâm của XP, trong đó hai điểm xoay quanh các chiến lược kiểm thử thông thường giúp kiểm tra hiệu quả hơn nhiều:
- Các lập trình viên viết các test riêng và cần viết trước khi viết mã.
- Thiết kế phát triển - Không được phá vỡ các thử nghiệm hiện có và cũng phải hỗ trợ phát triển gia tăng có thể mở rộng

* Các nét chính của XP
  - Phát triển "cực đoan" với các hoạt động lập trình và kiểm thử
  - Liên tục chuyển giao các phiên bản sản phẩm mới
  - Phải thực hiện kiểm thử tất cả các test cho mỗi phiên bản và chỉ chấp nhận nếu như tất cả các test thành công

* Lập trình đôi
  - Trong XP, lập trình viên làm việc theo từng cặp
    + Cùng ngồi viết mã
    + Nhờ vậy cùng làm chủ phần mã và lan truyền kiến thức cho cả đội
    + Khuyến khích refactoring
  - + Giảm thiểu rủi ro khi 1 thành viên rời nhóm

* Ưu điểm của XP
  - Cải thiện tinh đồng đội
  - Nâng cao năng lực
  - Làm cho công việc thú vị
  - TDD : cải thiện chất lượng
  - Cung cấp ni cô cụ
  - Cho khách hàng khả năng xem xét
  - K° lãng phí t/gian
* Nhược điểm
  - H/động thiết kế k° thể hiện rõ ràng
  - Rủi ro
  - Khó viết đc các kiểm thử tốt
  - Lặp → giảm chất lượng
  - Cần làm thường xuyên

Thg thường : y/cầu → lập trình → kiểm thử
TDD: y/cầu → xdựng kiểm thử → Lập trình
↳ thường ↑ theo vg lặp, chia nhỏ yêu cầu để xử lý trong mỗi vg lặp

Ưu điểm của XP

- Nếu được hoàn thành tốt, XP cải thiện tính đồng đội.
- xây dựng năng lực thực sự ở tất cả các thành viên trong nhóm.
- làm cho công việc thú vị.
- TDD (Test-driven development) hướng dẫn các nhà phát triển về cách viết mã chất lượng và cách cải thiện quan niệm về thiết kế; giúp cải thiện việc ước lượng.

Ưu điểm của XP

- Cung cấp nhiều công cụ cho quản lý , bao gồm khả năng dự đoán, tính linh hoạt của các nguồn lực, tính nhất quán và khả năng hiển thị về những gì thực sự đang diễn ra.
- Cung cấp cho khách hàng khả năng xem xét công ty có thể thực hiện công việc của mình hay không.
- Không lãng phí thời gian và không tạo ra nhiều tài liệu vô ích.

Nhược điểm của XP

- Thiết kế trở nên tiềm ẩn thay vì rõ ràng
- Dựa vào thiết kế là rủi ro
- Rất khó để viết các kiểm thử tốt
- Lặp lại quá thường xuyên có thể làm giảm chất lượng
- Để thực hiện tốt, cần phải làm thường xuyên

Nội dung

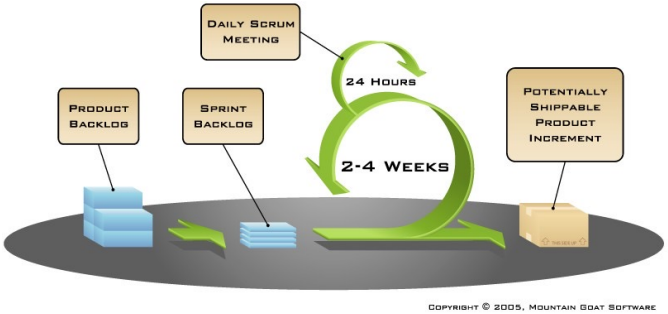
1. Khái niệm
2. Các nguyên lý cơ bản của phương pháp Agile
3. Ưu, nhược điểm của phương pháp Agile
4. Extreme Programming
5. Scrum
6. Các phương pháp Agile khác

Scrum

- ‘Scrum’ là thuật ngữ chỉ một nhóm người chơi với nhau trong môn bóng bầu dục.
- Trong phát triển phần mềm, công việc là đưa ra một bản phát hành.
- Nhiệm vụ: tăng tốc việc phát hành sản phẩm

Scrum

- Quá trình phát triển được chia thành một loạt các lần lặp được gọi là sprint.
- Trước mỗi sprint, các thành viên trong nhóm xác định các công việc còn tồn.
- Khi kết thúc sprint, nhóm sẽ xem xét để nêu rõ các kinh nghiệm và kiểm tra tiến độ.



* PP Scrum
  - Các công việc ↑ đc chia thành các gói (package)
  - Mỗi vòng lặp ↑ đc gọi là sprint
  - Đầu vào của mỗi vòng lặp gọi là backlog, gồm công việc hiện tại và công việc tồn đọng của vg lặp trc
  - Thường xuyên có các cuộc họp ngắn
  - Kết thúc sprint có rút kinh nghiệm và kiểm tra tiến độ

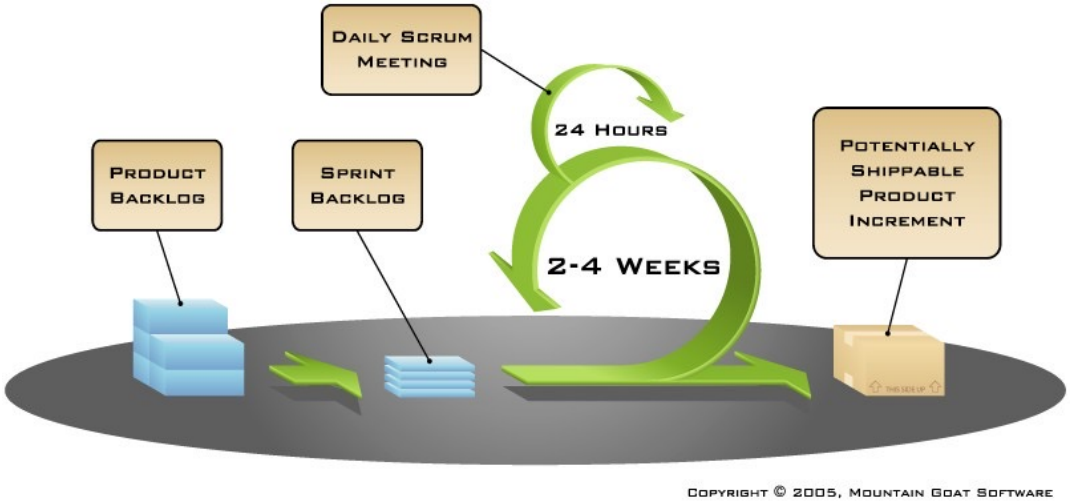
Các khái niệm chính

- **Burndown chart** biểu đồ cho thấy công việc còn tồn trong sprint (được cập nhật mỗi ngày) sử dụng để theo dõi tiến trình và quyết định khi nào các mục phải được loại bỏ khỏi sprint backlog và hoãn lại sprint tiếp theo.
- **Product backlog** là danh sách đầy đủ các yêu cầu - bao gồm lỗi, yêu cầu nâng cao, cải tiến khả năng sử dụng và hiệu suất
  - hiện không có trong bản phát hành sản phẩm.
- **ScrumMaster** là người chịu trách nhiệm quản lý dự án.
- **Sprint backlog** là danh sách các công việc được chỉ định cho một sprint, nhưng chưa hoàn thành.
  - Trong thực tế (phổ biến), không có công việc tồn nào của sprint phải mất hơn hai ngày để hoàn thành.
  - Sprint backlog giúp nhóm dự đoán mức độ nỗ lực cần thiết để hoàn thành một sprint.

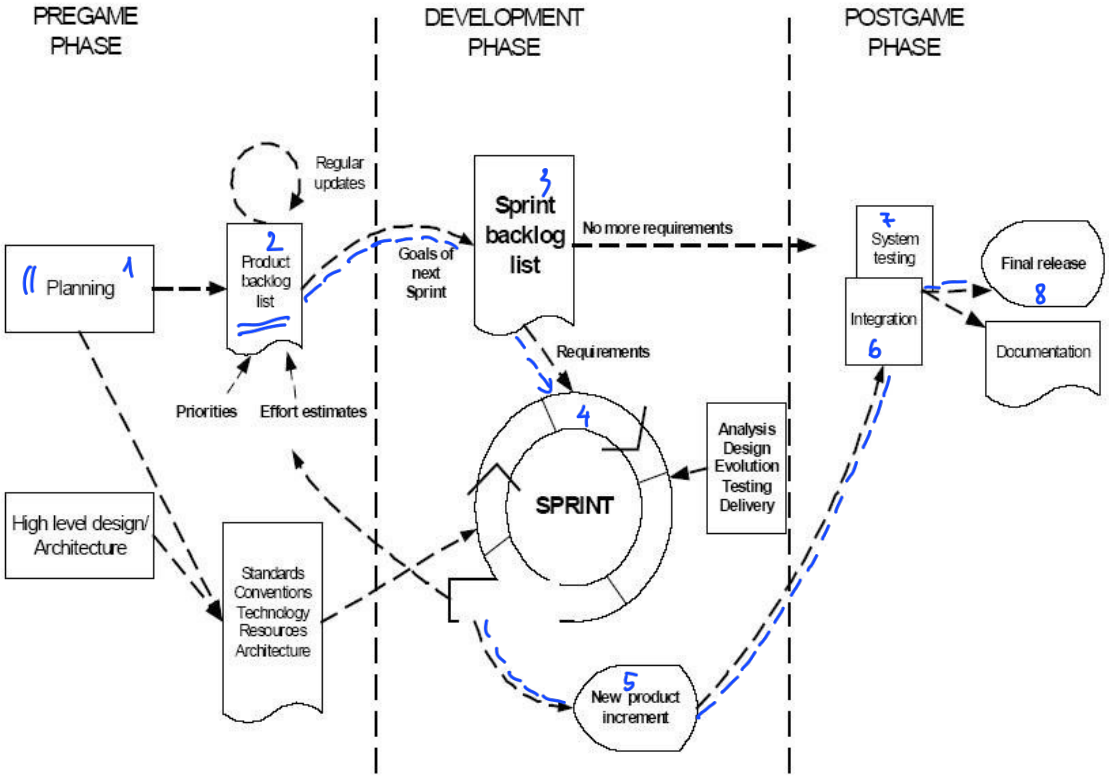
J Paul Gibson: Agile Methods October 2011

Scrum

- Các công việc tồn (là điểm mấu chốt): được điền vào trong pha lập kế hoạch của lần phát hành sản phẩm và xác định phạm vi của bản phát hành



Các pha của Scrums



Scrum Meetings

- Trong một spint, nhóm có một cuộc họp hàng ngày.
  - Mỗi thành viên mô tả công việc sẽ được thực hiện trong ngày hôm đó, tiến độ từ ngày hôm trước.
  - Để giữ cho các cuộc họp ngắn gọn, tất cả mọi người đều tham dự (họp đứng trong cùng một phòng).
- Khi các backlog đã được thực hiện để người dùng tin rằng bản phát hành đáng được đưa vào sản xuất, kết thúc quá trình phát triển, nhóm thực hiện kiểm tra tích hợp, đào tạo và làm tài liệu khi cần thiết để phát hành sản phẩm.

Scrum

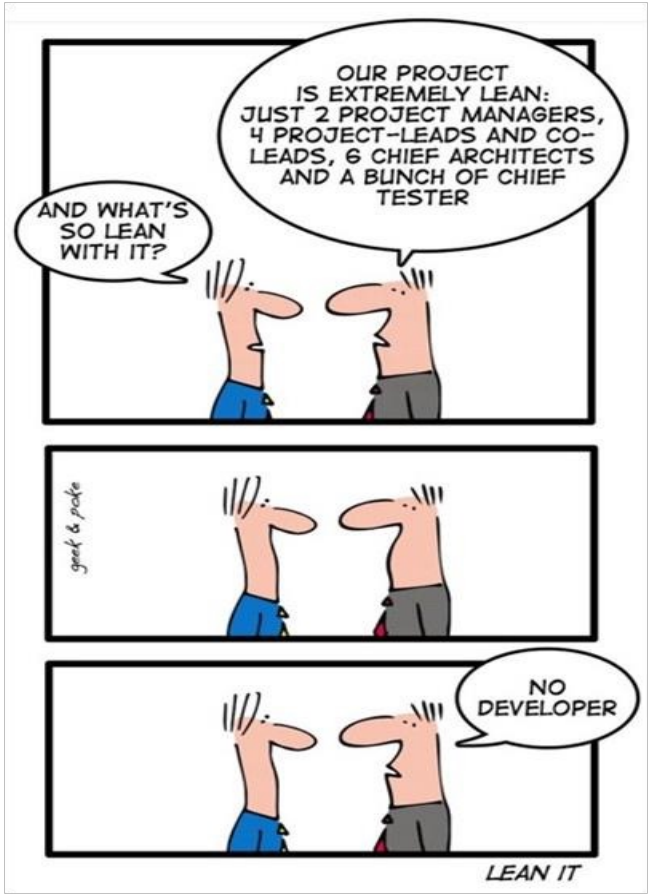
Ưu điểm (Pros)	Nhược điểm (Cons)
Scrum sử dụng Sprint nhỏ để chia hệ thống thành các thành phần nhỏ hơn một cách hiệu quả và phân chia cho các nhóm.	Scrum không hoạt động khi người quản lý "tin tưởng các nhóm phát triển". Scrum phụ thuộc vào khả năng kiểm soát, vì vậy, nếu đội ngũ phát triển còn trẻ và chưa trưởng thành thì Scrum trở nên rất rủi ro.
Một hoạt động chính trong scrum là "cuộc họp hàng ngày", giúp các thành viên trong nhóm đưa ra bằng chứng về việc hoàn thành nhiệm vụ và cho phép cải tiến liên tục, do đó cho phép áp dụng kỹ thuật từ dưới lên một cách nhanh chóng	Scrum được thiết kế lý tưởng cho công ty có "các phương pháp nhanh hiện có". Do đó, một công ty phải có kiến thức về phương pháp làm việc trước khi sử dụng Scrum.

Nội dung

1. Khái niệm
2. Các nguyên lý cơ bản của phương pháp Agile
3. Ưu, nhược điểm của phương pháp Agile
4. Extreme Programming
5. Scrum
6. Các phương pháp Agile khác

Lean development (phát triển tinh gọn)

J Paul Gibson: Agile Methods October 2011



Lean Software Development

- Mọi thứ sẽ dễ dàng hơn nếu khách hàng ngừng thay đổi suy nghĩ của họ.
- Hãy để khách hàng trì hoãn quyết định về những gì họ muốn càng lâu càng tốt và khi họ yêu cầu hãy để họ không có thời gian để thay đổi.
- Các thiết kế xuất hiện khi các nhà thiết kế phát triển sự hiểu biết về vấn đề, chứ không phải thu thập số lượng lớn các yêu cầu.
- Cung cấp hệ thống nhanh nhất có thể.

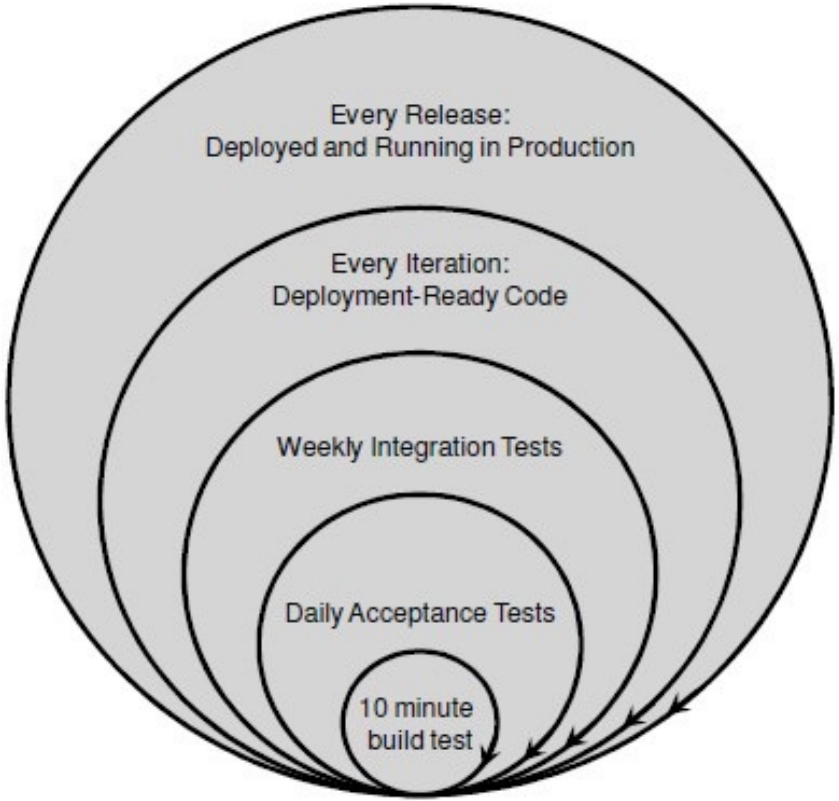
Lean Software Development

- **Eliminate waste.** Giảm thiểu sự lãng phí: Bất kỳ hoạt động nào không "tự trả giá" trong nỗ lực giảm thiểu ở những nơi khác trong hệ thống và cần được loại bỏ.
- **Amplify learning.** Nhấn mạnh vào việc học: Các nhà phát triển luôn cần học các phương pháp mới để tạo ra một hệ thống mạnh mẽ nhất.
- **Đưa ra quyết định muộn nhất có thể.** Tránh đưa ra quyết định sai lầm sớm và sau đó phải sửa chữa nó sau này.
- **Cung cấp nhanh nhất có thể.** Làm giảm đi việc thay đổi yêu cầu. Điều này có thể phải trả giá lớn nếu sự thay đổi đến muộn trong quá trình phát triển.

Lean Software Development

- **Để mọi người chịu trách nhiệm:** có động lực và gắn bó với nhau như một đội, cần phải chịu trách nhiệm về kết quả và được ủy quyền để biến nó thành hiện thực.
- **Duy trì tính toàn vẹn:** phải kiểm thử tích hợp, kiểm thử đơn vị và kiểm thử chung, đặc biệt là từ khách hàng.
- **Xem xét toàn bộ hệ thống:** đừng chia nhỏ hệ thống thành nhiều phần.

Tập trung vào kiểm thử



Lean Software Development

Ưu điểm	Nhược điểm
Tư duy toàn bộ hệ thống, mặc dù khó đối với hệ thống phức tạp, giúp đảm bảo tính nhất quán và toàn vẹn của hệ thống. Giảm thời gian tích hợp kể từ khi được phát triển.	Khó thực hiện việc hình dung cấu trúc của một hệ thống phức tạp là thông qua việc phân vùng hệ thống.
Nhóm làm việc thiết kế các quy trình của riêng mình, đưa ra các cam kết riêng, thu thập thông tin cần thiết để đạt được mục tiêu và tự lập chính sách để đạt được các mốc quan trọng của mình.	Lịch trình sẽ bị ảnh hưởng xấu bởi các quyết định muộn. Điều này có thể làm tổn hại đến sự phát triển song song và làm tăng thời gian thực hiện.

Tổng kết

- Phương pháp phát triển phần mềm nhanh: phát triển liên tục, nhanh chóng phân phối sản phẩm.
- Một số mô hình phát triển phần mềm nhanh:
  1. Extreme Programming
  2. Scrum
  3. Lean development


Q&A