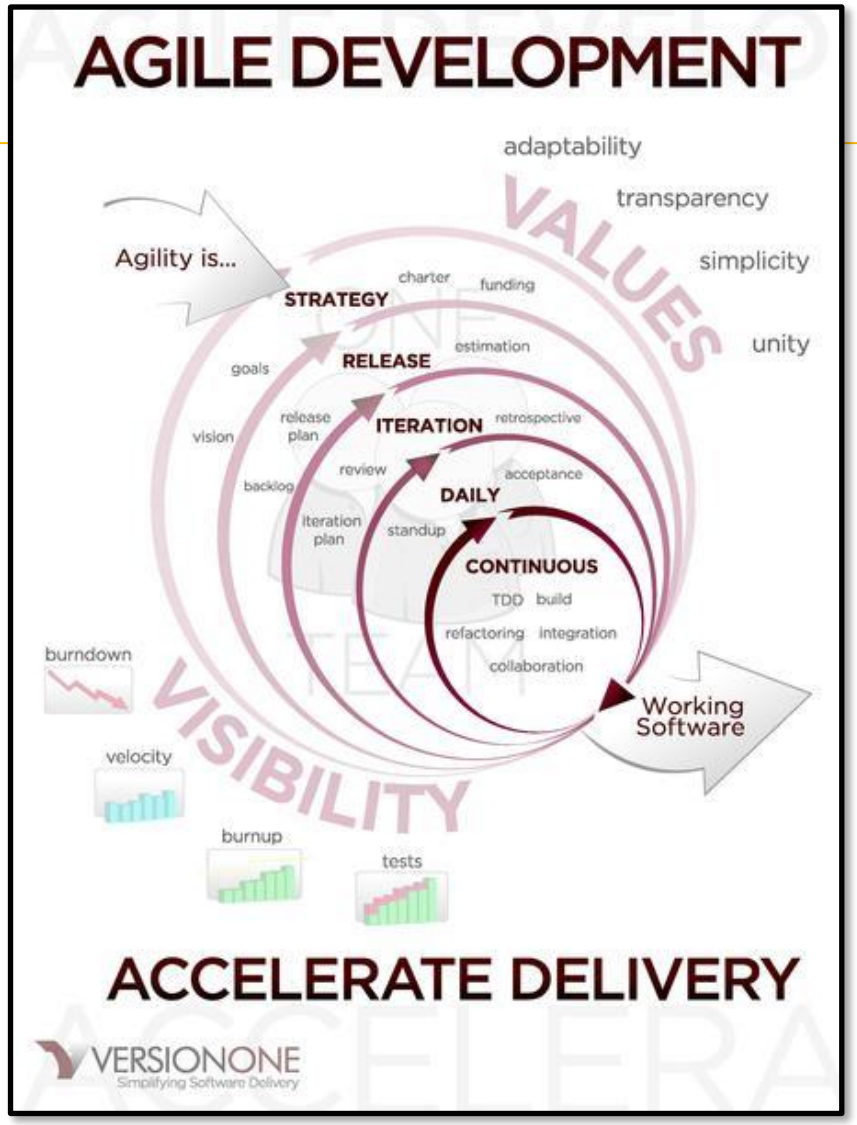


Kỹ thuật phần mềm

Quy trình Agile

Các chủ đề

- Các phương pháp agile
- Phát triển kiểu agile và theo kế hoạch
- Lập trình cực đoan
- Quản lý dự án agile
- Mở rộng quy mô các phương pháp agile

Rapid software development

- Phát triển và chuyển giao nhanh ngày nay thường là yêu cầu quan trọng nhất đối với các hệ thống phần mềm
  - các doanh nghiệp vận hành trong một bộ yêu cầu nhanh chóng thay đổi
    - trong thực tế không thể tạo ra một bộ yêu cầu phần mềm ổn định
  - phần mềm phải tiến hóa nhanh chóng để phản ánh nhu cầu thay đổi của doanh nghiệp.

Rapid software development

- Rapid software development
  - Các hoạt động đặc tả, thiết kế và cài đặt được tiến hành xen kẽ
  - Hệ thống được phát triển theo một chuỗi các phiên bản mà khách hàng tham gia đánh giá từng phiên bản
  - Giao diện người dùng thường được phát triển bằng cách dùng một môi trường phát triển tích hợp (IDE) và một bộ công cụ đồ họa.

Động cơ của agile

- Sự thất vọng với phụ phí của các phương pháp thiết kế phần mềm thập kỉ 1990 dẫn đến sự xuất hiện của các phương pháp agile. Các phương pháp này:
  - chú trọng vào mã chương trình thay vì thiết kế
  - dựa trên một cách phát triển phần mềm theo kiểu vòng lặp
  - nhằm nhanh chóng chuyển giao phần mềm hoạt động được và nhanh chóng tiến hóa nó để đáp ứng các yêu cầu thay đổi.

Mục đích của agile

- Giảm phụ phí trong quy trình phần mềm
  - ví dụ bằng cách giảm viết tài liệu
- Có khả năng đáp ứng nhanh chóng các yêu cầu thay đổi mà không cần phải làm lại quá nhiều.

Tuyên ngôn agile

- *Chúng tôi đang khám phá những phương pháp tốt hơn để phát triển phần mềm bằng cách tự tay phát triển và giúp những người khác làm việc đó. Qua công việc này chúng tôi đã đi đến chỗ đánh giá cao:*
  - *Các cá nhân và tương tác hơn các quy trình và công cụ*
  - *Phần mềm hoạt động được hơn tài liệu đầy đủ*
  - *Cộng tác của khách hàng hơn thương lượng hợp đồng*
  - *Đáp ứng các thay đổi hơn làm theo kế hoạch*
- *Nghĩa là, mặc dù các điểm bên phải có giá trị, nhưng chúng tôi đánh giá các điểm bên trái* ***cao hơn****.*

Nguyên bản tại http://agilemanifesto.org/

Các nguyên tắc agile

Nguyên tắc	Miêu tả
Khách hàng tham gia	Khách hàng nên tham gia chặt chẽ trong suốt quá trình phát triển. Vai trò của họ là cung cấp và quy định mức độ ưu tiên các yêu cầu mới về hệ thống, và đánh giá hệ thống tại các lần lặp (iterations of the system).
Chuyển giao tăng dần	Phần mềm được phát triển một cách tăng dần từng đợt (increment), trong đó khách hàng chỉ ra các yêu cầu cần được đưa vào mỗi đợt.
Con người thay vì quy trình	Kĩ năng của đội phát triển cần được ghi nhận và khai thác. Các thành viên của đội cần được tự do phát triển cách làm việc của riêng mình mà không cần đến các quy trình quy phạm định trước.
Chấp nhận thay đổi	Hiểu rằng yêu cầu hệ thống sẽ thay đổi nên thiết kế hệ thống sao cho nó có thể chấp nhận được các thay đổi đó.
Gìn giữ tính giản dị dễ hiểu	Chú trọng vào tính giản dị dễ hiểu của phần mềm đang được phát triển cũng như của quy trình phát triển. Chủ động nỗ lực loại bỏ sự phức tạp ra khỏi hệ thống bất cứ khi nào có thể.

Khả năng ứng dụng phương pháp agile

- Phát triển phần mềm dùng chung, trong đó một công ty phần mềm đang phát triển một sản phẩm cỡ nhỏ-trung bình để bán.
- Phát triển phần mềm đặt hàng (custom system development) trong phạm vi một tổ chức, trong đó
  - có một cam kết rõ ràng từ khách hàng về việc tham gia quy trình phát triển
  - có không nhiều các quy tắc và quy định từ bên ngoài có ảnh hưởng tới phần mềm.
- Do sự chú trọng vào các nhóm nhỏ làm việc một cách gắn kết, có vấn đề trong việc mở rộng quy mô của các phương pháp agile cho các hệ thống lớn.

Vấn đề với các phương pháp agile

- Có thể khó giữ sự quan tâm của khách hàng tham gia quy trình.
- Các thành viên trong đội có thể không phù hợp với cường độ làm việc đặc thù của các phương pháp agile.
- Khi có nhiều hơn một người có quyền xác định mức độ ưu tiên của các yêu cầu, có thể khó thay đổi mức độ ưu tiên đó.
- Việc gìn giữ tính giản dị dễ hiểu cũng đòi hỏi công sức.
- Cũng như các phương pháp phát triển lặp khác, hợp đồng có thể là vấn đề.

Agile và bảo trì phần mềm

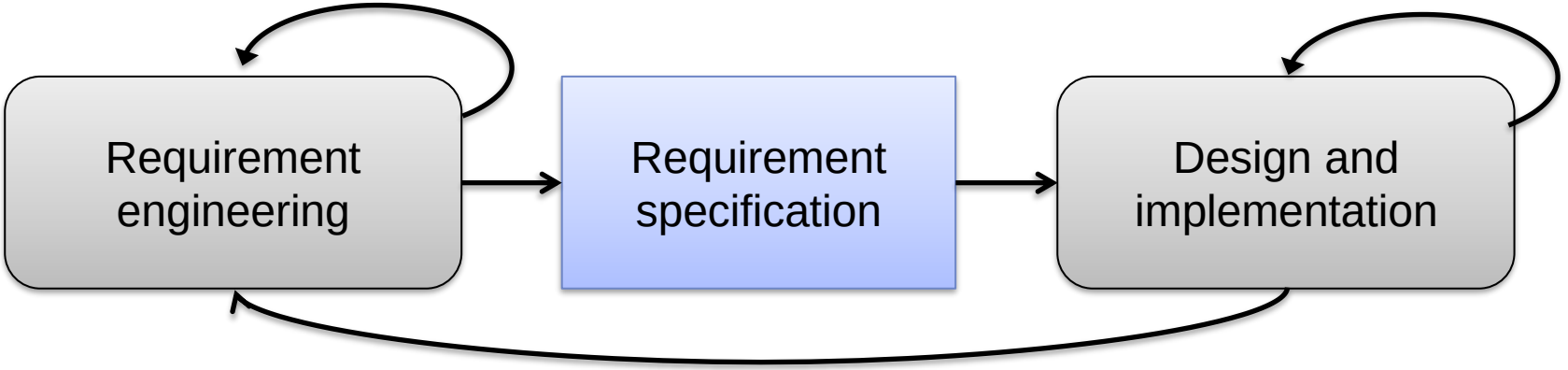
- Đa số các tổ chức chi nhiều tiền cho việc bảo trì một phần mềm cũ hơn là cho việc phát triển phần mềm mới.
  - do đó, nếu muốn các phương pháp agile thành công, chúng phải hỗ trợ bảo trì cũng tốt như với phát triển.
- Hai điểm quan trọng:
  - Các hệ thống phát triển bằng agile có bảo trì được không?
    - quy trình nhấn mạnh vào việc tối thiểu hóa tài liệu chính thức.
  - Có thể sử dụng hiệu quả các phương pháp agile cho việc tiến hóa một hệ thống để đáp ứng yêu cầu của khách hàng hay không?
- Rắc rối có thể nảy sinh nếu không thể giữ đội phát triển ban đầu.

Phát triển theo kế hoạch và phát triển agile

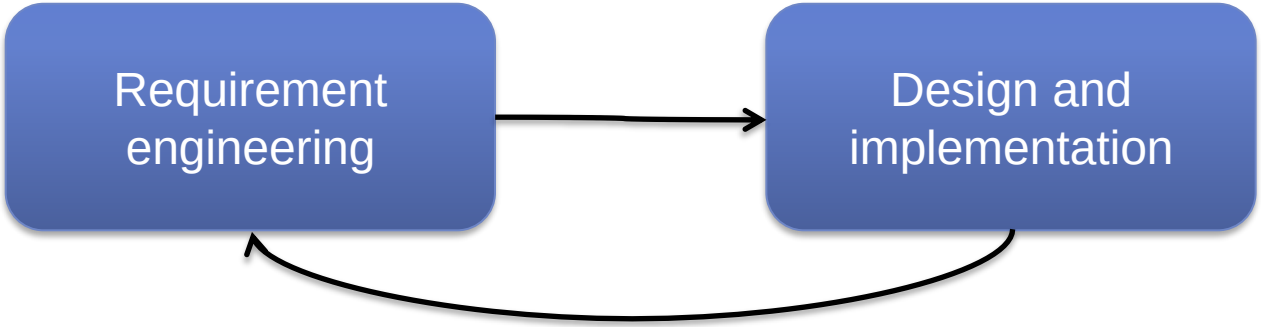
- Phát triển theo kế hoạch
  - Các giai đoạn phát triển tách biệt
    - Sản phẩm của mỗi giai đoạn được lập kế hoạch từ trước.
  - Không nhất thiết chỉ dành cho mô hình thác nước, mô hình phát triển tăng dần cũng dùng được
    - lặp xảy ra bên trong mỗi hoạt động.
- Phát triển agile
  - Các hoạt động phát triển được thực hiện xen kẽ
  - Sản phẩm cần thực hiện được quyết định bởi thương thuyết trong quá trình phát triển.

Phát triển theo kế hoạch và phát triển agile

Plan-based development



Agile development



Lựa chọn: Agile hay theo kế hoạch?

- Đa số dự án bao gồm các quy trình theo kế hoạch cũng như quy trình agile. Sự cân bằng được quyết định tùy theo:
  - Có cần có một đặc tả và thiết kế rất chi tiết trước khi chuyển sang cài đặt hay không?
  - Chiến lược chuyển giao tăng dần có thực tế hay không?
    - nếu có, nên xem xét việc sử dụng phương pháp agile.
  - Hệ thống cần phát triển lớn đến đâu?
    - Các phương pháp agile có hiệu quả nhất khi có thể phát triển hệ thống với một đội nhỏ làm việc cùng một chỗ và có thể giao tiếp thân mật với nhau.
    - Với hệ thống lớn đòi hỏi các đội phát triển lớn, khi đó có thể phải dùng cách tiếp cận theo kế hoạch.

Lựa chọn: Agile hay theo kế hoạch?

- Hệ thống cần phát triển thuộc loại gì?
  - Cách tiếp cận theo kế hoạch thích hợp với các hệ thống đòi hỏi nhiều công phân tích trước khi cài đặt
    - ví dụ hệ thống thời gian thực với yêu cầu phức tạp về thời gian.
- Thời gian sống của hệ thống?
  - Các hệ thống được sử dụng lâu dài có thể đòi hỏi nhiều hơn về tài liệu thiết kế để truyền đạt được chủ ý ban đầu của những người phát triển hệ thống cho nhóm hỗ trợ.
- Có công nghệ gì để hỗ trợ việc phát triển hệ thống?
  - Các phương pháp agile dựa vào các công cụ tốt để ghi lại dấu vết của một thiết kế liên tục biến đổi

Lựa chọn: Agile hay theo kế hoạch?

- Đội phát triển được tổ chức như thế nào?
  - Nếu đội phát triển làm việc phân tán hoặc một phần của sản phẩm được gia công ở bên ngoài, bạn có thể cần có các tài liệu thiết kế để truyền đạt giữa các thành viên trong nhóm hoặc giữa các nhóm phát triển.
- Trình độ của những người thiết kế và lập trình viên trong đội phát triển?
  - các phương pháp agile đòi hỏi kĩ năng cao hơn các phương pháp theo kế hoạch
    - ở kiểu theo kế hoạch, các lập trình viên chỉ đơn giản là dịch một thiết kế chi tiết thành mã chương trình
- Hệ thống có phải chịu sự chi phối của quy định của bên ngoài?
  - Nếu một hệ thống phải được duyệt bởi một nhân tố bên ngoài, nó cần đến tài liệu chi tiết (để an toàn).

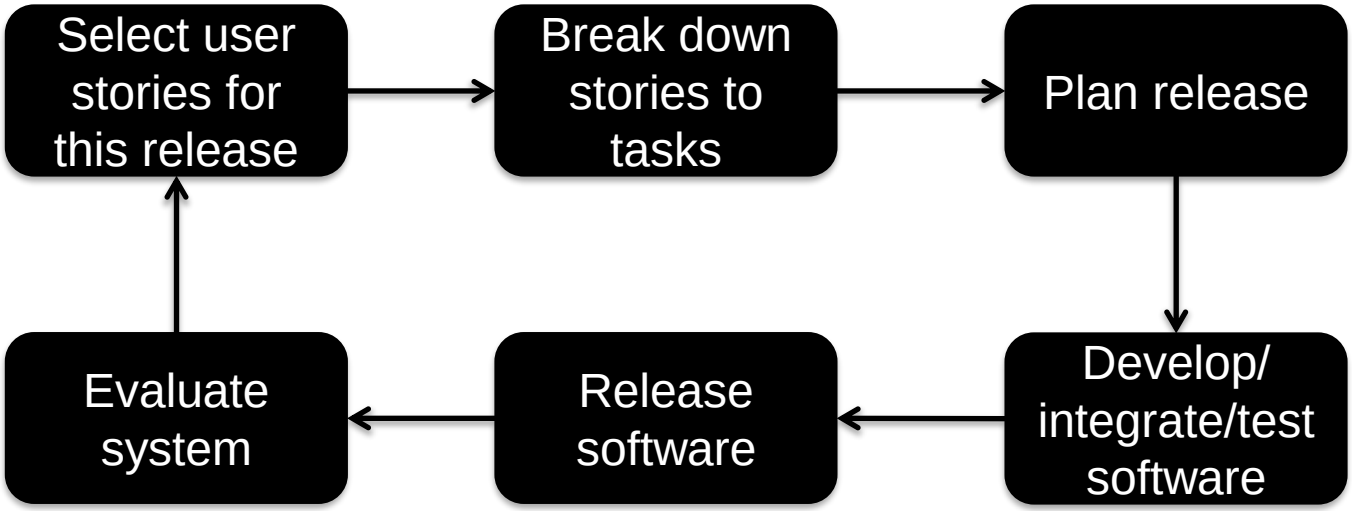
eXtreme Programming

XP

Extreme programming

- Phương pháp agile nổi tiếng nhất và được sử dụng rộng rãi nhất.
- Extreme Programming (XP) có một cách tiếp cận '**cực đoan**' đối với hoạt động phát triển vòng lặp.
  - Các phiên bản mới có thể được xây dựng **vài lần mỗi ngày**;
  - **Hai tuần một lần** chuyển giao sản phẩm đợt mới (increment) cho khách hàng;
  - Phải chạy tất cả các test cho từng phiên bản (build) và một phiên bản chỉ được chấp nhận nếu tất cả các test đều thành công.

XP release cycle



XP practices (a)

Nguyên lý hay thực hành	**Miêu tả**
Incremental planning (lập kế hoạch tăng dần)	Các yêu cầu được ghi lại trên các story card, quyết định xem các story nào được kèm theo một bản release là tùy vào thời gian và mức độ ưu tiên tương đối giữa chúng. Developer chia các story thành các tác vụ (development task).
Small releases (các bản release nhỏ)	Đầu tiên, tập chức năng hữu ích tối thiểu mang lại giá trị được phát triển. Các bản release của hệ thống được phát hành thường xuyên và bổ sung dần chức năng cho bản release đầu tiên.
Simple design (thiết kế đơn giản)	Chỉ thiết kế vừa đủ để thỏa mãn các yêu cầu hiện tại.
Test-first development (test trước)	sử dụng một framework cho unit test để viết test cho một chức năng mới trước khi cài đặt chính chức năng đó.
Refactoring (cải tiến)	Tất cả các developer liên tục cải tiến mã nguồn bất cứ khi nào tìm thấy điểm nào có thể cải tiến. Việc này làm cho mã nguồn trở nên đơn giản dễ hiểu và bảo trì được.

XP practices (b)

Pair programming (lập trình cặp)	Developer làm việc theo từng cặp, người này kiểm tra công việc của người kia và hỗ trợ để việc lúc nào cũng chạy.
Collective ownership (sở hữu tập thể)	Cặp developer làm việc trong mọi lĩnh vực của hệ thống, để không xảy ra tình trạng mỗi người chỉ thạo một vùng, và tất cả các developer chịu trách nhiệm cho toàn bộ mã nguồn. Ai cũng có thể sửa bất cứ cái gì.
Continuous integration (tích hợp liên tục)	Mỗi khi một tác vụ được hoàn thành, nó được tích hợp ngay vào hệ thống. Sau mỗi lần tích hợp như vậy, hệ thống phải chạy qua tất cả các unit test.
Sustainable pace (tiến độ bền vững)	Làm việc quá giờ quá nhiều không được chấp nhận do hệ quả thường là giảm chất lượng mã nguồn và giảm năng xuất trung hạn.
On-site customer (khách hàng tại chỗ)	Trong suốt thời gian làm việc của đội XP, luôn có một đại diện của người dùng hệ thống (khách hàng) sẵn sàng tham gia. Trong một quy trình XP, khách hàng là một thành viên của đội và có trách nhiệm giao các yêu cầu hệ thống cho đội để cài đặt.

Kịch bản yêu cầu

- Với XP, một khách hàng hoặc người dùng là thành viên của đội XP và có trách nhiệm đưa ra các quyết định về các yêu cầu.
- Yêu cầu của người dùng được diễn đạt dưới dạng các kịch bản hoặc **user story**.
  - được viết trên các story card
  - đội phát triển chia story thành các tác vụ cài đặt.
    - Các tác vụ này là cơ sở cho việc lập kế hoạch và ước lượng chi phí.
- Khách hàng lựa chọn các story cần được đáp ứng trong bản release tiếp theo
  - dựa theo đánh giá ưu tiên và ước lượng về kế hoạch của họ.

User story

- Mẫu

"As a <role>, I want <goal/desire> so that <benefit>"

- Ví dụ

Với vai trò một đại diện khách hàng, tôi muốn tra cứu các khách hàng của tôi theo tên và họ.

Khởi động ứng dụng
Ứng dụng bắt đầu bằng việc mở tài liệu cuối mà user đã dùng lần trước.

Nhà tư vấn sẽ nhập chi phí vào một form chi phí. nhà tư vấn sẽ nhập các mục trong form như dạng chi phí, miêu tả, số lượng, và các ghi chú về chi phí đó. Khi nào muốn, nhà tư vấn có thể thực hiện một công việc tùy chọn trong số dưới đây.

(1)Sau khi điền xong form, nhà tư vấn sẽ “Submit”. Nếu chi phí nhỏ hơn 50, chi phí đó sẽ được gửi thẳng cho hệ thống để xử lý.
(2)Nếu nhà tư vấn chưa điền xong form chi phí, nhà tư vấn có thể muốn “Save for later”. Form đó sau đó sẽ được hiện trong một danh sách (hàng đợi) dành cho nhà tư vấn với ghi chú trạng thái là “Incomplete”.
(3)Nếu nhà tư vấn quyết định xóa toàn bộ dữ liệu và đóng forrm, nhà tư vấn sẽ “Cancel and exit”. Dữ liệu soạn dở sẽ không được lưu lại ở bất cứ đâu.

A ‘prescribing medication’ story

Prescribing medication

The record of the patient must be open for input. Click on the medication field and select either 'current medication', 'new medication' or 'formulary'.

If you select 'current medication', you will be asked to check the dose; If you wish to change the dose, enter the new dose then confirm the prescription.

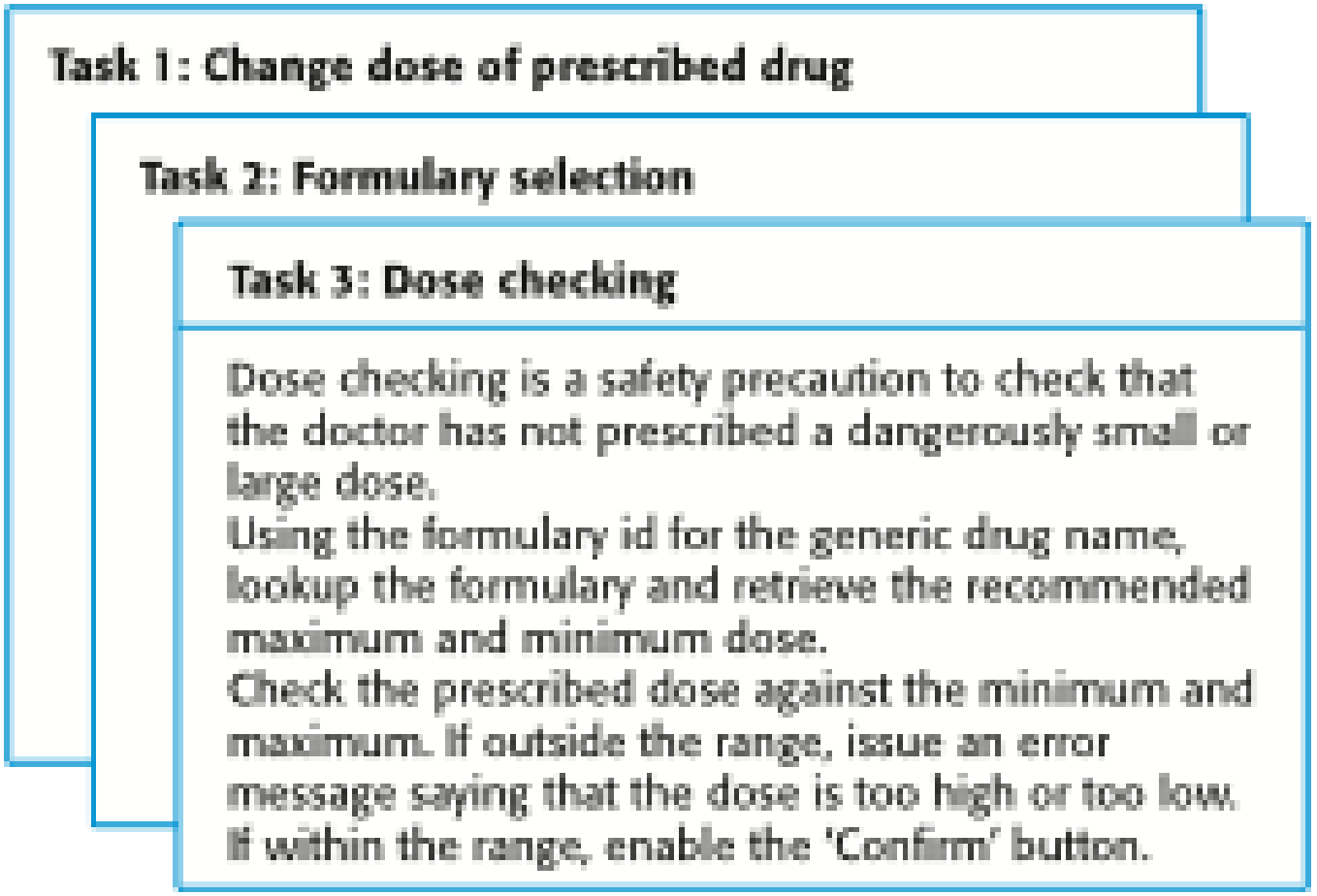
If you choose, 'new medication', the system assumes that you know which medication you wish to prescribe. Type the first few letters of the drug name. You will then see a list of possible drugs starting with these letters. Choose the required medication. You will then be asked to check that the medication you have selected is correct. Enter the dose then confirm the prescription.

If you choose 'formulary', you will be presented with a search box for the approved formulary. Search for the drug required then select it. You will then be asked to check that the medication you have selected is correct. Enter the dose then confirm the prescription.

In all cases, the system will check that the dose is within the approved range and will ask you to change it if it is outside the range of recommended doses.

After you have confirmed the prescription, it will be displayed for checking. Either click 'OK' or 'Change'. If you click 'OK', your prescription will be recorded on the audit database. If you click 'Change', you reenter the 'Prescribing medication' process.

Examples of task cards for prescribing medication



XP và sự thay đổi

- Tri thức truyền thống trong ngành công nghệ phần mềm là: thiết kế cho thay đổi (design for change).
  - dành thời gian và nỗ lực cho việc dự đoán các thay đổi là việc đáng làm vì sau này nó sẽ làm giảm chi phí.
- Còn XP khẳng định việc đó không đáng làm vì không thể dự đoán các thay đổi một cách đáng tin cậy.
  - liên tục cải tiến mã (refactoring) để thay đổi sau này sẽ dễ dàng hơn.

Refactoring

- Tìm kiếm các cải tiến có thể thực hiện cho phần mềm và thực hiện các cải tiến đó
  - thậm chí ngay cả ở nơi chưa có nhu cầu cải tiến ngay.
  - làm tăng tính dễ hiểu của phần mềm
    - dẫn tới giảm nhu cầu tài liệu.
  - dễ thực hiện các thay đổi hơn vì mã nguồn rõ ràng và có cấu trúc tốt.
- tuy nhiên, một số thay đổi đòi hỏi refactoring về thiết kế và điều này đòi hỏi chi phí cao hơn.

Ví dụ về refactoring

- Tổ chức lại cây phân cấp class để loại bỏ các đoạn mã nguồn lặp đi lặp lại
- Dọn dẹp và đổi tên các thuộc tính và phương thức để làm chúng trở nên dễ hiểu hơn.
- thay thế mã inline bằng các lời gọi các phương thức đã được kèm trong một thư viện chương trình.

Testing trong XP

- Testing là trung tâm của XP, và XP đã phát triển một cách tiếp cận mà theo đó chương trình được **test sau mỗi sửa đổi** đã được thực hiện.
- Các đặc điểm của XP testing:
  - Test trước nhất – test-first development
  - Hoàn thiện dần việc test từ các scenario.
  - Người dùng tham gia việc phát triển và thẩm định test.
  - Cơ chế chạy test tự động để mỗi lần xây dựng một bản release mới lại chạy lại tất cả các test thành phần (component test).

Test-first development

- Việc viết test trước khi viết code làm rõ các yêu cầu cần cài đặt.
- Các test được viết thành các chương trình thay vì chỉ là dữ liệu
  - Để có thể chạy chúng một cách tự động.
    - Mỗi test kèm theo chức năng kiểm tra xem chương trình đã chạy đúng hay sai.
  - Thường được phát triển dựa vào một testing framework chẳng hạn Junit.
- Tất cả các test cũ và test mới đều được chạy tự động mỗi khi bổ sung một chức năng mới cho sản phẩm
  - Nhờ vậy kiểm tra được xem chức năng mới có gây lỗi hay không.

Khách hàng tham gia test

- Để giúp phát triển các acceptance test cho các story cần được cài đặt trong bản release tiếp theo của hệ thống.
- Khách hàng tham gia đội phát triển viết các bộ test trong khi sản phẩm được phát triển.
  - Do đó tất cả mã mới được thẩm định để đảm bảo thỏa mãn nhu cầu của khách hàng.
- Tuy nhiên, những người đóng vai trò khách hàng có ít thời gian nên không thể làm việc toàn thời gian với nhóm phát triển.
  - Họ có thể cảm thấy rằng tham gia bằng cách cung cấp các yêu cầu là đã đủ, và có thể sẽ không hào hứng tham gia quy trình test.

Mô tả test case cho chức năng kiểm tra liều lượng (dose checking)

Test 4: Dose checking

Input:
1. A number in mg representing a single dose of the drug.
2. A number representing the number of single doses per day.

Tests:
1. Test for inputs where the single dose is correct but the frequency is too high.
2. Test for inputs where the single dose is too high and too low.
3. Test for inputs where the single dose * frequency is too high and too low.
4. Test for inputs where single dose * frequency is in the permitted range.

Output:
OK or error message indicating that the dose is outside the safe range.

Tự động hóa test

- Tự động hóa test nghĩa là test được viết dưới dạng các thành phần chạy được trước khi tác vụ (task) được cài đặt
  - Các thành phần testing component nên
    - Đứng độc lập (stand-alone),
    - Giả lập việc nhập các dữ liệu cần test
    - Kiểm tra xem kết quả có thỏa mãn đặc tả output hay không.
  - Một framework cho test tự động (ví dụ Junit) là một hệ thống tạo thuận lợi cho việc viết các test chạy được và chạy một bộ test.
- Khi testing được tự động hóa, luôn luôn có một bộ test có thể thực thi nhanh chóng và dễ dàng
  - Mỗi khi một chức năng được bổ sung cho hệ thống, có thể chạy các test và các vấn đề mà phần code mới đã gây ra có thể được nắm bắt lập tức.

Khó khăn trong XP testing

- Lập trình viên thích viết code hơn là test
  - Đôi khi họ đi đường tắt khi viết test.
    - Ví dụ viết các test không hoàn chỉnh không kiểm tra đủ các trường hợp ngoại lệ có thể xảy ra.
- Một số test có thể rất khó viết theo kiểu tăng dần.
  - Ví dụ, trong một giao diện người dùng phức tạp, khó viết unit test cho mã cài đặt 'lôgic hiển thị' và luồng chuyển tiếp giữa các màn hình.
- Khó đánh giá tính đầy đủ hoàn thiện của một bộ test.
  - Dù bạn có thể có rất nhiều system test, bộ test của bạn có thể vẫn không phủ đủ.

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.
- Các số liệu đo đạc cho thấy năng xuất của lập trình đôi tương đương với năng xuất của hai người làm việc độc lập.

Lập trình đôi

- Khi lập trình theo từng cặp, các lập trình viên ngồi cùng nhau trước một máy tính để phát triển phần mềm.
- Các cặp được sắp xếp một cách linh động để tất cả các thành viên đều có thời gian làm việc cùng nhau trong suốt quá trình phát triển.
- Sự chia sẻ kiến thức xảy ra trong quá trình lập trình theo cặp rất quan trọng
  - Nó giảm rủi ro cho dự án khi một số thành viên rời khỏi nhóm.

Ưu điểm của lập trình đôi

- Nó hỗ trợ quan niệm sở hữu tập thể và trách nhiệm tập thể đối với hệ thống.
  - Các cá nhân không chịu trách nhiệm đối với các vấn đề của mã chương trình.
    - Thay vào đó, cả đội cùng có trách nhiệm giải quyết các vấn đề đó.
- Nó hoạt động như là một quy trình review không chính thức vì mỗi dòng lệnh đều có ít nhất hai người xem.
- Tạo điều kiện cho refactoring – một quá trình trong việc nâng cấp phần mềm.
  - Khi áp dụng lập trình đôi và sở hữu tập thể, những người khác có lợi trực tiếp từ việc refactoring nên họ sẽ dễ hỗ trợ quá trình này.

Quản lý dự án agile

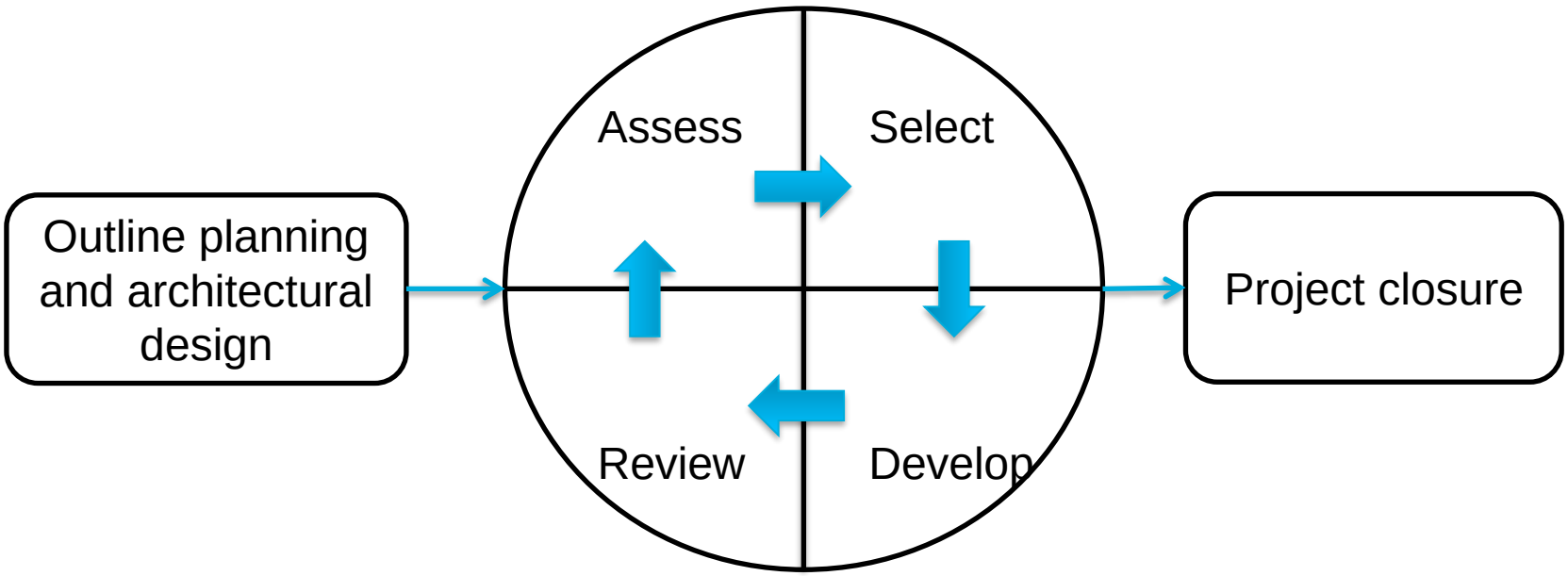
- Trách nhiệm chính của những người quản lý dự án phần mềm là quản lý dự án
  - Sao cho phần mềm được bàn giao **đúng hạn và không bị bội chi ngân sách**.
- Cách tiếp cận tiêu chuẩn đối với quản lý dự án là làm theo kế hoạch (plan-driven).
  - Các quản lý lập kế hoạch cho dự án, chỉ rõ
    - Cái gì cần bàn giao,
    - Khi nào cần bàn giao và
    - Ai sẽ làm các sản phẩm cần bàn giao của dự án.
- Agile đòi hỏi một cách tiếp cận khác, thích nghi với kiểu phát triển tăng dần và các thế mạnh đặc thù của các phương pháp agile.

Scrum

Scrum

- Scrum là một phương pháp agile tổng quát nhưng tập trung vào **quản lý phát triển tăng dần (iterative development)**.
- Scrum có ba pha.
  - Pha khởi đầu là lập kế hoạch phác thảo (outline planning)
    - Thiết lập các mục tiêu tổng quan cho dự án và thiết kế kiến trúc phần mềm.
  - Tiếp theo là một chuỗi các vòng **sprint** (sprint cycle),
    - Mỗi vòng phát triển một increment (bản tăng dần) của hệ thống.
  - Pha kết thúc dự án hoàn chỉnh các tài liệu cần thiết
    - Chẳng hạn trợ giúp hệ thống và hướng dẫn sử dụng
    - Đánh giá các bài học thu được từ dự án.

Quy trình Scrum



Sprint cycle

Vòng Sprint

- Một vòng Sprint có độ dài cố định, thường 2–4 tuần.
  - Tương ứng với việc phát triển một bản release của hệ thống theo kiểu XP.
- Điểm khởi đầu cho lập kế hoạch là product backlog,
  - Là danh sách các công việc cần thực hiện cho dự án.
- Pha chọn lựa (select) trong đó toàn đội phát triển làm việc với khách hàng
  - Chọn các tính năng và chức năng cần phát triển trong vòng sprint đó.

Vòng Sprint

- Sau khi thống nhất, đội **tự tổ chức** để phát triển phần mềm.
  - Tại bước này, đội hoạt động độc lập, không liên quan đến khách hàng và tổ chức,
  - Tất cả các liên lạc được đi qua kệnh có tên ‘Scrum Master’.
- Vai trò của Scrum master là để tránh cho đội phát triển khỏi bị phân tán bởi các tác động bên ngoài.
- Ở cuối vòng sprint, sản phẩm được review và trình bày cho bên đặt hàng.
  - Sau đó vòng sprint tiếp theo sẽ bắt đầu.

Làm việc theo nhóm với Scrum

- 'Scrum master' hỗ trợ
  - Sắp xếp các cuộc gặp hàng ngày,
  - Dùng backlog theo dõi các công việc đã được thực hiện,
  - Ghi lại các quyết định,
  - Đo đạc tiến độ so với backlog và
  - Liên lạc với khách hàng và quản lý ở ngoài đội.
- Toàn đội họp ngắn hàng ngày
  - Tất cả các thành viên chia sẻ thông tin,
  - Miêu tả tiến độ làm việc kể từ buổi họp trước, các vấn đề đã nảy sinh và kế hoạch cho ngày tiếp theo.
- Điều đó có nghĩa cả nhóm đều biết tình hình, và nếu vấn đề nảy sinh thì có thể lập lại kế hoạch ngắn hạn để đối phó.

Ích lợi của Scrum

- Sản phẩm được chia thành những phần quản lý được và hiểu được.
- Các yêu cầu không ổn định không làm đình trệ tiến độ.
- Cả đội "nhìn" thấy mọi việc và nhờ đó tăng tính tương tác và liên lạc giữa các thành viên.
- Khách hàng thấy các bản increment được giao đúng hạn và gửi phản hồi về sản phẩm
- Thiết lập sự tin tưởng giữa khách hàng và đội phát triển, xây dựng một môi trường văn hóa tích cực trong đó tất cả đều trông đợi dự án thành công.

Tools

- Use online tool
  - www.agilebench.com/
  - Manage your project using the tool
  - Send me the link
- Setup your own one:
  - www.agile-tools.net

Tổng kết

- Các phương pháp Agile là các phương pháp phát triển tăng dần mà tập trung vào các khía cạnh
  - Phát triển nhanh,
  - Thường xuyên ra các bản release của phần mềm,
  - Giảm phụ phí quy trình, và
  - Tạo mã chất lượng cao.
- Các phương pháp này đòi hỏi khách hàng trực tiếp tham gia quy trình phát triển.

Tổng kết (2)

- Việc nên chọn agile hay plan-driven tùy thuộc vào
  - Loại phần mềm được phát triển,
  - Năng lực của đội phát triển và
  - Văn hóa của công ty phát triển phần mềm.
- Lập trình cực đoan (XP) là
  - Một phương pháp agile nổi tiếng, nó tích hợp các kiểu hoạt động như
    - Thường xuyên ra các bản release của phần mềm,
    - Liên tục cải tiến phần mềm và
    - Khách hàng tham gia đội phát triển.

Tổng kết (3)

- Một thế mạnh đặc biệt của lập trình cực đoan là việc phát triển của các test tự động trước khi tạo một tính năng của chương trình. Tất cả các test đều phải thành công khi một phần bổ sung (increment) được tích hợp vào một hệ thống
- Phương pháp Scrum là một phương pháp agile cung cấp một framework cho quản lý dự án. Nó xoay quanh một tập các vòng sprint, mỗi vòng có độ dài cố định về thời gian và dành cho việc phát triển một increment cho hệ thống.
- Mở rộng quy mô phương pháp agile cho các hệ thống lớn là rất khó khăn. Các hệ thống lớn cần các thiết kế được hoàn thành từ sớm và cần tài liệu.