Phương pháp phát triển phần mềm linh hoạt

Agile Software Development

Nhóm nghiên cứu:

- Triệu Minh Tiến
- Nguyễn Viết Tùng
- Phạm Đức Khánh

(Chương trình Việt - Nhật * Đại học Bách Khoa Hà Nội)

HEDSPI **HUT**

Tài liệu tham khảo:

- ***Agile software development methods - Review and analysis***

Pekka Abrahamsson, Outi Salo & Jussi Ronkainen, 2002

- ***An Introduction to Agile Methods***

Steve Hayes (Khatovar Technology)

Martin Andrews (Object Consulting)

- ***Internet***

http://en.wikipedia.org/wiki/Agile_Software_Development

…

I. Tổng quan:

1. Sự cần thiết của một mô hình phát triển phần mềm mới

Chúng ta đã viết phần mềm được 30 năm nhưng những gì chúng ta đã làm được còn rất ít. Thành công của chúng ta được thúc đẩy bởi sự tưởng tượng, sáng tạo của con người. Chúng ta càng viết ra nhiều phần mềm thì sự đòi hỏi, yêu cầu của con người càng nhiều. Chính vì vậy, những nhà quản lý và phát triển phần mềm tiếp tục tìm kiếm phương thức phát triển phần mềm tốt hơn. So với 30 năm trước chúng ta đã có những chiếc máy tính rẻ hơn, nhanh hơn, nhiều ngôn ngữ lập trình mạnh mẽ ra đời, số lượng nhiều hơn cần thiết những công cụ hỗ trợ, sự đào tạo tốt hơn và có những hiểu biết sâu hơn về lý thuyết phần mềm. Internet đã thay đổi cách con người giao tiếp với nhau, thúc đẩy sự trao đổi thông tin và thay đổi một cách triệt để sự kỳ vọng của con người về cách thức phần mềm làm việc. Chúng ta cũng có số lượng đáng kể những phương pháp khác nhau giúp xác định

con đường phát triển phần mềm tốt nhất và đó chính là khía cạnh của việc phát triển phần mềm mà chúng ta sẽ tập trung vào trong thời gian tới.

a. Những hạn chế của mô hình phát triển phần mềm truyền thống

Đã có rất nhiều mô hình phát triển phần mềm được tạo ra trong những năm qua. Có thể kể đến như:

- Pure waterfall
- Code-and-Fix
- Spiral
- Modifed Waterfalls
- Evolutionary Prototyping
- Staged Delivery
- Evolutionary Delivery
- Design-to-Schedule
- Design-to-Tools
- Commercial Off-the-shelf Software

Khi xây dựng các phương pháp truyền thống người ta đã cố gắng trang bị cho chúng khả năng dự đoán trước. Với khả năng này ta có thể tạo ra một bản kế hoạch tại thời điểm đầu của dự án và xác định được thời gian hoàn thành dự án dựa theo bản kế hoạch này. Và vấn đề cố hữu trong quá trình thực hiện dự án vẫn là "sự thay đổi yêu cầu của người dùng". Thông thường khách hàng không thay đổi yêu cầu của họ bởi vì họ biết rằng chi phí để thay đổi rất đắt. Tuy nhiên phần mềm không phải là thứ hữu hình. Khách hàng không chỉ khó để xác định một cách chính xác cái gì là cần thiết mà cũng khó để hiểu tại sao việc thay đổi lại khó khăn như vậy. Họ mong đợi một phần mềm phải có tính mềm dẻo. Những phương pháp truyền thống đã đưa ra những thủ tục nhằm ngăn chặn sự thay đổi yêu cầu từ phía khách hàng. Điều này giúp duy trì bản kế hoạch dự án đã xây dựng ban đầu nhưng lại không đảm bảo rằng kết quả cuối cùng là những gì mà khách hàng mong muốn. Khả năng dự đoán trước có thể là điều ước ao nhưng ta chỉ có thể đạt được điều đó với giá phải trả là sự giảm sút chất lượng phần mềm không thỏa mãn được đòi hỏi của khách hàng.

Với những hạn chế như vậy của những phương pháp phát triển phần mềm truyền thống, ta thắc mắc rằng tại sao chúng lại được sử dụng và nếu chúng đã được sử dụng trong quá khứ thì lại từ bỏ chúng bây giờ? Phải chăng một vài thứ đã thay đổi. Trong suốt thập niên 80 những thay đổi cơ bản đã xảy ra và

kết quả là hình thành nên "thế giới nhanh" – thế giới của sự toàn cầu hóa và "thế giới chậm" của những ai tự tách mình ra khỏi quá trình toàn cầu hóa. Sự phát triển phần mềm diễn ra trong thế giới nhanh và sự thay đổi diễn ra đồng thời ở công nghệ, tài chính, thông tin và đi kèm với chúng là sự dỡ bỏ những hàng rào chính trị được duy trì suốt thời kỳ chiến tranh lạnh. Mọi việc diễn ra nhanh hơn. Những đối thủ xuất hiện, sự thành công hay thất bại chỉ là ranh giới mong manh. Vòng đời của sản phẩm ngắn hơn và người dùng thì hay thay đổi. Không có gì là ngạc nhiên khi những phương pháp phù hợp với thập niên 70 lại thất bại trong thập niên 80 và 90.

b. Sự nổi lên của phương pháp linh hoạt

Trong thập kỉ 90 nhiều người đã nhận ra mọi thứ đã thay đổi bằng cách này hay cách khác. Những người này đã quan tâm tới phương pháp phát triển phần mềm linh hoạt hơn, phù hợp hơn với môi trường làm việc luôn luôn vận động. Mặc dù chi tiết những phương pháp này là khác nhau nhưng chúng đều có chung một số nguyên tắc và trong một phạm vi nào đó những phương pháp này thường được nhóm lại với nhau dưới tên gọi "những phương pháp linh hoạt – agile methodologies".

2. Phương pháp linh hoạt

Phương pháp phát triển phần mềm linh hoạt được đưa ra vào giữa những năm 90 như là một phần của sự nỗ lực chống lại những phương pháp "nặng nề" – điển hình bởi những quy định khắt khe. Ban đầu chúng được gọi là "phương pháp nhẹ". Đến năm 2001, 17 thành viên nổi bật của cộng đồng phát triển phần mềm linh hoạt đã gặp gỡ tại Snowbird, Utah để thảo luận về cách thức tạo ra phần mềm linh hoạt hơn, nhanh hơn và hướng con người hơn. Họ đã thông qua tên gọi chính thức "**phương pháp linh hoạt**". Và cũng trong hội nghị này, Agile Manifesto – tuyên ngôn về phương pháp linh hoạt được đưa ra và được công nhận rộng rãi như là một định nghĩa chuẩn của phương pháp phát triển linh hoạt kèm theo những nguyên tắc cơ bản. Bản tuyên ngôn hướng tới những giá trị:

- **Sự độc lập và sự tương tác** dựa trên các quy trình và công cụ : *sự vận động linh hoạt nhấn mạnh mối quan hệ và cộng đồng các nhà phát triển phần mềm và vai trò của con người được phản ánh trong hợp đồng, trái với những quy trình đã được thể chế hóa và những công cụ phát triển. Trong thực tiễn nó tự thể hiện thông qua những mối quan hệ chặt chẽ trong nhóm, việc tạo ra môi trường làm việc gần gũi, và những thủ tục khác để nâng cao tinh thần của nhóm.*

- Vận hành phần mềm dựa trên tài liệu hướng dẫn toàn diện : *các mục tiêu quan trọng của nhóm phát triển phần mềm là liên tiếp đưa ra những phần mềm đã được kiểm thử. Những phiên bản mới thường được đưa ra hàng tháng thậm chí ở một vài phương pháp là hằng giờ hoặc hàng ngày. Những nhà phát triển luôn cố gắng giữ cho mã nguồn đơn giản, rõ ràng nhất có thể, bởi vậy gánh nặng về tài liệu hướng dẫn được giảm bớt.*

- Sự cộng tác với khách hàng dựa trên thương thảo hợp đồng : *mối quan hệ và sự hợp tác giữa những nhà phát triển và khách hàng được định rõ thông qua những bản hợp đồng chặt chẽ. Những dự án càng lớn thì càng cần một bản dự thảo hợp đồng chặt chẽ. Quá trình thương lượng nên được đánh giá như là phương tiện để đạt được và duy trì mối quan hệ. Nhìn từ quan điểm kinh doanh, phương pháp phát triển linh hoạt tập trung vào việc nhanh chóng đưa ra được những sản phẩm có thể đáp ứng những yêu cầu cơ bản của khách hàng ngay sau khi dự án được tiến hành; do đó làm giảm nguy cơ vỡ hợp đồng.*

- Đáp ứng với thay đổi dựa trên một kế hoạch theo sau : *nhóm phát triển gồm cả những nhà phát triển phần mềm và đại diện khách hàng nên được cung cấp thông tin đầy đủ, họ có đủ thẩm quyền và đươc ủy thác để xem xét những sự điều chỉnh cần thiết trong suốt vòng đời của quy trình phát triển phần mềm. Như vậy những người tham gia được chuẩn bị để đối mặt với những thay đổi và có thể đưa ra được bản hợp đồng hỗ trợ và cho phép sự thay đổi.*

Một số nguyên tắc đi kèm sau tuyên ngôn có thể kể đến như:

- Phần mềm chạy ổn định được bàn giao thường xuyên (hàng tuần hoặc hàng tháng)
- Những thay đổi yêu cầu dù muộn luôn được hoan nghênh
- Sự hợp tác gắn bó khăng khít giữa nhà kinh doanh và những nhà phát triển phần mềm
- Đối thoại trực tiếp là hình thức giao tiếp tốt nhất
- Dự án được tiến hành bởi những cá nhân nhiệt tình, tận tụy, đáng tin cậy
- Luôn luôn chú trọng tới kỹ thuật và thiết kế
- Sự đơn giản
- Các nhóm tự tổ chức
- Sự thích nghi với những những thay đổi

3. So sánh với các phương pháp khác

Phương pháp phát triển linh hoạt đôi khi bị đánh giá là thiểu tính kỉ luật. Những nhận xét như vậy gây ra sự hiểu lầm. Để hiểu vấn đề một cách đúng đắn, ta có thể hình dung rằng những phương pháp phát triển phần mềm hiện có nằm trên một trục đi từ "khả năng thích ứng" tới "khả năng dự đoán trước" thì phương pháp phát triển linh hoạt nằm về phía "khả năng thích ứng".

Những phương pháp nằm về phía "khả năng thích ứng" có thể thích nghi nhanh chóng với những thay đổi của thực tế. Khi mà những yêu cầu của dự án thay đổi, nhóm thực hiện cần phải có những điều chỉnh thích hợp. Họ sẽ gặp khó khăn để mô tả chính xác những gì sẽ xảy ra trong tương lai. Tương lai càng xa thì sự khó khăn đó càng lớn. Nhóm thực hiện có thể báo cáo chính xác công việc sẽ được tiến hành trong tuần tới nhưng chỉ có thể báo cáo những tính năng nào sẽ được xây dựng trong tháng tới. Và khi được hỏi về phiên bản phần mềm trong 6 tháng tiếp theo thì họ chỉ có thể đưa ra được những tính năng chung nhất hoặc đưa ra kinh phí dự kiến.

Trong khi đó những phương pháp nằm về phía "khả năng dự báo trước" trong hợp đồng với khách hàng, tập trung vào xây dựng một kế hoạch chi tiết cho tương lai. Nhóm thực hiện dự án có thể báo cáo chính xác những tính năng và công việc cần thực hiện trong toàn bộ quy trình phát triển phần mềm. Bản kế hoạch được tối ưu hoá cho những mục tiêu đã đặt ra lúc đầu và sự thay đổi có thể khiến cho công việc đã hoàn thành trở nên vô nghĩa. Nhóm phát triển dự án sẽ xây dựng một bảng kiểm soát những thay đổi để đảm bảo rằng chỉ những thay đổi có giá trị mới được xem xét đến.

a. Phân biệt với mô hình thác nước

Phương pháp phát triển linh hoạt có vài điểm chung nhỏ với mô hình thác nước. Hiện nay mô hình thác nước vẫn được sử dụng phổ biến. Nó được lên kế hoạch trước và được tiến hành lần lượt qua các bước nắm bắt yêu cầu, phân tích, thiết kế, viết code và kiểm thử một cách nghiêm ngặt. Vấn đề của mô hình thác nước là sự phân chia cứng nhắc dự án thành các giai đoạn riêng biệt và do đó rất khó khăn khi muốn thay đổi yêu cầu. Chi phí để thực hiện lại rất đắt. Điều đó có nghĩa là mô hình thác nước không thích hợp khi mà không thể xác định chính xác, rõ ràng yêu cầu của khách hàng hoặc những yêu cầu có thể thay đổi trong quá trình tiến hành dự án. Phương pháp phát triển linh hoạt, trong hợp đồng, sẽ nhanh chóng đưa ra sản phẩm hoạt động ổn định với những tính năng cơ bản giúp khách hàng sớm được sử dụng sản phẩm phục vụ mục

đích của họ. Sau đó nhóm phát triển tiếp tục nâng cấp sản phẩm trong suốt thời gian tiến hành dự án, hàng tuần hoặc tháng sẽ bổ sung những tính năng được được phát triển và kiểm thử toàn diện.

Theo khía cạnh này, những người chỉ trích phương pháp linh hoạt có thể quả quyết rằng những tính năng này không được xem xét trong quy mô toàn dự án. Nếu người tài trợ dự án lo ngại về thời gian hoàn thành toàn bộ dự án đã được định trước hay ngân sách đầu tư cho dự án thì phương pháp linh hoạt có thể không thích hợp. Tuy nhiên lời chỉ trích này gặp phải sự phản đối của cộng đồng phát triển phần mềm linh hoạt. Họ cho rằng với SCUM (một phương pháp phát triển linh hoạt sẽ được tìm hiểu kĩ hơn ở phần sau), nhóm phát triển có thể đẩy nhanh tiến độ thực hiện và liên tục cải thiện kế hoạch chiến lược.

Vài nhóm phát triển phần mềm linh hoạt sử dụng mô hình thác nước với quy mô nhỏ trong các giai đoạn của dự án.

b. Phân biệt với "Cowboy coding"

Khi những thành viên trong nhóm làm bất cứ điều gì họ cho là đúng, không tuân thủ kỷ luật, nguyên tắc thì người ta gọi đó là "Cowboy coding". Sự thường xuyên đánh giá lại các kế hoạch của phương pháp phát triển linh hoạt, sự chú trọng vào giao tiếp mặt đối mặt trực tiếp và vấn đề tài liệu hướng dẫn đi kèm khá ít đôi khi khiến cho người dùng lầm tưởng nó với "cowboy coding". Tuy nhiên thực tế là nhóm phát triển phần mềm linh hoạt luôn làm việc theo một quy trình đã được vạch rõ (và thường rất kỷ luật và nghiêm ngặt).

Giống như tất cả các phương pháp phát triển phần mềm, kỹ năng và kinh nghiệm của người sử dụng quyết định để mức độ thành công hoặc thất bại của sản phẩm. Càng có nhiều hệ thống kiểm soát khắt khe được nhúng vào trong quy trình phát triển thì trách nhiệm của người sử dụng càng được nâng cao.

II. Các phương pháp phát triển phần mềm linh hoạt

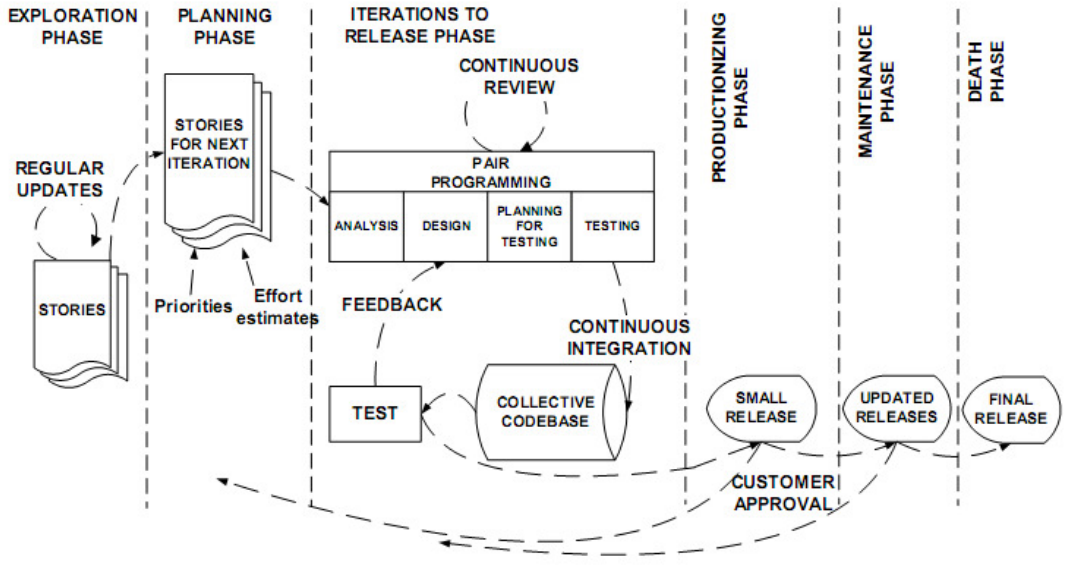
Các phương pháp phát triển phần mềm linh hoạt hiện nay bao gồm: Extreme programming (XP), Scrum, Crystal family of methodologies, Feature Driven Development (FDD), The Rational Unified Process, Dynamic Systems Development Menthod (DSDM), Adaptive Software Development, Open Sourse Software Development, ngoài ra còn có các phương pháp khác.

1. Lập trình cực hạn - Extreme programming (XP)

XP là một phương pháp xây dựng phần mềm mới, dựa trên lý thuyết phương pháp phát triển phần mềm linh hoạt được phát triển bởi Kent Beck, Ward Cunningham, and Ron Jeffries, nó nhấn mạnh vào sự cộng tác, tạo ra phần mềm một cách nhanh chóng, và phát triển mở rộng một cách khéo léo trong quá trình thực hành. Nó được cô đọng lại trong bốn giá trị: sự giao tiếp (communication), đơn giản hóa (simplicity), sự phản hồi (feedback), và thế mạnh (courage).

Nếu bạn làm việc trong một môi trường mà ở đó các nhu cầu được chờ để thay đổi và các khách hàng sẽ được lợi từ việc bàn giao phần mềm sớm và thường xuyên thì chắc chắn nên xem xét XP . Các nhóm làm theo XP sẽ thường xuyên nhận ra rằng họ đang bàn giao các sản phẩm phần mềm chất lượng cao với số lượng rất lớn và nhanh hơn trước đây rất nhiều. [http://vi.wikipedia.org/wiki/Lập_trình_cực_hạn]

XP bao gồm một tập hợp các luật giá trị và thực hành giúp người lập trình mô tả chi tiết các hành vi. Vòng đời của XP gồm có các pha: Khảo sát (Exploration), Lập kế hoạch (Planning), Các bước lặp để phát hành (Iteration to Release), Sản xuất (Productionizing), Bảo trì và kết thúc (Maintenance and Death).



Life cycle of the XP process

Theo mô tả của Beck's (1999b) thì pha ban đầu là pha "Khám phá". Trong pha này khách hàng viết ra các yêu cầu về sản phẩm vào các "story card". Mỗi "story card" sẽ mô tả một đặc trưng sẽ được thêm vào trong chương trình. Trong khi đó nhóm phát triển sẽ giới thiệu những công cụ, công nghệ, những thực thi mà họ sẽ sử dụng trong dự án đó. Công nghệ sử dụng cần được kiểm tra và kiến trúc có thể được phân tích bằng cách xây dựng một nguyên mẫu. Pha này có thể kéo dài vài tuần đến vài tháng tùy thuộc vào từng dự án và nhóm phát triển.

Ở pha thứ hai đó là pha "Lập kế hoạch" là tập hợp các thứ tự ưu tiên cho những yêu cầu và thống nhất nội dung cho phiên bản phần mềm đầu tiên. Người lập trình đầu tiên phải ước lượng khả năng đáp ứng các yêu cầu này và lập thời gian biểu cho việc thực hiện. Khoảng thời gian để đưa ra bản phát hành đầu tiên thường không vượt quá 2 tháng.

Pha "Lặp để phát hành" bao gồm một số bước lặp trong hệ thống để tạo ra bản phát hành đầu tiên. Thời hạn đã đặt ra trong bước lập kế hoạch có thể bị sụp đổ nếu như thời gian để thực thi các bước lặp từ một đến bốn tuần. Bước lặp đầu tiên tạo ra một hệ thống bao gồm kiến trúc của cả một hệ thống. Điều đó đạt được bằng cách thực thi những yêu cầu có tác động mạnh mẽ đến việc xây dựng cấu trúc của cả hệ thống. Khách hàng sẽ quyết định yêu cầu nào được chọn qua mỗi bước lặp. Các hàm kiểm tra được tạo bởi khách hàng sẽ chạy khi mỗi bước lặp kết thúc. Đến khi kết thúc bước lặp cuối cùng thì sản phẩm đã được hoàn thành.

Pha "Sản xuất" sẽ có các kiểm tra thêm đối với hoạt động của hệ thống trước khi tạo ra một hệ thống hoàn chỉnh giao cho khách hàng. Trong pha này, những thay đổi mới vẫn có thể được phát hiện và sự quyết đinh sẽ được đưa ra nếu chúng nằm trong bản phát hành hiện tại. Trong suốt pha này, các bước lặp cần được đẩy nhanh hơn giảm từ ba tuần xuống còn khoảng một tuần. Các ý kiến và yêu cầu bổ xung sẽ được ghi nhận lại và thực hiện ở pha tiếp theo.

Trong khi phiên bản đầu tiên đang được khách hàng sử dụng thì nhóm phát triển vẫn phải đồng thời vừa giữ cho hệ thống làm việc liên tục vừa duy trì những bước lặp mới để tạo ra các phiên bản kế tiếp. Để làm được việc này, pha "Phân tích" đòi hỏi những cố gắng để chăm sóc khách hàng. Vì vậy tốc độ có thể bị chậm lại sau khi sản phẩm đã hoàn thành. Trong pha này có thể yêu cầu kết hợp một số người mới làm thay đổi cấu trúc của nhóm lập trình.

Pha "Chết" bắt đầu khi khách hàng không còn yêu cầu nào nữa để thi hành. Lúc này khách hàng đã thỏa mãn với những chức năng mà phần mềm đem lại. Đây là lúc thích hợp để viết tài liệu cần thiết về hệ thống, lúc hệ thống đã ổn định, không có sự thay đổi trong kiến trúc, thiết kế hay lập trình. "Chết" cũng có thể xảy ra nếu như hệ thống không đạt được kết quả mong đợi hoặc nó quá đắt để phát triển tiếp.

Đối với người lập trình phải viết chương trình và kiểm thử một cách càng đơn giản và càng rõ ràng càng tốt. Điều đầu tiên tạo nên thành công của phương pháp phát triển phần mềm XP là sự giao tiếp và hợp tác giữa các lập trình viên khác và các thành viên trong nhóm.

Khách hàng viết ra yêu cầu và các hàm kiểm tra và quyết định khi nào các yêu cầu được thỏa mãn. Khách hàng tập hợp các thực thi ưu tiên cho các yêu cầu.

Kiểm định viên sẽ giúp khách hàng viết hàm kiểm tra. Họ chạy hàm kiểm tra một cách thường xuyên, thông báo rộng rãi kết quả kiểm tra và duy trì công cụ kiểm tra.

Người theo dõi sẽ đưa ra các phản hồi trong XP. Họ xác định các ước lượng được tạo bởi nhóm lập trình và đưa ra phản hồi trong việc làm thế nào nhóm lập trình có thể tuần tự cải thiện các ước lượng trong tương lai một cách chính xác. Người này cũng theo sát sự tiến triển của mỗi vòng lặp để ước lượng xem có hay không việc đạt tới kết quả trong phạm vi nguồn lực cho phép và thời gian ràng buộc hoặc có gi thay đổi cần thiết trong quá trình xử lý.

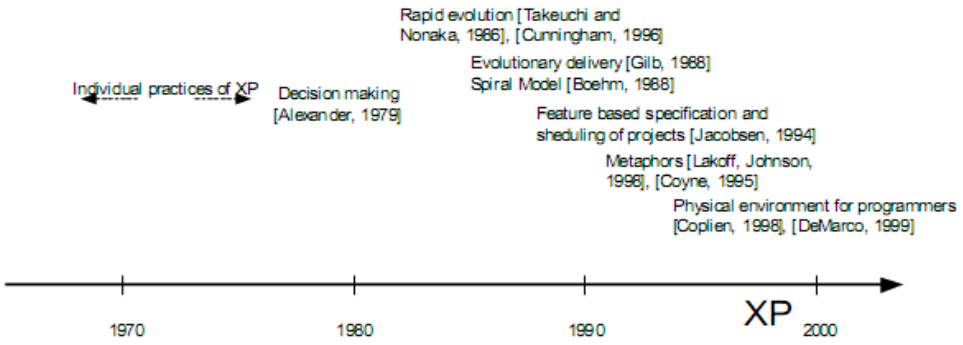
Huấn luyện viên là người chịu trách nhiệm cho toàn bộ quá trình phát triển phần mềm. Việc hiểu rõ XP có vai trò quan trọng cho phép huấn luyện viên có thể hướng dẫn cho những thành viên trong nhóm tuân theo.

Chuyên gia là những người có kiến thức chuyên biệt về vấn đề nào đó, họ có nhiệm vụ hướng dẫn nhóm lập trình giải quyết các vấn đề chuyên biệt của họ.

Người quản lý là người đưa ra những quyết định. Để làm được việc đó anh ta phải giao tiếp với nhóm lập trình để quyết định những tình huống tức thời, và để nhận định những khó khăn hoặc thiếu hụt trong quá trình thực hiện.

XP bao gồm một tập các ý tưởng và thực thi dựa trên những phương pháp luận đã có (Beck 1999a). Chính sự quyết định đã tạo nên cấu trúc. Trong khi khách hàng đưa ra những quyết định mang tính thương mại thì những người lập trình lựa chọn công nghệ, đó chính là ý tưởng của Alexander (1979). Loại hình

phát triển nhanh XP có nguồn gốc từ những ý tưởng hình thành sau Scrum (Takeuchi and Nonaka 1986) và ngôn ngữ mô hình của Cunningham (1986). Việc lập dự án sử dụng XP dựa trên những yêu cầu từ phía khách hàng được vẽ ra từ những tình huống sử dụng (Jacobsen 1994) và phương pháp phát triển phân phối sinh ra bởi Gilb (1988). Cũng như mô hình xoắn ốc, sự phản hồi ban đầu là mô hình thác nước cả hai đều có ảnh hưởng lên phương pháp XP. Phép ẩn dụ của XP khởi đầu từ nghiên cứu của Lakoff, Johnson (1998) và Coyne (1995). Cuối cùng thì môi trường làm việc vật lý đã được Coplien (1998), DeMarco và Lister (1999) tìm ra.



Roots of Extreme Programming.

Mục tiêu mà XP nhắm đến là việc phát triển phần mềm thành công cho dù có sự mập mờ và yêu cầu liên tục bị thay đổi trong một nhóm lập trình. Những bước lặp ngắn với phiên bản phát hành nhỏ và tốc độ phản hồi nhanh, sự tham gia của khách hàng, sự trao đổi và hợp tác, những bước lặp liên tục và kiểm định, chung quyền sở hữu phần lập trình, những tài liệu hạn chế và phương pháp lập trình theo cặp là những đặc trưng chính của phương pháp XP.

Thực thi của XP được biểu diễn theo cấu trúc của Beck (1999[a]). Đó là các bước Lập kế hoạch Bản phân phối nhỏ và ngắn Phép ẩn dụ Thiết kế đơn Tái tạo Lập trình theo cặp Quyền sở hữu tập thể Bước lặp liên tục 40 giờ mọt tuần Khách hàng có mặt Chuẩn lập trình Không gian làm việc mở Quy tắc, luật lệ.

Beck cho rằng phương pháp XP sẽ dần dần được chấp nhận:

"Nếu bạn muốn thử XP, cho những mục đích tốt đẹp thì đừng cố nuốt tất cả một lúc. Chọn ra vấn đề tồi tệ nhất trong xử lý hiện thời của bạn và cố xử lý nó với phương pháp XP."