Bảo Đảm Chất Lượng Phần Mềm

Khoa Công nghệ thông tin

Trường Đại học Nguyễn Tất Thành

Chương 1:
Tổng quan về BĐCLPM và Kiểm thử phần mềm

Khoa Công nghệ thông tin

Trường Đại học Nguyễn Tất Thành

Nội dung

1/ Các khái niệm về KTPM và BĐCLPM

2/ Nghề nghiệp QA/QC/Tester

3/. Các mô hình phát triển phần mềm và kiểm thử

4/. Định nghĩa Lỗi phần mềm

5/. Phân loại lỗi

6/. Vòng đời của lỗi

7/. Bảo đảm chất lượng phần mềm

1/. Các khái niệm về KTPM và BĐCLPM

- **Kiểm thử phần mềm** _ KTPM (Software Testing) là quá trình thực thi kiểm tra PM với mục tiêu tìm ra lỗi, từ đó khẳng định được chất lượng PM đang xây dựng.
- Kiểm thử phần mềm giống với việc kiểm tra các sản phẩm như: quần áo, giày dép, máy tính, điện thoại, động cơ điện, quạt máy, ... trước khi đem bán. Phần mềm cũng cần được kiểm tra trước, nếu không có kiểm tra thì làm sao có thể phát hiện có những lỗi, lỗ hổng sẽ xảy ra khi sử dụng và có thể gây ra tác hại to lớn.

1/. Các khái niệm về KTPM và BĐCLPM

Bảo đảm chất lượng phần mềm _BĐCLPM (Software Quality Assurance) là tập hợp các hoạt động đảm bảo chất lượng trong quá trình phát triển phần mềm. BĐCLPM bao gồm toàn bộ vòng đời phát triển phần mềm, với mục đích để đảm bảo quá trình phát triển và quy trình bảo trì liên tục được cải tiến để sản xuất ra những sản phẩm phần mềm có chất lượng, đáp ứng được các yêu cầu của khách hàng.

2/. Nghề nghiệp QA/QC/Tester

QA (Quality Assurance _ Kiểm định / bảo đảm chất lượng) :

- QA là người chịu trách nhiệm đảm bảo chất lượng sản phẩm thông qua việc đưa ra qui trình làm việc giữa các bên. Thuật ngữ QA để nói về quy trình được dùng trong đảm bảo chất lượng của PM.
- QA là những người làm những công việc đảm bảo chất lượng phần mềm đó là xác định, đề xuất, giám sát các quy trình và phương pháp kỹ thuật được sử dụng nhằm đảm bảo chất lượng cho sản phẩm phần mềm.

2/. Nghề nghiệp QA/QC/Tester

QC (Quality Control _ Kiểm soát chất lượng) :

- QC là người chịu trách nhiệm thực hiện công việc kiểm tra chất lượng sản phẩm phần mềm.
- QC làm những việc kiểm tra, đánh giá chất lượng phần mềm. Tức là tập trung vào đối tượng là sản phẩm và các thủ tục, các hàm, các thành phần trong phần mềm thay vì vào quy trình như QA.
- QC còn thực hiện những bài kiểm tra chất lượng (Test) để đảm bảo sản phẩm đáp ứng đúng và đủ những yêu cầu mà QA đề ra.

2/. Nghề nghiệp QA/QC/Tester

Người kiểm thử hay còn gọi là Tester, họ là những người có nhiệm vụ tìm ra lỗi của phần mềm, điều này sẽ phụ thuộc vào quy trình và các bên liên quan trong dự án. Trong ngành công nghệ phần mềm, các công ty lớn có đội ngũ chuyên chịu trách nhiệm đánh giá phần mềm phát triển trong bối cảnh các yêu cầu đã được đề ra trước đó.

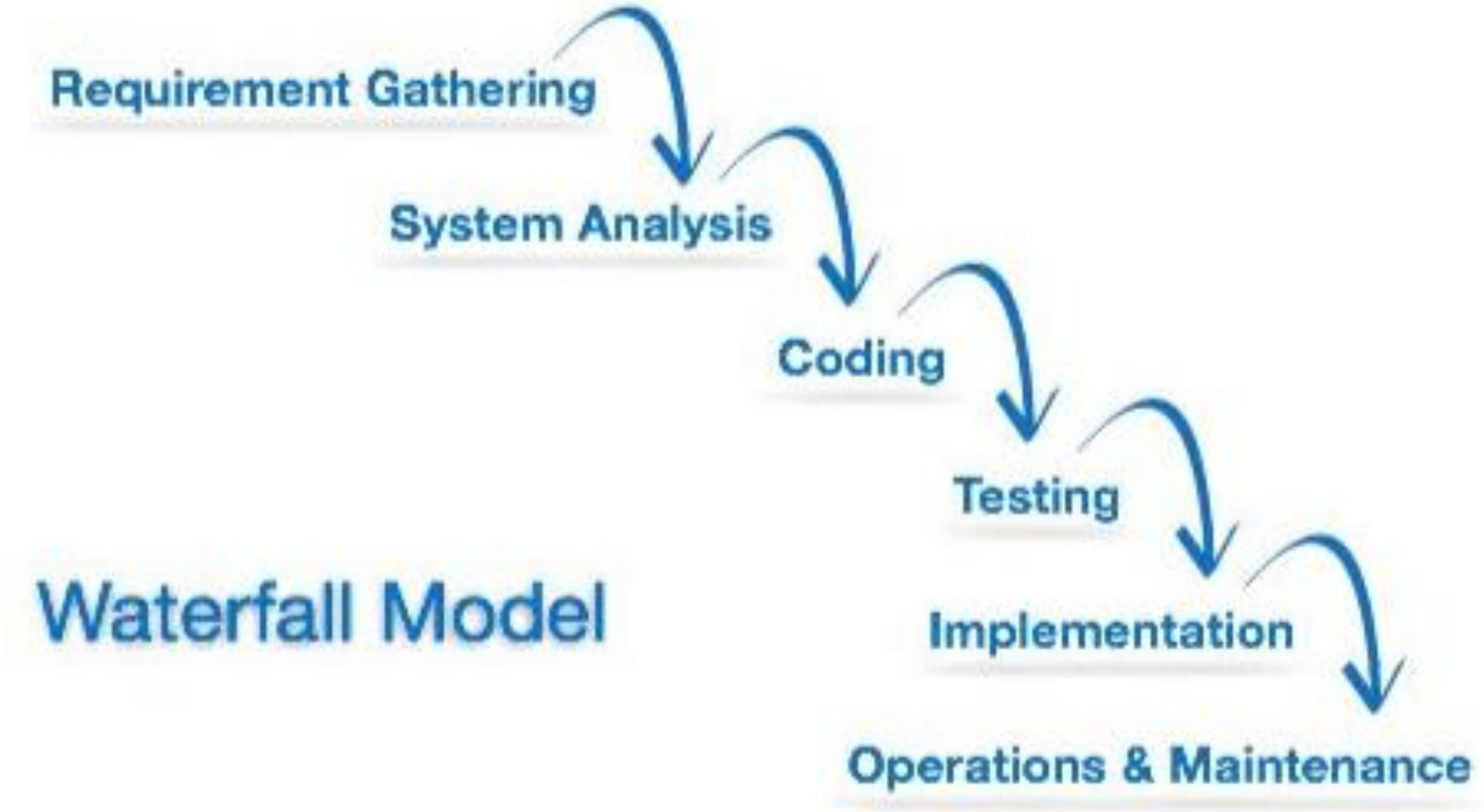
Hơn nữa, các nhà phát triển (Developer) cũng có thể tiến hành kiểm thử được gọi là Kiểm thử đơn vị (Unit Testing). Tester có thể còn bao gồm cả QA, QC.

3/. Các mô hình phát triển phần mềm và kiểm thử

- Mô hình phát triển phần mềm hay quy trình phát triển phần mềm xác định các pha/ giai đoạn trong xây dựng phần mềm. Có nhiều loại mô hình phát triển phần mềm khác nhau ví dụ như:
  - Mô hình thác nước (Waterfall model)
  - Mô hình xoắn ốc (Spiral model)
  - Mô hình Agile
  - Mô hình lặp (Iterative model)
  - Mô hình tăng trưởng (Incremental model)
  - Mô hình chữ V (V model)
  - Mô hình Scrum
  - RAD model (Rapid Application Development)

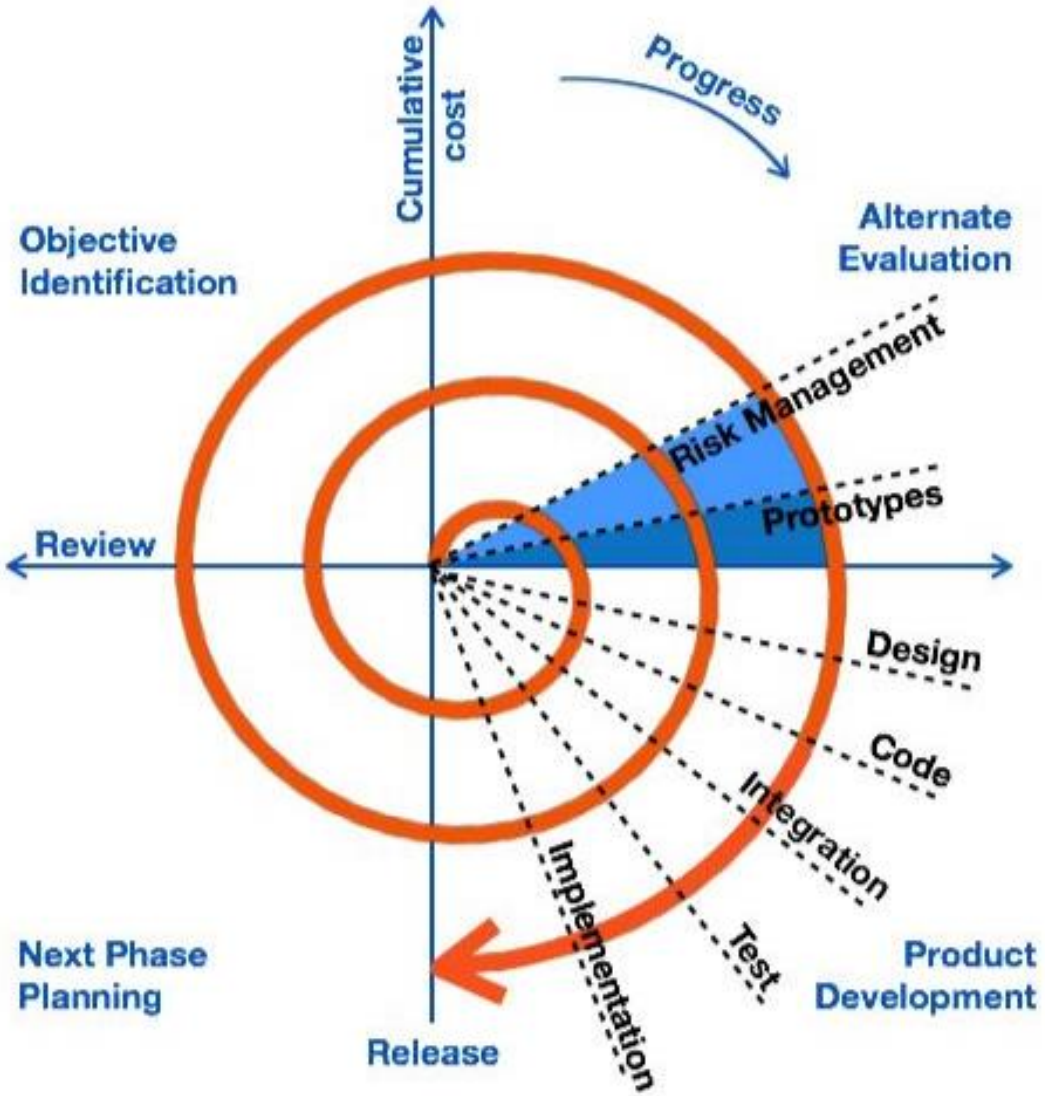
3/. Các mô hình phát triển phần mềm và kiểm thử

Mô hình thác nước:



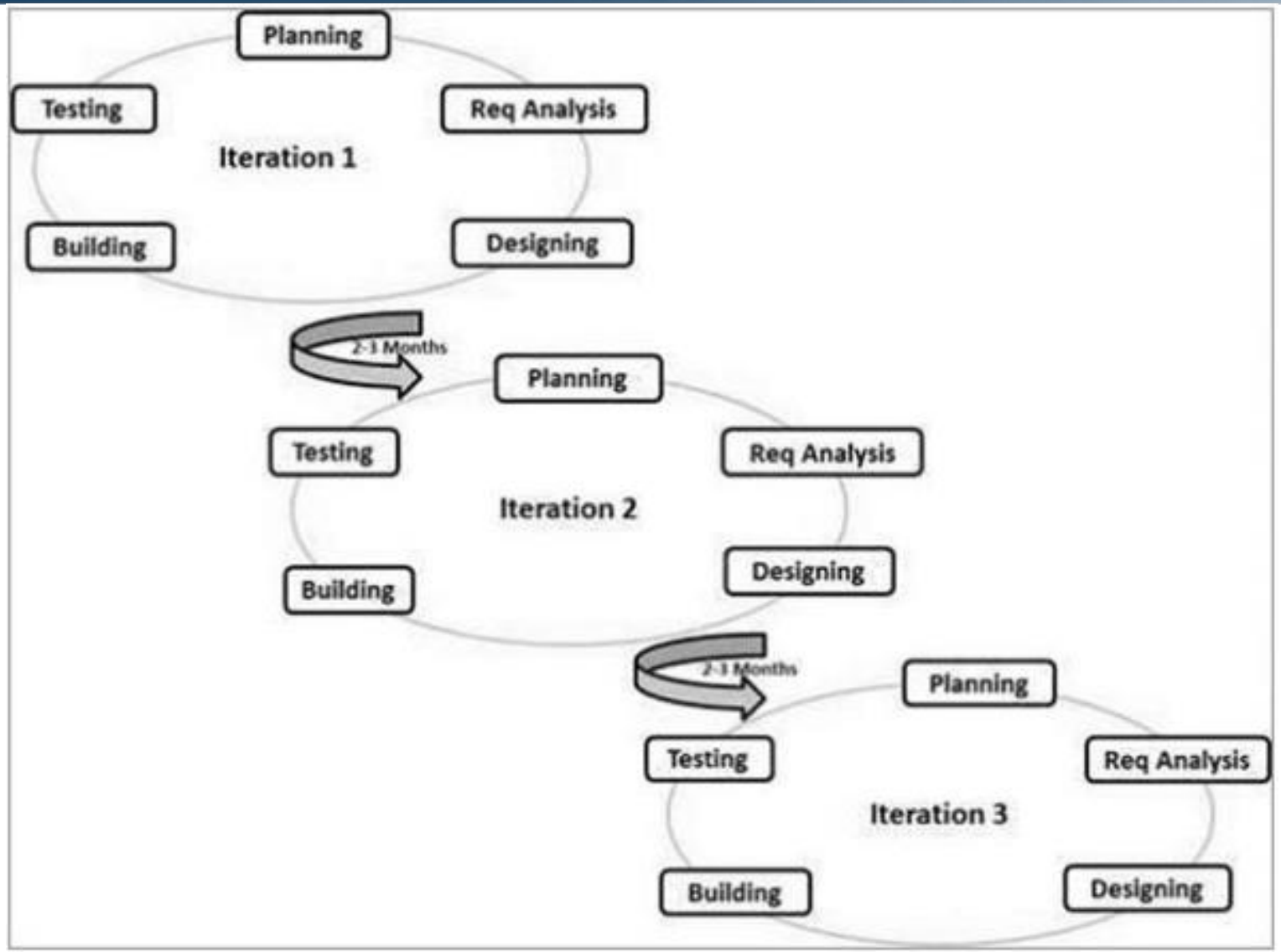
3/. Các mô hình phát triển phần mềm và kiểm thử

Mô hình xoắn ốc:



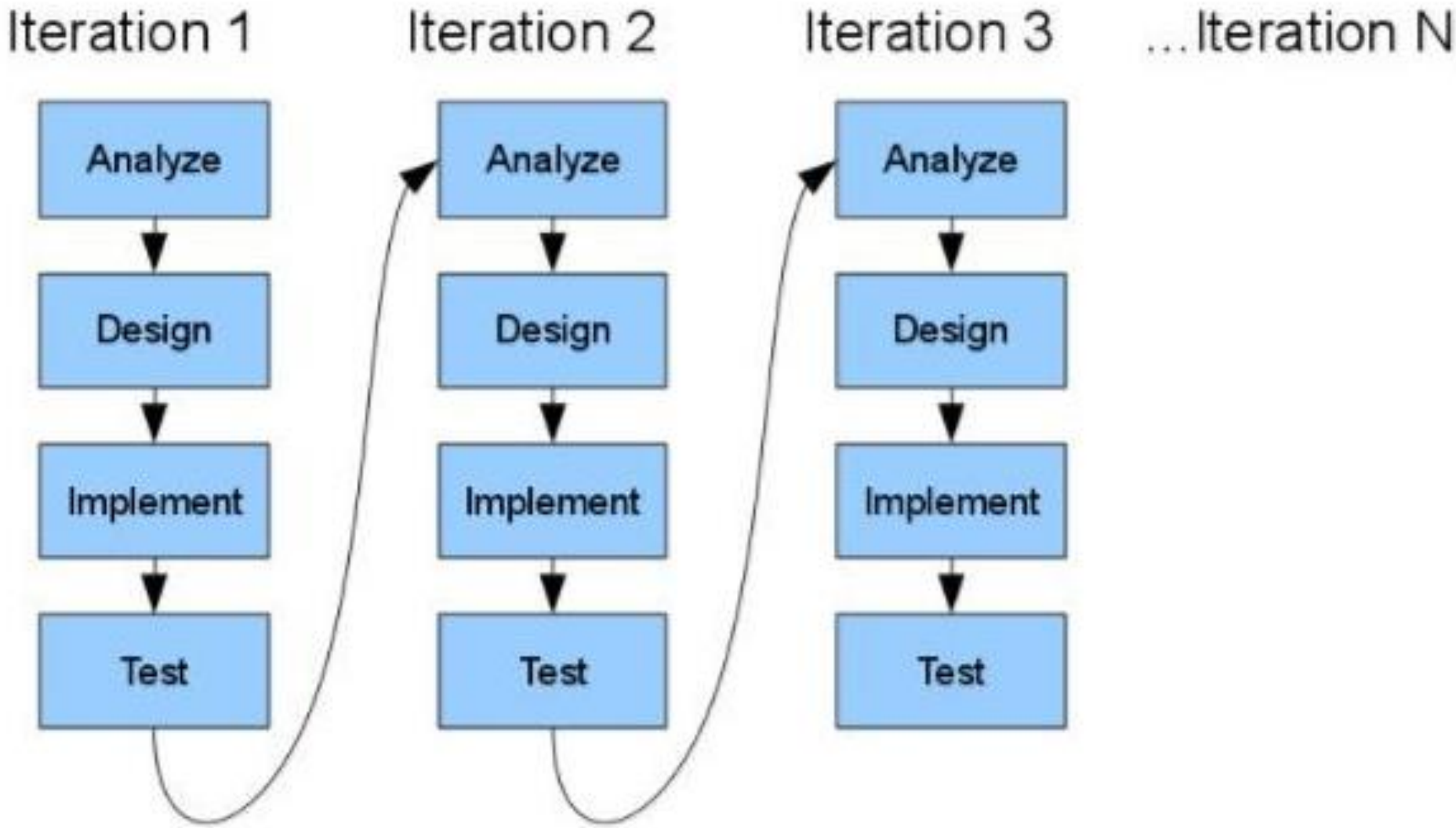
3/. Các mô hình phát triển phần mềm và kiểm thử

Mô hình Agile:



3/. Các mô hình phát triển phần mềm và kiểm thử

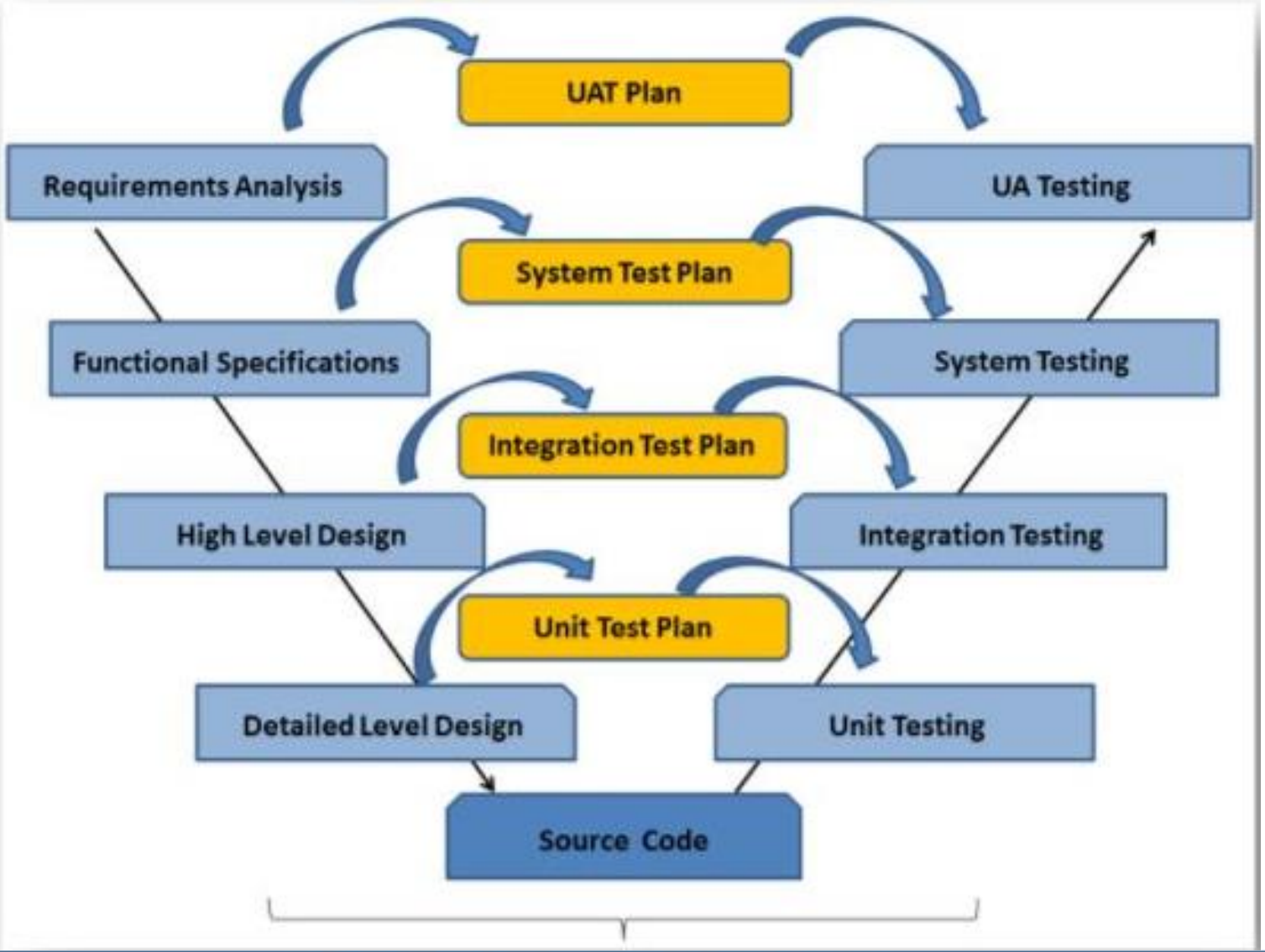
Mô hình Lặp:



3/. Các mô hình phát triển phần mềm và kiểm thử

Mô hình chữ V:

(UA: User Acceptance)

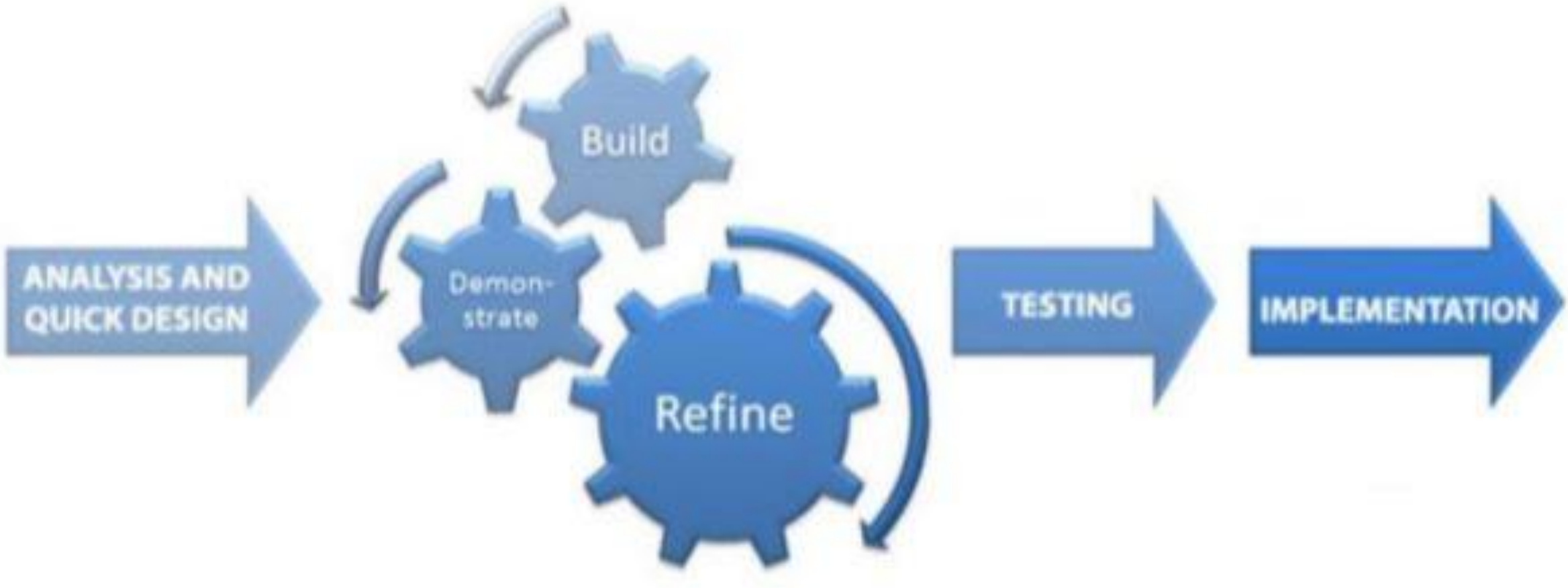


3/. Các mô hình phát triển phần mềm và kiểm thử

Mô hình Scrum:

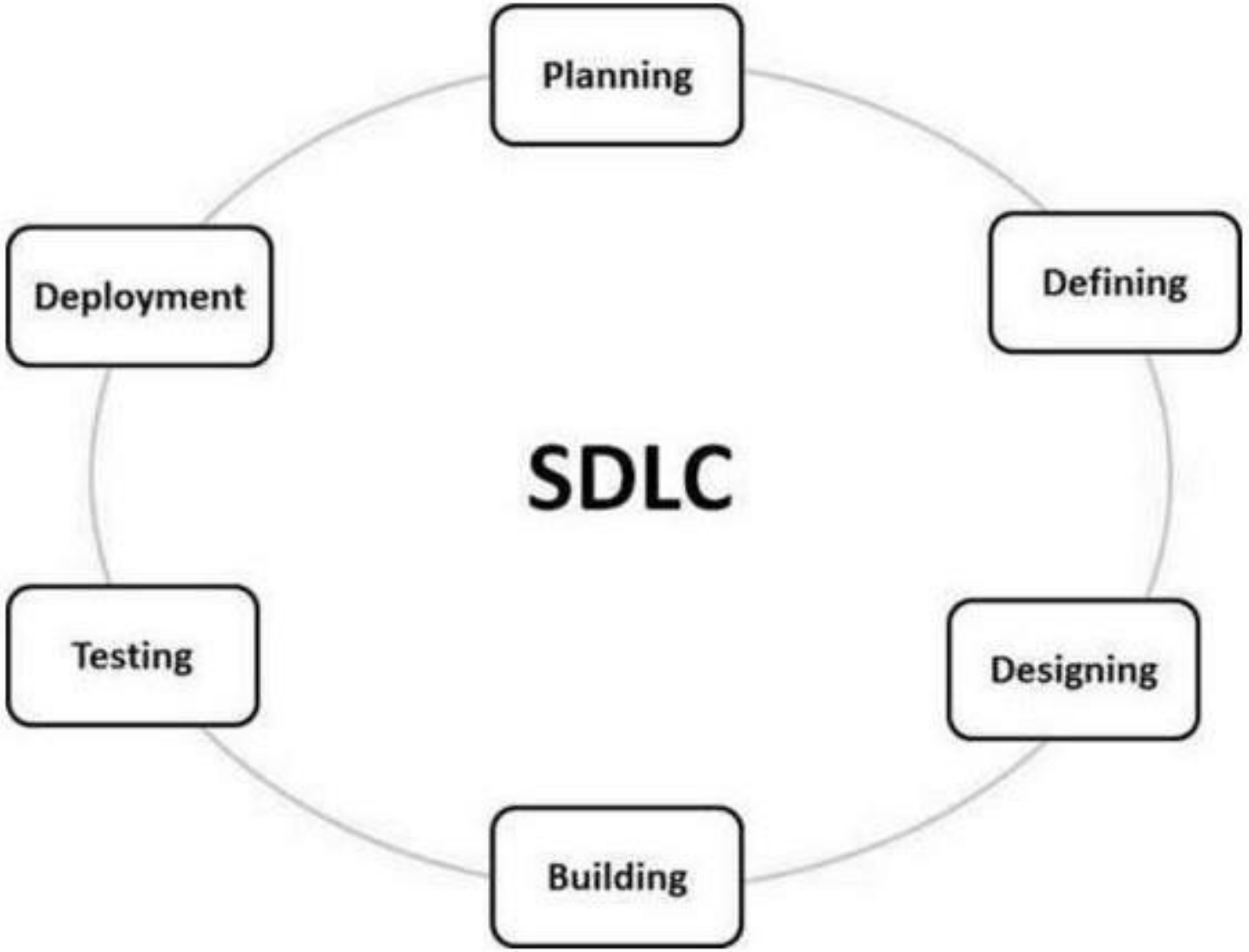
3/. Các mô hình phát triển phần mềm và kiểm thử

Mô hình RAD:



3/. Các mô hình phát triển phần mềm và kiểm thử

Vòng đời Phát triển PM (SDLC):



3/. Các mô hình phát triển phần mềm và kiểm thử

- Vòng đời phát triển PM _ SDLC (Software Development Life Circicle) là một quá trình của một dự án phần mềm.
- Mỗi mô hình phát triển PM sẽ có sự áp dụng khác nhau, tuy nhiên, mỗi mô hình có thể bao gồm tất cả hoặc một số giai đoạn / hoạt động / nhiệm vụ sau:

3/. Các mô hình phát triển phần mềm và kiểm thử

1/. Planning - Lập kế hoạch:

- Phân tích yêu cầu được thực hiện bởi các thành viên của nhóm với thông tin đầu vào được thu thập từ khách hàng, bộ phận nghiệp vụ, hồ sơ của hệ thống cũ và các chuyên gia trong ngành.
- Lập kế hoạch cho các yêu cầu đảm bảo chất lượng và xác định các rủi ro liên quan đến dự án.

2/. Defining - Xác định yêu cầu:

- Xác định rõ ràng và ghi nhận lại các yêu cầu.
- Thực hiện khảo sát, thu thập, xác định các yêu cầu thông qua một tài liệu SRS (Software Requirement Specification) bao gồm tất cả các yêu cầu sản phẩm được thiết kế và phát triển trong suốt vòng đời của dự án.

3/. Designing - Phân tích thiết kế hệ thống:

- Trong giai đoạn này, thiết kế hệ thống và phần mềm được chuẩn bị từ các đặc tả yêu cầu đã được nghiên cứu trong giai đoạn đầu tiên.
- Thiết kế hệ thống giúp xác định các yêu cầu phần mềm, yêu cầu phần cứng và kiến trúc hệ thống tổng thể.

4/. Building - Phát triển:

- Khi nhận được tài liệu thiết kế hệ thống, công việc được chia thành các module nhỏ và nhiệm vụ coding được bắt đầu.
- Đây là giai đoạn trọng tâm và dài nhất của vòng đời phát triển phần mềm.

5/. Testing – Kiểm thử:

- Sau khi coding được phát triển, nó được kiểm tra dựa trên các yêu cầu để đảm bảo rằng sản phẩm thực sự hoạt động đúng trong giai đoạn phân tích yêu cầu.
- Trong giai đoạn này tất cả các loại kiểm thử chức năng như: kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử hệ thống, kiểm thử chấp nhận được thực hiện cũng như kiểm thử phi chức năng cũng được thực hiện.

6/. Deployment - Phát hành / Triển khai:

- Sau khi kiểm thử thành công, sản phẩm được PM được phân phối / triển khai cho khách hàng để vận hành / sử dụng.
- PM có thể trải qua giai đoạn Test Beta, nếu có vấn đề xảy ra, thì khách hang sẽ báo cáo cho nhóm kỹ thuật để được hỗ trợ, khắc phục lỗi.

3/. Các mô hình phát triển phần mềm và kiểm thử

7/. Maintenance - Bảo trì:

- Giai đoạn này, hệ thống được đánh giá để đảm bảo nó được nâng cấp không lỗi thời.
- Hệ thống đã phát triển thì những vấn đề về cập nhật / nâng cấp / sửa đổi / cải tiến thực sự xuất hiện và cần được giải quyết.

4/. Định nghĩa Lỗi phần mềm

- **Lỗi phần mềm** là một lỗi hay hỏng hóc trong chương trình hoặc hệ thống máy tính khiến nó tạo ra kết quả không chính xác hoặc không mong muốn hoặc hành xử theo những cách không lường trước được. Quá trình tìm và sửa lỗi được gọi là "gỡ lỗi" và thường sử dụng các kỹ thuật hoặc phương pháp để xác định lỗi. Hiện nay, có một số hệ thống máy tính đã được thiết kế để ngăn chặn, phát hiện hoặc tự động sửa các lỗi máy tính khác nhau trong quá trình hoạt động.
- Có nhiều định nghĩa khác nhau về lỗi phần mềm như sau:

4/. Định nghĩa Lỗi phần mềm

- Lỗi PM là một sự bất thường trong phần mềm có thể khiến nó hoạt động không chính xác, và không theo đặc điểm kỹ thuật của nó.
- Một lỗi là kết quả của một lỗi mã hóa, một khiếm khuyết là một sai lệch so với yêu cầu.
- **Thảo luận về lỗi PM**

Lỗi PM không nhất thiết có nghĩa là có lỗi trong mã, nó có thể là một chức năng không được triển khai nhưng được xác định trong các yêu cầu của phần mềm.

5/. Phân loại lỗi

- Có nhiều tiêu chí phân loại lỗi khác nhau:

1/. Phân loại lỗi theo mức độ nghiêm trọng:

- Block (Lỗi nghiêm trọng): Ví dụ: ứng dụng bị crash, trang/ứng dụng treo, mật khẩu mới không được nhận, thanh toán đơn hàng bị từ chối, không thêm mục vào giỏ hàng được
- Major (Lỗi lớn): thực hiện chức năng lúc được lúc không, nội dung hiển thị không đúng,..
- Minor (Lỗi nhỏ): nội dung không hiển thị rõ, lỗi chính tả,..

5/. Phân loại lỗi

2/. Phân loại lỗi theo hình thức:

- **Lỗi chức năng (Functional Bug): đ**ây là lỗi quan trọng liên quan tới thao tác thực hiện. Lỗi này chỉ có thể tìm thấy khi thao tác gì đó và PM không phản hồi như mong muốn. Ví dụ: chuyển hướng tới trang 404, nút không hoạt động, bộ lọc không hoạt động,..
- **Lỗi đồ họa (Graphical Bug)**: ảnh / nội dung bị mờ, không cân đối,..
- **Lỗi từ ngữ (Wording Bug)**: lỗi chính tả, văn bản không đồng nhất,..
- **Lỗi hoạt động (Performance Bug)**: tải trang lâu, bị gián đoạn,..

5/. Phân loại lỗi

3/. Phân loại lỗi theo tần suất:

- **Luôn luôn: p**hần nhiều lỗi sẽ lặp lại nhiều lần (cho tới khi bạn khắc phục) và điều này là tốt để phát hiện.
- **Ngẫu nhiên**: những lỗi này là khó, không biết được là xảy ra trong điều kiện nào. Cần kiên nhẫn kiểm tra từng bước để phát hiện lỗi này.
- **Một** lần: các lỗi này chỉ xuất hiện 1 lần duy nhất, có thể đúng là lỗi thật nhưng điều kiện xảy ra lỗi cũng là bí ẩn, Tester thường bỏ qua lỗi này.

5/. Phân loại lỗi

4/. Phân loại lỗi theo đặc tả:

- Lỗi sai: Sản phẩm phần mềm được xây dựng khác với đặc tả.
- Lỗi thiếu: Các yêu cầu của sản phẩm phần mềm đã có trong đặc tả nhưng lại không có trong sản phẩm PM thực tế.
- Lỗi thừa: Sản phẩm PM thực tế có những tính năng không có trong tài liệu đặc tả.

5/. Phân loại lỗi

Thảo luận:

Sinh viên trình bày và phân loại các lỗi xảy ra khi truy cập Internet.

- ❑ Lỗi chức năng
- ❑ Lỗi giao tiếp
- ❑ Lỗi thiếu lệnh
- ❑ Lỗi cú pháp
- ❑ Lỗi xử lý lỗi
- ❑ Lỗi tính toán
- ❑ Lỗi luồng điều khiển

5/. Phân loại lỗi

5/. **Đánh giá mức độ nghiêm trọng và độ ưu tiên trong quản lý bug:**

- Trong kiểm thử phần mềm thì hai khái niệm Độ ưu tiên (Priority) và Độ nghiêm trọng (Severity) cũng không quá xa lạ, đặc biệt là trong quản lý bug. Hai khái niệm trên đã trở nên quá quen thuộc và phổ biến đến nỗi chúng ta hầu như không phân biệt được ý nghĩa cũng như sự khác nhau giữa hai khái niệm đó. Mặc dù hai yếu tố này không phải là yếu tố sống còn trong quản lý bug nhưng việc hiểu đúng sẽ giúp chúng ta tiết kiệm thời gian cũng như làm công việc hiệu quả hơn.

5/. Phân loại lỗi

- **Độ ưu tiên:** Mức độ ưu tiên xác định thứ tự mà chúng ta nên giải quyết một Bug. Chúng ta có nên sửa nó ngay bây giờ, hay nó có thể có thể để sau được không? Trạng thái ưu tiên này được xác định bởi kỹ sư kiểm thử để người lập trình có thể sắp xếp khoảng thời gian sửa lỗi vào giai đoạn nào cho phù hợp. Nếu bug có mức ưu tiên cao thì người lập trình phải khắc phục sớm nhất. Mức độ ưu tiên được xác định dựa trên yêu cầu của khách hàng.

Ví dụ: Nếu tên công ty sai chính tả trong trang chủ của trang web được xem là có độ ưu tiên thấp.

5/. Phân loại lỗi

Độ ưu tiên của con bug cũng thường được chia thành 3-4 cấp độ:

- Mức độ 1: Cao – Bug sẽ phải sửa ngay lập tức
- Mức độ 2: Trung bình – Bug sẽ sửa trong bản cập nhật lần tới
- Mức độ 3: Thấp – Bug không cần sửa trong bản cập nhật lần tới, có thể sửa sau nếu có thời gian

5/. Phân loại lỗi

Độ nghiêm trọng: Đó là phạm vi mà lỗi có thể ảnh hưởng đến phần mềm. Nói cách khác, nó định nghĩa tác động mà một khiếm khuyết nhất định có trên hệ thống.

Ví dụ: Nếu ứng dụng hoặc trang Web gặp sự cố khi ta click chọn liên kết , trong trường hợp này tác động của lỗi là nghiêm trọng. Vì vậy, mức độ nghiêm trọng là cao, nhưng ưu tiên là tùy thuộc vào nội dung của liên kết. Mỗi dự án hay sản phẩm phần mềm sẽ có tiêu chí đánh giá độ nghiêm trọng khác nhau.

5/. Phân loại lỗi

Thông thường sẽ có 4-5 mức độ khác nhau từ nghiêm trọng nhất đến ít nghiêm trọng hơn:

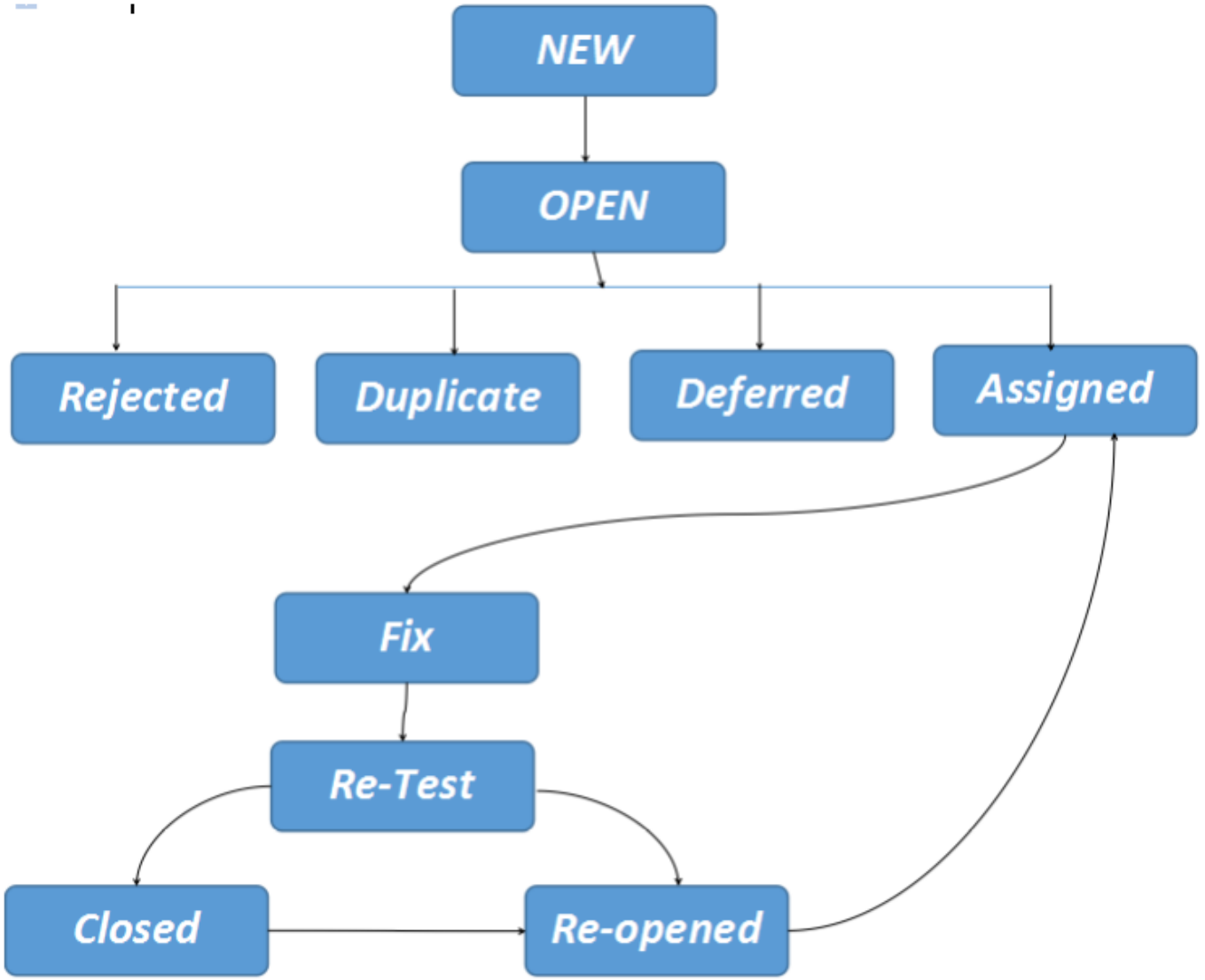
- Mức độ 1: Hệ thống sụp đổ, dữ liệu bị mất, ứng dụng không làm việc được …
- Mức độ 2: Chức năng chính của hệ thống không hoạt động
- Mức độ 3: Chức năng phụ của hệ thống không hoạt động
- Mức độ 4: Bug nhỏ, không quan trọng
- Mức độ 5: Yêu cầu cải tiến sản phẩm, thêm chức năng mới …
- Tương tự mức độ ưu tiên, mức độ nghiêm trọng cũng như ý nghĩa của chúng có thể sẽ khác nhau ở những sản phẩm, dự án khác nhau.

5/. Phân loại lỗi

Độ ưu tiên và độ nghiêm trọng chỉ là hai trong số các thuộc tính của phần mềm, chúng ta còn có nhiều thuộc tính khác như: môi trường của bug, mức độ lặp đi lặp lại, các bước mô tả con bug, phạm vi của bug v.v. Tuy nhiên, việc hiểu đúng về mức độ nghiêm trọng, độ ưu tiên của sản phẩm cho thấy người kiểm thử thực sự hiểu rõ và quan tâm đến chất lượng sản phẩm PM cũng như thể hiện sự chuyên nghiệp của một kỹ sư kiểm thử Hy vọng người học nhận thức rõ hơn về mức độ nghiêm trọng và ưu tiên trong quản lý bug.

6/. Vòng đời của lỗi

Mục tiêu của Tester không chỉ là tìm ra lỗi của PM mà còn phải theo dõi lỗi cho đến khi kết thúc lỗi đó. Hình bên là sơ đồ vòng đời của lỗi PM:



6/. Vòng đời lỗi

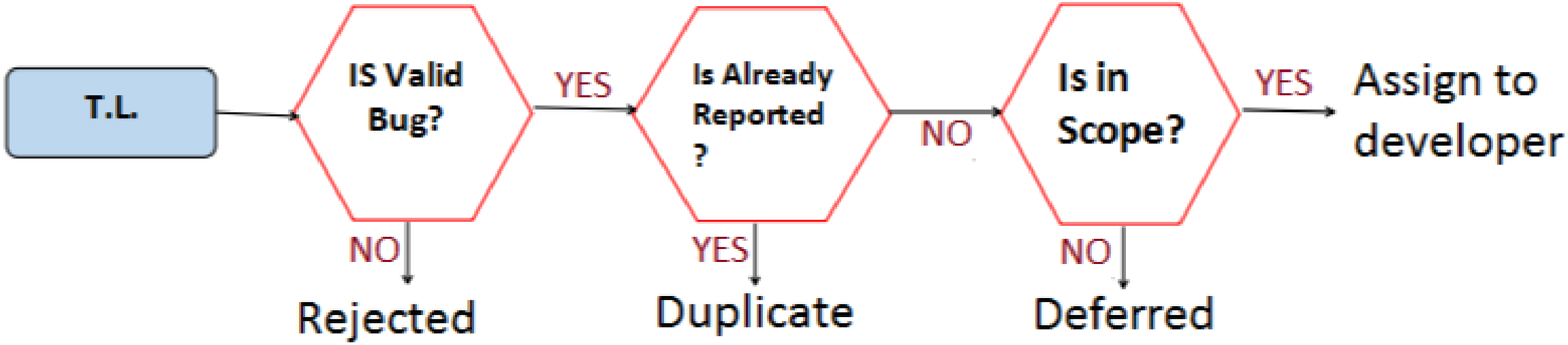
+ **New**: Khi Tester thực thi test case và đầu ra của test case đó không đúng như kết quả mong đợi, thì họ sẽ gọi đó là lỗi / bug.

Do đó lỗi này cần phải được sửa / fix. Nhưng tester không phải là người fix bug mà là nhóm lập trình sẽ sửa nó. Tester sẽ báo bug này như thế nào? Tester cần báo cáo lỗi này cho Team Lead và Team Lead sẽ giao cho Developer sửa lỗi này sau khi đã phân tích.

Khi Tester tìm thấy lỗi thì nó sẽ có trạng thái là NEW

6/. Vòng đời lỗi

+ **Open**: Lỗi được báo cáo lên bởi Tester. Team lead cần xác minh lại lỗi đó xem có đúng hay không ?, Nếu đúng thì lỗi có trạng thái OPEN. Sơ đồ dưới đây là những hoạt động được thực hiện bởi Team Lead (T.L):



6/. Vòng đời lỗi

+ **Rejected** (từ chối): Một lỗi được đánh dấu là Rejected khi lỗi đó không hợp lệ. Nghĩa là thỉnh thoảng Tester có thể hiểu sai chức năng và có thể đánh dấu chức năng đó là lỗi. Trong trường hợp này, lỗi sẽ bị Reject sau khi T.L kiểm tra lại.

6/. Vòng đời lỗi

+ **Duplicate** (trùng lặp): Nếu lỗi là hợp lệ, thì T.L sẽ kiểm tra xem lỗi đó đã được báo bởi người khác hay chưa. Nếu đã có người khác báo lỗi rồi, thì T.L sẽ đánh dấu nó là Duplicate. Còn nếu nó chưa được báo cáo bởi tester khác thì team lead sẽ thực hiện tìm kiếm nó trong phạm vi làm việc nhằm đảm bảo rằng, cùng một lỗi sẽ không được báo cáo 2 lần hoặc nhiều hơn thế.

Nếu cùng một bug dược báo cáo bởi hai hay nhiều tester thì lỗi được báo cáo sau sẽ được đánh dấu là Duplicate

6/. Vòng đời lỗi

+ **Deferred** (hoãn lại): Nếu lỗi không là Duplicate, nhưng lại không thuộc bản phát hành (release) hiện tại thì sẽ được đánh dấu là Deferred. Giả sử ta đang làm việc theo mô hình Agile, và họ chia yêu cầu dự án thành các sprint, ví dụ chia thành 10 sprint: sprint 1, sprint 2, ..., sprint 10. Hiện tại đang ở sprint 1, nhưng bug bạn tìm thấy lại có liên quan đến tính năng sẽ được phát triển ở sprint 2, thì bug này sẽ được đánh dấu là Deferred. Deferred bug là một lỗi, nhưng nó sẽ được sửa chữa trong bản release tương lai. Khi một lỗi được xác định thuộc về bản release tương lai thì nó sẽ được đánh dấu là Deferred.

6/. Vòng đời lỗi

+ **Assigned** (giao lỗi): Khi lỗi tìm thấy là hợp lệ, duy nhất và thuộc bản release hiện tại, thì T.L sẽ giao / chuyển lỗi đó cho Developer (xem để sửa).

+ **Fix** (sửa lỗi): Khi nhận được bug từ T.L, Developer sẽ thực hiện thay đổi để sửa lỗi cho đúng với yêu cầu, và chuyển lại cho Tester kiểm tra lại lỗi đó.

6/. Vòng đời lỗi

+ **Re-Testing** (kiểm thử lại): Sau khi sửa xong lỗi, và chức năng / tính năng đã sẵn sàng để kiểm thử, thì Tester sẽ thực hiện lại những test case lỗi và xác minh lại xem nó đã chạy đúng hay chưa. Việc này gọi là kiểm thử lại.

+ **Closed** (đóng lỗi): Khi lỗi đã được sửa, đã được kiểm thử lại và nó chạy đúng như yêu cầu thì Tester sẽ đánh dấu nó là Closed.

6/. Vòng đời lỗi

+ **Re-Opened** (mở lại lỗi): Có 2 tình huống mà chúng ta cần phải Re-Open lại lỗi:

- Tình huống 1: Khi Developer sửa lỗi và Tester thực hiện test lại nó, nhưng sau khi re-test, lỗi đó vẫn xảy ra thì Tester sẽ Re-Open lại lỗi và giao / Assign cho Developer,..
- Tình huống 2: Có trường hợp lỗi đã sửa và được Close, mà lỗi lại xuất hiện. Trong trường hợp này, Tester cần Re-Open lại lỗi đã Closed và gán nó lại cho Developer.

7/. Bảo đảm chất lượng phần mềm

- **Định nghĩa theo Daniel Galin**: Đảm bảo chất lượng phần mềm (Software Quality Assure) là một tập hợp các hành động cần thiết được lên kế hoạch một cách có hệ thống để cung cấp đầy đủ niềm tin rằng quá trình phát triển phần mềm phù hợp để thành lập các yêu cầu chức năng kỹ thuật cũng như các yêu cầu quản lý theo lịch trình và hoạt động trong giới hạn ngân sách.

7/. Bảo đảm chất lượng phần mềm

- **Thảo luận: BĐCLPM và KTPM**

7/. Bảo đảm chất lượng phần mềm

- Cả hai đều có mục tiêu, chức năng và phương pháp khác nhau để áp dụng. Đảm bảo chất lượng phần mềm dựa trên kế hoạch hợp lý để cung cấp ứng dụng không có lỗi bao gồm tài liệu Đặc tả yêu cầu phần mềm (Software Requirements Specification - SRS). Đảm bảo chất lượng phần mềm cũng tạo và thực hiện các phương pháp và quy trình để cải thiện chu trình phát triển tổng thể, trong khi đó Kiểm thử phần mềm tập trung vào việc xác minh và xác thực sản phẩm để phát hiện ra lỗi và xác định lỗi.

7/. Bảo đảm chất lượng phần mềm

- Đảm bảo chất lượng phần mềm là một cách tiếp cận được thiết kế để đảm bảo ứng dụng có qui trình đáp ứng các yêu cầu cụ thể. Đây là một chiến lược lập kế hoạch cho quá trình kiểm thử của một ứng dụng nhằm mục đích mang lại chất lượng sản phẩm. QA là tất cả về việc chuẩn bị một kế hoạch hoặc chiến lược phù hợp theo sau bởi những Tester để thực hiện các trường hợp kiểm thử khác nhau và làm cho ứng dụng không có lỗi.

7/. Bảo đảm chất lượng phần mềm

- Mặt khác, kiểm thử phần mềm là các hoạt động để kiểm tra / kiểm thử xem kết quả thực tế có khớp với kết quả mong đợi hay không. Nó cũng thực hiện các trường hợp kiểm thử khác nhau để kiểm thử phần mềm. Kiểm thử phần mềm xác định các lỗ hổng, lỗi và các yêu cầu còn thiếu và cố gắng khắc phục tất cả. Ngoài ra, kiểm thử phần mềm có thể được thực hiện bằng cách kiểm thử thủ công hoặc tự động.

7/. Bảo đảm chất lượng phần mềm

- Có thể có sự chồng chéo giữa đảm bảo chất lượng và kiểm thử trong các dự án, đặc biệt là các dự án quy mô nhỏ. Kiểm thử có thể cung cấp thông tin liên quan đến chất lượng quan trọng về sản phẩm, nhưng đảm bảo chất lượng là những gì sử dụng thông tin này để cải thiện các quy trình liên quan đến chất lượng.
- Trong một số dự án, Đảm bảo chất lượng còn có thể được tham gia để tiến hành kiểm toán các quy trình liên quan đến chất lượng.

The End