Ch(1+2)

Software Engineering

Mohammad JadAllah

Ch.1 8

بسم الله الرحمن الرحيم

Software System

Business system

بناء سوفتوير لمؤسسة أو شركة مثل مكتبة أو فندق ...
تكون الـ budget فيه معقولة والوقت محدود

Critical System

بناء سوفتوير بأنظمة حيوية جداً وذي فضاء يسبب خسائر كبيرة بالمال أو الأرواح - مثل أنظمة الأجهزة الطبية أو الطائرات وغيرها وتكون الـ budget عالية جداً ومدتها طويلة جداً

*Note that Software engineering is part from System engineering

⑧ Attributes of good Software :-

1. deliver the required functionality

⇐ يقوم الوظيفة المطلوبة منه

2. Acceptable performance to the user

⇐ يقوم أداء مقبول للمستخدم

3. should be maintainable

⇐ قابل للصيانة والتعديل بسرعة

4. dependable

⇐ يمكن الاعتماد على نتائجه

5. Usable

⇐ سهل الاستخدام من اليوزر

Attributes of a Good Software 2

Product characteristics	Description
Maintainability ①	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability and security ②	Software dependability includes a range of characteristics including reliability, security, and safety. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.
Efficiency ③	Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilization, etc.
Acceptability ④	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable, and compatible with other systems that they use.

يتم اعتماد هذه النقاط مع الشرح

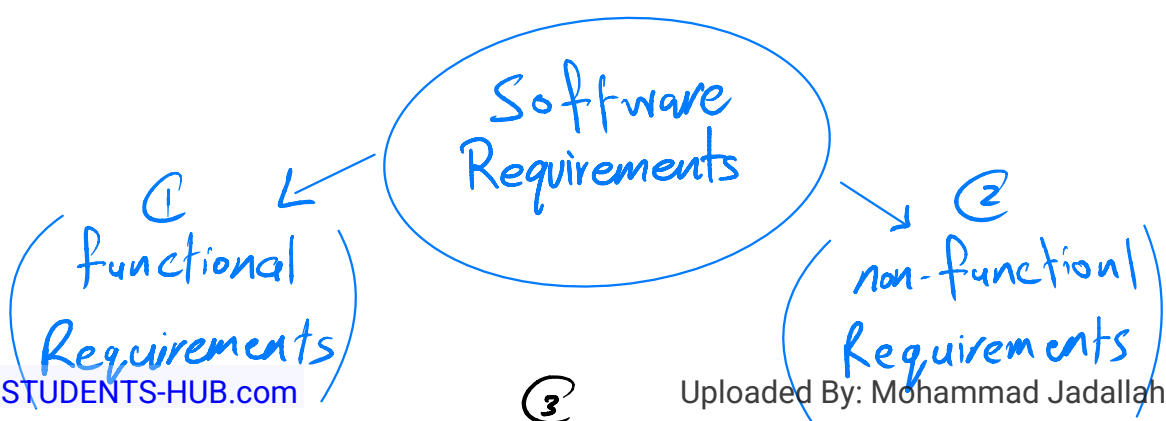
مقارنة للذكور في الصفحة السابقة

① القدرة على تطوير النظام وتصليح الأعطال (صيانة)

② الموثوقية والأمان

③ مدى استهلاك الـ Software للـ hardware (جديد)

④ مدى سهولة استخدام النظام (usability)



1 functional Requiremnts:

=> كل action بعملو ال user. بدخل في input, والنظام يعطيني output مثل: تسجيل الدخول، عرض تاريخ، استعراض جدول ...

2 non-functional Requiremnts:

=> هي صفات النظام بشكل عام (هي attributes of good software)

Case study about functional and non-functional

* Case A:

"Our new sales information system seems okay, the invoices are correct, the inventory records are correct, the discounts granted to our clients exactly follow our very complicated discount policy, but our new sales information system frequently fails, usually at least twice a day, each time for twenty minutes or more. Yesterday it took an hour and half before we could get back to work . . . . Imagine how embarrassing it is to store managers . . . . Softbest, the software house that developed our computerized sales system, claims no responsibility . . . ."

=> non-functional, Reliability

* Case B :~

"Believe it or not, our software package 'Blackboard' for schoolteachers, launched just three months ago, is already installed in 187 schools. The development team just returned from a week in Hawaii, their vacation bonus. But we have been suddenly receiving daily complaints from the 'Blackboard' maintenance team. They claim that the lack of failure-detection features in the software, in addition to the poor programmer's manual, have caused them to invest more than the time estimated to deal with bugs or adding minor software changes that were agreed as part of purchasing contracts with clients."

=> non-functional, also maintainability

* Case C :~

"The new version of our loan contract software is really accurate. We have already processed 1200 customer requests, and checked each of the output contracts. There were no errors. But we did face a severe unexpected problem – training a new staff member to use this software takes about two weeks. This is a real problem in customers' departments suffering from high employee turnover . . . . The project team says that as they were not required to deal with training issues in time, an additional two to three months of work will be required to solve the problem."

=> non-functional, usability

* من ضمن المعايير المختصة بالusability هي المدة التي يستغرقها المستخدم لتعلم استعمال البرنامج

other examples about non/functional requirements from Dr. Samer slides

Example

1. The user must be able to purchase tickets ans: functional
2. The user must be able to access traffic information ans: functional
3. The system must be provided feedback in less than one second ans: non-functional, performance
4. The colors used in the interface should be consistent with the company LOGO non-functional, usability
5. System should be easy to use since users could be of different ages non-functional, usability

- **Other nonfunctional requirements may include:**

1. using specific hardware platform for the system, non-functional
2. security requirements, non-functional
3. how the system should deal with failures and faults, non-functional
4. and how to provide backward compatibility with an old system that the client is unwilling to retire. non-functional

④ Why software projects fail?

Why software projects fail?

1 • Inaccurate understanding of customer needs
2 • Inability to deal with changing requirements
3 • Modules that do not fit together
4 • Software that are hard to maintain/extend
5 • Poor Quality
6 • Testing….normally should cost 40%
7 • Unacceptable performance
8 • Technology change and team-member change over time in long period projects

1] سوء فهم الـ customer

* ملاحظة: يجب إبقاء هامش للتغيير ولكن بحدود

2] تغيير ما الـ requirements من الـ customer

3] تجميع الـ components بعد بناء كل تيم لواحد ممكن أن يفشل

4] صعب التعديل عليه مستقبلاً (not maintainable)

5] جودة سيئة (bad non-functional requirements)

6] لم يجرب بشكل كافي

* ملاحظة: الـ Testing الطبيعي يأخذ 40% من الـ cost، إذا كان critical ممكن أن يصل إلى (70% - 80%)

7] تندرج تحت بند poor quality الأداء السيء

8] يجب أن يكون لكل component عضوين على الأقل يعرفو وضعه تحسباً لأي ظرف عند أحد الأعضاء (مرض، سفر، ظروف طارئة ...). ((صرح أحد المحاضرين))

What is successful software Projects?

So what is a successful software project?

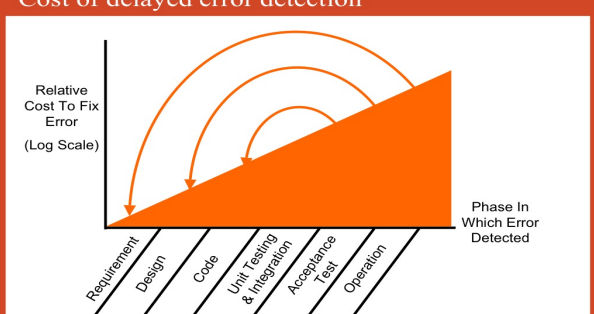
- Good software should:
  1. Deliver the required functionality
  2. Efficient: does not waste voluble resources, response time
  3. Usable
  4. Dependable: reliable, secure, and safe.
  5. Maintainable
  6. **Within budget and time**

← على عكس أسباب فشل المشاريع

← إضافة إلى ذلك الالتزام بالوقت والميزانية المتفق عليها.

Causes:

Cost of delayed error detection



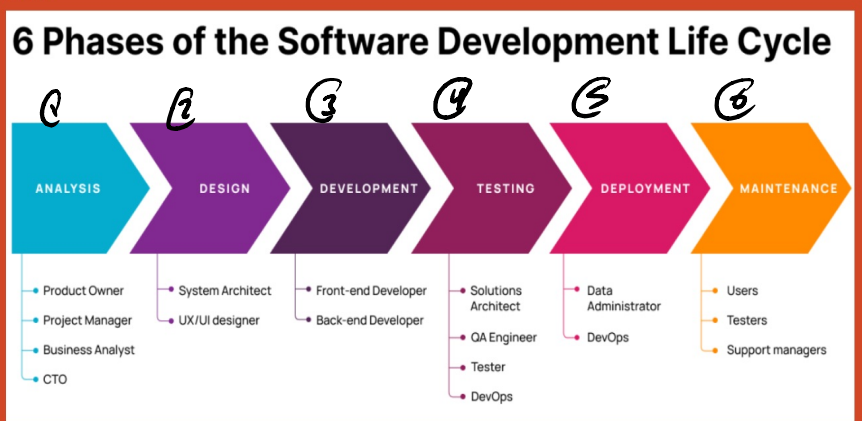
← صورة توضح العلاقة بين الـ cost والـ Fix Errors حسب المرحلة الحالية

* كلما تم اكتشاف الخلل (والخطأ) أبكر كلما كانت الخسائر
* يتم على نسخة Beta (تجريبية) قبل تسليم المشروع للتأكد من عدم وجود مشاكل
* تجربة المستخدم الطبيعي تختلف عن تجربة المبرمج لذلك يجب التأكد من عدم وجود مشاكل.

⑧ Software development process

Software Development Process

- A systematic approach for software development.
- It is a sequence of activities that leads to the production of software.



26

1. Analysis : what thy system should do. مرحلة جمع وتحليل وهي الأساس
2. Design : How the system will do. تعتبر المرحلة الأصعب
3. Development : Coding تعتمد على الـ Design
4. Testing : هي
5. Deployment : in the real enviroment (تسليم الـ Software)
6. Maintenance : Changes and updates (أطول مرحلة)

Different types of software

1) **Stand Alone** : office Products

سوفتوير جاهزة تعطى لأكثر من customer
يكون سعرها أقل والـ quality أعلى ((تجربة))

2) **Interative transaction based** : Ritaj

عبارة عن واجهات إما تحذف أو تضيف
على داتابيس موجودة

3) **Embedded systems** : car software systems

تشغيل السيارات والآلات

4) **Entertaiment** : Games

ألعاب وترفيه

5) **Data Collections** : Mars rovers

تحكم عن بعد في الآلات التي تبعث للفضاء

٥ مصطلحات سيتم استخدامها

Generic Technical Terms:

1. **Notation**: a graphical or textual set of rules for representing a model (UML)
2. **Method:** a repeatable technique that specifies the steps involved in solving a specific problem (Sorting Algorithm)
3. **Methodology:** a collection of methods for solving a specific set of problems.

١ شكل أو وظيفة أو حتى معين

٢ طريقة معينة للقيام بشكل معين

٣ مجموعة من الـ methods ضمن إطار معين

مثال: إحدى طرق جمع الـ requirements هي الـ interview طريقة أخرى documents كل طريقة هي method و مجموعهم معا methodology

⑬ What are the key challenges facing software engineering

Software engineering in the 22st century faces three key challenges:

- **Legacy systems** ①
  - Old, valuable systems must be maintained and updated
- **Increasing Diversity and Heterogeneity** ②
  - Systems are distributed and include a mix of different hardware and software
- **Dependability and Delivery** ③
  - Having trustworthy software with faster delivery of software (time-to-market)

① في حال وجود نظام قديم لا يريد الـ customer استبداله بالتالي سنضطر للتعامل معه لبناء الـ software ومن الممكن أن يكون من الصعب التعامل معه

② في تطبيقات الموبايل بناء تطبيق يصلح لجميع الأجهزة الحديثة مثلاً من الممكن بناء تطبيق android يصلح لأجهزة سامسونج وه يصلح لأجهزة شاومي وهكذا.

③ بناء برمجيات وتسليمه بدون مشاكل ضمن الوقت والميزانية المحددة.

Ch 2: Software Development Process

There are many different software processes but all must include four activities ~

1. Software specification
2. Design & implementation
3. Validation
4. Evolution

هو هاد هي كل Software Development Process

also in software process:

Software Processes: intro..2

- When we talk about processes, it is not only about the activities and their order.
- It is also about:
  - **Products: outcomes of each activity, example Software Architecture Document**
  - **Roles: programmer, designer, analyst, tester, team leader, etc.**
  - **Pre and Post Conditions: example**
    - **Software requirements must be ready before building software architecture.**
    - **UML models must be designed and reviewed.**

Critical and Business systems from Ch.1

- Critical Systems: a very structured development process is required
- A Business System with rapidly changing requirements, a less formal, flexible process is likely to be more effective

Note: Critical system very structured and linear
Business system is more incremental and agile
Ch2 في التفاصيل

Two Categories of Software Processes

All process activities are planned in advance
Progress is measured against this plan

Software Processes

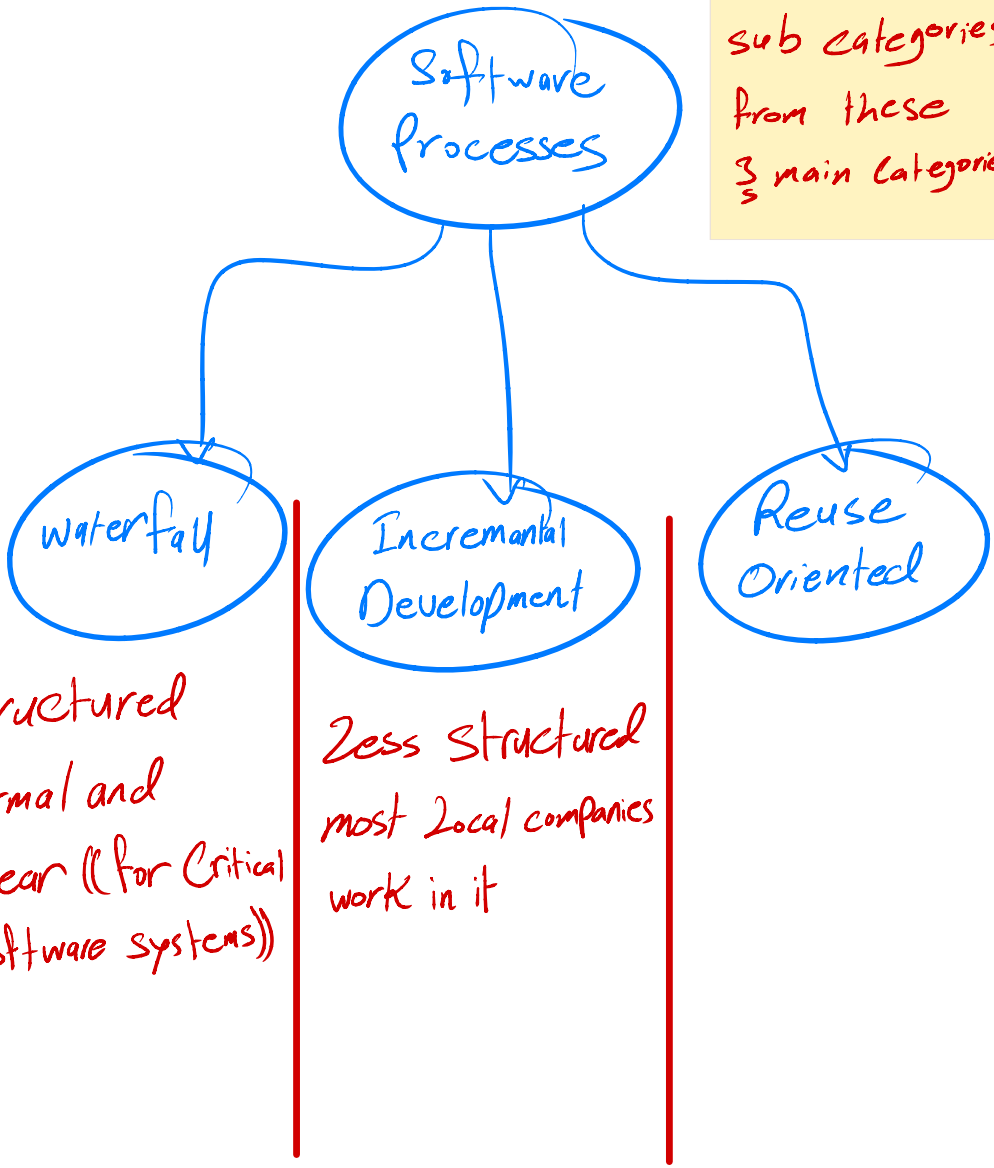
Plan Driven

Planning is incremental
Flexible
Cope with change

Agile

⭐ There is 3 main categories in Software Processes

There is also sub categories from these 3 main categories



Waterfall Model

=> Derived from more general engineering System
تم أخذها بالأساس من الهندسة الكهربائية والكهربية

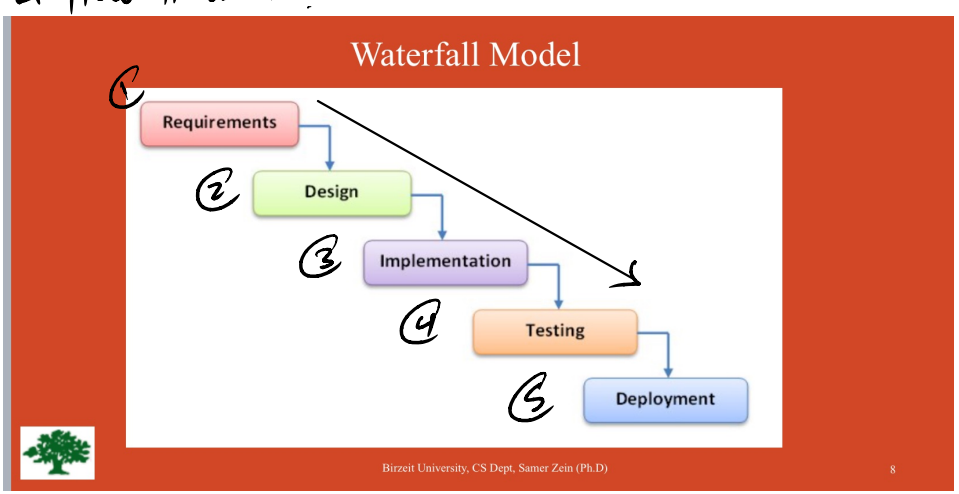
=> It is the best used for critical system
((requirements are well understood and will not change)) هو الأفضل لل critical system

Why?

=> it's very risky in business system

=> it's good for small projects

How it work?



لا يتم الإنتقال إلى المرحلة التالية إلا بعد الإنتهاء بشكل كامل من المرحلة السابقة

Why it's very risky for business systems ? (ليس دائماً سيئ لكنه خطر فال business)

ans: requirements are not clear, maybe fault and there is a great possibility that it will change (critical system عكس ال)

* سبب آخر ال Testing في نهاية المشروع ومن اعمق اكتشاف مشاكل كبيرة نحتاج الى وقت و budget لإصلاحها ونحن لا نملكها

⑧ Advantages of waterfall Model

Advantages of Waterfall Model

1. Developers and customers agree on what will be delivered early in the development lifecycle. This makes **planning and designing more straightforward**.
2. Progress is more **easily measured**, as the full scope of the work is known in advance.
3. Throughout the development effort, it's possible for various members of the team to be involved or to continue **with other work**, depending on the active phase of the project
4. customer presence is **not strictly required** after the requirements phase.
5. The software can be designed **completely and more carefully**, based upon a more complete understanding of **all** software deliverables

1. الخطة أوضح
2. مرجع للـ Management
3. مجال للموظفين استلام أكثر من مشروع في نفس الوقت مثلاً تم الانتهاء من مرحلة الـ Analysis في المشروع الأول استلام نفس المرحلة في مشروع آخر
4. لا يتدخل العميل customer بعد المرة الأولى كأني سلمته واضح
5. تصميم السوفتوير بشكل متكامل وشامل بسبب امتلاك الوقت بعد فهم كل التفاصيل

⑭ Disadvantages of Waterfall Model

Disadvantages of Waterfall Model

1. One area which almost always falls short is the effectiveness of **requirements**.
2. Gathering and documenting requirements in a way that is meaningful to a customer is often **the most difficult part** of software development.
3. Another potential drawback of pure Waterfall development is the possibility that the customer will be **dissatisfied** with their delivered software product
4. And **Testing** of whole system that only happens at end of project

1. سيكون هناك مشاكل في الـ requirements، إذا وضع في الإمكان الخطأ (business system)
2. مشاكل في عملية كتابة الـ requirements كذلك
3. إنه نقطع عن الـ customer لفترة طويلة
4. مرحلة الـ Testing تكون خالية من الأخطاء

⑮ Formal System Development

1. very expensive
2. high quality
3. for extreme critical

⇒ very strict waterfall system, to highly critical systems

بداية يتم جمع الـ requirements ثم تحويلها إلى mathematical models التي يتم استخدامها بعمل generate للـ code هذا الـ code يكون ذو جودة عالية وذلك للتخفيف من الأخطاء البشرية

طريقة الـ formal

Formal System Development

- An important variant of the waterfall model is formal system development, where a **mathematical model** of a system specification is created.
- This model is then refined, using mathematical transformations that preserve its consistency, into executable code
- Is particularly suited to the development of systems that have stringent **safety, reliability, or security** requirements.
- The formal approach simplifies the production of a safety or security case

Birzeit University, CS Dept, Samer Zein (Ph.D)

12 Incremental Development Model (Part of agile)

Incremental Development

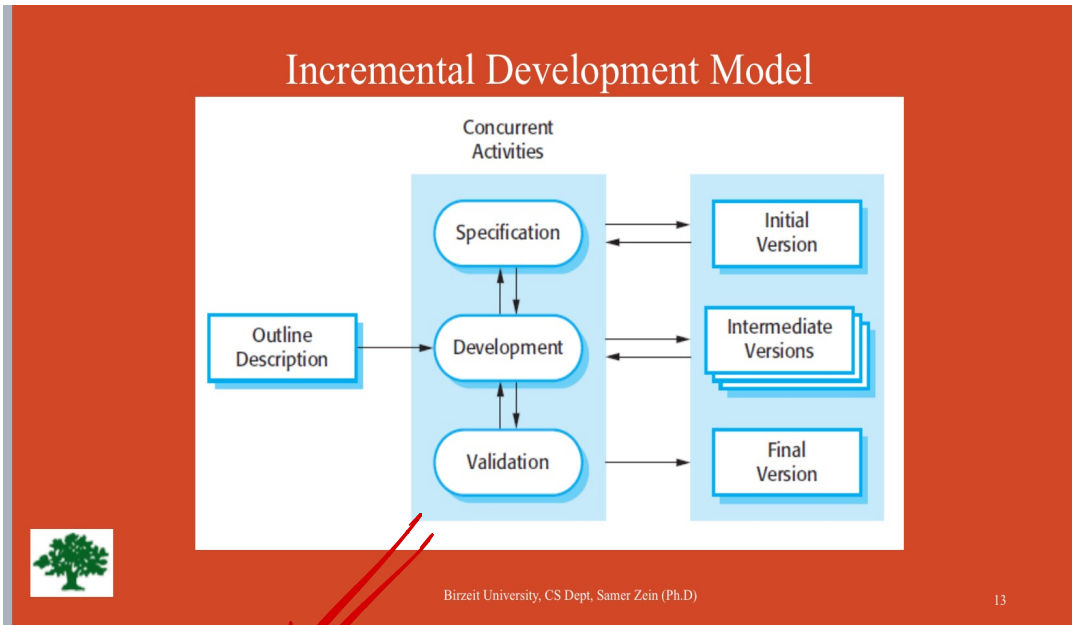
- Incremental development is based on the idea of
  - A) developing an initial implementation,
  - B) exposing this to user comment and
  - C) evolving it through several versions until an adequate system has been developed
- fundamental part of **agile** approaches
- Better than waterfall approach for most e-business, e-commerce, and personal systems.
- **Can be plan-driven, agile, or a mix of both!**

⇐ يتم تجزئة المشروع المراد عمله إلى أجزاء (Sprints)
⇐ كل جزء يتم عمله بشكل كامل بحيث يصبح جاهز للاستخدام ويتم تسليمه
⇐ البداية تكون بالأجزاء الأكثر أهمية والأكثر صعوبة وليس عشوائيا (highest priority and complixity)

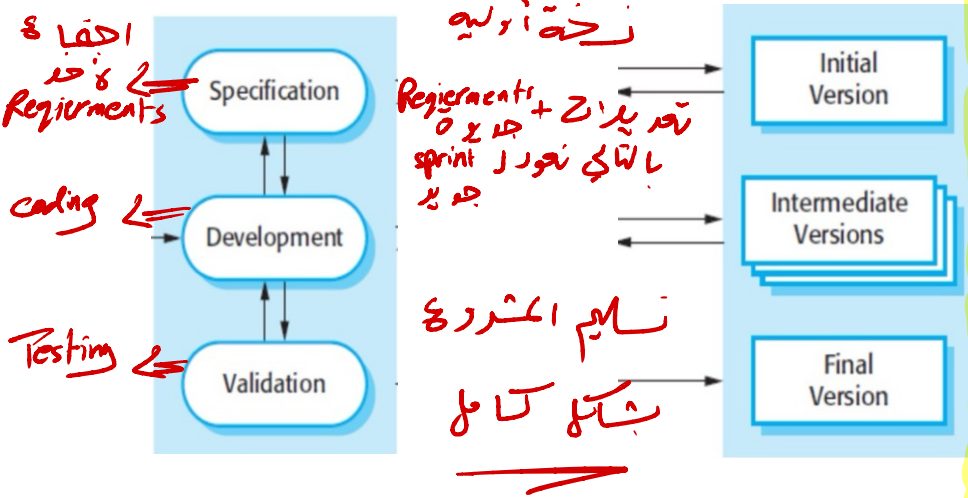
⇐ كل جزء بشكل كامل أي أنه الـ Testing يتم عمله أيضا قبل تسليمه
⇐ يعتبر أفضل من الـ Waterfall لأنه مع الـ business والمشاريع الشخصية
⇐ يتم اللقاء مع الـ customer بشكل دوري (كل أسبوعين تقريبا) بالتالي أي تغيير أو تعديل على الـ Requirements يمكن التعامل معه

⑧ كل sprint يتم تسليمه هو عبارة عن mini waterfall

@ Diagram show the process of incremental



(This is sprint)



نسخة أولية

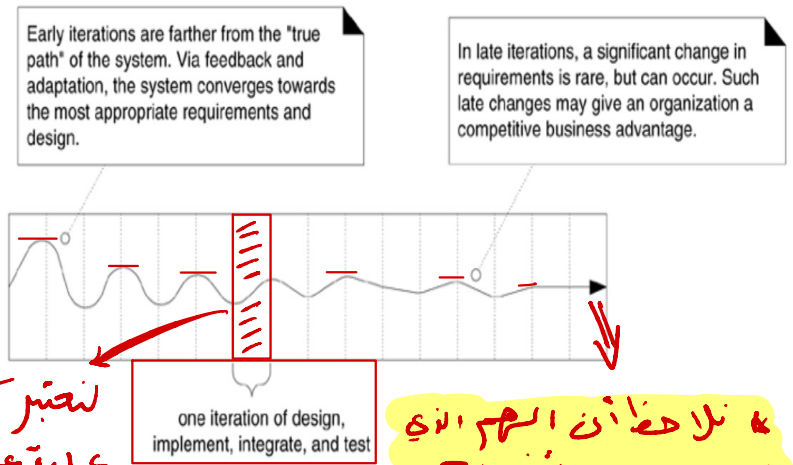
Requirements + جديد يتم إضافة

بالتالي نعود sprint جديد

تسليم المشروع بشكل كامل

يمكن تسليم أجزاء جاهزة للاستخدام قبل إنتهاء من المشروع بشكل كامل وذلك للإستفادة من الأجزاء الكاملة بدلاً من إنتظار (خصوصاً وأن أهم الأجزاء هي التي يتم البدء بها)

(8) صورة توضح العلاقة بين حجم التغييرات كلما تقدمنا في sprints



نعتبر كل sprint عبارة عن ستايل

* نلاحظ أن السهم الذي يشير إلى حجم التغييرات يتخفض مع مرور كل sprint بالتالي تقل التغييرات كلما تقدمنا في المشروع

* because we take the highest task Priority and Complexity in the first sprints

(9) Benefits of incremental

- Benefits of incremental development
  1. The **cost** of accommodating **changing customer requirements** is reduced.
  2. It is easier to get customer **feedback** on the development work that has been done.
  3. More **rapid delivery** and deployment of useful software to the customer is possible
  4. Better fit for short time-to-market

1) حل مشكلة الـ Requirements غير الواضحة بوضوح لتتماشى دورية مع الـ customer لأخذ التعديلات

2) أخذ رأي الـ customer في كل جزء يتم تجهيزه

3) ليس بالضرورة الإنتظار حتى نهاية المشروع بشكل كامل حيث يمكن تسليم الـ customer الأجزاء التي يتم تجهيزها

4) يساعد في عملية التوظيف للشركة في وقت قصير قبل أن يصبح هنالك منافسة على هذا النوع من الشاريع

⑦ Problems of incremental sw

- Problems with incremental development
  1. The process is **not visible**. Managers need regular deliverables to measure progress.
  2. System **structure** tends to degrade as new increments are added.
  3. **Additional unplanned** iterations may be needed.
  4. Customer may **not** have the required **free time** to be involved.

Birzeit University, CS Dept, Samer Zein (Ph.D) 16

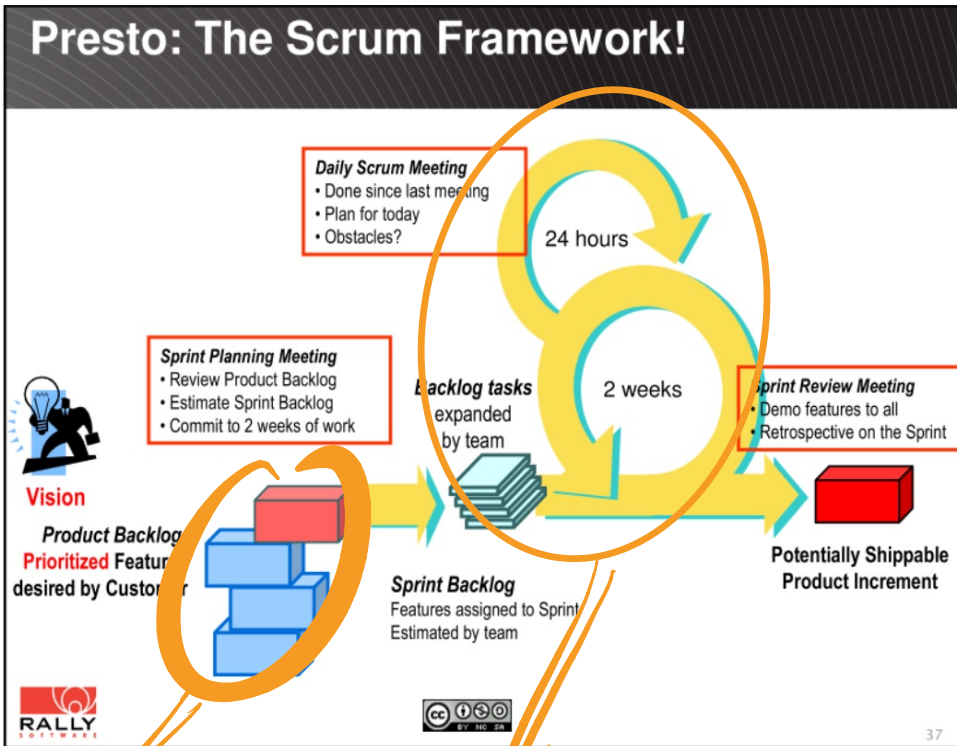
1. الـ Mangers لا يفضلوها لأنها ليست formal (في البداية يمكن إعطاء وقت تقديري للمشروع ولكن بعد البدء وعمل مجموعة من الـ sprints يمكن التقرير بشكل أفضل لأن الـ Managers يأخذ فكرة عن سرعة الفريق في إنجاز الـ sprints)
2. جودة الـ code لا تكون عالية جداً (من الممكن في البداية يتم انجاز القرارات بناءً على الـ Requirements الموجودة وبعد ذلك نكتشف أن هذه القرارات ليست مناسبة جداً لمرحلة أخرى وهذه من أهم الأسباب أيضاً التي تجعله غير مناسب لـ critical).
3. من الممكن ظهور sprints غير مخطط لها وهذه تأتي بسبب وجود (bugs) يمكن تجميعها في sprint إضافي
4. عدم تفرغ الـ customer لعمل لقاءات دورية
5. من الممكن أن يكون الفريق المشرف على المشروع غير متواجد في مكان واحد (مثلاً دولتين) بالتالي من الصعب تنظيم الاجتماع بسبب مشكلة فرق الوقت مثلاً.

⑧ يوجد اجتماع يومي للفريق (stand-up meeting) لمتابعة حالة المشروع وأية تطلعات على شاكلة كل شخص إن وجدت

⑨ Scrum frame work sw + (XP, Isanban, Rup)

⇒ one of the most popular ways to implemintaion incremental.

⇒ Process of scrum framework



User story Backlog:

main fetchers of Project and it will be written depend on Priority and complexity

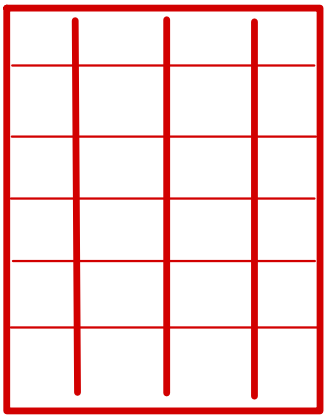
first sprint take 2weeks and there is meeting every day

in Scrum there is 5 days work in the week and 2 days off

كما ذكرنا سابقاً بعد مجموعة الـ sprints يمكن تسليم release جاهز للـ customer حيث يبدأ العمل به

There is a point for every user story that help to calculate speed of the team which is help to determine the time needed to end the Project after 2 3 sprints in more accurate.

يتم توزيع المهام ويختار كل شخص المهمة التي يريدها حسب قواعد Scrum وبعد وقت يخبرها وإذا كانت كبيرة إلى أكثر من ٤ ساعات نقسمها إلى أقسام لأنه حسب Scrum فإن الشخص لا يستطيع العمل لأكثر من ٤ ساعات يومياً العمل المتعهد به إلا إذا وليس عدد ساعات الدوام



* availability team schedule

There is a schedule for team members and the hours of work for every team member, and also total hours of work for whole team, when the member work automatically the time in the schedule decreased for this member

(12) There is many tools helped to organized all things and steps during the project, one of the most popular tool is Jira

=> more about points : as we said every user story measured by number of points depend on how much this user story was difficult after that we calculate the expected time we needed to finish every user story in specific way we will study it later on.

(13) Scrum Master : lead the process during work and find the solution for the problems wich facing the Team.

[3] Reuse-Oriented Software Engineering Model

=> تعتمدها غالباً الشركات التي تكون مخصصة لنوع معين من البرامج مثل شركات متخصصة في برامج المحلات التجارية

Process: divide projects to components wich i can reused this components in another projects that have the same type.

بما معناه يتم تقسيم البروجيكت إلى أجزاء هذه الأجزاء يتم إعادة استخدامها في بروجيكتات أخرى من نفس النوع (في المثال أعلاه ستشترك المحلات التجارية بنسبة كبيرة من الأمور المتشابهة مما يعطي فرصة لإعادة استخدام هذه الأجزاء من البروجيكتات السابقة.

Reuse-Oriented Software Engineering

- In the majority of software projects, there is some software reuse.

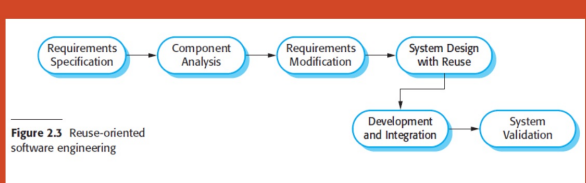


Figure 2.3 Reuse-oriented software engineering

* يتم تشبيك هذه الأجزاء مع بعض بعد الإطلاع على Requirements وعمل testing للنظام

① ليس شرطاً أن تكون الشركة هي من قامت ببناء جميع هذه الأجزاء من الممكن أن تكون بعض هذه الأجزاء قد تم شراؤها جاهزة.

هنا في أكثر من نوع يتم تمثيل الـ components فيها

Types of Reuse-Oriented Software Engineering

There are three types of software component that may be used in a reuse-oriented process:

1. Web services that are developed according to service standards and which are available for remote invocation.
2. Collections of objects that are developed as a package to be integrated with a component framework such as .NET or J2EE.
3. Stand-alone software systems that are configured for use in a particular environment.

Prototyping

- Method that used in all software process (it's not software process)
- Make a Sketches of Ui Ux to show to the customer how the system will do.
- one of the most important methods that used in software engineering

الزبون يمكن أن يكون من أي مجال محاماة أو تدريس أو أي شيء آخر بالتالي سيكون من الصعب عليه تخيل طريقة عمل البرنامج ، لذلك الـ Prototyping مهم لتوضيح طريقة عمله من خلال بناء سكتشات لهذا السوفتوير

من خلال ال Prototyping نقوم بعمل فحص دقيق للـ Requirements الي تم كتابتها.

ملاحظة: ال Prototyping ليس فقط عمل واجهات شكلي، وإنما يمكن عملها بحيث تكون dynamic (شغالة ويمكن تجربتها) ليست مجرد شكل شاشات.

أمثلة على برامج لعمل ذلك

1. Microsoft visio
2. Adobe XD

يتم كتابة ال Requirements ثم عمل الPrototyping ويمكن عمل أكثر من واحد وعرض جميعها على ال customer ليختار الأنسب

أحد فوائد ال Prototyping أنها تعطي مجال لتجربة ال Usability للبرنامج قبل بناؤه

4 Spiral Model of Bohem

=> Similar to waterfall for very high risk Projects

=> Used for Complex and expensive Projects which is no guarantee to complete this project successfully

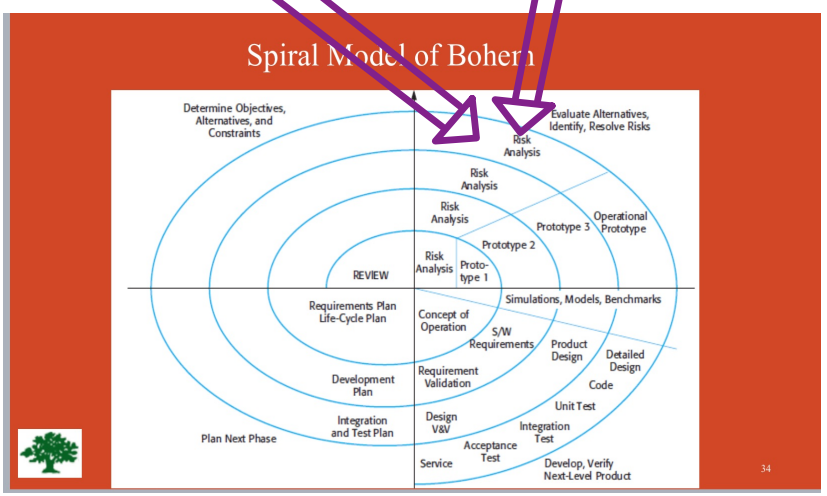
يعني بنستخدم هذا ال Model لما يكون عنا مشروع صعب ومكلف وفي خطورة فيه (risky) بحيث إنه احتمال كبير انو يفشل، فبنستخدم هاي الطريقة عشان باللحظة اللي بنتأكد انو مش حنقدر نعمل المشروع نوقف مباشرة؛ لنتجنب المزيد من الخسائر.

Spiral Model of Bohem

- A **risk-driven** software process framework
- Each loop in the spiral represents a phase of the software process.
- Thus, the innermost loop might be concerned with system **feasibility**, the next loop with **requirements** definition, the next loop with system **design**, and so on.
- includes explicit **risk management** activities
- to reduce these risks.

* كل loop بمثل مرحلة من مراحل عمل البروجيكت

* بعد كل مرحلة يتم إنجازها يجب عمل risk analasiys + حل هذه الrisks قبل الدخول فالمرحلة التالية



* بعض بعد عمل فحص لهذه المخاطر يجب عمل Prototype للتأكد من حلها قبل الذهاب للمرحلة التالية.

* إذا تم حل هذه المخاطر نكمل ونذهب للمرحلة التالية وهي الحالة إذا لم يتم حل هذه المخاطر يتم إيقاف المشروع مباشرة.

Case study about Software process

Software Engineering COMP433
Tutorial -3- Software Process Models

1. Which software process model (or models) you would apply for the following scenarios. Justify your Answer.

a) To develop a secure ATM sub-system to integrate with an existing banking system. The developed ATM sub-system will be deployed across a 1000 ATM machines. It should have an availability rate of 99%, accurate. It should also have a 99.9% accuracy money notes counting dispenser, and three-level security that requires a card, a pin code and a biometric code. WF

Waterfall + formal ⇒

b) To develop a mobile app to teach children ages 7-10 the basics of Arabic language. The app teaches the basics of Arabic grammar and basic words through well-defined set of exercises. The features of the app are well defined, clear, and understood by the development team. However, the app should be designed carefully to be suitable to children users. WF

Waterfall ⇒ because Req is clear + Prototype

c) To develop university student management/registration system that can support 75000 students, and up-to 15000 concurrent students' access, would not need more than 1 hour (student/user) training and need to be delivered in 2 years for operational use. AGILE

Agile ⇒

d) To develop a mobile app, that monitors health indicators (e.g. blood pressure, sugar level, pulse) of patients, by collecting readings through special medical sensors, and then provides medical advice based on the collected readings by an external medical decision system, which your system must be connected to it.

Waterfall ⇒

e) To develop a word processing application, that uses existing print, graphic, font styles, spelling check, and grammar check components. The application must be designed to be used by people with dyslexic/learning difficulties. Reuse

Reused ⇒

2. If you were a project manger responsible to develop a system for a product-line ordering system for a manufacturer of car parts. The system, should allow telephone and online ordering of car parts, and has a dedicated team to process the orders. The system should keep inventory of existing stock and be accessed by the manufacturer's product-line to manufacture parts according to sales. The users of the system are online users, who should register an account online and store users' information including their credit card details to enable them ordering online within a secure login sub-system. To enable users order online, the system will be required to connect to the respective credit card bank to authorize payment. Other system users also include salesmen who can place orders through telephone calls, and process payment through the system, and inventory users who manage availability of car parts and system administrators who manage the system database. The system should allow 10 concurrent salesmen and 10 inventory men to use the system.

How would you project manage the above scenario?- particularly which software process model (or models) would you apply and use? If you are applying more than one model, indicate for what parts/components of the project and at what stage of its development life cycle.

The whole system is agile + Prototype, but the Payment part is waterfall + formal

④ Process activities

تنضم بشكل أكبر الـ formal Models
مثل: Sperial, waterfall

II Software specification (Requirements engineering)

Process Activities: Software Specifications

- Software specification or requirements engineering is the process of understanding and defining:
  - what services are required from the system
  - and identifying the constraints on the system's operation and development

الخدمات التي سيقدمها → non Functional Requirements

- Requirements are usually presented at two levels of detail.
  - End-users and customers need a high-level statement of the requirements;
  - system developers need a more detailed system specification.

كتابتها بشكل عام (User Req)
كتابة التفاصيل (System Req)

الـ System Req هي تفصيل كامل لكل الـ User Req

مثال: يجب أن يكون هناك خانة للبحث عن كتاب في مشروع المكتبة (User Req إذا عام)

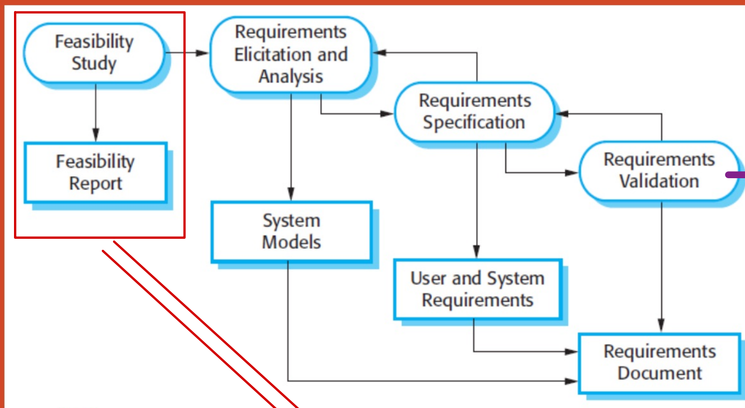
بينما يجب البحث عنه عن طريق عنوانه وراعه التسلسلي

(هنا تفصيل لطريقة البحث عنه، بما معناه تفصيل لـ User Req إذا هي System Req)

مثال آخر عن البحث عن كتب يجب أن تظهر مرتبة بناءً على سنة النشر

System Req

Process Activities: Software Specifications, 2



(Ph.D)

36

* فالبداية يتم النظر إلى المشروع وتحديد كل ما يحتاجه من وقت وجهد
ثم عمل دراسة لما يمكن عمله وتقديم عرض يشمل الأفكار، الأسعار، وغيرها (Phase1)
ثم بعد ذلك يتم جمع ال Requirements

Requirements Validation : التأكد من أن جميع ال Riquirements صحيحة بعد الانتهاء من جمعها.

* من ضمن ال Requirements Validation هي عمل اجتماعات مع فريق العمل الموكل في عمل هذا المشروع ومراجعة ال Requirements المكتوبة

باختصار: يوجد 3 مراحل لل Requirements
(1) جمعها من العميل (2) كتابتها وتفصيلها (3) التحقق (Validition) ... ال customer في بعض الأمور

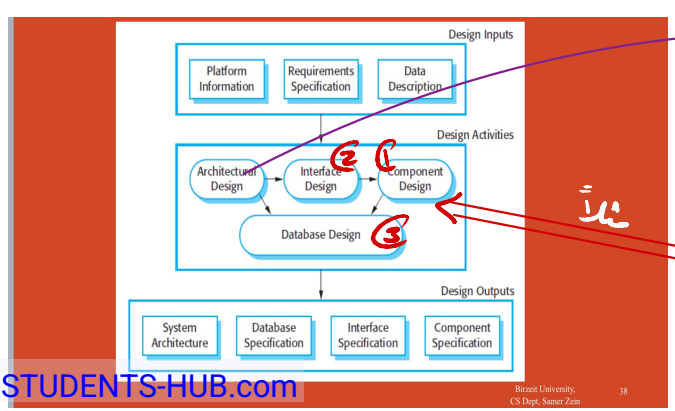
2] Software Design & Implementation

Process Activities: Software Design & Implementation

- A software design is a description of:
  - the structure of the software to be implemented,
  - the data models and structures used by the system,
  - the interfaces between system components
  - and, sometimes, the algorithms used.
- Creating detailed designs for critical systems.
- Generating code from designs and diagrams

مرحلة ② بناء models وتصف كيف سيكون الكود قبل كتابته
الـ design تتم.

⇐ تسبق عملية الكتابة والـ Validation للكود
⇐ تعد أصعب مرحلة في الـ Software ويقوم بها الـ Seniors ومن لهم خبرة طويلة فالعجال. Process



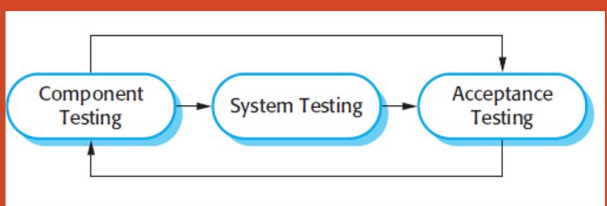
مثلاً

أول نقطة يتم بناؤها هي الـ Architecular وهي تقسيم الـ System إلى components لتسهيل بناؤه وتوزيع المهام على الـ team وتفصيل كل component عما بداخله وطريقة التواصل عليه بشكل كامل

Process Activities: Validation

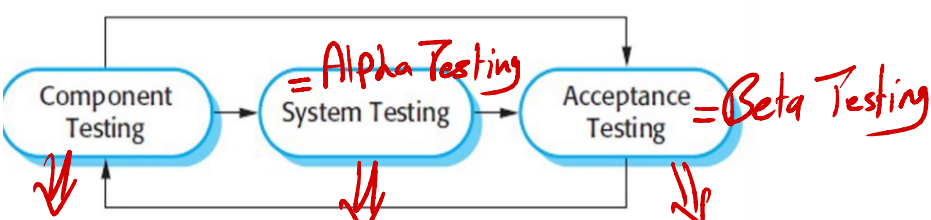
- intended to show that a system both conforms to its **specification** and that it meets the **expectations** of the system customer.
- Program testing, where the system is executed using simulated test data, is the principal validation technique.
- **Alpha** Testing VS **Beta** Testing

عليه انأكد من أن كل شيء يعمل بشكل صحيح ، و ن bugs (Testing)



Unit Testing : testing during write code

* مهم وضروري ويعمل عليه ال testing فيما بعد



= Alpha Testing

= Beta Testing

Testing after ending specific component

Testing after ending all components together

Testing the software with Customer

End of ch.2