***University of mustansiriyah /College of Education***

***Computer Science Department***

***Software Engineering 3rd Class***

***Chapter 2: Software Development Life cycle (دورة حياة اعداد المنظومة)***

Topics:

2.1 Software Lifecycle

2.2 Software Development

2.3 Software Process Models

**2.1 Software Lifecycle:**

Each software product proceeds to a number of distinct stages, these are:

• Requirements engineering

• Software design

• Software construction

• Validation and verification

• Software testing

• Software deployment

• Software maintenance

The software process used for the development of the software Product, these stages may occur in different orders, or frequency.

**2.1.1 Requirements Engineering** (requirement analysis and definition by using engineering approach)

Requirements engineering is the interface between customers and developers on a software project. Requirements should make explicit the ideas of the customer about the prospective system.

**2.1.2 Software Design**

The designers converts the logical software requirements from stage 1 into a technical software design by describe the software in such a way that programmers can write line of code that implement what the requirements specify.

**2.1.3 Software Construction**

Software construction is concerned with implementing the software design by means of programs in one or more programming languages and setting up a build management system for compiling and linking the programs.

**This stage content several steps, these are:**

**a. Software reuse**

1. Component based software engineering

2. Software product lines

**b. Security and reliability**

**c. Software documentation**

**d. Coding standards**

**a. Software Reuse**

The goal of software engineering is to achieve many features with little effort and few defects. Software reuse is believed to play an important role in achieving this goal by encapsulating effort in units of source code, which can be reused in other projects.

However, the effort needed to make something reusable may not be worth it, if it is only reused few times, or needs extensive adaptation for each reuse.

**a.1. Component Based Software Engineering**

Building software systems from prefab software components is an old dream of

software engineering.

**a.2. Software Product Lines**

Software systems are often part of a family of similar systems. The goal of a software product line is to maintain a set of reusable core artifacts that are common to all systems in the product line. Thus, code for a specific product can focus on the specifics of that product, reusing the common functionality.

**b. Security and Reliability**

Software must be dependable by making it reliable ( software should work very well under any environments ), secure and safety (by verifying from user authentication to using any system).

**c. Software Documentation**

• User documentation?

• Technical documentation?

• Documentation generation?

**d. Coding Standards**

Coding standards are important to ensure portability and make code maintainable by others than the original developer.

**2.1.4 Validation and Verification**

• Software inspection

• Software testing

**2.1.4.A) Software Inspection**

Software inspections are reviews of the code with the purpose of detecting

defects. In an inspection someone other than the programmer reads a program unit of limited size to determine whether it satisfies the requirements and specification. A formal process and checklist are used to ensure that no aspects are forgotten.

**2.1.4.B) Software Testing**

Testing each unit founded in this software, follow by testing software integration.

**2.1.5 Software Deployment**

After development, software should be put to use. That is, it should be released

and made available to users, who can then download, install, and activate it. These activities are captured under the common term *software deployment*. Richard S. Hall in the 'Software Deployment Information Clearinghouse' defines software deployment as follows: "The term software deployment refers to all of the activities that occur after a software system has been developed and made available for release. As such, software deployment includes activities such as packaging, releasing, installing, configuring, updating, and uninstalling a software system." and "Software deployment is the assembly and maintenance of the resources necessary to use a version of a system at a

particular site".

**The following deployment activities make up the software deployment process:**

\_ Release

\_ Packaging

\_ Transfer

\_ Installation

\_ Configuration

\_ Activation

\_ De-activation

\_ Update

\_ Adapt

\_ De-installation

­\_De-release

These activities are not necessarily performed sequentially. Many phases of the

deployment process are often performed manually. For example, downloading, building and installing a source distribution of a software package, requires a number of commands to be formulated and executed. Each such command requires knowledge of some sort about the activity.

**Manual deployment does not scale when deploying:**

\_ many applications

\_ applications composed from separately deployed components

\_ on multiple machines

\_ on different types of machines

**2.1.6 Software Maintenance**

As software evolves after its first release, software maintenance is needed to improve it, i.e., repair defects, and to extend it, i.e., add new functionality.

**2.2 Software development**

**Three phases to develop the software**

1- Definition

2- Design

3- Maintenance

**1- Definition**

1- What information to be processed

2- What design constrains exist.

3- What function and performance desired.

4- What interfaces are desired.

5- What validation criteria are required?

6- What is modeling?

**2- Design**

1- How data structures to be designed.

2- How procedural details to be implemented.

3- How design to be translated into language.

4- How testing is performed.

**3- Maintenance**

1- Error

2- Adaptation.

3- Modification

2.1 Software Development Life cycle

منذ بداية السبعينات بدا إعداد المنظومات يتم بطريقة صحيحة وفي شكل مشروع أسوة بالمشاريع الهندسية الاخرى ،وهذا المشروع يتكون من مجموعة مراحل عرفت باسم دورة حياة إعداد المنظومات (Software Development Lifecycle).

ودورة حياة إعداد اي منتج تمر بعدد من المراحل المتميزة، وهذه هى;

* (المتطلبات الهندسية ) Requirements engineering
* (تصميم المنظومة )Software design
* (بناء المنظومة ) Software construction
* (المصادقة والتحقيق ) Validation and verification
* (اختبار المنظومة)testing ( Software
* (نشر او توظيف المنظومة Software depl0yment (i
* (صيانة المنظومة )Software maintenance

2.1.1 **Requirements** Engineering (requirement analysis and definition by

:(using engineering approach

ان الهدف الاساسي من مرحلة التحليل هو فهم متطلبات المستخدم (user requirement) وتنظيم وتمثيل modeling هذه المتطلبات في شكل مخططات وجداول لتتم الموافقة عليها مع الزبون واعتبارها اساسا للمرحلة اللاحقة ألا وهي مرحلة التصميم .

-وفي مرحلة التحليل يتم ايضا جمع الحقائق حول النظام الحالى وفهم المشاكل التي تعترض سير العمل، وتستخدم عدة طرق للوصول لهذا الفهم مثل (المقابلة الشخصية والاستبيان والملاحظة والعرض التجريبي وغيرها) ومن ثم المتطلبات الوظيفية (Functional requirement) وغير وظيفية (non-functional requirement) للمنظومة المقترح تنفيذها.

ويمكن إعداد المنظومة باستخدام أحد المنهجيات الاتية

- المنهجية الهيكلية (Structured Methodology) : وتقسم هذه المنهجية المنظومة الى آجزاء وظيفية في شكل هرمي وتستخدم أدوات هيكيلة في التحليل والتصميم والبرمجة مثل مخطط انسياب البيانات DFD والمخطط الهيكلي Structure chart.

* المنهجية الشينية (Object Oriented Methodology) : وتقسم هذه المنهجية النظام الى كائنات لها وظائف وخصائص شبيهة بالكائنات التي نتعامل معها في حياتنا اليومية .وتستخدم ادوات شينية مثل مخطط الحالة UCD ومخطط الفصيلة class diagram.
* منهجية فصانل المكتبة **(Class** **library**) : حيث يستطيع من خلالها معد المنظومة تجهيز منظومة تجريبية او إعداد منظومة متكاملة في خلال فترة وجيزة (ساعات أو أيام ).وتحتاج بالطبع الى التدريب عليها ،وتحتوي على الادوات الضرورية لتحليل وتصميم وتنفيذ المنظومات **مشتملة** ايضا على كيفية تصميم واجهة المستخدم .ومن **امثلة** هذه الفصانل **ما** يسمى الفصانل الاساسية لميكروسوفت (Microsoft Foundation)

Class MFC : وهي عبارة عن **مكتبة** فصانل وتسمى ايضا (Application Frameworks).

وتتلخص أهم نشاطات التحليل فيما يلى :

* اكتشاف المجهول من قبل محلل النظم .لهذا وجب علية في البداية الاستماع للزبون والمستخدم وملاحظة مجريات الامور في النظام الحالي.
* الاستقصاء والتنقيح والفهم لمشاكل النظام الحالي وتحديد المتطلبات للنظام الحالي
* تمثيل ما فهمه بالمخططات وشاشات الحواروالتثبيت من البيانات المؤخدة بعناية
* بعد فهم المسالة تأتي كتابة وثيقة مواصفات المتطلبات لتصبح على شكل عقد او اتفاقية بين المحلل ) والزبون للأشياء المطلوب تنفيذها فى المنظومة المقترحة

**2.1.2 Software Design**

مرحلة التصميم : تأتي هذه المرحلة بعد تحديد المتطلبات من قبل الزبون ،وإعداد مواصفات دقيقة لهذه المتطلبات .حيث يتم في هذه المرحلة (مرحلة التصميم ) تمثيل الحل في شكل خرائط وجداول ومخططات تفصيلية نهانية (blueprint) للمنظومة المقترحة التي تلبي المتطلبات المدونة في وثيقة المتطلبات في المرحلة السابقة لها وتستخدم هذه المرحلة ادوات تماثل وتشبة الخرانط الانسانية والمعمارية التي تسبق عملية البناء وفي هذه المرحلة يتم

* تحديد المدخلات
* تحديد المخرجات
* تحديد الواجهات interface مع المنظومة التي تشتمل علي
* واجهة software
* **واجهة** hardware
* واجهة المستخدم

ثلاثة مراحللتطوير البرمجيات **(Software** **Development**)

1. التعريف **(Definition)**
2. التصميم**(Design)**
3. الصيانة**(Maintenance)**
4. التعريف**(Definition)**

ا\_ما هي المعلومات التي سيتم تجهيزها(معالجتها)

2\_ ما هي المعوقات الموجودة في التصميم .

3\_ ما وظيفة وأداء المطلوب ( المتطلبات)

٠ 4\_ ما هي الواجهات المطلوبة

. 5\_ ما هي المعايير (الشروط) المطلوبة للتحقق من الصحة

6- ما هي النماذج؟

1. التصميم **(Design)**
2. كيف تصمم هياكل البيانات
3. كيف تنفذ تفاصيل الإجرائية
4. كيفية تصميم الترجمة إلى اللغة.(كيف يترجم التصميم الى لغة).
5. كيف يتم تنفيذ الاختبار.
6. الصيانة**(Maintenance)**
7. الخطا(الاخطاء**)(Error** )
8. التكيف(**Adaptation**)
9. التعديل (**Adaptation)**

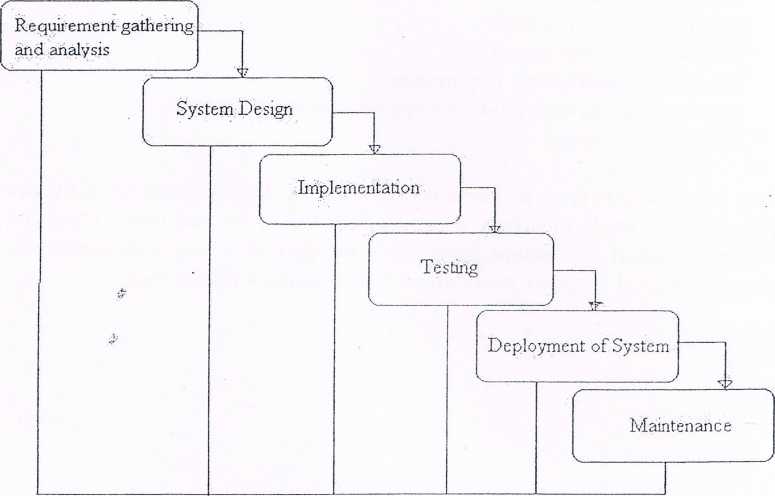
**2.3 Software Development Models**

**1-Waterfall Model**

The Waterfall Model was first Process Model to be introduced. I't is also referred to as a **linear sequential life cycle model.** It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of model is basically used for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model the testing starts only after the development is complete. In **waterfall model phases** do not overlap.

**Diagram of Waterfall-model:**

Generid Overview of "Waterfall Model'"



**Advantages of waterfall model:**

* This model is simple and easy to understand and use.
* It is easy to manage due to' the rigidity of the model - each phase has specific deliverables and a review process.
* In this model 'phases are processed and completed one at a time. Phases do not overlap.

. Waterfall model works well for smaller projects where requirements are very well understood.

**Disadvantages of waterfall model:**

* Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.,

. No working software is produced until late during the life cycle.

* High amounts of risk and uncertainty.
* Not a good model for complex and object-oriented projects.
* Poor model for long and ongoing projects.
* Not suitable for the projects where requirements are at a moderate to high risk of changing.

**When to use the waterfall model:**

* This model is used only when the requirements are very well known, clear and fixed.
* Product definition is stable.
* Technology is understood.
* There are no ambiguous requirements
* Ample -resources with required expertise are available freely
* The project is short.

Very less customer enter action is involved during the development of the product. Once the product is ready then only it can be demoted to the end users. Once the product is developed and if any failure occurs then the cost of fixing such issues are very high, because we need to update**٠**everywhere from document till the logic.

النمونجالتدفقي **Waterfall model**

يسمى النموذج الشلال او التدفقي waterfall model (وهو اول نموذج تم استخدامه بداية هن السبعينيات) ايفا يعرف بالنموذج التسلسلي Sequential model نظرا لتتابع المراحل وكان طوق النجاة في نهاية الستينيات وله الفضل في حل الكثير من المشاكل التي تمت مواجهتها عند اعداد المنظومات وساهم الى حد ما في حل مشكلة ازمة البرمجيات software crisis حيث كانت المنظومات يتم اعدادها بدون توثيق وتستخدم ادوات بدانية في التحليل والتصميم

وكان المخطط الانسيابي flowchart الاداة الوحيدة المستخدمة وبذلك كانت الجودة متدنية والاخطاء كثيرة والزبون في اغلب الاحيان غير راض عن المنظومة النهائية .

(متى يجباستخدامنموذج الشلال**) When to use the waterfall model**

يشترط عند استخدام هذا النمونج :

* عندما تكون المتطلبات معروفة جيدا، وواضحة وثابتة وكاملة.
* تعريف المنتج مستقر.
* التكنولوجيا مفهومة.
* لا توجد متطلبات غامضة
* تتوفر موارد (المصادر وغيرها) متوفرة و ذوي الخبرة المطلوبة متوفرة
* يجب ان يكون المشروع قصير.

وقد تم استخدام هذا النموذج كثيرا في السابق اعداد المنظومات وهو مقتبس من الطريقة الهندسية  **في التصنيع والانتاج**  والبناء من حيت عملية المراحل واسنخدام منهو م المشروع

**اما المراحل التى** تخص **النموذج. التدفقي فهي :**

1. مرحلة التخطيط Planning phase
2. مرحلة التحليل Analysis phase
3. مرحلة التصميم Design phase
4. موحلة التنفيذ Implantation phase
5. مرحلة الاختبار Testing phase'
6. مرحلة الصيانة Maintenance phase

**Advantages of waterfall model: (**المزايا**)**

* هذا النمونج هو بسيط وسهل الفهم والاستعمال.
* فمن السهل إلادارة نظرا لصلابة هذا النموذج — لان كل مرحلة لها مخرجات محددة وعملية تنقيح (عملية مراجعة)
* في هذا النموذح المراحل **يتم** معالجتها والانتهاء منها **في** وقت واحد. اي ان المراحل لايحدث فيها عملية تداخل (**overlap**).
* نمونج الشلال يعمل بشكل جيد للمشاريع الصغيرة عندما يكون هناك فهم جيد ودقيق للمتطلبات

**Disadvantages of waterfall model;** عيوب نموذج التدفقي )**)**

* بمجرد التطبيق في مرحلة الاختبار، فإنه من الصعب جدا الرجوع وتغيير شيء ما لم يكن مدروسة جيدا في مرحلة الفهم (مرحلة التحليل وجمع المتطلبات).
* لايتم إنتاج أي نسخة من المنظومة الا في مرحلة متاخرة من دورة حياة اعداد المنظومة
* في هذا النموذج يوجد الكثير من المخاطر وعدم اليقين (الشك).
* ليس نموذجا جيدا للمشاريع المعقده' ومشاريع المنهجية الشينية (**object-oriented** **projects**).
* نموذج ضعيف غير مناسب للمشاريع الطويلة والمستمرة.
* لا تصلح للمشاريع حيث المتطلبات معتدلة (محدودة المدى- متوسطة) إلى عالية و المخاطر متغيرة.

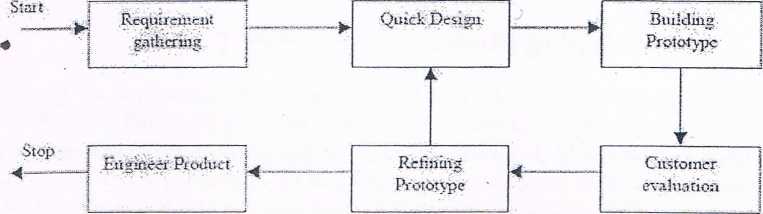
ان هذا النموذج Waterfall Model وكما اشرنا سابقا متداول في اغلب المشاريع **منذ**  بداية السبعينيات اما الان فاصبح هذا النمونح لا يواكب تعقد وتنوع المنظومات خصوصا بعد ظهور الشبكات والانترنت والوسانط المتعددة ونظم قواعد البيانات

والاهم من ذلك لم يحل مشكلة المتطلبات التى في اغلب الاحيان لا تكون دقيقة ولا كاملة . لهذا تم تعديل هذا النموذج بحيث يمكن الرجوع الى المراحل السابقة في حالة تعديل او حدوث تغيير في المتطلبات . وهذا النموذج المعدل هو ايضا لم يحقق الغرض المطلوب .لهذا اصبحت الحاجة لاستخدام نماذج اكثر تطورا ضرورة حتمية ،وسار ع المختصون بهنسة البرمجيات الى البحث عن حل بديل وذلك باعداد نماذج اخرى اكثر تطورا وتراعي مشكلة عدم اكتمال ودقة المتطلبات .

**2- Prototype model:**

The basic idea here is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. This prototype is developed based on the, currently known requirements. By using this prototype, the client can get an “actual feel” of the system, since the interactions with prototype can enable the client to better understand the requirements of, the desired system. Prototyping is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements. The prototype is usually not complete systems and many of the details are not built in the prototype. The goal is to provide a system with overall functionality.

Diagram of Prototype model:



**Prototyping Model**

**Advantages of Prototype model:**

* Users are actively' involved in the development.
* Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.
* Errors can be detected much earlier.
* Quicker user' feedback is available leading to better solutions.
* Missing functionality can be identified easily**.**
* "Confusing or difficult functions can be identified Requirements validation. Quick implementation of, incomplete, but functional,-application**.**

**Disadvantages of Prototype model:**

* Leads to implementing and then repairing way of building systems.
* Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
* Incomplete application may cause application not to be used as the full system was designed Incomplete or inadequate problem analysis.

**When to use Prototype model:**

* Prototype model should be used when the desired system needs to have a lot of interaction with the end users.
* Typically, online systems, web interfaces have a very high amount of interaction with end users, are best suited for Prototype model. It might take a while for a system to be built that allows ease of use and needs minimal training for the end user.
* Prototyping ensures that the end users constantly work with the system and provide a feedback which is incorporated in the prototype to result in a useable system. They are excellent for designing good human computer interface systems.

**نموذج العرض التجريبي Prototyping Model**

جاء نموذج العرض التجريبي Prototyping Model كمحاولة لحل قصور (عدم نجاح) النموذج التدفقي في حل مشكلة المتطلبات الخاصة بالزبون ٠ ويقوم معد المنظومة باعداده ثم بعد ذلك يعوض منظومة تجريبية (نموذح تجريبي pilot system) كعينة sample على الزبون (والمستخدم) يمثل او يشابه المنظومة النهانية المطلوبة ولكن بصورة اصغر (اي صورة مبدئية) ومدة قصيرة في الاعداد. وتستخدم هذه العينة عادة اما لتوضيح المتطلبات او كبداية لاصدار عدة نسخ متتالية للوصول للمنظومة النهائية التي ستسلم للزبون فيما بعد . وفي هذا النموذج يتم عرض وظانف محددة من المنظومة على الزبون في اقرب وقت ممكن وان كان ذلك، في مستوى منخفض من الجودة ، والفكرة هنا هي جعل الزبون يرى عينة من المنظومات النهائية في مرحلة مبكرة

متىنستحدمالعرضالتجريبي **When to use Prototype model:**

نستخدم العرض التجريبي Prototype للاسباب الاتية ;

1 - هذا النموذج يستخدم عندما يحتاج النظام المرغوب الكثير من التفاعل مع المستخدمين النهائيين

2- عادة، انظمة الانترنت (online systems, web interfaces) تحتوي على تفاعل عاليء مع المستخدمين النهانيين، فهذا النموذج مناسب لها .

3\_ لتوضيح وتحديد وتنقيح المتطلبات ٠

4- اقناع الزبون بالجدوى الفنية عندما تكون المخاطرة الفنية عالية

5\_ محاولة فهم المسالة المراد حلها قبل الشروع في رصد الموارد والمستلزمات ٠

6\_ نسخة بدائية للتوصل الى النسخة النهائية او ما يسمى بالاصدار (المنظومة النهائية).

والميزة هنا ان الزبون يعطى فرصة لتغيير متطلباته في مرحلة مبكرة من المشروع حيث تعتبر تكلفة التغيير قليلة نسبيا. وتستخدم طريقة العرض المبدئي كحل لأهم **مشكلة** في النمونج التدفقي (Waterfall model) وهي **مشكلة** عدم اكتمال المتطلبات .

**Disadvantages of Prototype model:**

**عيوب العرض التجريبى :**

1- يؤدي إلى تنفيذ ومن ثم إصلاح طريق أنظمة البناء.

2\_ عمليا،هذا المنهجية قد يزيد من تعقيد النظام ونطاق النظام قد يتوسع خارج الخطط الاصلية

3- طلب غير مكتمل قد يتسبب بتطبيقات لا تستخدام مثلا تصميم نظام كامل على اساس تحليل غير كافي وغير كامل للمشكلة.

4- احيانا يكون من الصعب اقناع الزبون بأن المنتوج المبدنى هو للعرض فقط (نظام تجريبي) وقد يطلب الاسراع في استلام المنتوح ونتيجة لنلك يستلم منظومة غير مكتملة الجودة لتبدا مرحلة الصيانة مبكرا.

5- لاعداد منظومة تجريبية سريعة قد يستخدم المبرمج خوارزميات قليلة الكفاءة ويستخدم ادوات بسيطة في التحليل والتصميم والبرمجة وهذه الادوات عادة ما تصبح هي الادوات الاساسية في اعداد المنتوج النهائي مما قد يؤدي الى اعداد منتوج قليل الجودة.

6\_- تخلق احساسا خادعا من التفاؤل (تفاؤل ان المنظومة تم الانتهاء من اعدادها).

7- قلة استخدام التوثيق.

**3- Incremental model**

In incremental model the whole requirement is divided into various builds. Multiple development cycles take place here, making the life cycle a “multi-waterfall” cycle. Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first module, so you have working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved:'

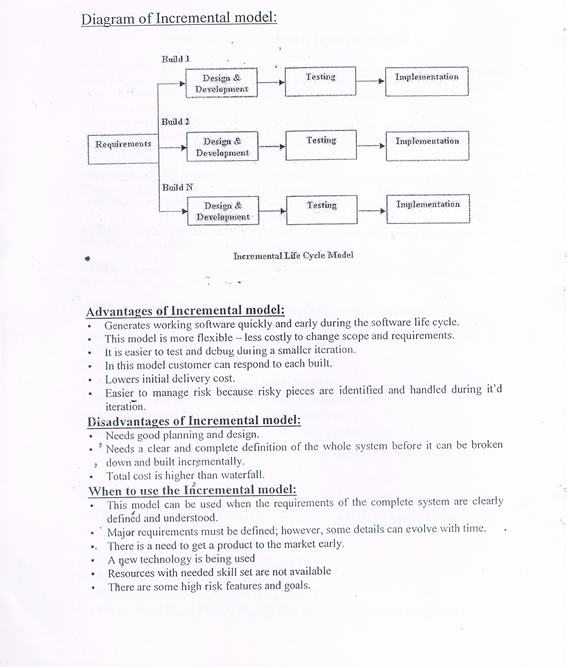
****

Diagram of Incremental model:

**Advantages of Incremental model:**

* Generates working software quickly and early during the software life cycle.
* This model is more ﬂexible — less costly to change scope and requirements.
* It is easier to test and debug during a smaller iteration.
* In this model customer can respond to each built.
* Lowers initial delivery cost.
* Easier\_ to manage risk because risky pieces are identiﬁed and handled during it’d iteration.

**Disadvantages of Incremental model:**

* Needs good planning and design.
* Needs a clear and complete deﬁnition of the whole system before it can be broken down and built incrementally.
* Total cost is higher than waterfall.

**When to use the incremental model:**

* This model can be used when the requirements of the complete system are clearly deﬁned and understood.
* Major requirements must be deﬁned; however, some details can evolve with time.
* There is a need to get a product to the market early.
* A new technology is being used**.**
* Resources with needed skill set are not available**.**
* There are some high risk features and goals.

خصائص النموذج التزايدي

* **يتكون المنتوج عادة من عدة اجزاء او بناء.**
* **كل جزء (بناء) يتم تصميمه وكتابته واختباره ثم تسليمه للزبون.**
* **يتكون البناء من قطع برمجية تتفاعل لتعطي وظيفة معينة .**
* **يعطي المنتوج للزبون تدريجيا (على دفعات ) جزءا بعد الاخر .**
* **بخلاف نموذج العرض المبدئي يعتبر كل جزء في هذا النموذج منتوجا (اصدار release) قابلا للعمل في موقع الزبون.**

**Disadvantages of Incremental model:** عيوبالنموذج التزايدي

* الاحتياج الى التخطيط الجيد والتصميم.
* يحتاج الى تعريف واضح وكامل للنظام كله قبل أن'يتم التقسيم والبناء بشكل متزايد.
* التكلفة الإجمالية أعلى من نموذج الشلال

**When to use the Incremental model: متى يستخدم نمونج التقديم التزايد ي**

* يمكن استخدام هذا النموذح عندما متطلبات نظام كامل تكون واضحة التعريف ومفهومه
* يجب تحديد المتطلبات الرئيسية؛ ومع ذلك، يمكن لبعض التفاصيل تتطور مع مرور الوقت
* وهناك حاجة للحصول على المنتج إلى السوق (انزاله الى السوق) في وقت مبكر.
* تكنولوجيا جديدة بدات تستخدم
* الموارد مع مجموعة المهارات اللازمة غير متوفرة
* هناك بعض الميزات عالية المخاطر والأهداف.

**4- Spiral model:**

The spiral model is similar to the **incremental model,** with more emphasis placed on risk analysis. The spiral model has four phases: **Planning, Risk Analysis, Engineering and Evaluation؛** A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spiral, starting in the planning phase, requirements is gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral.

**Planning Phase:** Requirements are gathered during the planning phase. Requirements like **‘BRS’** that is ‘Business Requirement Specifications' and **‘SRS’** that, is ‘System Requirement specifications’.

Risk **Analysis:** In the rjsk analysis phase, a process is undertaken to identify risk and alternate solutions. A prototype is produced at the end of the risk analysis phase. If any risk is found during The risk analysis then alternate solutions are suggested and implemented.

**Engineering Phase: In** this phase software is **developed,** along with testing at tlie end of the phase. Hence in **this** phase tire **development and testing** is done.

**Evaluation phase:** This phase allows tire customer to evaluate tire output of the project to date before the project continues to tire next spiral.

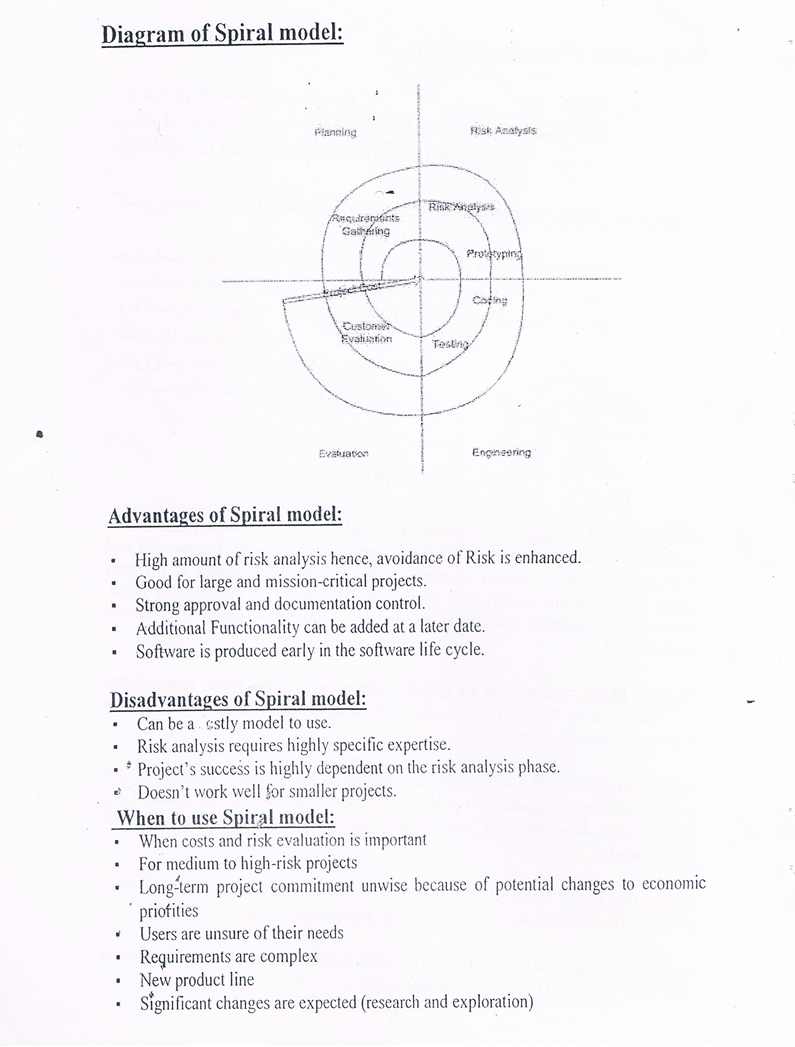
****

Diagram of Spiral model:

**Advantages of Spiral model:**

-

High amount of risk analysis hence, avoidance of Risk is enhanced.

•

Good for large and mission -critical projects.

•

Strong approval and documentation control.

•

Additional Functionality can be added at a later date.

•

Software is produced early in the software life cycle.

**Disadvantages of Spiral model:**

-

Can be a

costly model to use.

•

Risk analysis requires highly specific expertise.

•

Project's success is highly dependent on the risk analysis phase.

**When to use Spirul model:**

•

When costs and risk evaluation is important

-

For medium to high -risk projects

•

Long-term project commitment unwise because of potential changes to economic

priorities

•

Users are unsure of their needs

•

Re9uirements are complex

•

New product line

•

Significant changes are expected (research and exploration)

* Doesn't work well for smaller projects.

**4- Spiral Model (النموذج الحلزوني او اللولبي)**

يشبه النموذج الحلزوني النموذج التدريجي، مع التركيز بشكل أكبر على تحليل المخاطر. يتكون النموذج الحلزوني من أربع مراحل: التخطيط وتحليل المخاطر والهندسة والتقييم؛ ومشروع البرمجيات يمر مرارا عبر هذه المراحل في التكرارات (تسمى اللوالب في هذا النموذج). يتم تجميع المتطلبات الأساسية، بدءا من مرحلة التخطيط، ويتم تقييم المخاطر. كل دوامة لاحقة يبني على دوامة خط الأساس.

**يتكون النموذج اللولبي من اربع مراحل وهي** :

1. مرحلة التخطيط: يتم جمع المتطلبات خلال مرحلة التخطيط. متطلبات مثل "BRS" التي هي " مواصفات متطلبات الأعمال" و "SRS" ، هو "مواصفات متطلبات النظام".
2. تحليل المخاطر: في مرحلة تحليل المخاطر، يتم إجراء عملية لتحديد المخاطر والحلول البديلة. ويتم إنتاج نموذج أولي في نهاية مرحلة تحليل المخاطر. إذا تم العثور على أي خطر خلال تحليل المخاطر ثم يتم اقتراح الحلول البديلة وتنفيذها.
3. مرحلة الهندسة: في هذه المرحلة تم تطوير البرمجيات، جنبا إلى جنب مع اختبار في نهاية تيلي المرحلة. وبالتالي يتم في هذه المرحلة تطوير الإطارات والاختبار.
4. مرحلة التقييم: تسمح هذه المرحلة لعميل الاطارات بتقييم انتاج الاطارات للمشروع حتى تاريخه حتى يستمر المشروع في الاطارات دوامة المقبل.

**5- Fourth generation techniques (4GT)**

The term "fourth generation .techniques" (4GT) encompasses a broad array of software

tools that have one thing in common. Each enables the software engineer to specify some

characteristic of software at a high level, the tool then automatically generates source

\_

code based on the developer's specification.

The 4GT paradigm for software engineering focuses on the ability to specify software

using specialized language forms or a graphic notation that describes the problem to be

solved in terms that the customer can understand.

Currently, a software development environment that supports the 4GT paradigm includes

some or all of the following tools:

•

Non -procedural languages for database query

•

Report generation

•

Data manipulation

•

Screen interaction

•

Definition & code generation; high-level graphics capability

•

Spreadsheet capability

Like other paradigms, 4GT begins with a requirements gathering step, the customer

would describe requirements and these would be directly translated into an operational

prototype.

**Advantages**

**of (4GT):**

•

Simplified the programming process.

•

Use non -procedural languages that encourage users and programmers to specify

the results they want, while the computers determines the sequence of instruction

that will accomplish those results.

•

Use natural languages that impose no rigid grammatical rules.

**Disadvantages of (4GT):**

Us.

•

Less flexible that other languages

.t•

Programs written in 4GLs are generally far less efficient during program

execution that programs in high-level languages. Therefore, their use is limited to

projects that do not call for such efficiency.

***-***

**Rapid Application Development model (RAD)**

RAD model is Rapid Application Development model. It is a type of incremental model. In RAD model the components or functions are developed in parallel as if they were mini projects. The developments are time boxed, delivered and then assembled into a working prototype. This can quickly give the customer something to see and use and to provide feedback regarding the delivery and their requirements.

The phases in the rapid application development (RAD) model are:

1- Business modeling: The information flow is identified between various business functions

1. Data modeling : Information gathered from business modeling is used to define data objects that are needed for the business
2. Process modeling: Data objects defined in data modeling are converted to achieve the business information flow to achieve some specific business objective. Description are identified and created for CRUD of data objects
3. Application generation: Automated tools are used to convert process models into code and the actual system.
4. Testing and turnover: Test new components and all the interfaces.

**Team#1**

**Business Modeling**

**Data Modeling**

**Process Modeling**

**Application generation**

**Testing& Turnover**

**Team#2**

**Business Modeling**

**Data Modeling**

**Process Modeling**

**Application generation**

**Testing& Turnover**

**Team#3**

**Business Modeling**

**Data Modeling**

**Process Modeling**

**Application generation**

**Testing& Turnover**

|  |
| --- |
| Days 90- 60 |

**Diagram of RAP Model**

**Advantages of the RAD model:**

* Reduced development time**.**
* Increases reusability of components**.**
* Quick initial reviews occur
* Encourages customer feedback
* Integration from very beginning solves a lot of integration issues.

**Disadvantages of RAP model:**

* Depends on strong team and individual performances for identifying business' requirements.
* Only system that can be modularized can be built using RAD
* Requires highly skilled developers/designers.
* High dependency on modeling skills
* Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.

**When to use RAD model:**

* RAD should be used when there is a need to create a system that can be modularized in 2-3 months of time.
* It should be used if there’s high availability of designers for modeling and the budget 'is high enough to afford their cost along- with the cost of automated code generating tools.
* RAD SDLC model should be chosen only if resources with high business knowledge are available and there is a need to produce the system in a short span of time (2-3months).