Chapter two

2.1 Software Lifecycle

Each software product passing to a number of distinct stages as shown in Figure (2-1) which are:

1- Planning
2- Software analysis
3- Software design
4- Software construction
5- Software evaluation or testing
6- Software deployment
7- Software maintenance

Software Implementation

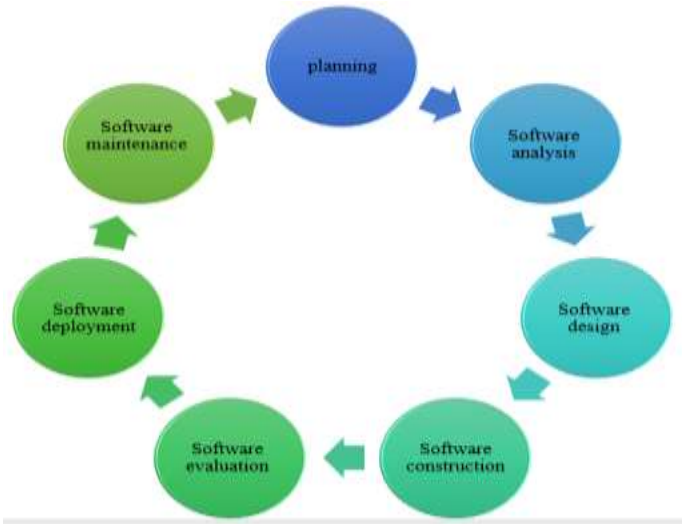

Figure (2-1)

1- Planning

This stage involve:-

a- **Communication:-** User initiates the request for a desired software product by contacts with the service providers to provide them with his requirement and submits the request to the organization.

يقوم المستخدم بطلب software product بواسطة التواصل مع مزودي الخدمة لتزويدهم بمتطلباته وتقديم الطلب للمؤوسسة المسؤولة.

b- **Feasibility Study:-** the team analyzes if a software can be designed to fulfill all requirements of the user, and if there is any possibility of software being no more useful. It is also analyzed if the project is economically, practically, and technologically feasible for the organization to take up.

يقوم فريق العمل بتحليل ما إذا كان من الممكن تصميم software بحيث يلبي جميع متطلبات المستخدم ، وإذا كان هناك أي احتمال لعدم جدوى Software .كما يتم عمل دراسة جدوى لتحديد فيما اذا كان مجدياً اقتصاديًا وعمليًا وتكنلوجباً بالنسبة للمؤسسة

2- Software analysis

a- The functionality of the software and constraints on its operation must be defined.
b- The developers try to bring up the best software model suitable for the project.

في هذه المرحلة يتم فهم software product وتحديد المتطلبات (الاهداف) بشكل كامل وتحديد المحددات على عمله. حيث يقوم المطورون بعمل software model مناسبة لعملهم.

3- Software design

Convert by developer

Logical software requirements ---------- technical software design

This happened by describe the software in such a way that programmers can write line of code that implement what the requirements specify. Engineers produce meta-data and data dictionaries, logical diagrams, data-flow diagrams, and in some cases pseudo codes.

يقوم المصممون بوضع تصميم للنظام (تصميم واجهات انظام system interface ،قواعد البيانات) بالاعتماد على احد design approaches ووضع خوارزمية algorithm للنظام كامل بشكل يحقق requirements ليقوم المبرمج بكتابة code.

4- Software construction

is concerned with implementing the software design by means writing the programs in one or more programming languages and then compiling and linking them. This stage content several steps:

a. Software reuse

The goal of software engineering is to achieve many features with little effort and few defects. Software components 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.

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

Software must be properly documented and understandable so as to minimize the cost of updates that may occur

d. Coding standards

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

5- Software evaluation or testing

In this phase, the software is reviewed with the purpose of detecting defects.

يتم اختبار عمل كل جزء من البرنامج ثم توحيدها في برنامج واحد واختباره لاكتشاف فيما اذا كان هنالك اخطاء.

Starting with verification ---------- ending with validation

Verification: - checking that the software (not the actual final product) meet the specified requirements for that phase of life cycle, someone other than the programmer reads a program unit of limited size to determine whether it satisfies the requirements.

في هذة المرحلة تتم مراجعة software للتاكد انه يحقق requirements المتطلبات الخاصة بتلك المرحلة من دورة حياة software، يقوم بهذة العملية شخص غير المبرمج بمراجعة code للتاكد انه يحقق requirements.

Software validation, checking that the software (during or at the end of life cycle) is what the customer requires or meets intended use.

في هذه المرحلة يتم فحص software للتاكد انه يحقق متطلبات المستخدم user requirements خلال او في نهاية دورة حياة software وطبقا للمعايير العالمية للجودة.

6- Software deployment

Refers to all of the activities that accrue after a software system has been developed and made available for release and use for users

- the following deployment activities
- Release, packaging, transfer, installation, update, download

7- Software maintenance

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

يصبح هنالك تغييرات معينة بعد اول اصدار للبرامجيات ، نحتاج الى صيانة البرامجيات وذالك لغرض تحسينها، وتشمل هذه المرحلة الدعم، تصحيح الاخطاء، واضافة خصائص او وظائف جديدة .

2.2 Software layered technology

Software engineering is a layered technology. Any software can be developed using these layered approach. Various layers on which the technology is based are quality focus layer, process layer, methods layer, tools layer. As shown in figure (2-2)

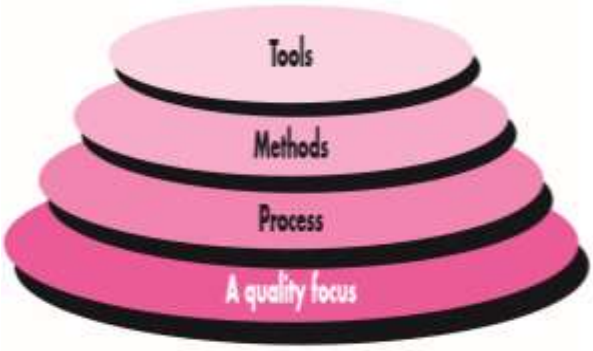

Figure (2-2)

1- A quality focus: a disciplined quality management is a backbone of software engineering technology.
2- Process layer: is a foundation of software engineering. Basically, process defines the framework for timely delivery of software.
3- Method layer: the actual method of implementation is carried out with the help of requirement analysis, designing, program construction, testing and maintenance.
4- Tools: are used to bring automation in software development process.

So, software engineering is a combination of process, methods and tools for development of quality software.

2.3 Software Process models

When you build a product or system, it's important to go through a series of predictable steps—a road map that helps you create a timely, high-quality result. The road map is called a 'software process.'

عند بناء منتج أو نظام ، من المهم اتباع سلسلة من الخطوات ، وهي عبارة عن خريطة طريق تساعدك في الحصول على نتيجة عالية الجودة في الوقت المناسب .تُعرف هذه الخريطة بأنها "software process"

Software process is a set of activities, actions and tasks whose goal is the development of software.

There are four fundamental process activities (1. Software analysis, 2. Software design, 3. Implementation, 4. Software maintenance), these different activities may organized in different ways and described at different level of detail for different types of software.

A software process model (also referred as software Development process Models) is an abstract representation of a process. It presents a description of a process from some particular perspective.

Each process model follows a life cycle in order to success in process of software development. A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required. There are variety different process models such as:-

1- **The Waterfall Model (linear sequential model)**

Was first process model introduced in software engineering. This model is known as waterfall model because all these phases overlap and feed information to each other, the second phase should not start until the first phase has finished (when defined set of goals are achieved). The waterfall model should only be used when the requirements are well understood and unlikely to change radically during system development, Figure (2-3) showing waterfall model.

وهو اول نموذج تم تقديمه ويعرف هذا النموذج باسم نموذج الشلال لأن كل هذه المراحل تتداخل لتمنح المعلومات لبعضها البعض ، ولا ينبغي أن تبدأ المرحلة الثانية حتى تنتهي المرحلة الأولى (عند تتحقق اهداف المرحلة) .يجب استخدام نموذج الشلال فقط عندما تكون المتطلبات مفهومة بشكل جيد ومن غير المرجح أن تتغير بشكل جذري خلال تطوير النظام.

This model divided into separate process phases:-

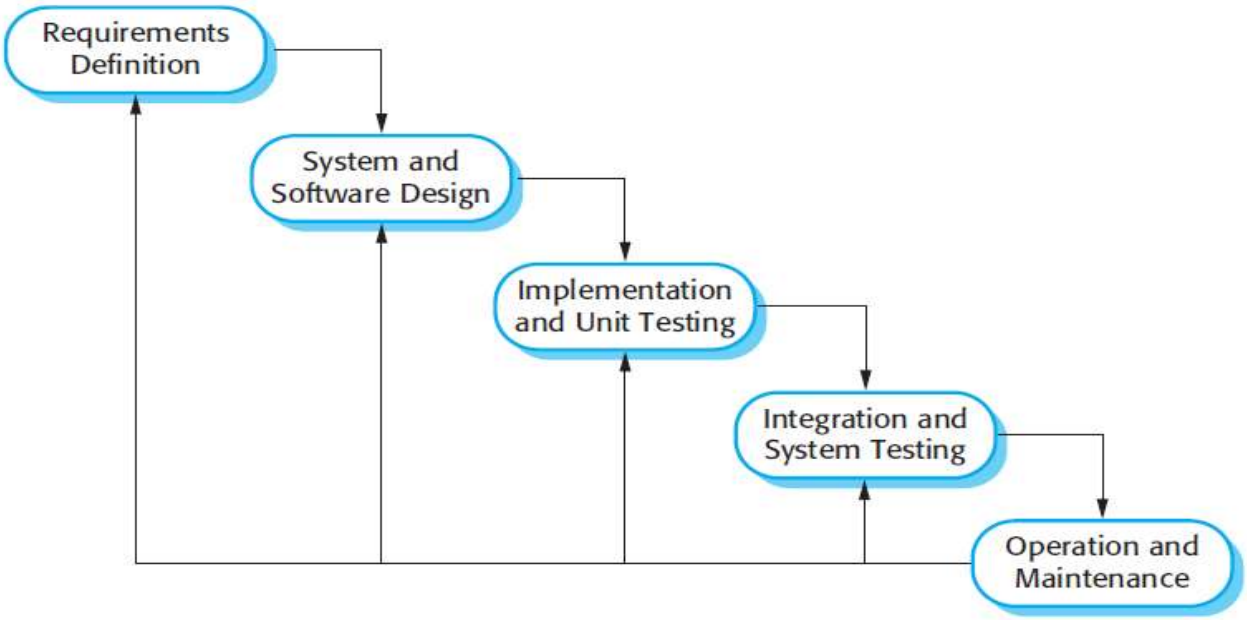

Figure (2-3) waterfall model

1- Requirement gathering and analysis phase

The system's services, constrains and goals are established by consultation with system users. They are then defined in detail and serve as a system specification.

يتم تعريف جميع المتطلبات بهذة المرحلة ويجب فهمها بشكل جيد بواسطة software engineer الذي يسمى Analyst ايضا ثم يتم تعريفها بشكل مفصل.

2- System and software design

Partitions the requirements to either hardware or software systems. It establishes overall system architecture. Software design involves identifying and describing the fundamental software system abstractions and their relationships, design focus on program attribute such as (data structure, software architecture, interface representation, algorithm details).

تحديد الهيكلية العامة للنظام كما مر ذكره.

3- Implementation and unit testing phase

During this stage, the software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specification.

تكتب البرامج على شكل اجزاء ويتم اختباركل جزء منها لتحقق بأنها تلبي المتطلبات

4- Integration and system testing

The individual program units or programs are integrated and tested as a complete system to ensure that the software requirements have been met. After testing, the software system is delivered to the customer.

يتم دمج واختبار program units الفردية أو programs كنظام كامل لضمان تلبية المتطلبات .بعد الاختبار ، يتم تسليم software system إلى العميل.

5- Operation and maintenance

Is the longest life cycle phase. When the system is installed and put into practical use, then the error may get introduced, correcting such errors is the major purpose of Maintenance, and also improving the system's services as new requirements are discovered.

هي أطول مرحلة في life cycle عندما يتم تثبيت النظام واستخدامه عمليًا ،عندها تبدأ الاخطاء بالظهور ، وتصحيح هذه الأخطاء هو الغرض الرئيسي لهذه المرحلة ، وكذلك تحسين خدمات النظام عند اكتشاف متطلبات جديدة.

- Benefits of waterfall model

1. the documentation it produce at each phase.
2. Fit with other engineering process models.

- Drawbacks (problems) of waterfall model

1. The requirement analysis is done initialy and sometimes it is not possible to state all therequirements explicitly in the beginning. This lead to increase of cost.

من المهم اكمال جميع requirements في مرحلة requirement Gathering في هذا model لكن غالبا requirements تضاف حتى بعد اول مرحلة لذا فان تكاليف requirements المضافة والتي لم تضاف في المرحلة الاولى يتحملها المهندس المسؤول فتزداد cost

2. The customer can see the working model of the project only at the end, after reviewing of the working model; if the customer gets dissatisfied then it causes serious problems.

يمكن للعميل رؤية نموذج العمل الخاص بالمشروع فقط في النهاية ، بعد مراجعة نموذج العمل ؛ إذا كان العميل غير راض ، فإنه يسبب مشاكل خطيرة.

3. The linear nature of waterfall model leads to "blocking states" in which some project team members must wait for other members of the team to complete dependent tasks. In fact, the time spent waiting can exceed the time spent on productive work.

Problems في one phase لا تحل تماما عند هذة phase وبالتالي يتم الانتقال الى phase الاخرى وبالتالي سيستغرق long time لحل المشكلة

4. Waterfall model is not appropriate for work with is fast-paced and subject to a never-ending stream of changes

غير مناسب للاعمال سريعة الخطى والخاضعة الى تغييرات مستمرة

2- Prototyping model

The prototype model is using for many reasons, such as:

a) A customer define a set of general objectives for software but does not identify detailed input, processing, or output requirements.

الزبون customer يعرف مجموعة من الاهداف العامة للsoftware ولا يحدد بالتفاصيل كل متطلبات input, processing or output

b) The developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human/machine interaction should take.

Developer يكون غير متاكد من كفاءة algorithm وهل operating system المعتمد مناسب وهل سيتفاعل معه الموظفين

When implementing a prototype model, you first develop the parts of the system you understand least, then you begin by developing the parts of the system you understand best.

The stages of prototyping model

1- requirements gathering (listen to customer)

Developer and customer meet and define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is mandatory.

يجتمع المطور والزبون لتعريف الأهداف العامة لل software ، وتحديد أية متطلبات معروفة، والمجالات التي تتطلب تعريفات أكثر

A **"quick design"** then occurs. The quick design focuses on a representation of those aspects of the software that will be visible to the customer/user (e.g., input approaches and output formats).

ثم يوضع بعد ذالك "Quick design" يركز التصميم السريع على تمثيل نواح محددة من software، وخاصة تلك التي ستكون مرئية للزبون أو المستخدم (input and output and interface) .

2- Construction of prototype (build / revise mock-up)

Writing the software code depending on the quick design information.

3- **Evaluation (customer test drives mock-up):-**

The prototype is evaluated by the customer/user and used to refine requirements for the software to be developed. Iteration occurs as the prototype is tuned to satisfy the needs of the customer, while at the same time **enabling the developer to better understand what needs to be done.**

يتم تقييم النموذج الأولي من قبل الزبون أو المستخدم ويستخدم ذلك التقييم لتحسين المتطلبات للبرامجيات التي يجب تطويرها . ويحدث iteration من خلال ضبط النموذج الأولي لتحقيق متطلبات الزبون، وهذا يمكّن المطور في الوقت ذاته من إيجاد فهم أفضل لتحقيق الاهداف.

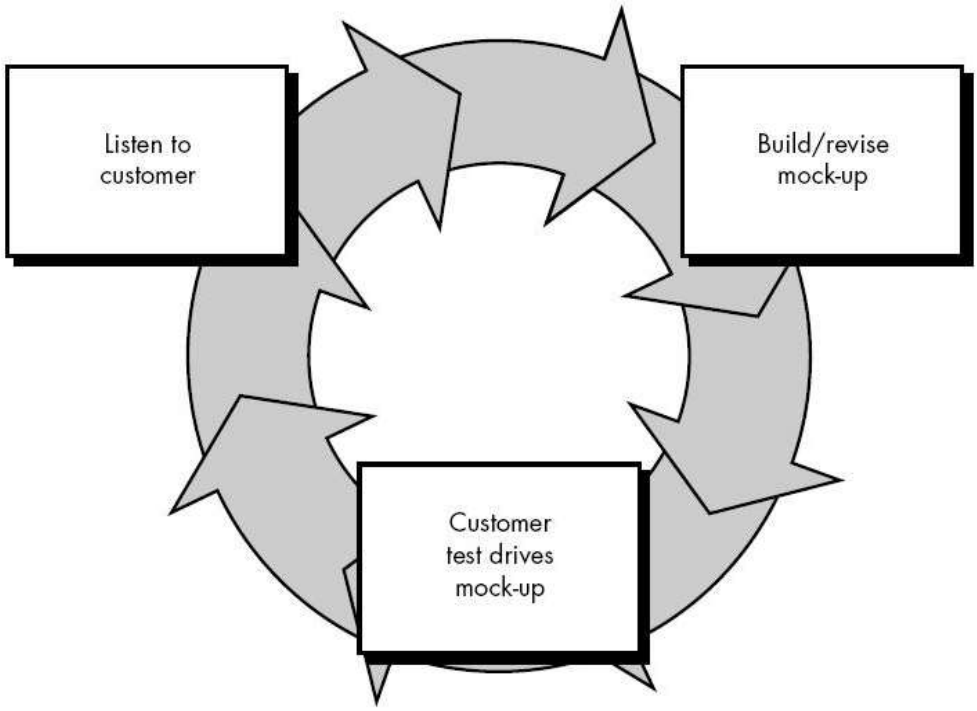

Figure (2-4): The prototyping Paradigm

- Benefits of using Prototype Model:

1. Improved system usability.
2. A closer match of the system to users' needs.
3. Improved design quality.
4. Improved maintainability.
5. Reduce development effort

(Detect errors earlier and save time and money).

- Drawbacks (problems) of Prototyping model

The prototyping can also be problematic for the following reasons:

1- The customer sees what appears to be a working version of the software, unaware that in the rush to get it working no one has considered overall software quality or long-term maintainability.

يرى الزبون في prototype ما يبدو انه نسخة صحيحة (عاملة) من software وهو غير مدرك إن هذا النموذج الأولي مترابطا ترابطا واهيا جدا، ولا يدرك إننا بسبب السرعة في انجازه لم نأخذ بالحسبان quality الكلية لل software أو لقابلية صيانته(maintainability) على المدى الطويل.

2- The developer often makes implementation compromises in order to get a prototype working quickly. An inappropriate operating system or programming language may be used simply because it is available and known; an inefficient algorithm may be implemented simply to demonstrate capability.

غالبا ما يجري المطور بعض التجاوزات في انجاز النموذج الأولي لجعله يعمل بسرعة. فقد يستخدم نظام تشغيل أو لغة برمجة غير مناسبين، فقط لأنهما متوافران ومعروفان، وقد يعتمد خوارزمية غير كافية، فقط لإيضاح الإمكانيات.

3- It may be impossible to tune the prototype to meet some important requirements that were ignored during prototype development, such as performance, security, robustness and reliability.

قد يكون من المستحيل ضبط النموذج الاولي لتلبية بعض المتطلبات المهمة التي تم تجاهلها أثناء تطوير النموذج الاولي ، مثل الأداء والأمان والقوة والموثوقية .

4- Rapid change during development inevitably means that prototype is undocumented. Only the design specification is the prototype code. This is not good enough for long-term maintenance.

التغيير السريع خلال التطوير يعني حتما أن النموذج الاولي غير موثق. وهذا ليس جيدا بما فيه الكفاية للصيانة على المدى الطويل.

3. Evolution software process model

Evolutionary models are iterative. They are characterized in a manner that enables software engineers to develop increasingly more complete versions of the software. There are several types of that model, these are:

نماذج ال Evolutionary هي تكرارية وهي تتميز بطريقة تمكن مهندسي البرمجيات من تطوير إصدارات أكثر اكتمالا للبرامجيات software .هناك عدة أنواع من هذا النموذج وهي:

لقد صمم لحالات التطوير المباشر (خط مستقيم) تفترض انه سيتم تسليم كامل النظام بعد اكتمال هذا التتابع الخطي اما النموذج (prototyping model) صمم لمساعدة الزبون او المطور على فهم المتطلبات ولم يسلم عموما لغاية لتسليم نظام نهائي.

b- The incremental model
c- The spiral model
d- Exploratory model

a- The incremental model (calendar time)

The incremental model combines elements of linear sequential model + iterative of prototyping model

- The incremental model applies linear sequences as calendar time progresses.
- The incremental model delivers a series of releases (increments) to the customer, more and more functionality is associated with each increment.
- The first increment is called a **core product**, in this release the basic requirements are implemented and then in subsequent increments new requirement are added.

يحقق ال **core product** المتطلبات الاساسية ويتم استخدامه وتقييمه من قبل الزبون، ثم يتم التخطيط لل next increment حيث تجرى التعديلات اللازمة لل **core product** لتلبية متطلبات المستخدم اوالزبون واضافة مزايا ووظائف اضافية. حيث تتكرر هذه العملية بعد تسليم كل increment حتى يتم تقديم المنتج بصورة كاملة complete product.

The incremental process model, like prototyping and other evolutionary approaches, is iterative in nature. But unlike prototyping, the incremental model focuses on the delivery of an operational product with each increment.

هذا النموذج هو تكراري مثل النموذج الاولي prototyping model ولكن يختلف عنه لانه يقدم operational product في كل increment

- For example, word-processing software developed using the incremental model as bellow:

1) In the first increment only the document processing facilities (the basic file management, editing, and document production functions) are available.
2) In the second increment, more sophisticated document production and processing facilities.
3) In the third increment, Spelling and grammar checking.
4) Advanced page layout capability in the fourth increment.

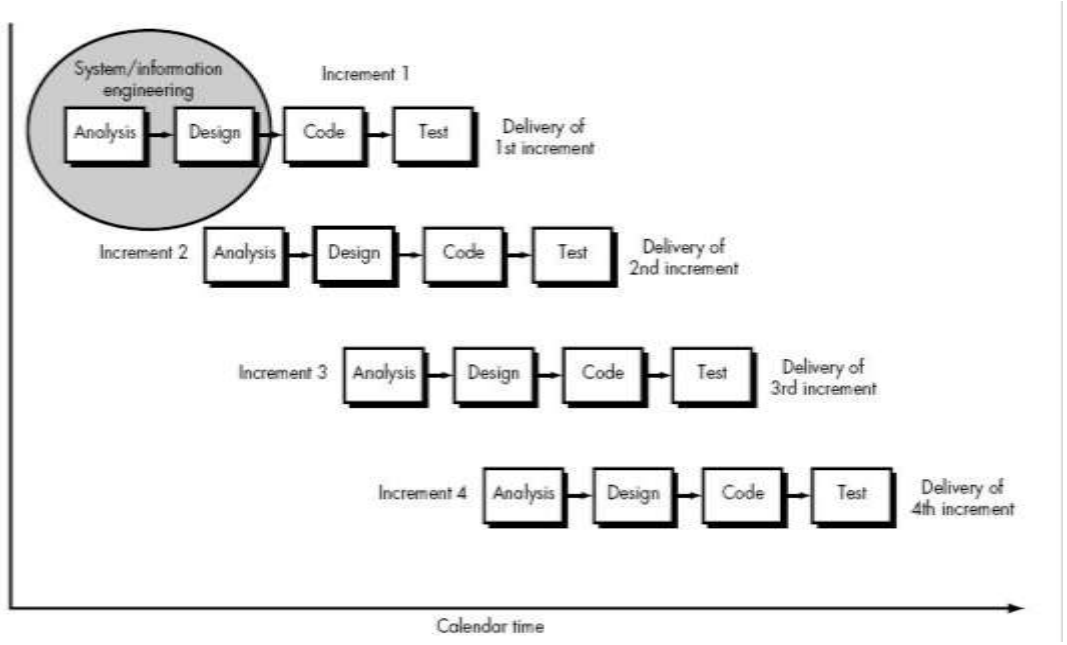

Figure (2-5): The incremental model

When to choose it?

1. When requirements are reasonably well-defined.
2. When overall scope of the development effort suggests a purely linear effort.
3. When limited set of software functionality needed quickly.

- Incremental development is particularly useful when:

1- Staffing is unavailable for a complete implementation by the business deadline that has been established for the project. Early increments can be implemented with fewer people. If the core product is well received, then additional staff (if required) can be added to implement the next increment.

عند عدم توفر فريق كاف من الموظفين لانجاز الأعمال في الموعد النهائي الذي تم تحديده للمشروع يمكن انجاز التزايدات الأولية Early increments بعدد قليل من العاملين، وإذا جرى تقبل core product باستحسان، يمكن عندئذ إضافة موظفين جدد (إذا كان ذلك ضروريا) لانجاز التزايد التالي next increment.

2- Increments can be planned to manage technical risks. For example, a major system might require the availability of new hardware that is under development and whose delivery date is uncertain. It might be possible to plan early increments in a way that avoids the use of this hardware, thereby enabling partial functionality to be delivered to end-users without inordinate delay.

يمكنها التخطيط لادارة technical risks مثلا، قد يتطلب نظام رئيسي توفر new hardware ما يزال قيد التطوير، وتاريخ تسليمه غير مؤكد. قد يكون ممكنا تخطيط early increment بطريقة ما بحيث يمكن تجنب استخدام هذا hardware، وبالتالي يمكن تقدم جزء من الوظائف للمستخدم دون حصول تأخير مفرط.

b. The Spiral Model

- proposed by Boehm, it couples the iterative nature of prototyping + the controlled and systematic aspects of the linear sequential model.

- This model gives efficient development of incremental versions of the software. In this model, the software is developed in a series of increments.

- A spiral model is dividing into a number of framework activities called task regions, contain six task regions as shown in figure (2-6):

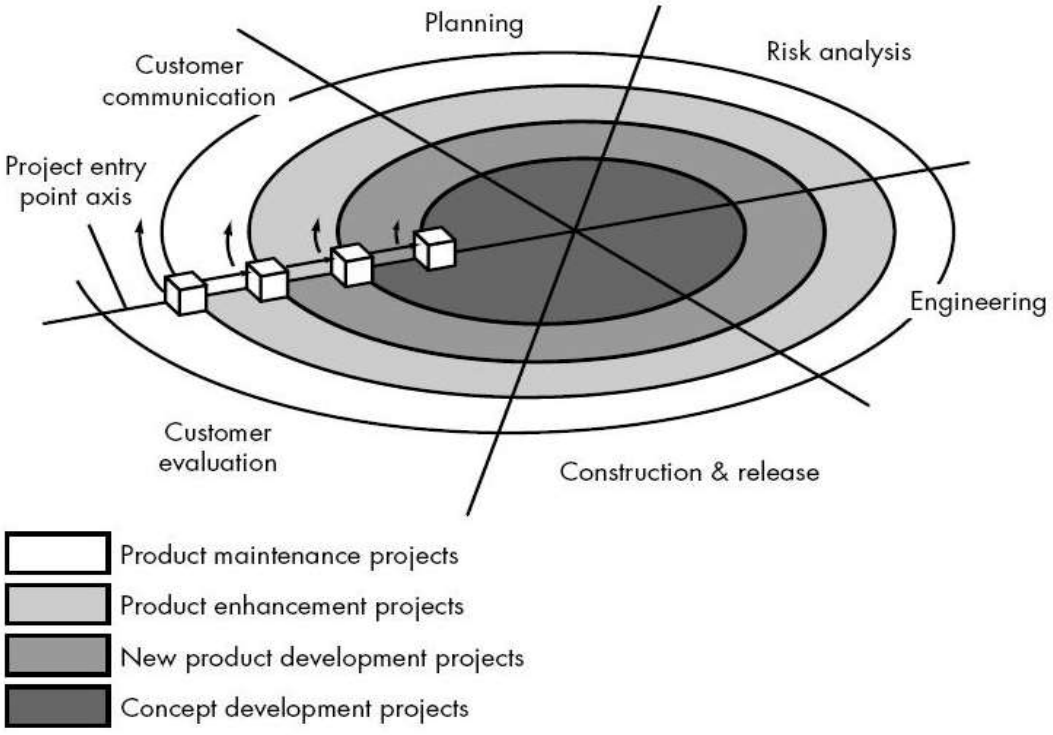

Figure (2-6): Spiral model

1. Customer communication

-tasks required to establish effective communication between developer and customer.

التواصل الفعال بين المطور والزبون.

2. Planning

-tasks required to define resources, timelines, and other project-related information.

مطلوبة لتعريف المعلومات عن المشروع مثل resource, timelines

3. Risk analysis

-tasks required to assess both technical and management risks. لتحديد المخاطر التقنية و الادارية

4. Engineering

-tasks required to build one or more representations of the application. Application مطلوبة لبناء نماذج لل

5. Construction and release

-tasks required to construct, test, install, and provide user support (e.g., documentation and training).

تعليم المستخدم وتدريبه على كيفية التعامل معه

6. Customer evaluation

-tasks required to obtain customer feedback based on evaluation of the software representations created during the engineering stage and implemented during the installation stage.

العودة للزبون Customer feedback من اجل الحصول على تقيمه لل software بعد تشغيله والعمل عليه

- The software engineering team moves around the spiral in a clockwise direction, beginning at the center.

يتحرك فريق هندسة البرمجيات حول spiral باتجاه عقارب الساعة بدءاً من المركز قد ينتج عن الدورة الأولى حول spiral تطوير مواصفات المنتج، وقد تستعمل الدورات التالية حول spiral لتطوير نموذج أولي prototype، ثم شيئاً فشيئاً تعد نسخ من البرامجيات أكثر تطورا ينتج عن كل مرور عبر planning region يؤدي الى ضبط خطة المشروع، ويجري ضبط الكلفة والجدول الزمني بالاعتماد على feedback الزبون إضافة إلى ذلك يضبط مدير المشروع عدد التزايدات iterations المخطط لها والمطلوبة لانجاز software.

محور نقطة دخول المشروع (point project entry) يمثل كل مكعب يوضع على هذا المحور نقطة البداية لمشروع جديد آخر.

يبدأ مشروع تطوير المفاهيم (concept development project) في core of the spiral ويستمر (تحدث تزايدات متعددة multiple iterationsعلى مسار spiral الذي يحيط بالمنطقة المظللة المركزية) حتى يكتمل تطوير المفهوم ، تتقدم عملية البرمجة عبر المكعب

التالي (نقطة دخول المشروع تطوير منتج جديد(new product development project entry point)) إذا كان المطلوب تطوير المفهوم ليصبح منتجا حقيقيا وتوضع في بداية لتطوير مشروع جديد، يتطور المنتج الجديد في عدد من التزايدات حول spiral متبعا المسار الذي يحيط بالمنطقة التي لها تظليل اخف من تظليل ال core ، ويحدث تدفق مشابه لعملية برمجة أنواع أخرى من المشاريع .

إن النموذج الحلزوني طريقة واقعية لتطوير أنظمة وبرمجيات واسعة النطاق، ولان البرمجيات تتطور مع تقدم عملية البرمجة، فانه من الأفضل للمطور والزبون فهم المخاطر والقيام بالإجراء المناسب لها في كل مستوى من مستويات التطور، يستخدم النموذج الحلزوني النمذجة الأولية prototype كآلية لتقليل المخاطر، ولكن الأهم من ذلك هو انه يمكن المطور من تطبيق نموذج النمذجة الأولية prototyping approach في أي مرحلة من مراحل تطور المنتج فهو يحافظ على الطريقة التدريجية النظامية waterfall التي تقترحها دورة الحياة التقليدية، ولكن يطبقها بإطار تكراري iterative framework يعكس واقع العالم الحقيقي ويتطلب spiral model اعتباراً مباشراً للمخاطر التقنية في جميع مراحل المشروع، وإذا طبق بالوجه المناسب، وجب أن يقلل المخاطر قبل أن تصبح مشكلة يصعب حلها.

Advantages of spiral model

1. Requirement changes can be made at every stage.
2. Risks can be identified and rectified before they get problematic.

- Drawbacks of spiral model:

1. It may be difficult to convince customers (particularly in contract situations) that the evolutionary approach is controllable.

2. Requires risk assessment expertise and relies on this expertise for success.

3. The model has not been used as widely as the linear sequential or prototyping paradigms.

١- قد يصعب إقناع الزبون (وخاصة في حالات توقيع عقود) بأنه يمكن التحكم بهذا النموذج.

٢- يتطلب خبرة جيدة في تقدير المخاطرة risk، ويعتمد على هذه الخبرة في نجاحه، فإذا لم تعالج المخاطر الرئيسة، فان المشاكل ستقع دون شك.

٣- أخيراً النموذج بحد ذاته جديد نسبياً ولم يستخدم على نطاق واسع مثل the linear sequential or prototyping paradigms

4- Exploratory model

The approach is based on the idea of developing an initial working system exposing this to the user and modifying system through many stages until performs adequately is the approach for systems where it is very difficult or impossible to establish a detailed system specification. Therefore has been used for the development of A.I system. it use a very high level programming language such as LISP or PROLOG for software development

The exploratory model phases are:-

1- initial specification development :

يستخدم immediately available information لخلق starting point

2- system construction / modification (development)

بناء وخلق وتعديل للنظام طبقا لما موجود من معلومات

3- system test (validation)

اختبار النظام لمعرفة كيف يعمل وكيف يتم تعلمه وكيف يتطور

4- system implementation

بعد عدة تكرارات iteration للخطوتين السابقتين system finished

-The problem associated with the exploratory model

1- It is difficult to measure or predict its cost, effectiveness. من الصعب قياس أو التنبؤ بتكلفته وفعاليته

2- it is limited to use with very high level language يقتصر استخدامه على اللغات عالية المستوى

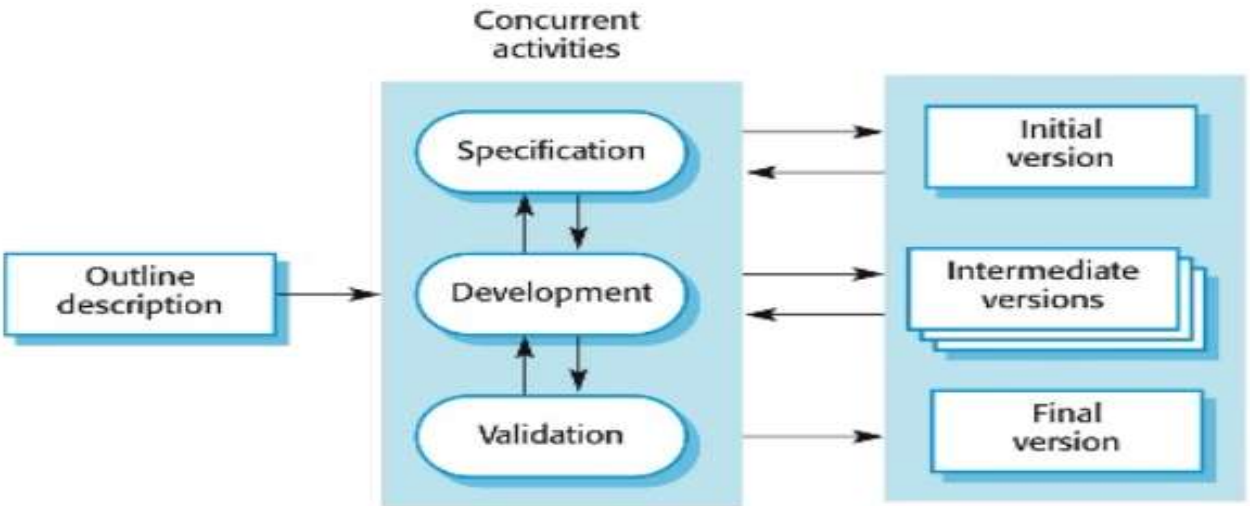

Figure (2-7): Exploratory model