AL- Mustansirya University
College of Education
Department of Computer Science

Course: Software Engineering
Lecturer: Iman Hussein
Third Class

CHAPTER 3

Software Requirement Engineering

هندسة المتطلبات البرمجية

Topics:

3.1 Introduction
3.2 Requirement and its Problems
3.3 Software Requirement Engineering and its Objectives
3.4 Software Requirement Engineering Activities
- Requirement Elicitation
- Requirement Analysis
- Requirement Specification
- Requirement Validation
- Requirement Management

3.1 Introduction

After system engineering is completed, software engineers need to look at the role of software in the proposed system. Software requirements analysis is necessary to avoid creating software product that fails to meet the customer's needs. Data, functional, and behavioral requirements are elicited from the customer and refined to create specification that can be used to design the system. Software requirements work products must be reviewed for clarity, completeness, and consistency.

مقدمة :

ان احد الاسباب الاساسية لفشل معظم المنظومات البرمجية هو وجود مشاكل في تحديد المتطلبات(Requirements) ويمكن ان نقول بأن تقريبا 85% من الاخطاء في المنظومات مرده ومرجعه عدم وضوح المتطلبات وبالتالي المساعدة على عرقلة المراحل التي تلي مرحلة التحليل وفي النهاية الحصول على منظومات غير ذات جودة ولا تلبي احتياجات المستخدم بالكامل .

لقد كان الاهتمام في :

- الستينيات بالبرمجة .
- السبعينيات بالتصميم .
- الثمانينيات والتسعينيات (والى حد الان) بالمتطلبات والمواصفات .

وتعتبر المرحلة المتعلقة بجمع المتطلبات Requirement Gathering من اهم مراحل دورة حياة اعداد المنظومة لان جمع المتطلبات اذا لم يتم بالصورة الصحيحة والكاملة فأن كل المراحل التي تليها ستكون بالطبع غير كاملة .ومهما يكن مستوى التصميم جيدا فلن تنجح المنظومة الا بجودة المتطلبات اولا .

وهناك مفهوم خاطىء عند بعض الناس مرده ان المتطلبات موجودة في رأس المستخدم او الزبون . والحقيقة التي تم اكتشافها هي ان جمع المتطلبات عمل جماعي يتم اداؤه من قبل ما يسمى Stakeholders وهم الزبون والمستخدم ومعد المنظومة .

لهذا بدأ العلماء والمتخصصون والهيئات العلمية ذات العلاقة بهذا المجال في التفكير جديا بالتعامل مع هذه المسألة وايجاد الحلول لها ومن ثم ظهر مصطلح هندسة المتطلبات البرمجية Requirement Engineering يطفو على السطح في بداية التسعينيات وهو نفس النمط الذي ظهر في نهاية الستينيات عندما ظهر مصطلح هندسة البرمجيات كحل لأزمة البرمجيات Software Crisis التي لم تحل بالكامل حتى الان .

فهندسة المتطلبات البرمجية ستساهم مع غيرها من الطرق لحل هذه المشكلة التي شعرت بها الشركات المتخصصة لاعداد المنظومات وادركت ان العواقب ستكون وخيمة وقوية ومؤثرة على الاعداد والتكلفة والصيانة .

وقبل ان نبدأ في شرح محتويات هذا الفصل نورد بعض التعريفات المهمة للأشخاص الذين لهم علاقة بهندسة المتطلبات :

- المستخدم النهائي End-User : وهو الشخص الذي سيستخدم المنظومة لأداء الوظائف المنوطة به.
- مهندس البرمجيات Software Engineer : وهو الشخص الذي له الخبرة والمعرفة بالتحليل والتصميم والبرمجة والاختبار واعداد النماذج التجريبية وغيرها .
- مهندس المتطلبات Requirement Engineer : وهو الشخص المسؤول على التعرف على المتطلبات وتوثيقها .

نستعرض في هذا الفصل معنى ومشاكل المتطلبات ثم نعرج على خمس نشاطات رئيسية (وظائف هندسة المتطلبات البرمجية)خاصة بالوصول الى متطلبات صحيحة وكاملة وواضحة وهذه النشاطات تشتمل على :

وظائف هندسة المتطلبات البرمجية (نشاطات) :

1) استنباط المتطلبات Requirement Elicitation
2) تحليل المتطلبات Requirement Analysis
3) توصيف المتطلبات Requirement Specification
4) اعتماد المتطلبات Requirement Validation
5) ادارة المتطلبات Requirement Management

3.2 Requirement and its Problems

المتطلبات ومشاكلها

المتطلبات(Requirement) : ما هي الا ملخص لمجموعة طلبات يرغبها الزبون في شكل وظائف وقدرات وخصائص ومعايير جودة خاصة بالمنظومة المعلوماتية المطلوبة وذلك لجلب منفعة وقيمة للمستخدم .او بتعبير اخر هي قدرات برمجية مطلوبة في المنظومة لحل مشكلة المستخدم.

انواع المتطلبات البرمجية :

يمكن تصنيف المتطلبات البرمجية حسب الاتي :

1- متطلبات وظيفية Functional Requirements
المتطلبات الوظيفية هي الافعال (الوظائف) المرغوب اداؤها بواسطة المنظومة البرمجية .او هي المعاملات الرئيسية التي تحدث في المنظومة من حيث قراءة ومعالجة البيانات والحصول على النتائج في صورة مخرجات .

2- متطلبات الأداء Performance Requirements
وتسمى متطلبات زمن الاستجابة مثل : كم عدد المعاملات التي يتم تنفيذها في فترة زمنية محددة او ان زمن الاستفسار عن صنف معين لا يتعدى ثلاث ثوان او نوع وكمية البيانات التي يتم التعامل معها ومعالجتها فمثلا :
قد تكون احدى المتطلبات ما يلي :95% من المعاملات يجب اداؤها في زمن اقل من الثانية الواحدة .

3- متطلبات البيانات Data Requirements
البيانات المطلوبة في المنظومة الجديدة .

4- متطلبات التشغيل Operational Requirements
يمكن ان تشتمل على مؤهلات وقدرات المستخدم ومعايير واجهة المستخدم المطلوبة .

5- متطلبات القبول Acceptance Requirements
وتتضمن خصائص الجودة المطلوبة في المنظومة البرمجية مثل الاعتمادية وسهولة الصيانة والحماية والكفاءة وغيرها .

المتطلبات : المشاكل والأسباب

تتلخص اهم المشاكل والأسباب التي تتعلق بالمتطلبات في الآتي :

- الزبون او المستخدم لا يعرف بالضبط ماذا يريد وما هي احتياجاته الفعلية .
- التحليل في اغلب الاحيان ناقص وغير كامل .
- التغيير الكبير في تقنية الحاسوب (البرمجياتSW والعتادHW) وعالم التجارة والاعمال زاد من الضغط على تقليص مدة اعداد المنظومة .
- معد البرمجيات يجد صعوبة في ترجمة او تحويل المتطلبات الى برامج وقواعد بيانات .
- التغيير في المتطلبات من جانب الزبون بشكل متكرر وهو في بعض الاحيان غير مبرر.
- الميزانية والجدول الزمني غير معقول وغير مقبول.
- التغيير في اللوائح والقوانين زاد من مشكلة المتطلبات .
- عدم تعريف نطاق العمل المراد اداؤه بصورة صحيحة وكاملة .
- التعرف على كامل المتطلبات وتوثيقها صعب للغاية .
- عدم قدرة المستخدم على فهم اهمية وغرض مواصفات المتطلبات .
- عدم القدرة على فهم منهجية صحيحة عند اعداد المنظومات .
- الاعتقاد السائد بين المدراء بأن مرحلتي البرمجة والاختبار هما المجهود الحقيقي والمهم في اي مشروع برمجي دون مراعاة اهمية المراحل الاخرى .
- نقص في مدراء مشاريع البرمجيات المهرة وذوي الخبرة .

3.3 Software Requirement Engineering and its Objectives

هندسة المتطلبات البرمجية واهدافها

هندسة المتطلبات البرمجية Software Requirement Engineering هي كل النشاطات المستخدمة للتعرف على المتطلبات ثم تحليل هذه المتطلبات للوصول الى متطلبات اضافية ومن ثم توثيق واعتماد هذه المتطلبات لتلبي احتياجات المستخدم .

اما الاهداف التي ترمي اليها فهي :

1- فهم اساسيات هندسة المتطلبات .
2- التعرف على طرق استنباط المتطلبات .
3- تحليل المتطلبات البرمجية .
4- تكوين وكتابة مواصفات المتطلبات .
5- تقييم واعتماد المتطلبات البرمجية .
6- ادارة اي تغييرات تحصل لهذه المتطلبات .

.....

3.4 Software Requirement Engineering Activities

ولنبدأ بوظائف هندسة المتطلبات البرمجية (النشاطات) :

اولاً- استنباط المتطلبات Elicitation of Requirements

يعتبر هذا النشاط وهو جمع واستنباط المتطلبات من المستخدم والزبون الوظيفة الاولى في مرحلة التحليل ويتم فيه الاستفادة من كل الذين لهم علاقة بالمنظومة Stakeholders المراد اعدادها من اجل اكتشاف واستنباط وفهم احتياجات المستخدم User needs .

ان اصعب مهمة في عملية جمع المتطلبات ليس توثيق ما يريده الزبون ولكن محاولة مساعدة الزبون لفهم ما يحتاجه .

ويتم في هذا النشاط استخدام مجموعة طرق وادوات لهذا الغرض منها :

1- المقابلة الشخصية Interview

تعتبر المقابلة الشخصية احدى اهم طرق جمع المتطلبات الخاصة بالمنظومات المعلوماتية وفي بداية المشروع يقضي محلل النظم وقتا كافيا لعمل مقابلات شخصية مع الزبون والمستخدم من اجل فهم طبيعة العمل والبيانات والمعلومات المتداولة في بيئة الزبون وايضا القوانين واللوائح المستخدمة لتسيير دفة العمل .

2- الاستبيان Questionnaire

بالرغم من اهمية المقابلة الشخصية كطريقة من الطرق الرئيسية في الحصول على المتطلبات الا انها مكلفة ويتم اجراؤها على عدد صغير ومحدود بمعنى انه لن تكون الصورة واضحة على كل ما يجري .
لهذا فان طريقة الاستبيان Questionnaire هي النقيض حيث تجري على عدد هائل من الناس في زمن قليل . وعادة يتم تجهيز الاستبيان على ورق الا انه يمكن اجراؤه على الهاتف او الانترنت .
ويحتوي الاستبيان عادة على مجموعة من الاسئلة القصيرة والطويلة او الاسئلة ذات الاجابة المحددة من بين مجموعة اجابات Multiple Choice . وتجري الاستبيانات وتسمى ايضا المسح Survey للوصول الى هدف معين . وعلى سبيل المثال يمكن استخدامها لمعرفة رأي المستخدمين في عمل النظام (او المنظومة) الحالي .

3- الملاحظة Observation

تعتبر طريقة مباشرة لتفقد ومعرفة النظام تحت الدراسة وتساعد على الوصول للمعلومة بشفافية وبموضوعية كما هي وليس كما يعتقدها الزبون والمستخدم . ولكن من عيوبها ان عملها ليس دائم ومستمرا لهذا قد يلتجيء المستخدم للتصنع والعمل المثالي لحظة ملاحظة المحلل وبالتالي تكون المعلومات المجمعة لا تعكس الواقع المعيش.
لهذا من الافضل اداء هذا العمل (في بعض الحالات) من دون علم الموظف القائم بالعمل وهذا لا يعني انه في حالات معينة يمكن ان يتفاعل محلل النظم مع الشخص المعني وان يوجه الاسئلة او يشاهد ويجمع بعض النماذج المستخدمة .

4- جمع وتحليل العينات والوثائق Document Gathering & Sampling

تعتبر طريقة جمع وتحليل النماذج Forms والوثائق Documents هامة ايضا ومجدية لاننا قد نجد طرقا اخرى تساعدنا على معرفة طريقة عمل النظام كما هي على ارض الواقع او كما تم شرحها اثناء المقابلة والملاحظة .
لكن السؤال هو هل يعمل المستخدم بالطريقة المثالية والصحيحة المدونة بالقوانين واللوائح في شكل وثائق او لا ؟
وللاجابة عن السؤال اقول : عند جمع عينات من الوثائق الخاصة باللوائح والملفات والنماذج والتقارير والمخططات التنظيمية للمؤسسة وتحليلها يمكن معرفة ومقارنة طريقة عمل النظام بالطريقة المثالية وطريقة عمل النظام بالطريقة الفعلية . وبالطبع سيستفيد محلل النظم من هذا عند تحيد المتطلبات للمنظومة الجديدة .

5- تصميم التطبيق المشترك Joint Application Design (JAD)

بدأت هذه الطريقة في أواخر السبعينات من قبل شركة IBM لغرض جمع كل من الزبون (المدير ورؤساء الاقسام مثلا) والمستخدمين ومحللي النظم والمبرمجين ومقرر الجلسات في عملية تحليل النظام . وانتشرت هذه الطريقة JAD بعد ذلك كأداة مهمة تستعمل في اعداد المنظومة وذلك بأشراك الزبون في معظم المراحل .
وفي مرحلة التحليل يتم عمل ورش عمل (جلسات) لجمع المتطلبات من قبل الذين لهم علاقة بالنظام (Stakeholders)في مكان واحد وفي وقت واحد ولمدة قد تستغرق اسبوعا . وهي جلسات مكثفة لحل المشاكل او على الاقل معرفة السبب في صعوبة ايجاد الحلول . ان مشاركة المستخدم في اغلب مراحل المشروع اصبح من المفاهيم الحديثة والمرغوبة . وقد تم ايجاد علاقة بين مشاركة المستخدم ورضاه عن المنظومة حين تسلمها لان افضل الافكار الابداعية للمنتجات الجديدة وتحسينها تأتي عادة من الزبون وليس من معد المنظومة . ويجب اخذ الملاحظة بأن مكان الاجتماع يجب ان يكون بعيدا من مكان عمل الزبون ويجب اعداد الحجرة التي ستعقد فيها الجلسات جيدا بمعنى وجود السبورة البيضاء وجهاز الحاسوب للمقرر وطابعة لطباعة الوثائق التي قد توزع على الحاضرين وجهاز

عرض الشرائح باستخدام الحاسوب . اما بعد المكان فمرده الى الابتعاد عن الازعاج والارباك وزحمة العمل ليتسنى التركيز على العمل الذي كلفوا به **. وتتلخص النقاط في هذه الجلسات في ما يلي :**

1- لمحة على النظام الحالي والمشاكل التي تعترضه .
2- مناقشة لتصميم النماذج والتقارير للمنظومة المقترحة (واجهة المستخدم).
3- مناقشة حول وظائف المنظومة المقترحة وخصائص الجودة المطلوبة في هذه المنظومة .

6- تخطيط المتطلبات المشترك Joint Requirement Planning (JRP)

هذه الطريقة JRP لجمع المتطلبات تعتبر حالة خاصة من طريقة JAD وهي مصطلح خاص بالتحليل والمتطلبات بينما JAD لجميع المراحل . وهي جلسة عمل جماعية بعكس الطريقة التقليدية المتمثلة في : المقابلة التي يتم فيها اجتماع منفرد مع الزبون او المستخدم .

فهذا الاجتماع الجماعي للاشخاص الذين لهم علاقة بالمنظومة الجديدة له هدف واحد يتمثل في التعرف على المشاكل التي تعترض النظام الحالي وتحليلها ومحاولة تحديد المتطلبات للمنظومة الجديدة .

وتتركز فكرة هذه الطريقة على مبدا الاجماع والشورى في تحديد المشاكل والمتطلبات والاهداف ويتم الاجتماع عادة في اماكن مجهزة لهذا الغرض . وفي هذه الجلسات يمكن اثارة وتوليد افكار لحل المشاكل التي تعترض النظام الحالي باستخدام طريقة اثارة الافكار Brainstorming .

7- العرض التجريبي Prototyping :

يعتبر العرض التجريبي Prototyping من اهم الطرق الفعالة والناجحة لجمع المتطلبات الوظيفية فهي طريقة لعرض منظومة تجريبية (لنسخة مبدئية من المنظومة المعلوماتية) ذات الوظائف المحدودة تبين قدرات المنظومة على اداء وظائفها . وهي اداة تواصل جيدة بين جميع الاطراف ذات العلاقة بالمنظومة وتبين بشكل مبدئي شكل المنظومة النهائي حيث نرى ان المتطلبات التي قد تم جمعها في المقابلة الشخصية تجسدت واقعيا في المنظومة .

8- حالة الاستخدام والسيناريو Use Case and Scenarios

تستخدم طريقة حالات الاستخدام Use Cases للتحديد والتعرف على المتطلبات الوظيفة للمنظومة المزمع تنفيذها . وتسمى هذه الوظائف (المعاملات) بحالات المستخدم .

وفي هذه الطريقة يقوم محلل النظم بالاستفادة من المقابلات الشخصية والملاحظة بالتعرف على الذين يقومون بأداء الوظائف ومن ثم رسم مخطط حالة الاستخدام UCD بمساعدة المستخدم .

ثم بعد ذلك يقوم المستخدمون بشرح كيفية اداء كل معاملة او وظيفة خاصة بالمنظومة على شكل وصف نصي لكل حالة استخدام على حده ويسمى هذا النص بالسيناريو Scenarios وتلحق بهذا المخطط وبالتالي الحصول على المتطلبات الوظيفية Functional Requirements .

ويفضل استخدام ورش العمل Workshops في شكل جلسات يحضرها المستخدم والزبون ومحلل النظم وفريق العمل الاخر . وهذه الطريقة تشجع عملية الاجماع حول المتطلبات ومن ميزاتها رخص تكاليف استعمالها .

وهي طريقة تفاعلية بين كل المستفيدين والمتأثرين بالمنظومة وبها يتم ادماج المستخدم في المنظومة منذ البداية .

9- جلسة توليد الافكار Brainstorming :

وتعتبر طريقة جلسة توليد واثارة افكار Brainstorming ذات اهمية حيث تكوين وجمع واقتباس المتطلبات من مجموعة من الاشخاص ذوي العلاقة بالمنظومة (Stakeholders) في مدة قصيرة وتوليد عدة افكار بخصوص المزايا المرتقبة من المنظومة .

ثم بعد ذلك يتم تصنيفها حسب الاهمية ويتم ايضا اكتشاف متطلبات مخفية ولم يتم التعرف عليها بالطرق الاخرى . وهذه الطريقة تشجع على التفكير المنظم واعطاء فرصة للابداع.

10- البحث والتطبيقات المشابهة Research & Similar Applications

تعتبر طريقة البحث Research من الطرق المهمة لتجهيز محلل النظم للعمل في المنظومات الجديدة التي ليس له سابق خبرة بها ولا بمصطلحاتها ومفرداتها . يتضمن البحث عادة المكتبة والانترنت ودراسة السوق ومراجعة تطبيقات مشابهة Similar Applications في الشركات واقسام الحاسوب بالجامعات والمعاهد التقنية .

ثانياً: تحليل المتطلبات Requirements Analysis

ان فكرة التحليل اساسا هي تقييم احتياجات المستخدم للوصول الى تعريف محدد للمتطلبات البرمجية المراد تجهيزها .

ونعني **بتحليل المتطلبات (Requirements Analysis):** عملية تفكيك وتجزئة المتطلبات العامة (العالية المستوى) وتحويلها الى متطلبات وظيفية تفصيلية (متدنية المستوى) حيث يتم استخدام ادوات مناسبة لتمثيلها و نمذجتها Modeling .

وكملاحظة في هذا السياق يجب ان اقول انه يجب تصنيف وترتيب هذه المتطلبات ليتم تنفيذها حسب الاهمية .
وهذا التصنيف للمتطلبات يمكن ان يأخذ شكل :

- متطلبات ضرورية .
- متطلبات مشروطة .
- متطلبات اختيارية .

- وعند تحليل المتطلبات اي نمذجتها نستخدم الادوات بناء على المنهجية التي يتم اختيارها لعملية التحليل للمنظومة.

وتوجد منهجيتان مشهورتان على نطاق واسع وهما المنهجية الهيكلية و المنهجية الشيئية .

المنهجيات المستخدمة في تحليل المتطلبات :

اولاً: المنهجية الهيكلية Structured Methodology
ثانياً : المنهجية الشيئية Object-Oriented Methodology
ثالثاً- منهجية أجل Agile Methodology
رابعاً: منهجية اطار عمل الحلول Ms-Solution Framework Methodology (MSF)

اولاً: المنهجية الهيكلية Structured Methodology:

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

هذه المنهجية تركز اكثر على وظائف المنظومة (المعالجة) حيث يتم استخدام الادوات المستخدمة في التحليل الهيكلي Structured Analysis لتمثيل ووصف نمذجة هذه الوظائف .

ومن هذه الادوات المستخدمة في المنهجية الهيكلية :

1- مخطط انسياب البيانات (DFD) Data Flow Diagram : وهو مخطط هيكلي رسومي يبين صورة لحركة انسياب البيانات داخل النظام بين مخازن البيانات والمعالجة والكيانات الخارجية .إن الشكل الأساسي للـ DFD يعرف باسم (Flow graph أو Bubble chart) الرسم البياني لتدفق البيانات أو مخطط الفقاعة.
يستخدم لغرضين : اولا ، يعطينا مؤشرات حول كيفية انتقال البيانات خلال النظام. **ثانيا،** يوضح الدوال التي تقوم بتناقل البيانات او تحويل البيانات.

وهناك نوعان من مخطط DFD :

الأول : يستخدم لتمثيل النظام كما هو على ارض الواقع مشتملا على المعلومات والمواد ويسمى المخطط الانسيابي المادي Physical DFD وهو وسيلة تفاهم بين المستخدم ومحلل النظم حتى يفهم محلل النظم الدورة المستندية وتحرك المواد عبر النظام ليتسنى له فهم عمل النظام وبالتالي تزال عملية الغموض لديه .

الثاني : يستخدم ليكون اساسا لمرحلة التصميم ويسمى المخطط الانسيابي المعنوي Logical DFD ويشتمل على انسياب البيانات فقط محذوفا منه انتقال المواد .

وعملية رسم مخطط انسياب البيانات تبدأ برسم مخطط عام وعالي المستوى (وبدون تفاصيل) يسمى المخطط البيئي Context Diagram يبين تفاعل المنظومة مع الكينونات الخارجية (البيئة الخارجية للمنظومة).

ثم يقوم محلل النظم بتفصيل المخطط البيئي ليشمل على مخططات تفصيلية تسمى DFD level 1 و DFD level 2 وهكذا تتم التجزئة تباعا . ومن مزايا هذه التجزئة الهيكلية ازدياد سهولة فهم النظام او المنظومة من خلال المناقشة حولها بين جميع الاطراف ذات العلاقة .

ويعتبر مخطط انسياب البيانات بسيط التكوين الا انه اداة قوية لتمثيل (نمذجة) وظائف النظام (او المنظومة).
ويعتقد العديد من محللي النظم ان هذا المخطط هو كل ما يحتاجونه لمعرفة التحليل الهيكلي والحق انه بدون استخدام ادوات اخرى مساعدة لعملية التحليل تصبح هذه الاداة غير ذات جدوى في حد ذاتها .

لهذا يجب جمع هذه الاداة مع ادوات اخرى مثل قاموس البيانات Data Dictionary والانجليزية الهيكلية Structured English ومخطط الكائنات العلائقية Entity Relationship Diagram وشجرة القرار Decision Tree وغيرها.

2- قاموس البيانات Data Dictionary

يمكن تعريف اداة قاموس البيانات Data Dictionary بأنه **قائمة او مستودع لكل عناصرالبيانات (data objects) الخاصة بالمنظومة او وصف لمخازن البيانات وانسيابها والموجودة في مخطط DFD .**

ونظرا لأهمية هذه الاداة لتحديد متطلبات البيانات Data Requirements الخاصة بالمستخدم فأنها يجب ان تكون دقيقة وواضحة لكي تساعد على التفاهم المشترك بين كل من المستخدم ومحلل النظم والمصمم والمبرمج وابعاد اي لبس حول مدخلات ومخرجات المنظومة .
وتعتبر هذه الاداة من الاساسيات التي سيستفاد منها في تصميم قاعدة البيانات في مرحلة التصميم .

3- الانجليزية الهيكلية Structure English

تعتبر الانجليزية الهيكلية Structure English اداة تحليل نصية تستخدم جزء محدود من اللغة الانجليزية لتوضيح الخطوات المراد اداؤها لوظائف (عمليات) منظومة معلومات .

وتستخدم افعال مثل : read ,write , print , sort , add, subtract.....etc وعناصر بيانات مثل Customer-no , item-no , price وتركيبات البرمجة الهيكلية مثل While-Do أو If-then-else وعادة ما تستخدم الانجليزية الهيكلية بعد رسم مخطط انسياب البيانات لتوضيح كل معالجة Processes في DFD .
وتعتبر الانجليزية الهيكلية اداة تواصل مع المستخدم لأزالة الغموض حول كل عملية . وهي اساس كتابة شبه الشفرة Pseudo code في مرحلة التصميم .

4- جدول القرار Decision Table

يعتبر جول القرار Decision Table أداة تحليل في شكل جدول (مصفوفة) يبين الافعال Actions المحتملة بناءا على شروط Conditions معينة .

- وتساعد هذه الأداة (Decision Table) في توضيح القرارات التي تستخدم في الحالات المعقدة .

ويستخدم محلل النظم هذه الاداة لتبيين سياسة عمل النظام مثل :

- سياسة التخفيضات في نظم المبيعات .
- سياسة تنسيب طلبة الثانوية للجامعات والمعاهد .
- سياسة تقييم المستوى الأكاديمي للطالب بناء على أدائه في الامتحانات والواجبات المدرسية .

5- شجرة القرار Decision Tree

شجرة القرار Decision Tree أداة تحليل على شكل شجرة تبين الحالات (الشروط) Conditions والافعال Actions ذات العلاقة بهذه الشروط وهي تبين سياسة عمل النظام . وهي تشبه الى حد بعيد عمل جدول القرار الا ان تفرعاتها يجب ان تكون محدودة وهي بديل لجدول القرار في النظم الغير معقدة .

6- مخطط الكائنات العلائقية Entity Relationship Diagram (ERD)

يعتبر مخطط الكائنات العلائقية ERD (Entity Relationship Diagram) بداية صحيحة من قبل محلل النظم لفهم متطلبات البيانات Data Requirements الخاصة بالمنظومة تحت الاعداد .ويتكون مخطط الكائنات العلائقية من اشكال هندسية تشبه المخطط الانسيابي تبين الكائنات Entities والعلاقة Relationships بين هذه الكائنات وايضا الخصائص Attributes (عناصر البيانات) لكل كائن أو علاقة . ويعتبر هذا المخطط أيضا أساسا لتصميم قاعدة البيانات .

- وهو أداة تواصل بين محلل النظم والمستخدم لفهم وتدوين عناصر البيانات المستخدمة وانتمائها للكيانات .

- وفي الوقت الحاضر وبأستخدام العرض التجريبي Prototyping أصبح الحصول على عناصر البيانات عملا ميسرا .

ثانياً : المنهجية الشيئية Object-Oriented Methodology:

المنهجية الشيئية (Object Oriented Methodology) : وتقسم هذه المنهجية النظام الى كائنات لها وظائف وخصائص شبيهة بالكائنات التي نتعامل معها في حياتنا اليومية .وتستخدم ادوات شيئية مثل مخطط الحالة UCD ومخطط الفصيلة class diagram.

يمثل التحليل الشيئي Object-Oriented Analysis تغيرا دراميا مقارنة بالتحليل الهيكلي حيث يتم التعامل مع النظام على أساس أنه مجموعة من الكائنات (المادية والمعنوية) .

أما التحليل الهيكلي فيعتبر البيانات منفصلة عن العمليات التي تحصل على هذه البيانات . بمعنى أن البيانات ليس لها أهمية بالغة في التحليل الهيكلي Structured Analysis . حيث يتم تقسيم المنظومة الى وظائف رئيسية وتجزأ هذه الوظائف الى وظائف فرعية وهكذا .

أما غرض التحليل الشيئي OOA فهو ربط البيانات والعمليات في مكان واحد وهو الكائن Object أو الفصيلة Class علما بأن الكائن حالة خاصة من الفصيلة . وفي هذا النوع من التحليل يمكن تعريف الفصائل والعلاقات بين هذه الفصائل والنظام **. ويجب اتباع الخطوات التالية للحصول على هذا التحليل :**

1- يبدأ محلل النظم في الحصول على بعض المتطلبات من الزبون والتي من اهمها المتطلبات الوظيفية ويستخدم مخطط استخدام الحالة Use Case Diagram (UCD) كأداة هامة لتحديد هذه المتطلبات . ويستخدم السيناريو Scenario النصي لوصف كل حالة استخدام .

2- التعرف على الفصائل Classes الخاصة بالنظام (أيضا الخصائص Attributes والطرق Methods لكل فصيلة).

3- رسم مخطط الفصائل Class Diagram ويتم ذلك اما يدويا او بأستخدام احدى ادوات Case .

4- رسم مخطط السلسلة Sequence Diagram ليعبر عن وصف تفصيلي لكل حالة استخدام .

الادوات المستخدمة في المنهجية الشيئية:

وفي هذا البند نقوم بشرح **ثلاث مخططات شيئية تستخدم في مرحلة التحليل :**

1- مخطط حالة الاستخدام Use Case Diagram (UCD) :

ويعتبر مخطط حالة الاستخدام UCD أداة تحليل شيئية مهمة لتوضيح المتطلبات الوظيفية للنظام . ويتكون من اشكال هندسية تعبر عن حالة الاستخدام Use Case وهي المعاملة او الوظيفية التي يؤديها النظام والممثل او الفاعل Actor وهو الذي يقوم بأداء هذه المعاملة (حالة الاستخدام) .

2- مخطط الفصيلة Class Diagram

مخطط الفصيلة Class Diagram هو أداة تحليل شيئي رسومي يبين هيكلية الكائنات الساكنة للنظام . وهذه الهيكلية تبين فصائل الكائنات والعلاقة بين هذه الكائنات . ويبين مخطط الفصيلة الحركة الساكنة للمنظومة الشيئية .

3- مخطط السلسلة Sequence Diagram

يعتبر مخطط السلسلة Sequence Diagram أداة تحليل شيئية تبين الكائنات والتواصل بين هذه الكائنات بأستخدام الرسائل المتبادلة بينهم عند تنفيذ حالة الاستخدام . ويبين مخطط السلسلة الحركة المتحركة (الديناميكية) للمنظومة الشيئية .

ثالثاً: منهجية أجل Agile Methodology :

تعتمد منهجية Agile على اختيار افضل الادوات والطرق المناسبة لأداء المهام الخاصة بالمنظومة .
وتعتمد على استعمال عدة منهجيات في المشروع الواحد مثل استخدام الاداة DFD مع الاداة UCD في نفس المشروع اي المنهجية الهيكلية والمنهجية الشيئية .
وبعبارة اخرى يمكن استخدام منهجيات وطرق مختلفة والهدف هو تسريع عملية اعداد المنظومة وهذه المنهجية تفاضل بين الانتاجية والجودة .

رابعاً: منهجية اطار عمل الحلول Ms-Solution Framework Methodology (MSF)

في هذه المنهجية MSF التي تستخدمها شركة Microsoft في اعداد منظوماتها يستطيع محلل النظم ان يصمم عدة نماذج Models منها على سبيل المثال لا الحصر :

- نموذج المعالجة Process Model
- نموذج البيانات Data Model
- نموذج ادارة المخاطرة Risk Management Model

وكل نموذج يساهم في تحليل وتصميم المنظومة تحت الاعداد .
وتستخدم منهجية MSF ادوات OOA مثل UCD وادوات CASE .
ونعيد هنا القول أنه : للحصول على منظومة ذات جودة يجب ادماج والحاق المستخدم جنبا الى جنب مع معد المنظومة في هذه المرحلة المهمة الا وهي التحليل .

.....

ثالثاً: المواصفاتSpecifications او توصيف المتطلبات (Requirements (Specification

في هذا النشاط يتم كتابة وتجهيز وثيقة هامة من وثائق المنظومة في نهاية مرحلة التحليل وتسمى وثيقة مواصفات المتطلبات Requirement Specification Document والتي تشتمل على كل متطلبات المنظومة المقترحة . وتلعب وثيقة المتطلبات دورا مهما في دورة حياة المنظومة لانها تقودنا الى مراحل التصميم والتنفيذ وتعتبر اساسا للتعاقد بين الزبون ومعد المنظومة .

ولقد ثبت ان حوالي 85% من اخطاء البرمجيات كان مرده الى المتطلبات ومشاكلها والتي هي :

- 49% افتراضات تتعلق بمتطلبات غير صحيحة .
- 29% متطلبات محذوفة (غير معلنة).
- 13% متطلبات متضاربة .
- 5% متطلبات غامضة وغير واضحة .

وقد ثبت من الاحصائيات بسبب مشكلة تحديد المتطلبات ايضا ان حوالي 30% من المشاريع يتم الغاؤها قبل ان تنتهي وان حوالي 50% من المشاريع تكلف الضعف من التقديرات الاولية .

خصائص مواصفات المتطلبات البرمجية :

تناول العالم Boehm (1984) خصائص مواصفات المتطلبات البرمجية الجيدة فيما يلي :

- كاملة Complete
- دقيقة وقابلة للقياس Measurable
- صحيحة Correct
- واضحة Unambiguous
- قابلة للاختبار Testable
- متناغمة (غير متضاربة) Consistent
- موجزة ومحددة Concise
- قابلة للتحقق Verifiable
- سهولة التعديل Changeable
- بدون علاقة بالتصميم Design-free

وقبل لن نشرع في هذه الوثيقة نشرح معنى واهداف التوثيق :

التوثيق Documentation

يعتبر التوثيق عنصرا مهما في اعداد البرمجيات واستمرار عملها بعد الاعداد . ويمكن تعريف التوثيق بأنه مجموعة اوصاف نصية ورسومية وشروح للمنتوج البرمجي (المنظومة البرمجية).

وقد يشمل التوثيق ما يلي :

- سرد أو نص Narratives
- مخططات Charts
- جداول Tables
- الصوت Voice
- قصاصات فيديو Video clips
- صور متحركة Animations
- تعليقات في البرنامج Comments in program

ويمكن ان تكون الوثيقة على شكل ورقة او تكون مخزنة في الحاسوب .

<u>اهداف ووظائف التوثيق :</u>

يؤدي التوثيق الوظائف التالية :
1- مرجع تاريخي .
2- مرجع ارشادي وتوضيحي .
3- متابعة جودة المنتوج البرمجي .
4- التواصل بين مراحل اعداد المنظومة .
5- التواصل بين المهام داخل المرحلة الواحدة .
6- اتفاق بين المستخدم او الزبون ومعد المنظومة .

<u>استخدام التوثيق :</u>

يستخدم التوثيق المعد بصورة عامة من قبل :

- الادارة لغرض المراجعة .
- القائمين على صيانة البرنامج .
- فريق التفتيش .
- فريق المراجعة غير الرسمية من قبل زملاء العمل .
- موظفي التحقق والمصادقة .

نستعرض الان محتويات وثيقة مواصفات المتطلبات التي يجب ان يعدها محلل النظم في نهاية مرحلة التحليل ولتتم مراجعتها من قبل الادارة لاتخاذ احد القرارات الاتية :

- الاستمرار في المشروع وتنفيذ المرحلة التالية وهي التصميم .
- وقف استمرار المشروع .
- اجراء بعض التعديلات ثم الاستمرار في المشروع .

<u>وثيقة تحديد المتطلبات Requirement Specification Document :</u>

تعريف وثيقة تحديد المتطلبات : هي وثيقة يتم اعدادها في نهاية مرحلة التحليل تتضمن وظائف المنظومة المراد تنفيذها وخصائص الجودة المتعلقة بها . وهذه الوثيقة يجب ان تكون صحيحة ودقيقة وكاملة ومتناسقة وقابلة للقياس والاختبار .

<u>بنود وثيقة مواصفات المتطلبات :</u>

a) Introduction (المقدمة)

1- Overall description وصف عام :

- problem definition تعريف المسألة
- objectives of the system الاهداف
- interfaces of the system البينيات (الواجهات)
- scope of the system حدود المنظومة
- constraints of the system القيود

2- الوصف الوظيفي Functional description :

- list of system functions قائمة الوظائف
- Narrative for each function وصف كل وظيفة

b) Data/ Information description وصف البيانات :

- Entity relationship diagram مخطط ERD
- Data dictionary قاموس البيانات

c) Process/ logic description وصف المعالجة والمنطق:

- (DFD) Data Flow Diagram مخطط انسياب البيانات
- (UCD) Use Case Diagram مخطط استخدام الحالة
- Structured English الانجليزية الهيكلية
- Decision tree شجرة القرار
- Decision table جدول القرار

d) Performance Requirements متطلبات الاداء:

- Response Time زمن الاستجابة
- Memory الذاكرة

e) Validation / Acceptance Criteria معيار التحقيق والقبول:

- Types of test انواع الاختبارات
- Quality attributes required خصائص الجودة المطلوبة
- Deliverables البنود القابلة للتسليم

f) Solution Strategyاستراتيجية الحل:

- on-line/off-line
- Graphic / text رسومات أم نص
- database / files قاعدة بيانات أم ملفات

ملاحظات عن وثيقة مواصفات المتطلبات :

1- تصف مشاكل وليس حلولا .
2- هي منتوج وليس عملية معالجة .
3- وثيقة بين الزبون والمحلل وتستخدم فيما بعد في التصميم .
4- تقوم بتحويل الاحتياجات الى متطلبات .
5- يجب مراجعتها من قبل المستخدم ومعد المنظومة .
6- تبين ما هو المتوقع من المنظومة وليس كيفية العمل .

.....

رابعا :اعتماد المتطلبات Requirement Validation

يعتبر هذا النشاط مهما للغاية يهدف في النهاية الى التأكد Confirmation من ان مواصفات المتطلبات التي تم تجهيزها في البند السابق تتوافق مع المعايير Standards في كتابة وثيقة المتطلبات وجاهزة لان تكون أساسا لعملية التصميم في المرحلة اللاحقة لمرحلة التحليل .

ويستخدم في هذا التحقق والاختبار عدة طرق للفحص والمراجعة والتأكد والتي منها :

1- المراجعة النهائية Review او مراجعة البرمجيات (Software Review) :

تعتبر مراجعة البرمجيات مصفيا ضروريا ومهما لعملية إعداد البرمجيات حيث تتم مراجعة توثيق نهاية كل مرحلة : التحليل والتصميم والبرمجة . والهدف هو كشف الاخطاء ،وهي لضمان الجودة عن طريق فحص التوثيق للمنظومة تحت الإعداد لغرض إيجاد أي أخطاء فيه . وفي المراجعة التي تتم بحضور الادارة يتم تجهيز تقرير يتم إعداده في نهاية كل مرحلة لاتخاذ قرار بخصوص المنظومة تحت الإعداد.

ونتيجة المراجعة يتم اتخاذ أحد القرارات التالية :
1- الاستمرار في المرحلة التي تليها .
2- التوقف عن المشروع بالكامل .
3- إجراء تغييرات على التوثيق ثم الاستمرار في المرحلة التي تليها.

والمراجعة بصفة عامة تعتبر طريقة تأكيد ومصادقة تحضرها الإدارة ويتم فيها :
- تحديد التحسينات الضرورية للمنتوج البرمجي الذي تم إعداده من قبل فريق إعداد المنظومة.
- تحديد وتأكيد الأجزاء التي لا حاجة ولا ضرورة لتحسينها.
- جعل العمل التقني أكثر ملائمة ويتوافق مع ما هو متوقع .

2- الفحص او التفتيش Inspection او فحص الشفرة او التفتيش الرسمي (Inspection (Formal/ Code Inspection) :

إلى جانب المراجعة على التوثيق في نهاية كل مرحلة يتم الفحص او التفتيش على الشفرة .
فالفحص هو عملية لكشف الأخطاء في شفرة جزء برمجي وليس تصحيحها .

ومن أمثلة هذه الأخطاء : الأخطاء المنطقية في الشفرة .

ويتم فحص الشفرة في جلسة تعقد بحضور معد الجزء البرمجي وفريق الفحص (التفتيش) المكون من المنسق والشخص المراجع او المختبر وقارئ للجزء البرمجي . وتستغرق الجلسة الواحدة زهاء الساعتين .

ويتم توفير الآتي لهذه الجلسة :
- الشفرة المراد فحصها .
- المعايير المستخدمة في إعداد المنظومة ليلم فريق الفحص بها وباستخدامها في عملية التفتيش .
- قائمة بالاخطاء المتوقعة (Check list errors) يتم تجهيزها بواسطة أخصائيين ذوي خبرة ويتم استخلاصها من منظومات سابقة .

ومن أمثلة بنود هذه القائمة ما يلي :
1- هل تم إعطاء قيم ابتدائية للمتغيرات المستخدمة في الجزء البرمجي (الشفرة)؟
2- هل كل حلقة (loop) لها نهاية (end)؟

3- هل عدد البارمترات المستخدمة في الإجراء (procedure) أو الدالة (function)صحيح ؟
4- هل التداخل بين جمل التحكم والحلقات صحيح ؟
5- هل جمل التحكم مكتوبة بشكل صحيح وتوافق لغة البرمجة ؟
6- هل جمل الإدخال والإخراج صحيحة ؟
7- هل العمليات الحسابية في الجزء البرمجي تمت بطريقة صحيحة ؟
-هذا النوع من المراجعة قد يكلف الوقت والجهد، ولكنه يقلل من تكلفة الاختبارات بعد تنفيذ المنظومة .
وكما ذكرنا سابقا فان مهمة فريق الفحص هي تحديد واكتشاف الاخطاء وليس تصحيحها فالذي يقوم بتصحيح هذه الاخطاء هو نفس الشخص الذي أعد هذا الجزء البرمجي .
وتتم جلسة أخرى للتاكد من عدم اكتشاف أخطاء اخرى ناتجة عن تصحيح الأخطاء المكتشفة سابقا.
لهذ نقول بأن عملية الفحص عملية تكرارية .
ملاحظة : رغم أن المراجعة والفحص لهما نفس الهدف، وهو إنتاج منتوج ذي جودة عالية إلا أنهما يختلفان في طريقة العمل .

	المراجعة (Software Review)	الفحص (Inspection)
1	**تتم المراجعة عادة بصورة عامة**	**يتم بشكل اكثر تفصيلا وتكرارا**
2	**يتم توجيه تقريرها إلى الإدارة لأخذ القرارات اللازمة حياله.**	**يتم توجيه التقرير الناتج عنه إلى معد الجزء البرمجي للتصحيح**

3- المراجعة السريعة (التصفح) Walkthrough :

هو عبارة عن طريقة أخرى للمراجعة الغير رسمية وتهدف أصلا للحصول على الجودة في المنتوج البرمجي، شأنها شأن المراجعة والفحص ، ولكنها تختلف عنهما بأنها عملية لمراجعة المنتوج البرمجي بواسطة الزملاء في فريق العمل (Peers) وذلك بهدف كشف الأخطاء (بدون تصحيحها) بطريقة غير منتظمة أي لا ضرورة من معرفة مدير المشروع بها ولا ضرورة لتوثيقها بل تاتي عفوية بين أعضاء فريق المنظومة فيما بينهم .

مثال : قد يعد أحد المحللين مخطط انسياب البيانات (DFD) وللتأكد من صحته قد يتصفحه ويراجعه محلل أخر من نفس الفريق لتحديد الأخطاء في هذا المخطط دون التعرض لطريقة تصحيحه .

والجدول التالي يعرض الفرق بين التصفح (Walkthrough) و الفحص (Inspection)

	التصفح (Walkthrough)	الفحص (Inspection)
1	**يزيد من تواصل اعضاء فريق المشروع** (تأتي عفوية بين أعضاء فريق المنظومة فيما بينهم)	**يستخدم قائمة أخطاء متوقعة**
2	**جلسات غير رسمية**	**تستخدم جلسات رسمية**
3	**جلسات غير منتظمة**	**أسلوب منتظم للبنود**
4	**يتم بواسطة زملاء فريق اعداد المنظومة فهو أداة تعليمية ممتازة لأعضاء الفريق الجدد .**	**يتم بواسطة عناصر مؤهلة**

4- التحقق والتصديق (Verification &Validation)

- التحقق Verification :

يستخدم التحقق Verification للتأكد من صحة توثيق المنتوج البرمجي في مراحل التحليل والتصميم والبرمجة (ويتم على الورق بدون جهاز الحاسوب)، والتأكد من أن أي وثيقة في نهاية مرحلة ما مقتنبسة من المرحلة السابقة لها .

فمثلا يمكن التاكد من ان شبة الشفرة (Pseudocode) التي يتم إعدادها في مرحلة التصميم مقبسة من المرحلة السابقة لها وهي مرحلة التحليل وبالتحديد مقتبسة من الانجليزية الهيكلية Structure English وهكذا.ويسمى التحقق بالكشف الساكن او الاختبار الساكن (Static Testing or Static Checking) ويعني الكشف عن الاخطاء على التوثيق (وثيقة المتطلبات أو وثيقة التصميم مثلا).

- اما التصديق و الاعتماد Validation: فهو اختبار يتم عند تنفيذ المنظومة Implementation، ويتم من قبل فريق عمل مستقل ،حيث يجري اختبار يسمى (Validation test) ويسمى أيضا (Dynamic Testing) لأنه يجري على جهاز الحاسوب (بدل التوثيق) للتأكد بان المتطلبات الوظيفية والمدونة في وثيقة المتطلبات قد تم تنفيذها بالكامل وبطريقة صحيحة على جهاز الحاسوب.
والهدف الوصول الى برمجية ذات جودة عالية وتلبي متطلبات المستخدم بالكامل وبطريقة صحيحة ومرضية .

ونلخص الفرق بين الاعتماد والتحقق في الآتي :

	التصديق و الاعتماد Validation	التحقق Verification
1	هو عملية تقييم المنظومة في نهاية المشروع أي في نهاية مرحلة التنفيذ والتأكد من ان المنظومة تؤدي وظائفها طبقا للمتطلبات الواردة في وثقية تحديد المتطلبات.	هو عملية تحديد ما إذا كانت مواصفات مرحلة معينة (مثلا مرحلة التصميم) تحقق مواصفات مرحلة سابقة لها (مثلا مرحلة التحليل) أي مقتبسة منها .
2	تستخدم آلية اختبار الاعتماد على جهاز الحاسوب في عملية الاختبار اي اختبار ديناميكي Dynamic Testing	يتم على الورق بدون جهاز الحاسوب اي اختبار ساكن Static Testing
3	يتم من قبل جهة او فرق عمل مستقل	يتم من قبل جهة او فرق عمل مستقل

ملاحظة : ترتيب التسلسل الزمني للطرق السابقة هو كالتالي :
1- **التصفح (Walkthrough)**
2- الفحص (Inspection)
3- التحقق Verification
4- التصديق و الاعتماد Validation
5- مراجعة البرمجيات (Software Review)

.....

خامسا : ادارة المتطلبات Requirement Management

ان ادارة المتطلبات : هي دراسة واستخدام الاجراءات والسياسات والعمليات التي تحكم كيفية التعامل مع التغير في المتطلبات وبمعنى أدق :
1- كيفية تقديم مستند طلب تغيير Change Request .
2- كيفية تحليل هذا الطلب ومعرفة تأثيره على التكاليف والجدول الزمني وحدود المشروع .
3- كيفية المصادقة والموافقة على اجراء التغيير او رفضه .
4- كيفية تنفيذ التغيير بعد اخذ الموافقة عليه .

ويهتم هذا النشاط في هندسة المتطلبات ايضا بالتخطيط Planning والمتابعة Controlling لنشاطات جميع المتطلبات والتحليل والمواصفات والتحقق .

ومن المهام الادارية الخاصة بأدارة المتطلبات Requirement Management ما يلي :

- ادارة النسخ الخاصة بالمنظومة والتغيير Managing versions and change
- تخزين خصائص المتطلبات Storing requirement Attributes
- التواصل مع الذين لهم علاقة بالمنظومة Stakeholders

وتوجد برمجيات جاهزة لادارة المتطلبات Automated Requirement Management من قبل شركات متخصصة . ومن ابرز هذه البرمجيات :

- Doors
- Requisite Pro
- RTM Workshop
- Caliber-RM

ونظرا لاهمية المتطلبات والتعامل معها فقد أنشأت بعض الشركات ادارة تعهد اليها بمتابعة التغييرات التي تحدث في المتطلبات ومتابعة اصدار النسخ والاصدارات لهذه البرمجيات والتي تسمى ادارة مكونات البرمجيات .

.....