

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

الفصل الأول

المقدمة

1-1 تمهيد :

لقد دخل الحاسب الآلي في مجالات الحياة وظهر أثره في حل العديد من المشاكل التي تعاني منها الشعوب والأفراد. ومن تلك المشاكل القدرة على تخزين كم هائل من البيانات وما يترتب عليه من أسلوب حفظ واسترجاع وفهرسة تلك البيانات والتي تتطلب جهد وتكلفة تحتاج إلي وقت طويل وعلى سبيل المثال أسلوب حفظ بيانات , ولقد كان الاعتماد حتى الآن على العنصر البشري فهو الذي يقع عليه العبء كله تقريباً وتتعدد مسؤوليته ابتداء من وضع استقبال البيانات ثم فهرستها لحفظها ناهيك عن العوامل الطبيعية التي تتعرض لها أوساط الحفظ الورقية وكما نرى نجد أن هذه العملية الروتينية تستغرق وقت طويل ويعتبر الوقت هو العامل الأساسي والحاسم لجميع الأعمال في هذا العصر.

من ذلك خسارة جزء من السوق بفقدان عدد من الزبائن. فالأداء معنى بتحديد مدى فعالية و إنتاجية النظام للمستخدم. يعرف الأداء كما يلي " إلى أي مدى يستطيع النظام موافقة الوظائف المحددة في متطلبات التصميم مثل السرعة و الدقة و استخدام الذاكرة" (IEEE 1991) و نخلص إلى إن للأداء اتجاهين اثنين هامين هما : التوسعية و تعنى إمكانية مواكبة التغيرات المستقبلية في النظام، و الاستجابة . استجابة النظام يعبر عنها بزمن الاستجابة أو بإنتاجية الاستجابة و يتم قياسهما بمدى و سرعة استجابة النظام للأحداث أو بعدد الأحداث التي يمكن للنظام معالجتها في إطار زمني محدد. هناك تقارير تؤكد إن 25% من مشاكل الأداء في أنظمة الحاسوب مرتبطة بعمارية و تصميم النظام يمكن اكتشافها أثناء بنا النظام (أثناء دورة تصميم النظام) برناردو و هيلستون و آخرون (Bernardo, Hillston et al. 2007) يعرف هندسة أداء البرمجيات بأنها "طريقة قياس كمي لأداء البرمجيات تتسم بالمنهجية و التنظيم مما يساعد على بناء نظام يلبي متطلبات الأداء الموضوعة مسبقا ". و يقوم أسلوب هندسة أداء البرمجيات على إجراءات وموجهات. الإجراءات يتم عن طريقها تجميع مواصفات الأداء ، و الموجهات

تساعد على تحديد نوع التقييم الواجب تطبيقه في كل مرحلة من مراحل بناء النظام. و بما أن هذه الطريقة تقوم على بناء نماذج الأداء من نماذج البرنامج ،فان النماذج الناتجة يمكن تقييمها و مقارنتها مع متطلبات الأداء المحددة مسبقا.

في منحى اخر و من اجل تجاوز صعوبات بناء الأنظمة المعقدة انتشر و بصورة واسعة أسلوب البرمجة أو التطوير المدفوع بالنموذج(OMG 2007). هذا الأسلوب لعب دورا مهما في استخدام المخرجات الكمية بصورة فاعلة في إعادة ضبط معمارية و تصميم النظام ليلبي متطلبات الأداء. و جدير بالذكر إن هذا الأسلوب في التطوير يتمحور حول النموذج (التطوير المتمحور حول النموذج) كعنصر أولى يستخدم في تطوير الأداء و بالتالي يتجنب إخفاقات الأسلوب الأخر المتمحور حول الشفرة (التطوير المتمحور حول الشفرة). عليه، يمكن الاستفادة من التطوير المتمحور حول النموذج في التوقع أو التحقق من استيفاء المتطلبات و ذلك ببناء نماذج للأداء من نماذج تطوير النظام و مقارنة النموذج الناتج بتلك المتطلبات.

مما سبق يتضح لنا جليا أهمية التوقع لأداء البرمجيات أثناء مراحل تطوير النظام لتجنب أوجه القصور التي تظهر عند التنفيذ و بالتالي ضرورة إتباع الأسلوب المنهجي و المنظم لضمان اعلي مستوى من الدقة ،كما تعرضنا إلى لمحة عامة حول التطوير المتمحور حول النموذج و أهميته في جعل مهمة التطوير أكثر سهولة و يسر و بالتالي إمكانية التوقع بالأداء بأقل جهد. بقية الورقة مرتبة كما يلي : الفقرة 2 تتناول موضوع هندسة أداء البرمجيات،مميزاتها،و آلية عملها.الفقرة 3 تتعرض لموضوع تطوير النظم المدفوع بالنماذج و كيفية توظيفه في تحليل و توقع أداء النظم. الفقرة الأخيرة تتناول توقع الأداء في نظم هندسة البرمجيات المؤسسة على المكونات ، المشاكل ، الطرق ، المقارنة و التحليل للمقارنة.

1-2 مشكلة الدراسة:

تتمثل مشكلة الدراسة في استخدام هندسة البرمجيات في تقييم الكفاءة و الأداء لدوال قواعد البيانات النشطة Active Database, في الوصول للبيانات في لغة أوراكل . في منحى آخر و من اجل تجاوز صعوبات بناء الأنظمة المعقدة انتشر و بصورة واسعة أسلوب البرمجة أو التطوير المدفوع بالنموذج . هذا الأسلوب لعب دورا مهما في استخدام المخرجات الكمية بصورة فاعلة في إعادة ضبط معمارية و تصميم النظام ليلبي متطلبات الأداء. و جدير بالذكر إن هذا الأسلوب في التطوير يتمحور حول النموذج (التطوير المتمحور حول النموذج) كعنصر أولى يستخدم في تطوير الأداء و بالتالي يتجنب إخفاقات الأسلوب الأخر المتمحور حول الشفرة (التطوير المتمحور حول الشفرة). عليه، يمكن الاستفادة من التطوير المتمحور حول النموذج في التوقع أو التحقق من استيفاء المتطلبات و ذلك ببناء نماذج للأداء من نماذج تطوير النظام و مقارنة النموذج الناتج بتلك المتطلبات.

يعتبر أداء البرمجيات احد أهم العوامل في تحديد مدى جودة المنتج البرمجي ، و ابعد من ذلك، يعتبر نجاح أو فشل المنظمة رهين بالوفاء بالأداء المطلوب للبرمجيات(Williams L.G. 2002). و الفشل في هذا يتمثل في الإهدار المالي بزيادة الصرف على المعدات الصلبة و على تطوير و إعادة تطوير النظام ، و تجاوز الوقت المخطط لتنفيذ النظام.

1-3 أهمية الدراسة :

تستمد الدراسة أهميتها من أهمية المتغيرات التي طرأت على عمل دوال قواعد البيانات النشطة الموجودة في لغة أوراكل و استخدام الطرق المختلفة في هندسة البرمجيات لقياس و تقييم الأداء و كفاءة المخرجات.

1-4 أهداف الدراسة :

تسعى الدراسة إلى تحقيق الأهداف الآتية :

1- المساهمة الجادة في كيفية استخدام هندسة البرمجيات في تقييم الأداء في العمل و قياس المخرجات في تطبيقات دوال قواعد البيانات النشطة في لغة أوراكل .

2- بيان مفهوم الفرق في عمل الدوال و قياس المخرجات باعتبارها المعطيات الأساسية لتقييم الأداء و أثرها في تقدير درجة الكفاءة.

1- 5 منهج الدراسة :

اعتمدت الدراسة المنهج الوصفي التحليلي و التطبيقي إذ مثل التحليل المنطقي أساس الجانب النظري من الدراسة في حين كانت خوارزميات تأمين حماية البيانات أساس الجانب التطبيقي منها .

1- 6 فرضيات الدراسة :

- 1- تأمين و حماية البيانات باسم المستخدم و كلمة المرور يعتبر جزء من جودة سرية النظام.
- 2- نظام قواعد البيانات النشطة يتبع السياسات العالمية (NIST , ISO) في تأمين و حماية البيانات.
- 3- النسخ الاحتياطي من أهم مؤشرات جودة تأمين و حماية البيانات و الوصول إليها عند الحاجة .

1- 7 حدود الدراسة:

أ- التطبيق على قواعد البيانات النشطة .

1- 8 المراجع و المصادر:

تتنوع المصادر و المراجع في هذا البحث ما بين الكتب العلمية و الدراسات السابقة و المنشورات

و الدوريات العلمية و المواقع العلمية على شبكة الانترنت و التلقي الشفوي من ذوي العلم

و المعرفة ، والتي تدور جميعها حول موضوع البحث.

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

الفصل الثاني

الإطار النظري

المبحث الأول : نظم إدارة قواعد البيانات :

بدأت معظم الشركات التجارية خاصة في البلدان المتقدمة تخزين وحفظ ملفاتها على الكمبيوتر منذ عام 1960 وقام علماء وخبراء الحاسبات في تطوير نظريات وأساليب لتطوير كيفية إعادة استخدام وزيادة كفاءة استخدام هذه الملفات المخزنة داخل الحاسب و التي تسمى ملفات آلية وبالتالي ظهرت واستحدثت مصطلحات كمبيوترية تعبر عن استخدامات هذه الملفات وأيضاً استخدمت طرق لمعالجة هذه الملفات آلية. و تخزن الملفات الكبيرة في قاعدة كبيرة وتحتوي على جميع البيانات المسجلة و التي يمكن استخدامها في زمن لاحق هذه القاعدة تسمى قاعدة بيانات Database . ولأن قواعد البيانات مهمة ومؤثرة جداً في جميع المجالات و الأنشطة الرئيسية . لذلك يلزم وجود نظم معينة لتنظيم وإدارة البيانات المخزنة . وهو ما يطلق عليه نظم إدارة قواعد البيانات Database Management Systems.

وتعرف نظم إدارة قواعد البيانات : بأنها هي البرامج التي تساعد على إنشاء قواعد البيانات و التعامل معها وتشغيل البيانات المخزنة بها . فمثلاً بعد إضافة عملاء جدد لدليل التليفون فإنك تحتاج إلى ترتيب الأسماء من جديد أبجدياً أو ترتيب عناوينهم . بمعنى آخر تتيح للمستخدم إضافة بيانات جديدة وتحديث البيانات وطباعة التقارير على الشكل التي تريده مثل القوائم و الجداول و النماذج و التقارير .

وقد تكون قاعدة البيانات كبيرة جداً وتحتوي على آلاف من البلايين من الكلمات وهي أكبر من الذاكرة الموجودة ونتيجة لذلك كانت لـ DBMS أن تعالج وتدير البيانات في الذاكرة الثانوية ومن البرامج التي صممت لهذه الشأن كثيرة منها التي تعمل على الحاسبات الكبيرة Mainframes أو التي تعمل على الحاسبات الشخصية. PCs ومثل

هذه البرامج DBASE IV : و Clipper و Paradox و Oracle و FoxBASE و FoxPro و SQL

و DMS و IDMS و برنامجنا الجميل MS Access و الكثير من هذه البرامج بمختلف الإصدارات .

ويتكون نظام إدارة البيانات من مجموعة من الملفات بالإضافة إلى البرنامج أو مجموعة البرامج التي تتضافر لحل مشكلة أو لتحويل نظام يدوي إلى نظام يعمل بالحاسب مثل تحويل نظام حسابات العملاء أو حسابات المخازن من نظام الدفاتر اليدوية إلى نظام وملفات تستخدم بواسطة الحاسب فإن هذه البرامج مع ملفات النظام يطلق عليه نظام إدارة قاعدة البيانات أو قد يشتمل على مجموعه من البرامج بالإضافة إلى ملفات النظام وفي هذه الحالة فإن البرامج مجتمعه يطلق عليها نظام إدارة قاعدة البيانات⁽¹⁾.

وبعيدا عن التعقيدات ودون الدخول في تفاصيل وفرت نظم إدارة قواعد البيانات المرنة المطلوبة عن نظم إدارة الملفات التي كانت تستخدم من قبل.

ولكن كل ما نود أن نعرفه أن الملفات المسلسلة و الملفات الثنائية و العشوائية لها الدور الأكبر في الانتقال من نظام الملفات إلى نظام قواعد البيانات .

(1) Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

Royce, W. (2011), "Managing the Development of Large Software Systems : Concept and Techniques", Proc. WESCON, August 2011.

2-1 أنواع نظم إدارة قواعد البيانات :

ولهذا فإن نماذج البيانات هي تمثيل بيانات العالم الحقيقي بصورة يسهل استخدامها بواسطة الحاسب وهناك أنواع من نماذج البيانات تتوقف على نظام إدارة قواعد البيانات المستخدم وكذلك على طبيعة البيانات وتبعاً لأنواع نماذج البيانات فهناك ثلاثة أنواع شائعة من نظم إدارة قواعد البيانات وهي.

1- نظم إدارة قواعد البيانات الهرمية .

2- نظم إدارة قواعد البيانات الشبكية .

3- نظم إدارة قواعد البيانات العلائقية .

قواعد البيانات الهرمية أو النظم الهرمية Hierarchical DBMS تقوم بتنظيم البيانات على شكل هرمي أو علي شكل شجرة مقلوبة أي جذرها في القمة وتخرج منها الفروع . شأن هذه التركيبة شأن شجرة الأسرة فلها جد واحد و الجد له عدة أبناء و الأبناء هم آباء الأحفاد ويستحيل وجود حفيد له أكثر من أب . وهذا شكل توضيحي ليوضح النظم الهرمية وتفرعاتها .

والملفات الهرمية هي ملفات لها نفس البناء الشجري ولها نفس العلاقات بين السجلات مثلاً لبعض أنواع السجلات التي يمكن أن تتواجد في تكوين هرمي فهناك سجلات مبيعات متعددة لكل بائع حيث يوجد سجل إحصائيات واحد لكل عملية جارية كما يوجد أيضاً سجلات عديدة للعملاء لكل بائع حيث أن كل بائع له عملاء محددين ويمكن أن يكون لكل عميل عدة سجلات حسابات مدينين سجل واحد لكل عملية شراء لم يتم تسديد ثمنها.

ومن المهم أن نفهم انه ليس من الضروري أن تتصل كل الملفات الموجودة في قاعدة البيانات مع بعضها . وكل ما هو مطلوب أن تتصل الملفات التي تستخدم كمجموعة مع بعضها في التطبيقات .

وسجلات المبيعات السابقة لها مثل هذه العلاقة المنطقية تسمى فئة . و الفئة Set عبارة عن مجموعة من السجلات متصلة مع بعضها منطقياً.

وعلى هذا تصبح قاعدة البيانات الهرمية عبارة عن تجميع لملفات وفئات ملفات متصلة مع بعضها منطقيا.

ويستخدم نظام إدارة المعلومات IMS الذي أعدته شركة IBM التكوين الهرمي وهو من اكبر نظم إدارة قواعد البيانات dbms الموجودة حاليا واعقدها . ولهذا السبب فأنه يتطلب مستوى رفيع من الخبرة لإمكانية بنائه وعلى أي حال فهو قوي واثبت كفاءة كبيرة في معاملة قواعد بيانات كبيرة جدا كما انه يقدم إجراءات استرجاع و أمن جيدة هذا بالإضافة إلى إمكانية استخدامه في نظام الاتصال النشط من خلال شبكة الاتصالات.

2-3-1 نظم إدارة قواعد البيانات الشبكية:

رغم أن كلمة الشبكة استخدمت كثيرا في شبكات الحاسب ومعالجة البيانات فقد وجد من الأفضل استخدام مسمى قواعد البيانات الضفيرة Plex رغم أن مسمى قواعد البيانات الشبكية لازال شائع الاستخدام.

ويتغلب هيكل بيانات التركيب الشبكي على معوقات التكوين الهرمي الذي لا يسمح للابن أن يكون له أكثر من أب واحد ويظهر ذلك في الشكل التوضيحي للتكوين الشبكي حيث نلاحظ أن للسجل رقم (4) عائلان هما السجل رقم (2) و السجل رقم 3 .

ومثل هذا النوع من قواعد البيانات حل كثيرا من مشاكل العلاقات فإذا فرضنا أن هناك أكثر من مورد يورد قطع غيار فإن كل مورد قادر على توريد أكثر من نوعية قطعة غيار وبالتالي فإن كل قطعة غيار يوردها أكثر من مورد مما يحتم لفهم المثال عرض العلاقة بين قطعة الغيار و الموردون على النحو الموضح في الشكل التالي .

ومن هذا الشكل يتضح لنا بما لا يقبل الشك أن تبسيط العلاقة الشبكية إلى علاقة هرمية أوجد تعقيدات أكثر حيث حولها إلى نوعين من شجرة العلاقات وفي هذا جهد إضافي في التنفيذ.

إن ما عرضنا حول العلاقات الشجرية (الهرمية) وقواعد البيانات الشبكية يؤكد أن كلاهما يمكن تحقيقه وان كانت بعض حزم إدارة قواعد البيانات يمكنها التعامل فقط من الشكل الشجري كما أن البعض الآخر يمكنه التعامل مع

النوع الشبكي كما أن هناك تنوع من برامج إدارة قواعد البيانات فبعض برامج إدارة قواعد البيانات الهرمية لا تتعامل مع العلاقات البسيطة و البعض يمكنه التعامل مع العلاقات المعقدة.

و أوجه التشابه بين نظم قواعد البيانات الشبكية و نظم قواعد البيانات الهرمية إنها تتطلب إلى ذاكرات ذات أحجام كبيرة وعادة تحتاج إلى لغات راقية لبرمجتها وهي صعبة التعلم ولها مزايا كثيرة فهي بالطبع أكثر كفاءة من قواعد البيانات العلائقية ويتعاملان مع كم كبير جدا من البيانات و المعلومات بالإضافة إلى إنها توفر بناء على طريقة تنظيم البيانات التي تتبعها مساحات كبيرة من وسائط تخزين البيانات.

2-3-2 نظم إدارة قواعد البيانات العلائقية:

أثبتت الأيام صحة القول الشائع أن الأبسط هو الأجل والأكفأ . فكلما كان سبك بسيط وكلما عشت في بساطة و بعدت عنك المشاكل وكلما كانت الآلة بسيطة سهلت إدارتها وصيانتها . وهذا ما أكدته التعامل مع قواعد البيانات الهرمية و الشبكية التي تعقدت ملفاتها وأساليب إدارتها لدرجة كادت تؤدي بها كلما أضيفت تطبيقات جديدة أو متطلبات جديدة تحتاج مؤشرات جديدة مما ضخم منها وعقدها.

وهذه المشاكل كانت المنطلق للبحث عن حلول تحقق جملة أهداف منها:

1- يمكن فهم قاعدة البيانات لمن لم يدرسوا علوم الحاسب .

2- يمكن تعديل وإضافة وحذف بيانات دون تغيير المخطط المنطقي للقاعدة .

3- 3 تتيح للمستخدم اعلي درجة من المرونة في التعامل مع البيانات .

في عام 1970 أستحدث E.E.Codd أسلوبا لتنظيم وفرز بيانات قواعد البيانات . وهي قواعد البيانات العلائقية . وقد وجد العالم الأمريكي E.E.Codd أن هذا لا يتحقق إلا برص البيانات على هيئة جداول لان الإنسان تعود على الجداول منذ طفولته بداية من جدول الحصص إلى جدول الضرب إلى كشف الأسماء و الدرجات. و هذه النظم تتعامل مع أكثر من ملف في نفس الوقت وتعامل البيانات داخل الملف كما لو كانت جدولا مكونا من صفوف و

أعمدة ويسمى علاقة Relation وتمثل أعمدة الجدول حقول قاعدة البيانات Fields وتسمى أيضا Attributes بينما تمثل صفوفها سجلات قاعدة البيانات وتسمى Tuples و النظام العلائقي Relation يقوم بربط البيانات بين العلاقات بناء على حقل مشترك بينهما .

والنظم العلائقية قامت أساسا علي النظريات العلائقية في الرياضيات وقد بدأ تطبيقها على الحاسبات الكبيرة أولا مثل

DBaseII . ORACLE . SQL ثم ظهرت عدة نظم علائقية على الحاسبات الشخصية PCs مثل برامج DBaseII

. FoxPro . FoxBase . DBaseIV ويمكن القول عن هذا النوع من قواعد البيانات مايلي :

1- تنظيم البيانات في قواعد البيانات العلائقية في جداول ذات بعدين ويمكن اعتبار كل جدول ملف ويستخدم

مصطلح ملف مسطح Flat File لان محتويات الملف مرتبة على محورين س , ص فقط.

2- نشأت مجموعة جديدة من المصطلحات تستخدم في وصف قواعد البيانات العلائقية هذه المصطلحات التي

تستخدم في وصف قواعد البيانات الهرمية أو الشبكية ففي النموذج العلائقي يستخدم مصطلح نموذج بيانات

علائقي جزئي أو رؤية لبيانات علائقية Relational Data Submodel or View بدلا من المخطط

الجزئي Subschema ومصطلح رؤية View مناسب فهو لجزء المستفيد من قاعدة البيانات.

3- كما استخدمت بالإضافة إلى ذلك أسماء لوصف مكونات الملفات المسطحة ويوضح الجدول التالي عينة

لملف ويشار إلى أعمدة الملف بأنها مسطح رأسي (نطاق) والى الصفوف بأنها مسطح أفقي و الجدول

عبارة عن تجميع من المسطحات الأفقية خاصة بموضوع معين و الجدول خاص بالبائع ويمكن استخدامه

في توفير أسمائهم ومبيعاتهم منذ بداية العام.

4- وقد وضعنا المشاكل السابق ذكرها نظراً لأنه أمكن تجنبها في قواعد البيانات العلائقية فالتكوين العلائقي

تكوين منطقي بحيث يستخدم علاقات ضمنية بدلاً من استخدامه لعلاقات صريحة وهي التي تستخدم في

كل من قواعد البيانات الهرمية و الشبكية⁽²⁾

وحتى نوضح مفهوم العلاقات الضمنية بين ملفات قاعدة البيانات العلائقية وكيفية استخدامها في تجميع البيانات مع بعضها من ملفات منفصلة عن بعضها نفرض أن لدينا جدولين في قاعدة البيانات جدول [أ] و جدول [ب] جدول [أ] يعرف منطقة المبيعات لكل بائع باستخدام رقم البائع كحقل مفتاحي و الجدول [ب] يحدد اسم كل بائع و الجدولان منفصلان عن بعضهما أي لا يوجد أي اتصال طبيعي بينهما وتحدد العلاقة ضمناً وذلك بإدخال حقل رقم البائع في كل من الجدولين.

ولعل أحد الأسباب الرئيسية في وجود أنظمة متخصصة لإدارة قواعد البيانات ، هو الحاجة الفعلية لدعم قاعدة بيانات مركزية لتطبيقات متعددة بالإضافة إلى مستخدمين متعددين على نفس التطبيق.

وتأتي أنظمة (DBMS) بأصناف عديدة ، وبمزايا مختلفة إلا أنها بشكل عام تسعى لتحقيق ثلاثة أهداف هامة:

• دمج البيانات :

وهذا الهدف يشير إلى إمكانية ضم أو توحيد ملفات البيانات المنفصلة في بنية مركزية ، وتخزين البيانات بصيغة خالية من الفائض ؛ الذي ينشأ في قاعدة البيانات عندما يخزن في موقعين أو أكثر ، فمثلاً قد نجد التخصص العلمي للموظف مخزناً ليس فقط في جدول البيانات الشخصية وإنما نجده أيضاً في جدول الوظائف ، و جدول

(2) Abiteboul, S, Hull, R, and Vianu, V., Foundations of Database, Addison Wesley 2005

Boehm, B. (1988), 'A Spiral Model of Software Development and Enhancement', IEEE Computer 21, 5, 61-72.

التاريخ الوظيفي ، وعندها نكون أمام قاعدة بيانات غير مركزية تحوي معلومات زائدة ، ولذلك ينبغي إذا أردنا أن نبني نظاماً مثالياً ومتكاملاً وخالياً من الفائض أن يحتوي على تخصص الموظف في جدول واحد.

• المشاركة على البيانات:

أي قدرة النظام على السماح لعدة مستخدمين بالوصول إلى أجزاء مستقلة من البيانات ضمن قاعدة البيانات في نفس الوقت ، وهذه خاصية تتميز بها تطبيقات DBMS بما يعرف بالتوازي. (Concurrency)

حماية البيانات: (Data Protection) أي قدرة الـ DBMS على المحافظة على سلامة البيانات أمام الحوادث الطارئة خلال المعالجة (فشل البرنامج أو توقفه فجأة .. الخ) ، إذ ينبغي على الـ DBMS أن تمتلك القدرة على إعادة البيانات إلى حالتها السابقة قبل التعديل غير الكامل عليها أو قبل حدوث الخطأ فيها وتسمى هذه العملية أحياناً بالتراجع (undo)⁽³⁾

2-4 تطبيق قواعد البيانات:-

عملية تصميم وتطوير تطبيقات قواعد بيانات أوراكل أو كما يطلق عليها المطورون (تصميم النظام) عملية معقدة وليست بالبساطة التي يمكن حصرها في إنشاء الجداول وتشغيل معالج إنشاء الفورمات والتقارير بل يمتد لأبعد من ذلك كثيرا فيجب للمطور الإلمام بكل التفاصيل التي يحتاجها تطبيقه بدأ من فهم وتحليل المطلوب واختيار أفضل الطرق لتنفيذه ، الموضوع لايركز على اساسيات تحليل النظم بل على الناحية البرمجية للتطبيقات, اى على المستوى المعرفى بأدوات أوراكل التي يجب على مطور التطبيقات معرفتها لاختيار ، فقاعدة بيانات أوراكل تتيح العديد من

(3) Bhargava, B., and ed, Concurrency and Reliability in Distributed Systems, Van Nostrand-Reinhold, 2007

Ronald J. Leach,, "Introduction to Software Engineering", CRC Press, 1999.

Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.

الأدوات التي يمكن أن تؤدي عدة أغراض أفضل من الأخرى , فمثلا بدون معرفة الفرق بين الجدول العادي Table View و ال Materialized View سيكون التطبيق حتما غير مكتمل الجوانب وربما سيعانى من مشكلات حقيقية فى التصميم تؤدي لمشكلات اكبر فى عمله من ناحية availability وربما يؤثر على قاعدة البيانات ككل , فالمطور يجب أن يأخذ فى الاعتبار متى يحفظ البيانات فى جدول عادى أو جدول عشوائي ومتى يستخدم external tables كما يجب عليه كيف يتوجب عليه القراءة من وحدة ال DUAL والتي يستخدمها المطورون بكثرة دون مراعاة لسرعة التطبيق ودون محاولة استخدام شفرات ال pause والتي تتيح إزالة الضغط على قاعدة البيانات وزيادة لسرعة التطبيق وغيرها , سأحدث بإيجاز عن بعض النقاط التي يجب لمطوري تطبيقات قواعد البيانات مراعاتها بشده وقبل كل شي يجب على كطور التطبيقات الإلمام بكل الأدوات التي تساعده فى عمله متوفرة داخل الاوراكل أمثال Tables وال indexes و Views وال synonym و snapshot وغيرها .

حسب إحصائيات اوراكل فإن 50% من مشاكل سرعة واستقرار التطبيقات تعتمد أساساً على أخطاء فى تصميم التطبيق واستخدام أدوات بدلا عن أدوات بعينها مثلا كاستخدام جدول بدلا عن View أو جعل التقرير يقوم بعمليات تجميع مجهزة لقاعدة البيانات كل مره بدلا من تجهيز البيانات التي يحتاجها التقرير فى جدول واحد بقدر الإمكان على أن تتم ذلك فى مرحلة تصميم التطبيق.

مستخدمو ومدخلو بيانات تطبيقات قواعد البيانات لأيهم ما الذي تحتويه التفاصيل الداخلية للبرنامج مقارنة بسرعة استجابة التطبيق وسهولة استخدامه مع الوضع فى المعيار بتغطية التطبيق لاحتياجات العمل , فلا يجب عند البدء فى تصميم البرنامج وضع كل الاهتمام على النظريات البرمجية وإغفال الجانب الذي يخص سهولة تعامل المستخدم مع التطبيق من كافة النواحي فيجب تبسيط التصميم لأقصى ما لا يمكن , فمثلا فى الأنظمة المالية يمكننا اخذ إدخال تفاصيل عنصر الفاتورة فالعملية ربما تقوم بعملية بحث عن بيانات أو تفاصيل بجدول آخر كما يمكن أن تقوم بأكثر من عملية إدخال فى عدة مواقع , هذا على نطاق المستخدم الواحد ربما يكون ليس ذي شأن فى بال مطور التطبيق , لكن يجب عليه إعطاءه اشد الانتباه فى حالة أن الإدخال يتم بأكثر من مستخدم واحد فيسبب هذا

مشكلات ضخمة للتطبيق وسيدخله في حلقة مغلقة من عمليات ال Locks وإيقاف المستخدمين والتطبيق أحيانا , هنا وجب تغيير التصميم واستخدام أدوات ال Materialize views وغيرها لدورها الفعال في حل هذه المشكلات.

إلغاء وتقليل مناداة البيانات من اوراقل , المعروف انه عند عمل اى إجراء على جهاز الكمبيوتر فان المعالج يقوم بعمل معين لتنفيذ الإجراء والوصول للبيانات على القرص هذه العملية نطلق عليها Physical reads وعى عملية مجهدة جدا ويجب تجنبها متى ماسنح ذلك , دعنا ننظر للمثال التالي : سأفترض أن مصمم تطبيقات قاعدة بيانات يقوم بالقراءة من جدول dual مثلا قيمة الوقت عدة مرات دون الوضع الاعتبار أن هذا الجدول بالذات ليس جدولاً عادياً بل , oracle internal table فإذا قمت مثلاً بقراءة الزمن من هذا الجدول فاعلم أن بيانات هذا الجدول يتغير 86,400 مره فى اليوم وهذا المصمم قام باستدعاء بيانات عدة مرات داخل تطبيقه من داخل هذا الجدول وتطبيقه هذا يستخدمه خمسون مستخدم فتخيل عدد مرات ال physical read التى ستتم وكما سبق وذكرنا أن هذا الجدول ليس جدول قواعد بيانات لنقول أنها Logical reads بل Physical reads فى هذه الحالات يمكن لمطور التطبيق استعمال sleep على مستوى نظام التشغيل باستخدامه لحزمة (DBMS_LOCK.SLEEP) عن طريق plsql حتى يجنب قاعدة البيانات اى أعمال لا تحتاج إليها فعلاً هذا بعض تحسين عملية الوصول لهذه البيانات.

تجنب تكرار تنفيذ العمليات التى قمت بتنفيذها مسبقاً بقدر الإمكان , على المصمم فهم أهمية استعمال ال procedures ومخرجاتها وأهمية تخزين النتائج التى يحتاجها المستخدمون بكثرة فى التطبيق بدلا عن إجراء نفس عمليات ال sort و ال group كل مره , فأحيانا تجد ببعض التطبيقات غير ذات التصميم الجيد أن المطور يقوم بعمل إجراء أكثر من مره للحصول على نفس النتيجة على الرغم من إمكانية الوصول للبيانات السابقة والتي لم يحدث تغيير بها بقاعدة البيانات , فمثلا عند بناء procedures استخدم القيم التى لا تتغير أو الثابتة كثوابت فى إجراء ولا تقم بتخزينها فى قاعدة البيانات , لأنك بذلك تقوم بعدة عمليات لاشيء , مجموعة ال physical reads و waitings ربما تنفذ من قبل 30 مستخدم تخيل أنه يمكنك الاستغناء ببساطه عن كل هذا.

فى تصمىم التقارير نغم بتخزىن البىانات التى تحتاجها التقارير فى جدول لتكون جاهزة مسبقاً، تعطى الأولوية دائماً للسرعة مقارنة بالمساحة ، فمثلاً تخيل إننا نرى عمل تقرير يعتمء على برامىتر، فعكس ماىتخيله معظم مطورى التطبيقات يفصل تماماً تجهىز هذه البىانات على الجدول مباشرة ومناءاتها بدون بارامىتر أفضل من إجراء عملية الفلتر Select أثناء تشغيل التقرير والفارق سىكون ضخم جداً وسىلاحظه المستخدم العاءى بسهولة جداً فى سرعة التطبيق وقلة الجراء الذى ستقوم قاعءة البىانات بعمله فهى فقط ستقوم بعمل تحءىث refresh لل materialized View أو الجدول.

لا تقم بالاتصال بقاعءة البىانات بأكثر من مره ، عادة عملية الاتصال بقاعءة البىانات تأخذ وقت أكثر من مجرد ال Connect الذى يعلمها المطور فىجب على المطور معرفة كىفئة عمل Reuse لل Connection وبالمقابل ترك الاتصال مفتوحاً داخل التطبيق.

ضرورة استخدام ال indexes وعدم الإكثار فىها ، معروف لتأثير المباشر لل indexes على سرعة تطبيقات قواعد البىانات فعلمة الوصول لسجل على جدول يحتوى مليون سجل من دون index بمكن أن تأخذ 15 ثانية بالإضافة للتأثير السىئ جداً على التطبيق وقاعءة البىانات ككل لكن فى ظل وجود index يمكن لهذه العملية أن تأخذ فقط 3 ثوان وما أسهل من إنشاء ، index لكن بالمقابل فىجب عدم الإكثار من استخدام ال Index أو عدم استخدامه فى مكان غير مكانه الحقىقى ، كمثل لا فىجب استخدام ال Bitmap Index مع تطبيقات ال OLTP .

7- تجنب عملیات ال sort الغير ضرورية مءمثلة فى ، Order by مثلاً تخيل جدول ضخم من غير index والمصمم یرىء إجراء عملية Sort لحقل معین فسیتم إجراء عملية محءءه یتم فىها عمل scan لكل الجدول ومن ثم فىقوم بعمل sort للنواتج !!! وعلمة ال sort تنفذ معها عملية select لكل البىانات الموجودة بالجدول ككل ونقلها الى خارج ال data file لا خلاء العملية ، وكامءءاء لهذه العملية تجنب استعمال Union واستعمل Union all نسبة لان union تقوم بعمل Sort أیضاً ولىس. Union all استخدم ال ، Partitioning كمطور تطبيقات فىجب

عليك استعمال Partitioning مع الجداول الكبيرة لأثر ذلك في تسريع التطبيق وتخفيف الضغط على قاعدة البيانات.(4)

استعمال ال Materialized View لتقسيم وتصنيف الإجراءات التي يحتاجها المستخدم من الجداول , فعند استخدام ال Materialized view يقوم الاوراكل بعمل نسختين من البيانات نسخة تتاح عند إدخال بيانات جديدة ونسخة أخرى تخصص لعمليات ال Query وغيرها بالإضافة لإمكانية ضم الجداول وإنشاء جداول حسب query معين مما يسهل ويقلل الوقت الذي سيحتاجه التطبيق لتوفير البيانات.

اختبار التطبيق مع كمية ضخمة من البيانات وعدد كبير من المستخدمين لان في هذه الحالة فقط ستخرج بنتائج صحيحة , لان تجربة النظام مع مستخدم واحد ليعطى النتائج المرجوة والمتعلقة بجوانب السرعة واستقرار التطبيق.

الاستخدام الصحيح لل Data types , فلا يجب استخدام ال Varchar2 مع بيانات ثابتة مثلا مفاتيح الدولة التي ستكون فقط من حرفين يمكن استعمال النوع char بطول 2 , وكمثال آخر وظائف استعمال raw و BLOB وأماكن استعمال كل واحده منهما .

(4) [2]Bhargava, B., and Helal, Efficient Reliability Mechanisms in Distributed Systems, CIKM 2003.

Ante, S, and Grow, B, Meet the Hackers, 2006.

[2]Barker, W, Introduction to the Analysis of the Data Encryption Standard(DES), Aegean Park Press, 2001.

2-6 عمليات نظام إدارة قاعدة البيانات:

يشتمل نظام إدارة قاعدة البيانات اوراكل على العمليات التالية

1- أوامر لغة تعريف البيانات (DDL) Data Definition Language

تستخدم هذه اللغة في تعريف وإنشاء الكائن Object ، ويمكن أن يكون الكائن ملفات وجداول بيانات

، فيمكننا إنشاء وتعديل وحذف الكائن ويمكننا إنشاء امتياز لمستخدم معين ، أو إنشاء كائن خيارات لفحص وإضافة تعليقات إلى قاموس البيانات ومن هذه الأوامر CREATE , DROP and ALTER

2- أوامر لغة معاملة البيانات (DML) Data Manipulation Language

تتيح هذه الأوامر التعامل مع البيانات وتعديلها ضمن الكائن الموجود Object ومن هذه الأوامر SELECT ,

DELETE, UPDATE and INSERT

أوامر لغة التحكم في البيانات (DCL) Data Control Language

تتيح هذه الأوامر التحكم في قاعدة البيانات وأدائها كالصلاحيات والمستخدمين والحقوق وغالبا ما تكون هذه الأوامر

مخصصة للاستخدام من قبل مدير قاعدة البيانات (DBA) ومن هذه الأوامر GRANT and REVOKE :

2-7 تعريف الزنادات : (5)

تعرف الزنادات على إنها الحدث الذي ينفذ أثناء حدوث تغيير على جدول معين بقاعدة البيانات لتنفيذ مجموعة من

التعليمات أو لأخذ معلومات وإضافتها بجدول آخر أو استدعاء PROCEDURE أو حتى استدعاء FUNCTION

تم إنشائها على قاعدة البيانات. TRIGGERS.

الزناد Trigger هو مجموع من أكواد برمجية يتم تنفيذها عند حدوث حدث معين.

المبحث الثاني: مفهوم ومكونات ومراحل تنفيذ الزنادات⁽⁶⁾ :

جدول (1.2) الزنادات

المكونات	الزناد
فيها يتم تحديد الحدث الذي سيتم عنده تنفيذ هذه الاكواد البرمجية.	Type of trigger
فيه يتم كتابة الكود البرمجي الذي سيتم تنفيذه.	code of trigger
فيه يتم تحديد المدى الذي سيتم تنفيذه عليه(هل سيتم تنفيذه على عنصر محدد فقط أو على تلك بيانات data block محدد فقط أو على البرنامج Module كله. و يتم تحديد مدى التريجر scope of trigger) من خلال موضع الزناد في البرنامج.(Module)	Scope of trigger

المصدر.: Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

(6)Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

يكون مدى الزناد على المستويات التالية:

جدول (2.2) مستويات الزناد

المستوى	الشرح
Forms-Level Triggers -1	و هذا يتم تنفيذه في أحداث خاصة بالنموذج فقط. و يؤثر في كل مكوناته.
Item-Level Triggers-2	و هذا يتم تنفيذه في أحداث خاصة بهذا العنصر فقط.
Report-Level Triggers - 3	و هذا يتم تنفيذه في أحداث خاصة بالتقارير

المصدر: Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

2-7-2 أنواع الزنادات : (7)

جدول (2.3) أنواع الزنادات

النوع	الشرح
On_Trigger	ينطلق أثناء حدوث الحدث
Pre_Trigger	ينطلق قبل حدوث الحدث
Post_Trigger	ينطلق أثناء الحدث مباشرة
When_Trigger	ينطلق بعد الحدث

المصدر: Comparing Simple Role-Based Access Control Models and Access Control Lists, Proceedings of the second ACM Workshop on

Role-Based Access Control, 2009.

(7)Comparing Simple Role-Based Access Control Models and Access Control Lists, Proceedings of the second ACM Workshop on Role-

Based Access Control, 2009.

2-7-3 مراحل تنفيذ الزنادات :

1- مدى الزناد و تدرج التنفيذ :

عندما يوجد أكثر من Trigger من نفس النوع على مستويين مختلفين فغن الـ Form Builder يقوم بإطلاق الـ Trigger الأكثر فاعلية في الموضع الحال للمؤشر. و خاصية تدرج التنفيذ Execution Hierarchy تحدد ما الذي يحدث عندما يوجد أكثر من Trigger من نفس النوع في مستويين مختلفين, و هذه الخاصية من خصائص الـ Trigger و هي تحتوي على ثلاثة خيارات هي:

- Override فقط الـ Trigger الأكثر فاعلية في موضع المؤشر سينطلق أولا.
- After فإن الـ Trigger سينطلق بعد الـ Trigger الآخر.
- Before خصائص الـ Trigger قبل الـ Trigger الآخر.

جدول (2.4) مراحل تنفيذ الزنادات

الشفرة	شرح الشفرة
create trigger ledger_bef_upd_row	تسمية الزناد
before update on leader	يحتوي هذا الأمر اسم الجدول الذي يتعامل معه القادح. و تعني هذه التعليمة إن القادح سينفذ قبل تثبيت اثر عملية التعديل في قاعدة البيانات
for each row	أي أن القادح سينفذ من اجل كل سطر
when (new.Amount/old.Amount>1.1	هذا يعني أن القادح سينفذ فقا عل الأسطر التي فيها ازدادت قيمة العمود Amount بمقدار 10 %

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

الفصل الثالث

تقنية هندسة البرمجيات

المبحث الأول : مفهوم ومبادئ وأدوات هندسة البرمجيات :

يوجد العديد من التعريفات التي تم اقتراحها من العديد من المهتمين بهندسة البرمجيات ويمكن إجمال هذه التعريفات في أن "هندسة البرمجيات" هي : علم يهتم ببناء الأنظمة البرمجية الكبيرة والمعقدة بواسطة فريق من مهندسي البرمجيات بإتباع طرق ومبادئ هندسية منظمة وصحيحة واستخدام وسائل وأدوات وتقنيات تجعل من تصميم وتطوير وبناء واختبار هذه النظم عملية منظمة يمكن تكرارها في حدود الإمكانيات المتاحة مادياً وبشياً وزمناً ، وذلك من خلال دراسة دورة حياة النظام مع إعطاء أشكالاً متعددة لعمليات تصميمه وتطويره وبنائه واختباره بحيث يكون المنتج البرمجي النهائي على درجة عالية من الجودة والموثوقية وقليل التكاليف بقدر الإمكان مع تسليمه للعميل في الوقت المناسب على أن يعمل بكفاءة على أجهزة الحاسب الآلي المختلفة.

وبصفة عامة ، يهتم علم هندسة البرمجيات بجميع نواحي إنتاج نظم البرمجيات التي تلبى متطلبات المستخدمين تحت قيود معينة ، وذلك من خلال تطبيق النظريات والمعرفة بأسلوب فعال وكفاءة موثقة، وتتدخل الطرق المستخدمة في هندسة البرمجيات في جميع أطوار دورة حياة نظم البرمجيات من المراحل الأولى لتحليل النظام حتى صيانتها مروراً بعمليات وضع المواصفات ، والتصميم ، والبناء ، والفحص ، والتشغيل. كما يوظف علم هندسة البرمجيات الطرق والأساليب والعمليات والقياسات الهندسية لتحقيق الأهداف وإتمام الفائدة من الأدوات المعدة لإدارة عمليات تطوير البرمجيات ، مع الاهتمام بالجودة والتحكم في النوعية.

وبالتدقيق في مصطلح "هندسة البرمجيات" تجده يحتوي على شقين :

الشق الأول : هندسي : وفيه يطبق المهندسون النظريات والأساليب والأدوات الملائمة بفعالية للحصول على حلول مثالية للمشاكل التي تواجههم مع التقيد بالقيود التنظيمية والمالية التي تحيط بالمشكلة ، ودائماً ما يحاولون اكتشاف

الحلول للمشاكل حتى عندما لا توجد نظريات أو أساليب مطبقة لتدعيمهم ، وذلك من خلال تبنينهم أسلوب تصنيفي منظم لعلهم مع التركيز على اختيار الطريقة الأكثر ملائمة لتنفيذ النظام البرمجي طبقاً للمواصفات المطلوبة والطريقة الأكثر ابتكاراً.

الشق الثاني : فهو برمجي : وفيه يتم الاهتمام بجميع مظاهر إنتاج البرمجيات ، حيث إن هندسة البرمجيات لا تهتم فقط بالعملية الفنية لتطوير البرمجيات ولكن تهتم أيضاً بالنشاطات الأخرى مثل إدارة مشروع البرمجيات ، وتطوير الأدوات والأساليب والنظريات لدعم إنتاج البرمجيات.

وبناءً على ما سبق يتضح أن دور هندسة البرمجيات هو : تطبيق منهج مرتب وقابل للقياس لعمليات تطوير وتشغيل وصيانة البرمجيات ، أي تطبيق الهندسة على البرمجيات ، وهو ذات التعريف الذي وضعه معهد المهندسين الكهربائيين والإلكترونيين (IEEE) لهندسة البرمجيات ، وذلك من خلال الإجابة على العديد من الأسئلة التي تخص هندسة وتطوير البرمجيات ، والتي من أهمها :

- ما هي الطرق والمبادئ الهندسية المنظمة والصحيحة التي يمكن تطبيقها على بناء البرنامج؟.
- ما هي الوسائل والأدوات والتقنيات التي تجعل من تصميم وتطوير وبناء واختبار البرنامج عملية منظمة يمكن تكرارها في حدود الإمكانيات المتاحة مادياً وبشرياً وزمنياً؟
- كيف نتمكن اقتصادياً من بناء برنامج موثوق يعمل بكفاءة ليس على جهاز حاسب واحد بل على العديد من أجهزة الحاسب المختلفة.
- ما هي القياسات والإجراءات التي يجب إجراؤها لضمان جودة المنتج البرمجي؟.
- كيف سيتم دعم البرنامج على المدى الطويل عندما يطلب المستخدم إجراء تعديلات أو تحسينات عليه؟.
- ما هو المنهج الواجب إتباعه لدراسة تقنيات الإدارة اللازمة لتخطيط المشاريع البرمجية وتنظيمها واستخدامه كآلية للتحكم في المشروع بشكل يقود إلى تسليم منتج برمجي عالي الجودة قابل للتشغيل في الوقت المحدد.

1-3 المهام النموذجية لهندسة البرمجيات:

هناك مجموعة من المهام النموذجية التي تتعلق بهندسة البرمجيات يجب أن تتبع في تطوير المشاريع

البرمجية من قبل فريق متكامل من مهندسي البرمجيات ، وهذه المهام هي كالتالي :

جدول (2.5) تطوير المشاريع البرمجية

تحليل المشكلة.	تحديد المتطلبات.	تصميم البرمجية.
الترميز (عملية البرمجة).	اختبار وتكاملية الشفرة البرمجية.	
التثبيت وتوزيع البرمجية.	توثيق البرمجية.	صيانة البرمجية.
التدريب	تقدير الموارد	إدارة المشروع

المصدر: Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley

وبنظرة ثاقبة للمهام المذكورة أعلاه نجد أن هندسة البرمجيات تهتم بجميع أطوار دورة حياة تطوير البرمجيات (Software Life Cycle) بدءاً من المراحل الأولى لتحليل النظام حتى صيانتها، وسوف نتناول أطوار دورة حياة تطوير البرمجيات بالتفصيل لاحقاً ، ولكن سوف تغطي فكرة بسيطة هنا عن أهم الخطوات المتبعة (الأطوار) في تطوير البرمجيات من وجهة نظر هندسة البرمجيات :

مبدئياً لا بد من التعرف على متطلبات النظام فإذا كان هناك زبون أو عدة زبائن فإنه لا بد من حدوث مقابلة بين الزبائن ومتخصصي دراسة وتحليل المتطلبات في فريق التطوير ، وإن لم يكن هناك زبون محدد ولكن هناك متطلبات لفئة عامة من الزبائن فإن دراسة متطلبات واحتياجات السوق يتم عملها وهذه الطريقة مناسبة لتطوير برمجيات عامة.

بعد دراسة وتحليل المتطلبات والانتهاء من كتابة وثائق التحليل يتم الانتقال إلى وضع التصميم الأمثل لترجمتها إلى شفرة برمجية قابلة للتنفيذ ، وذلك من خلال المصممين في فريق العمل ولا بد أن يكونوا ملمين إماماً كاملاً بطرق التصميم المختلفة والفروق المختلفة بينها ولا بد أن يكون لديهم إمام بخواص النظم الموجودة مثل أنظمة التشغيل ، ونظم المعلومات المختلفة المتاحة ، وواجهات التطبيق المختلفة الرسومية والبرمجيات المساعدة التي

يمكن أن تساعد في عملية الترميز (البرمجة).

بعد الانتهاء من تخطيط وثائق التصميم يتم الانتقال إلى توزيع المهام على المبرمجين في فريق العمل لتحويل تلك المخططات وترجمتها إلى شفرة برمجية قابلة للتنفيذ باستخدام إحدى لغات البرمجة عالية المستوى (High Level Programming Languages) ، ومن ثم تبدأ عملية التكامل للنظام وهي عملية تجميع المهام البرمجية بعد ترميزها عن طريق فريق التطوير مع برمجيات جاهزة تخدم في نفس المشروع.

بعد الانتهاء من مرحلة برمجة وتكامل النظام تبدأ مرحلة اختبار النظام البرمجي من حيث الأخطاء البرمجية ، وكذلك من حيث مطابقته للمواصفات المطلوبة وجودة الأداء. ومصطلح جودة البرمجية (Quality Assurance) يتعلق بعملية تطوير المنتج البرمجي والطريقة التي تم تطويره بها بحيث تكون وفقاً لمعايير مقاييس الجودة والتي يتم وضعها من خلال مجموعة مراقبة الجودة من فريق التطوير بعد تحديد متطلبات المشروع البرمجي مباشرة وقبل البدء في التنفيذ الفعلي للمشروع، وتتم عملية مراقبة الجودة عند الانتهاء من أي جزئية أو مكون من المشروع حيث يتم اختباره وفقاً للمعايير المتفق عليها سابقاً ، وعند وجود أي خلل أو خطأ فعلى الفريق المختص إصلاحه.

عملية التوثيق ليست مرحلة منفصلة ولكنها تتم بالتوازي مع المراحل السابقة ، وهي لا تشمل عملية التعليق على الشفرة البرمجية فقط ولكنها تشمل العديد من العمليات مثل كتابة ملفات المساعدة ، وكتيب المستخدم ، وكتيب التدريب، وكتيب المرجع ، وكتيبات التثبيت ولا غرابة أن يكون حجم ما يكتب من سطور لتوثيق النظام يزيد على مرتين من عدد سطور النظام نفسه.

بعد عمليات التحليل ، والتصميم ، والبرمجة ، والاختبار والتوثيق ، لا بد أن توزع هذه البرمجيات وتثبت على أجهزة العملاء.

ومع بداية إصدار وتوزيع المنتج البرمجي تبدأ مرحلة الصيانة، ومصطلح الصيانة هنا يصف النشاطات التي تتم بعد إصدار المنتج البرمجي والتي تشمل عملية تصحيح الأخطاء ، وتطوير البرمجية لتعمل مع أوساط جديدة وحاسبات جديدة وأنظمة تشغيل جديدة ، وزيادة فاعلية البرمجية، وفي كثير من الحالات تكون الصيانة هي الأكثر كلفة من

ناحية المصاريف والوقت في دورة حياة تطوير البرمجية. كل العمليات السابقة لا بد من إدارتها من خلال إدارة واعية وعلى دراية كاملة بجميع مراحل تطوير مشاريع البرمجيات ، حيث تمثل عملية إدارة المشروع النشاط المهم والحرص لتطوير البرمجيات.

3-2. مبادئ هندسة البرمجيات :

لهندسة البرمجيات أهمية كبيرة ، إذ أنها تساعد في زيادة المردود لشركات النظم البرمجية ، ومستخدمي البرمجيات، من خلال التوزيع السليم للموارد المتوفرة ، والتصميم الجيد لهذه البرمجيات مما يقلل من تكاليف وزمن إنتاجها ، مع زيادة وتحسين مستوى جودتها وموثوقيتها ، ويتم ذلك من خلال تطبيق مجموعة من المبادئ الأساسية التي تتميز بها والتي من أهمها ما يلي :

الدقة والصورية: يمكن الحصول على منتج موثوق ومضبوط الكلفة من خلال الدقة في عملية الإنتاج وتوجد درجات متنوعة للدقة أعلاها الصورية التي تتطلب أن تكون المنتجات مصممة ومقيمة بطريقة رياضية، وبالتالي تقتضي الدقة الصورية ولكن العكس غير صحيح إذ يمكن أن نكون دقيقين بطرق غير صورية (رياضية). يحتاج أي عمل لتنفيذه إلى خطوات وإذا نفذنا هذه الخطوات اعتماداً على خبرات وتجارب ونظريات عندها ستزداد الدقة، فإذا لم تتوفر هذه الاعتمادات (الخبرات والتجارب والنظريات) نعتمد عند ذلك على التمثيل الرياضي وهذه هي الصورية واللغات البرمجية أدوات صورية للتعبير عن وظائف ثم تحول المترجمات اللغة الصورية إلى لغة الآلة.

فصل الاهتمامات: يعتمد هذا المبدأ على أتباع سياسة " فرق تسد" حيث يتم تقسيم المشكلة إلى مشاكل أصغر غير مرتبطة ببعضها البعض أو يكون ارتباطها صغيراً ، ثم التركيز على حل كل مسألة على حدة. ويتم فصل الاهتمامات حسب الطرق التالية : حسب الزمن : جدولة الأعمال وفق محور الزمن.

حسب وجهات النظر: مثلاً عند تحليل متطلبات نظام برمجي ، قد يكون من المفيد التركيز على المعطيات التي تتعامل معها البرمجية من جهة والتركيز على الوظائف والتحكمات التي تقوم بها البرمجية من جهة أخرى.

حسب الخواص والمواصفات : يتضمن معالجة المواصفات المطلوبة من البرمجية على حدة، فعلى سبيل المثال إذا كان المطلوب بناء نظام برمجي فعال وصحيح ، فيتم أولاً تصميمه بطريقة متأنية ومنظمة تضمن صحته ثم يتم التغيير في النظام لتحسين فعاليته.

حسب الأقسام : وهو عبارة عن فصل التعامل مع الأقسام المكونة للنظام البرمجي كل على حدة. التجزئة: وهي تقسيم النظام البرمجي المعقد إلى أقسام أبسط تسمى كتلاً ، ويهدف مبدأ التجزئة في هندسة البرمجيات إلى :

القدرة على تجزئة النظام البرمجي إلى كتل (التجزئة التنازلي (الشجري) للنظام البرمجي).

القدرة على تركيب النظام البرمجي من كتل (التركيب التصاعدي للنظام).

القدرة على فهم الكتل كلاً على حدة لتعديلها (الاستقلالية).

يجب أن يتحقق في الكتل :

تماسك قوي داخلي للكتل (الكتلة مبنية بشكل منطقي لتحقيق هدف محدد).

ترابط ضعيف بين الكتل (الاستقلالية).

التجريد: وهو تحديد المظاهر المهمة وتجاهل تفاصيلها أثناء دراسة مسألة معينة (فصل الشيء المهم عن الشيء غير المهم في المسألة) ، وهي نظرة نسبية مرتبطة بالهدف الذي نعمل عليه.

توقع التغيير: وهو إحصاء الأماكن التي يمكن أن يتم تعديل عليها في البرمجيات ، ومن أكثر هذه الأماكن تغييراً .

الخوارزميات : يمكن أن تكون هناك عدة خوارزميات تقوم بنفس الوظيفة (الفرز مثلاً) وبالتالي فاستبدال إحدى هذه الخوارزميات بأخرى أفضل منها، هو أمر محتمل ، ولذلك يفضل كتابة الخوارزمية في وحدة برمجية منفصلة.

تمثيل المعطيات : يتغير أداء البرنامج بتغيير بنية المعطيات التي يستخدمها.

الآلات التجريدية: يمكن اعتبار البرامج التي نكتبها يتم تنفيذها على آلات تجريدية تفهم التعليمات والأوامر المكتوبة بلغة عالية المستوى، أي أن لكل لغة عالية المستوى آلة تجريدية تفهم تعليمات اللغة وتنفذها بشكل تجريدي وبعيد عن العتاد الصلب.

الوسط الاجتماعي : التغيير في الوسط الاجتماعي يؤدي إلى تغيير النظام البرمجي المرافق (مثلاً تغيير العملة يؤدي إلى تعديل النظم البرمجية البنكية الموجودة) .

الطرفيات : يتعلق هذا النوع بالبرمجيات التي تحتاج للتعامل مع طرفيات خاصة كأنظمة التحكم. وبالتالي تغيير الطرفيات يتطلب تغيير النظم البرمجية المرافقة.

العمومية: يجب علينا عند طرح مشكلة أو مسألة أن نحاول إيجاد مسألة أعم تحوي المسألة المطروحة، لأنه قد يكون حل المسألة الأعم أسهل من المسألة الأصلية، ويمكن أن يكون الحل العام قابلاً لإعادة الاستخدام. وكما يمكن أن يكون الحل العام موجوداً في المكتبات البرمجية الجاهزة، ويعتمد هذا المبدأ عندما يكون الهدف من إنتاج النظام البرمجي تسويقياً.

إعادة الاستخدام: يشير هذا المصطلح إلى تطوير برمجية معتمده على استخدام أجزاء جاهزة تم إعدادها وفقاً لمعايير الجودة حيث تم اختبارها من قبل مما يزيد من إنتاجية المشروع وتوفير المزيد من الوقت، ويمكن النظر لهذا المصطلح - أيضاً - على أنه استخدام ما هو متاح فعلياً لتحقيق ما هو مطلوب (8)

(8) Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley

Shari Pfleeger, "Software Engineering - Theory and Practice", 2nd Edition.

Ronald J. Leach,, "Introduction to Software Engineering", CRC Press, 1999.

3-3 أنواع البرمجيات :

البرمجيات تحيط بنا في كل مكان : في المناطق الصناعية ، وفي الاستعمالات الخاصة ، وفي أنظمة الاتصالات ، وفي أنظمة النقل ، وفي مجالات أخرى، فالبرامج تأتي مختلفة في أشكالها وأحجامها، فمنها ما يدمج في الهواتف المحمولة ، ومنها ما تقوم بتصميم مركبات الفضاء. أما عن تصنيف البرمجيات ، نستطيع أن نميز نوعان رئيسيان:

برمجيات النظام : هي البرامج التي تمثل الأدوات التي تساعد في بناء أو دعم البرمجيات التطبيقية. وتتميز برمجيات الأنظمة بالتفاعل الكثيف مع عتاد الحاسب الآلي وباستعمال كثيف والتشارك في الموارد من قبل المستخدمين ، مثل : أنظمة التشغيل ، مترجمات اللغات ، وخلافه.

البرمجيات التطبيقية (: هي البرامج التي تساعد في تأدية بعض المهام المفيدة أو الممتعة بشكل مباشر وأمثلة ذلك : الألعاب ، برامج الصرافة الآلية (ATMs) ، برامج التحكم في الطائرة وبرامج البريد الإلكتروني وبرامج معالجة النصوص ، وبرامج الجدولة.

وضمن تصنيف البرمجيات التطبيقية ، من المفيد أن نميز أصناف البرمجيات التالية:

برمجيات الألعاب (Games Software).

برمجيات نظم المعلومات (Information Systems Software) : وهي من أهم وأوسع مجالات التطبيقات البرمجية ، حيث تقوم هذه النظم بالاتصال بقاعدة بيانات كبيرة أو أكثر تحتوي على كميات ضخمة من البيانات والمعلومات الخاصة بهذه النظم وبطريقة تسهل الحصول على المعلومات واتخاذ القرارات ، والتفاعل مع المستخدم لإتمام عملياته ببسر وموثوقية كبيرة ، ومن هذه الأنظمة على سبيل المثال : نظام حجز مقعد بالخطوط الجوية ، نظام المعاملات البنكية ونظام شؤون الموظفين والرواتب ، خلافه من الأنظمة.

برمجيات نظم الوقت الحقيقي (Real – Time Systems Software) : هي الأنظمة التي يجب أن تعطي إجابة ضمن شروط زمنية محددة ، وتتألف مكونات برمجيات الزمن الحقيقي من : مجسات لجمع وتصفية

البيانات من المحيط الخارجي ، ومحوّل يقوم بتحويل البيانات إلى الشكل الذي يقبله النظام ، بالإضافة إلى مكوّن تحكم وإخراج إلى المحيط الخارجي ، ومراقب تحكم ينسق بين جميع هذه المكونات لتستمر المحافظة على الاستجابة بالزمن الحقيقي. ومثال على ذلك برمجيات التحكم في مراكز الطاقة ، والأقمار الصناعية ، والعمليات الصناعية المختلفة.

البرمجيات المدمجة بالأجهزة (Embedded Software) : تقوم هذه البرمجيات بأداء وظائف خاصة ضمن الأجهزة الذكية من خلال برمجة ذاكرات للقراءة فقط بهذه البرمجيات وتركيبها ضمن هذه الأجهزة. مثال على هذه الوظائف : التحكم في سرعة محرك الغسالة الكهربائية ، وكذلك التحكم في اختيار برامج تشغيلها.

البرمجيات المكتبية (Office Software) : وهي برمجيات موجهة لأجهزة الحاسبات الشخصية (PC Computer) لأتمتة الأعمال المكتبية مثل : مجموعة برمجيات ميكروسوفت أوفيس كبرنامج معالجة الكلمات وتنسيق النصوص ، وبرنامج الجداول الحسابية ، وتقديم العروض التقديمية ، وخلافه . وكذلك برمجيات الإنترنت والبريد الإلكتروني ، وخلافه من هذه البرمجيات.

البرمجيات العلمية والهندسية : تتميز هذه الأنظمة بمحاكاة الأنظمة وتعتمد على الخوارزميات العددية ، وتتراوح تطبيقاتها من إيجاد جذور معادلة من الدرجة الثانية إلى تحليل إحداثيات مركبات الفضاء وإجراء التفاعلات النووية .

برمجيات الذكاء الاصطناعي : تعتمد برمجيات الذكاء الاصطناعي على مجموعة من الحقائق (Facts) والعلاقات (Relationships) . وتقوم هذه البرمجيات بتحديد الحل المناسب من خلال الاستنتاج والاستدلال المنطقي (Logical Deductions) وبكفاءة عالية ، حيث يتم تمثيل العلاقات بين الأشياء وتجميعها وتنظيمها للوصول إلى استنتاج منطقي للحقائق التي تمثلها تلك العلاقات. ومثال ذلك برمجيات التعرف على الأشكال (الصور والصوت).

البرمجيات المغلفة :

يطلق هذا المصطلح (البرمجيات المغلفة) على البرمجيات التطبيقية المُعدة للتوزيع التجاري (Commercial

Software Packages) (مثل : ميكروسوفت أوفس). ويتم تحويل البرمجيات التطبيقية المُعدة للاستخدام

الداخلي (In-House Application Software) إلى برمجيات تطبيقية مُعدة للتوزيع التجاري (برمجيات

مغلقة) بإضافة العديد من الوحدات البرمجية الجديدة الصغيرة إليها مثل برمجية حماية الشيفرة ، وبرمجية

التثبيت وبرمجية الرخص ، وخلافه ، وبعد ذلك يتم وضعها كحزمة برمجية تطبيقية تجارية قابلة للتثبيت على

أقراص مدمجة مغلقة تُعد للبيع التجاري. ومن أهم الخصائص أو الاعتبارات الهامة التي يجب أن تؤخذ في

الاعتبار عند تطوير هذه البرمجيات ما يلي :

1- سهولة التكيف مع منصات تشغيل الحاسب الآلي :

لا بد أن يراعى في عملية التطوير للحزمة بأن تكون صالحة للعمل على منصات التشغيل المختلفة للحاسب

(Platform Independent) ، وإذا كانت غير موائمة لمنصات معينه ، يجب العمل على تكييفها للعمل معها عن

طريق إضافة وحدات برمجية تقوم بهذا الغرض.

2- الخيارات الإقليمية وخيارات اللغة :

يجب أن تتوافق هذه البرمجيات مع منصات التشغيل ذات اللغات المتعددة (Multi-Languages Platforms)

عن طريق إضافة وحدة برمجية (Multi-Languages Service Pack) يتم من خلالها اختيار الخيارات

الإقليمية وخيارات اللغة بسهولة تامة مما يساعد على انتشار البرمجية وبالتالي على تسويقها.

3- إدارة عملية الترقية :

وهي العملية الخاصة بتنسيق عمليات التحديثات اللازمة للإصدارات ، وكيفية توزيع الرقيعات (Patches)

إلى العملاء لسد الثغرات التي تنشأ في الحزمة أثناء عمليات التشغيل (مثل الرقيعات الأمنية التي توزعها شركة

ميكروسوفت لسد الثغرات الأمنية التي تكتشف بعد توزيع نظام التشغيل) ، وكيفية حماية العملاء من أي فقد للبيانات أو بيانات التشكيل ، وكيفية تحديث الحزمة.

4- رخصة الاستخدام وعملية التنشيط للمنتج :

عملية تنشيط المنتج البرمجي هي عبارة عن إجرائية للتحقق من صلاحية رخصة استخدامه (License Activation Procedure) لمنع استخدام المنتج لغير المرخص لهم ، لذا يجب تضمين هذه العملية في البرمجيات التجارية للحفاظ على حق الملكية ، ويجب أن تتضمن أيضاً طريق سهلة لعملية التنشيط من قبل العملاء، ويوجد العديد من أنواع رخص التشغيل وهي رخص محددة بوقت معين وتتم لنسخ المنتج التقويمية (Time Limited Evaluation Edition) (مثل البرمجيات المشتركة التي توزع من خلال الإنترنت للاستخدام لمدة شهر) ، أو رخص استخدام غير محددة الوقت (مثل رخص التشغيل الخاصة بحزمة ميكروسوفت أوفس).

5- حماية البرنامج المصدر :

لا بد من حماية برنامج المصدر من العابثين الذين يحاولون الحصول عليه ، وذلك باللجوء إلى وسائل التشفير والحماية المختلفة.

6- وجود وسائل المساعدة :

لا بد من احتواء الحزمة على طرق لمساعدة العميل في استخدام البرنامج بمعظم اللغات العالمية المشهورة ، وكذلك إرشادات في حالة حدوث مشاكل (Troubleshooting) ، ويجب أيضاً تخصيص مواقع على شبكة الإنترنت للمنتج لتقديم المساعدة للعملاء.

تثبيت البيانات :

وفقاً للأبحاث الحديثة الخاصة بتثبيت البرمجيات وجد أن حوالي 30% من فشل حزم البرمجيات ينشأ أصلاً من عدم التثبيت الصحيح ، لذا لا بد من التفكير جيداً في طريقة قوية لعملية التثبيت لتجنب الأخطاء التي تؤدي إلى فشل التثبيت ، ولا بد من الأخذ في الاعتبار أنواع الحاسبات المختلفة ، والتفكير في العمليات غير العادية في عملية التثبيت كوضع مفاتيح للتسجيل والرخص ، وعملية التنشيط⁽⁹⁾.

(9) روجر بريسمان ، "هندسة البرمجيات" ، الدار العربية للعلوم ، مركز التعريب والبرمجة ، الطبعة الأولى ، 1425هـ - 2004م.

مهندس إياد هلاي ، "هندسة البرمجيات" ، المركز الألماني السوري لأعمال الانترنت (gsibc.net)

هندسة البرمجيات و أدوات القياس المستخدمة في هندسة البرمجيات:

3-4 المقاييس التي تستخدم لقياس تعقيد المنتج البرمجي .

1- Line Of Code (LOC).

حساب خطوط الشفرة باعتبار حجم البرنامج أفضل مؤشر لتعقيده ولكن أثبتت الدراسات قصور هذا المقياس

في العديد من المواضع , ومن عيوبه:

1- عدم تناسق (Consistent) هذا المقياس مع لغات البرمجة والتطبيقات والمطورين. وقيمه تتأثر بهذه العوامل

وليس موحده.

2- تعقيد البرنامج لا ينعكس من خلال هذا المقياس. فليس دائماً البرنامج الأصغر أقل تعقيداً والتعقيد قد يتسبب

من جوانب أخرى كما سنرى لاحقاً .

لذلك هذا المقياس لا يعتبر مؤشر جيد لتقويم الجودة.

2 - نموذج هولستد:

يعتبر من أقدم المحاولات لبناء مقاييس جودة البرمجيات, أثبتت الدراسات نجاح مؤشرات هذا النموذج في تقويم تعقيد

المنتجات البرمجية وجوانبها الأخرى .استندت مؤشرات نموذج هولستد للقياس على التركيب الداخلي للنص البرمجي .

واستندت المؤشرات على تقسيم النص البرمجي إلى مؤثرات (Operations) وعوامل (Operands)ومن تحديد

ذلك تحدد الحجم القياسي (Standard Volume) للنص البرمجي الذي يعتمد على الخوارزمية ولا يتأثر باللغة أو

عمليات التحويل أو أي مؤثر آخر.

وجميع المقاييس في نموذج هولستد للقياس تستند على المقارنات بين الحجم القياسي والحجم المثالي للنص البرمجي

(Potential Volume).

ومن عيوبه:

أهمل العديد من مواطن التأثير في التعقيد والجودة للنص البرمجي مثل تأثير نقاط اتخاذ القرار (جمل التكرار والجمل الشرطية وغيرها) وتباين تأثير البني اللغوية المختلفة وتماسك وتخلخل النص البرمجي .

3- مقياس للبيانات :

يستند مقياس مكيب على نقاط اتخاذ القرار ويهمل المكونات الأخرى وكأن ليس لها تأثير.

يستخدم هذا المقياس في تحديد تعقيد البرنامج ومتابعته .(Traceability) والهدف الأساس لمقياس مكيب هو توفير

مقياس لقابلية الفحص (Testability) وقابلية فهم البرنامج.(Understandability)

بعد ظهور مفهوم التوجه الكينوني (Object Orientation) فرض واقع جديد على فكر الجودة للبرمجيات. لذلك

سنتطرق لأهم الخصائص المميزة والمؤثرة في بناء المقاييس الكينونية التوجه:

وهناك خمس خواص تقود توجه وتخصيص المقاييس الكينونية نتعرض لها فيما يلي:

أ. التوجه المحلي (Localization): التوجه المحلي، هو خاصية البرمجيات في التعبير والإشارة (Indicate)

لسلوك المعلومات للتمركز ضمن البرنامج بينما كان تمركز المعلومات حول الوظائف في التوجه الوظيفي.

ب. احتواء كائنات البيانات :

به امتلاك الكائنات لذاتها، التي أطلقنا عليها تجريد البيانات .(ADT) لذلك ولتقويم جودة البرمجيات الكينونية

نحتاج لمقاييس تختبر احتواء الأصناف على البيانات والإجراءات الضرورية والكافية لتمثيل الوليد (أي الكائن)

الذي ينشأ بفعلها، تغير اهتمام وتركيز مبدأ القياس من الوحدة البرمجية إلى قياس رزمة من البيانات والإجراءات

تشجيع القياس كي يكون من مستوٍ عالٍ من التجريد: مثل قياس عدد العمليات (أي الوظائف) النافذة للبيانات المعرفة لصنف ما.

ج. إخفاء المعلومات :

كتم التفاصيل الإجرائية لمكونات البرنامج والسماح فقط في إباحة ما هو ضروري للتواصل مع أجزاء أخرى.

د. الوراثة للكائنات :

خاصية الإرث تمنح القدرة على اشتقاق صنف جديد بإكسابه صفات و سلوك صنف أو أكثر (كلاً أو جزءاً) موجودة فعلاً. وهذه القابلية ممكن استثمارها في مختلف المستويات للفئات المنتظمة ضمن تشكيل هرمي بفعل خاصية الإرث. و أثارت هذه الخاصية اهتمام الباحثين لبناء مقاييس لتقويمها. ومن أمثلة هذه المقاييس: عدد الأصناف الأساس وعدد الأصناف المشتقة وعمق الاشتقاق في شجرة الإرث.

هـ. أساليب تجريد الكائنات :

أسلوب التجريد آلية تتيح للمصمم التركيز على التفاصيل الأساس الضرورية لأجزاء البرامج والاستغناء عن الإسهاب والتفصيل. وكما أشار الباحثون "التجريد فكرة نسبية. فكلما ارتقينا إلى مستوى أعلى من التجريد، تجاهلنا التفاصيل شيئاً فشيئاً. وكلما انحدرنا إلى المستويات الدنيا، حددنا تفاصيل الفكرة أو المبدأ بشكل

أدق (10).

(10) أسماء المنقوش ، "دورة هندسة البرمجيات" ، جامعة القاهرة ، مصر ، 2009.

مهندس عبد الحميد بسيوني ، "أساسيات هندسة البرمجيات" ، دار الكتب العلمية للنشر والتوزيع ، القاهرة ، 2005 م.

3-5 المعايير الدولية لجودة البرمجيات :

هي مجموعة القواعد والإجراءات والخطوات والعمليات التي وضعت من قبل جهات عالمية مختصة لكي تحقق في حالة الالتزام بها أن يتم التفكير والتخطيط للبرمجيات ثم تصميمها وإنتاجها بطريقة واضحة تحدد مدي كفاءة البرنامج في العمل وخلوه من العيوب والمشكلات المختلفة واتساقه التام مع الهدف الموضوع من أجله، وإتباع معايير واضحة ومعتمدة في أسلوب التصميم والبناء وكتابة كود البرنامج بحيث يسهل بعد ذلك تصنيف البرنامج من حيث مستوي جودته وقدرته علي العمل في الظروف المختلفة، وأن يكون البرنامج موثقا توثيقا كافيا وجيدا في مجموعة وثائق ورقية أو الكترونية بشكل يشرح كل تفاصيله وطريقة بنائه بحيث يمكن لأي مطور أو مدير مشروع أن يعمل بهذا البرنامج دون اللجوء بشكل أساسي لمن قام بالتصميم.

ولا تقتصر معايير الجودة علي البرنامج نفسه من الناحية التقنية بل تشمل كل شيء داخل شركة البرمجيات من العمليات الإدارية والتنظيمية المتبعة داخل الشركة وكيفية وضع مشروع لإنتاج برنامج معلومات وكيفية متابعتها وحتى كيفية التعامل مع المبرمجين في أمور التدريب الداخلي الحضور والانصراف وغيرها.

معييار ايزو:ISO

صدرت المعايير القياسية للأيزو بخصوص إنتاج وصيانة البرمجيات عام 1991. والقراءة السريعة لهذه المعايير تؤكد أن هناك الكثير مما يجب إنجازه من قبل الحاسوبيين العرب لتحقيق هذه المعايير خاصة في المجالات التالية: مسؤولية الإدارة - نظم الجودة - مراجعة العقود - تخطيط التطوير - تخطيط الجودة - الاختبار والمعايرة - التوريد و التركيب - الاستلام - الصيانة- متابعة التوثيق - والقياسات. ويجب على الشركات الكبرى أن تولى اهتماماً بتكوين كوادر متخصصة في هذا المجال.

أسس التقييس وفقا للمنظمة الدولية:ISO (International Organization for Standardization)

بني التقييس على أربعة أسس :وفيما يلي تلخيص ماذا يعني كل من هذه الأسس:

1- تبسيط البيانات :

عرفته المنظمة الدولية للتقييس (S.O) بأنه : "اختصار عدد نماذج المنتجات إلى العدد الذي يكفي لمواجهة الاحتياجات السائدة في وقت معين ، وذلك عن طريق اختصار أو استبعاد النماذج الزائدة أو استحداث نموذج جديد ليحل محل نموذجين أو أكثر على ألا يخل ذلك بحاجة المجتمع .

ويهدف التبسيط إلى عدم تعدد وتنوع النماذج المختلفة من السلع شائعة الاستعمال ، لما في ذلك من إسراف في التكاليف ، وزيادة في الجهود الإنتاجية ، لذا فهو يؤدي إلى زيادة في حجم الإنتاج وخفض التكاليف ، مع تحسين كبير في الخدمات المتاحة له من حيث توفر السلع والسرعة في استلامها ، وسهولة إصلاحها وصيانتها ، بالإضافة إلى ارتفاع مستوى وخفض رأس المال المستثمر نتيجة لتقليل الآلات والمعدات وقطع الغيار المستخدمة في الإنتاج.

2- التتميط :للبيانات:

عرفته المنظمة الدولية (International Organization for Standardization)I.S.O بأنه : "توحيد مواصفتين أو أكثر لجعلها مواصفة واحدة حتى يمكن للمنتجات الناتجة أن تكون قابلة للتبادل عند الاستخدام " ولقد أدخل التتميط تطورا هائلا على أساليب الصناعة فأليه يرجع الفضل الأكبر في إمكان الإنتاج على نطاق واسع وهو يؤدي عامة إلى نتائج مماثلة لما يؤدي إليه التبسيط فهو يقلل من مساحة التخزين ، ويزيد من دوران الموجودات بالمخازن ، فيقل بذلك حجم المخزون الراكد كما أن له تأثيرا كبيرا في تبسيط القيد في السجلات . كذلك فهو يؤدي إلى زيادة الإنتاجية والى تيسير احكم ضبط الجودة وتحقق كل هذا المزايا خفضا كبيرا في تكاليف الإنتاج مع الارتفاع بمستوى جودته .

3- توصيف البيانات:

عرفته المنظمة الدولية للتقييس (I.S.O) بأنه : " البيان الموجز لمجموعة المتطلبات التي ينبغي تحقيقها في منتج أو مادة أو عملية ما مع إيضاح الطريقة التي يمكن بواسطتها التحقق من استيفاء هذه المتطلبات كلما كان ذلك ملائماً. "

فالتوصيف يعني تحديد خصائص المواد والمنتجات وكذلك الطرق والوسائل الكفيلة لتحقيق توفر هذه الخصائص ، وقد لا يكون هذا التحديد يسيرا فقد يستلزم مثلا الاستعانة بكثير من الرسومات الهندسية أو المنحنيات أو الجداول وقد يحتاج إلى إجراء الكثير من البحوث الصناعية ، ولذلك فإن تحقيق مبدأ الحرية المطلقة يصبح ضروريا لإطلاق الحرية للتطورات التقنية عن طريق عدم التدخل في طرق التصنيع ما أمكن ، ويتم بدلا من ذلك التركيز على مستوى الأداء للسلعة ، فتحديد الحدود الدنيا لمقاومة الضغط أو الثني في نوع معين من الصلب مثلا أفضل كثير من النص على أسلوب تصنيعه.

وقد أزال هذا المبدأ التناقض الذي يمكن أن يحدث نتيجة التطور التقني واصطدامه بقيود تفرضها المواصفات وأزال عن التقييس دعوى وقوفه حجر عثر في سبيل التطور أو تقليصه حرية المنتج والمستهلك في اختيار السلعة التي تتلاءم مع أغراضه .

4- تحقيق الملاءمة للاستعمال:

ويتخلص هذا التحقيق في أن الجودة ليست مطلقة وإنما يجب أن ترتبط بظروف الاستخدام . فما هو جيد في مكان معين وتحت ظروف معينة قد يكون غير جيد في أمكنة أخرى أو تحت ظروف مخالفة .

ونظرا لضرورة هذا المبدأ فإنه يجب الاهتمام بوضع المواصفات الوطنية في كل بلد دون نقل للمواصفات الأجنبية مهما كان مشهورة⁽¹¹⁾.

3-6 أهداف التقييس وفوائده :

أن الأسس الأربعة السابقة والتي يضمنها التقييس لها آثار بعيدة المدى في جميع أنشطة الحياة . فالتقييس ليس غاية في حد ذاته بل انه وسيلة فعالة لتحقيق أهداف ضخمة من أهمها:

1- خفض التكاليف:

إنه من الطبيعي أن يتحقق خفض في تكاليف الإنتاج نتيجة لخفض الأموال المستثمرة فيما يلي :

شراء آلات ومعدات ذات كفاءة عالية.

خفض سعر شراء الخامات والمواد نتيجة لشرائها بكميات كبيرة.

وفر في النفقات الإدارية نتيجة لتقليل وتبسيط الإجراءات المكتبية.

(11) مهندس إباد هلاي ، "هندسة البرمجيات" ، المركز الألماني السوري لأعمال الانترنت [5] (gsibc.net) أحمد شعبان دسوقي وآخرون ، "أساسيات الحاسب الآلي

وتطبيقاته في التعليم" ، مكتبة الرشد ، الرياض ، الطبعة الأولى ، 1427هـ - 2006م.

2 - زيادة الكفاية الإنتاجية :

إن الاقتصاد على عدد محدد من النماذج والأنواع يؤدي إلى طول فترات تشغيل الآلات أي إلى زيادة في إنتاجيتها ، كذلك فإن انخفاض عدد العمليات الصناعية يؤدي إلى زيادة كفاءة العمال والآلات على حد سواء ، بالإضافة إلى أن تحسين ضبط الجودة يؤدي إلى تخفيض نسبة المرفوضات أي زيادة الكفاية الإنتاجية.

3- تحسين جودة المنتجات :

إن تركيز أعمال التصميم والإنتاج على عدد أقل من المواد والأجزاء ، وازدياد خبرة العمال قد هياً للإنتاج مستوى عال من الجودة بالإضافة إلى انه أمكن اقتناء أجهزة اختبار دقيقة وثمينة ، كان من الصعب شراؤها في حالة صغر حجم الإنتاج نظرا لارتفاع ثمنها وعدم وجود مبرر اقتصادي لذلك . وبالطبع فان استخدام مثل هذه الأجهزة الدقيقة يعمل على أحكام ضبط الجودة ورفع مستواها.

4- الحفاظ على المواد والموارد :

إنه من الطبيعي أن يحقق التقييس وفرا كبيرا في الخامات والمواد للأسباب التالية:

تحسين تصميم المنتجات نتيجة التركيز على إنتاج عدد أقل من الأنواع والأحجام والمقاسات.

حسن استغلال المواد مع استخدام المواد البديلة نتيجة للأبحاث اللازمة قبل وضع المواصفات.

5 -التبادلية لبيانات البرمجيات:

كان نتيجة التبسيط هي انخفاض التنوع في المقاسات والأحجام والنماذج ، ولقد فرض هذا الانخفاض مبدأ التبادلية أي قدرة الصانع على إنتاج عدد كبير من الأجزاء المتماثلة في الحجم والشكل والأداء إلى حد يضمن استبدال جزء منها بجزء آخر له نفس درجة الأداء.

وحيث انه لا يمكن لجزأين أن يتماثلا تماما فمن واجب التقييس أن يحدد التفاوت المقبول مع المحافظة على قابلية التبديل.

6- السلامة للبيانات:

يوجد العديد من المقاييس المنتجات التي أعدت خصيصا لحماية حياة الإنسان وصحته ، ومن أمثلتها أحزمة المسافرين في السيارات والملبوسات الواقية في مجال الصناعة ، وأحزمة النجاة لاستعمالها في البحر وغيره(12).

3-7 جودة البرمجيات :

أصبحت مصطلحات الجودة وتأمين الجودة للمنتجات في العالم الصناعي اليوم مهمة وشائعة الاستخدام وخاصة في صناعة البرمجيات وذلك للاعتماد المطلق على هذه الصناعة في جميع مناحي الحياة. وكحد أدنى يمكننا القول بأن برمجية ما عالية الجودة إذا ما اتسمت بالسمات العامة التالية :

تلبية احتياجات المستخدم , خالية من العيوب , يمكن الاعتماد عليها في العمل , يمكن إجراء صيانة لها .

وتُعرف الجودة (Quality) وفقاً لمقياس أيزو المعياري "ISO Standard 8204" بأنها : المجموع الكلي لميزات وخصائص منتج أو خدمة ما والتي تؤثر على قدرته / قدرتها على تلبية حاجات المستخدم المحددة أو الضمنية.

فمثلاً نعني بجودة التصميم (Quality of Design) مجموعة الميزات التي يحددها المصممون لمنتج ما ، فعندما يُطلب إنتاج منتج ذي مواصفات أداء عالية ، لا مناص من زيادة جودة تصميم المنتج إذا جرى تصنيعه وفقاً للمواصفات المطلوبة. أما جودة التوافق (Quality of Conformance) فهي درجة إتباع مواصفات التصميم خلال مرحلة التصنيع ، وكلما ازدادت درجة التوافق ارتفع مستوى جودة التوافق ، وبالتالي جودة المنتج.

(12) Ian Somerville , "Software Engineering", Addison Wesley, 2001.

ويعرف المنتج ذو الجودة - أيضاً - بأنه المنتج الذي يلبي ويستمر في تلبية الاحتياجات التي من أجله تم إنتاجه.

والجودة لها سمة أساسية وهي أنها يمكن قياسها ، وفي هندسة البرمجيات هذه القياسات يشار لها بالمقاييس

(Metrics) ، فعلى سبيل المثال يمكن قياس التكاليف والحجم الذي تشغله البرمجية ، وتعقيد البرمجية.

ولإنتاج برمجيات ذات جودة عالية يجب الأخذ في الاعتبار مجموعة من السمات الأساسية من أهمها :

مدى كبر أو حجم البرمجية ،مدى تعقيد عناصر مكونات البرمجية ،مدى الاعتماد على البرمجية.

تكاليف إنتاج البرمجية ، الجهد اللازم لتغيير البرمجية لتلبي احتياجات المستخدم.

حيث إن ذلك يسهم إلى حد كبير في : التنبؤ بجودة البرمجية خلال مرحلة تطويرها أو إنتاجها.

قياس جودة جزء من برمجية تم إنتاجها ، مراقبة وتحسين عملية إنتاج البرمجية ، المقارنة بين الطرق المختلفة

لإنتاج البرمجية واختيار الأفضل منها.

ويمكن الأخذ في الاعتبار سمات الجودة سابقة الذكر في واحد أو أكثر من الأوضاع التالية :

1. خارج عملية التطوير لتوضيح الأهداف.

2. خلال عملية التطوير لتوجيه عملية التطوير لتحقيق الأهداف.

3. في النهاية لتقويم إنتاج البرمجية.

ويمكن إيجاز أهم السمات التي تحدد مدى جودة البرمجيات فيما يلي:

1. الصحة البرمجية: وهي تمثل المدى التي تفي فيه البرمجية بمتطلبات المستخدم.

2. الاعتمادية للبرمجيات: وهي تمثل الحد الذي يمكن أن تعمل فيه البرمجية باستمرار دون حدوث توقف أو

فشل.

3. الكفاءة البرمجية : وهي كمية RAM ووقت المعالج الذي تستخدمه البرمجية.

4. التكاملية للبيانات : وهي الدرجة التي تمكن فيها البرمجية المستخدم من الدخول وتنظيم إدخال المعلومات.

5. الاستخدام : سهولة استخدام البرمجية.

6. الصيانة : وهي تعبر عن الجهد المطلوب لاكتشاف خطأ وإصلاحه.

7. المرونة: وهي تعبر عن الجهد اللازم لتغيير البرمجية لمقابلة التغيرات المطلوبة.

8. الاختيارية : وهي تعبر عن الجهد اللازم لاختبار البرمجية بفعالية.

9. الانتقالية: تعبر عن الجهد اللازم لنقل البرمجية إلى حاسبات ذات مكونات وأنظمة تشغيل مختلفة.

10. إعادة الاستخدام: تعبر عن الحد الذي يمكن به إعادة استخدام البرمجية أو جزء منها من خلال برمجية أخرى.

11. تضافرية العمل : تعبر عن الجهد اللازم لجعل البرمجية تعمل بالتعاون مع برمجية أخرى.

وكما هو ملاحظ فإن هناك العديد من هذه السمات متناقض مع بعضه فعلى سبيل المثال لإنتاج برمجية ذات كفاءة عالية فإن إمكانية النقل أو الحمل ربما تتناقض معه ، ولهذا فإن لكل مشروع تحقيق مجموعة من الأهداف المحددة توضع في الاعتبار قبل عملية التطوير والإعداد.

ولكن عندما تنتهي من إنتاج برمجية كيف تتأكد من جودتها من عدمه؟ ، هناك طريقتان للإجابة عن هذا التساؤل :

(1) قياس خواص البرمجية التي تم إنتاجها.

(2) مراقبة وتنظيم عملية تطوير البرمجية.

ولتوضيح الرؤيا دعنا نلاحظ أوجه الشبه بين إنتاج برمجية وإعداد وجبة غذائية من الطعام، فإذا أردت إعداد وجبة معينة فإنك تقوم بالخدمة عليها وبناءً عليه تحصل على وجبة ذات جودة يمكن معرفتها بقياس الطعم ، ورؤية اللون ، وساخنة أم لا ، وإذا كان فيها شيء غير مضبوط فإنه ليس هناك شيء يمكن عمله لضبط جودتها ، وعلى هذا فإنه يعاد التحكم في الكميات للمكونات والطريقة لضبط جودة الوجبة، ولإعداد وجبة ذات جودة يجب أن تمر عملية الإعداد بعدة مراحل بداية من ناحية إعداد المكونات وحتى الطهي بمراحله المختلفة ، وفي كل خطوة يمكن تصحيح الخطأ إذا ما تم شيء غير صحيح على سبيل المثال يمكن شراء مكونات جديدة إذا ما اتضح أن أحدها

فاسد أو غير طازج أو أننا نعيد غسل أحد المكونات إذا ما أتضح أن نظافته غير كافية ، وهكذا فإن جودة الوجبة يمكن تأكيدها بعمل اختبارات خلال عملية الإعداد، ويمكن تطبيق نفس المبدأ في عملية تطوير البرمجيات حيث يمكن إجراء قياسات ومراجعات مماثلة للتأكد من جودة البرمجية المنتجة. وكذلك فإنه لإعداد وجبه يلزم (وصفة للإعداد) والتي يمكن إتباعها وهذا مماثل تماما لاستخدام وسائل وطرق جيدة خلال عملية تطوير البرمجية.

8-3 ميزات أدوات هندسة البرمجيات بمساعدة الحاسوب :

يمكن تلخيص أهم ميزات أدوات هندسة البرمجيات كالتالي :

1- السرعة الفائقة (Increased Speed) : تتميز أدوات هندسة البرمجيات بالسرعة الفائقة في أتمتة العمليات وتخفيض الوقت اللازم لاستكمال العديد من المهام خاصة التي تتطلب المخططات الرسومية والبنود الخاصة بها. وتقدر التحسينات في الإنتاجية بعد التطبيق في مدى يتراوح من (200 - 300%) .

2- الدقة المتزايدة : تساعد أدوات هندسة البرمجيات في اكتشاف الأخطاء والذي تُعد خطوة مهمة في إزالة هذه الأخطاء حيث يوفر اكتشاف الأخطاء وإزالتها الوقت والجهد في المراحل المبكرة من إعداد النظام ، و كلما زاد حجم النظام أصبح من الصعب اكتشاف الأخطاء مما يؤدي إلى زيادة الوقت والجهد المطلوب للتنسيق والإدارة بين فرق العمل المختلفة.

3- تخفيض الوقت اللازم للصيانة: وكنتيجة للتحليل الجيد، التصميم الجيد، توليد الشفرات آليا، اختبار البرمجيات آليا، التوثيق الآلي، يتحسن أداء النظام وبالتالي فإن الجهد اللازم للصيانة ينخفض إلى حد كبير. وكذلك يمكن توفير العديد من المنابع لتطوير أنظمة جديدة ، وتقوم أدوات مساعدة البرمجيات (برمجيات إعادة الهندسة Re-engineering Programs) باكتشاف الأجزاء من البرمجيات والتي يمكن إعادة استخدامها مما يزيد من كفاء العمل وتقليل الجهد المطلوب.

4- التوثيق الأفضل: وباستخدام أدوات هندسة البرمجيات المساعدة هناك العديد من كميات التوثيق التي يتم توليدها لتمثل ملاحظات على كيفية تطوير النظام وعمل صيانة له.

5- البرمجة في أيدي غير المبرمجين :

مع التطور السريع والاتجاه نحو تكنولوجيا البرمجة الشيئية ، وقواعد بيانات خادم العميل يمكن أن تتم عملية البرمجة بأناس ليس لديهم خلفية كاملة عن البرمجة، حيث يصبح المهم هو فهم الهدف الأساسي من البرنامج والقدرة على تحليل مكونات البرنامج وتفصيله والتي تستخدم في توليده عن طريق هذه الأدوات (أدوات التطوير منخفضة المستوى).

5- الفوائد الغير ملموسة: تفيد أدوات التطوير في مشاركة المستخدم والتي تساهم في قبوله الجيد للنظام الجديد وهذا يساهم إلى حد كبير في تخفيض منحى التعليم الأولي (13) .

(13) Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.

Shari Pfleeger, "Software Engineering - Theory and Practice", 2nd Edition.

3-8-1 قصور أدوات هندسة البرمجيات بمساعدة الحاسوب :

يمكن تلخيص أهم القصور الموجودة في أدوات هندسة البرمجيات بمساعدة الحاسب في التالي :

1- مزج الأدوات: من المهم جدا اختيار ملائم لمزيج من الأدوات للحصول على مميزات وتكاليف مخفضة. ولا بد من اختيار الأدوات غير المعتمدة على نوعية العتاد وإمكانية مشاركة نتائج أداة تطوير تم استخدامها مع أدوات أخرى لضمان تكاملية الأدوات مع بعضها.

2- التكاليف: الأدوات المساعدة ليست رخيصة الثمن ، وفي الحقيقة فإن شركات التطوير على نطاق ضيق لا تستخدم هذه الوسائل معتقدين بأنها مفيدة فقط في تطوير الأنظمة الكبيرة ، حيث أن تكلفة تزويد مطوري الأنظمة بهذه الوسائل مكلف. ويعتبر (العتاد ، البرمجيات ، الاستشارات) كلها عوامل تدخل في معادلة التكلفة.

3- منحى التعلّم: وفي معظم الحالات فإن إنتاجية المبرمج تصل إلى أدنى مرحلة لها في المرحلة الأولية للتطبيق وذلك لاحتياج المستخدمين للوقت الكافي للتعلم ، بالرغم من أن مستشاري هذه الأدوات يقدمون العديد من الخبرات لمستخدميها حيث يقومون بالتدريب وكذلك وجود العديد من مواقع الخدمات الخاصة بهذه الأدوات على شبكة الإنترنت والتي تساعد في تعجيل منحى التعليم لهذه الأدوات.

3-8-2 التصنيف العام لأدوات هندسة البرمجيات بمساعدة الحاسوب :

1- أدوات التطوير المساعدة عالية المستوى:

وهي تمثل أدوات أنشطة العمليات المبكرة للمتطلبات ، التحليل والتصميم. وتساعد المحللون في تخزين ، وتنظيم ، وتحليل نماذج العمل ، وترتيب الأسبقيات التالية :

• إستراتيجيات العمل الحالية والمستقبلية.

- النظم المكتملة وإستراتيجيات تطبيقها.
- قواعد البيانات والشبكات المراد تطويرها.
- التطبيقات المراد تطويرها.

3-8-3 التصنيف العام لأدوات هندسة البرمجيات المساعدة

1- أدوات التطوير المساعدة منخفضة المستوى:

وهي تمثل أدوات دعم الأنشطة التالية مثل البرمجة واكتشاف العلل والاختبارات حيث تساعد المبرمجين في زيادة الإنتاجية والجودة ، و تمتد هذه الأدوات إلى تفاصيل التصميم للمساعدة في توليد التطبيقات من خلال الخدمات التالية :

- المساعدة في سرعة اختبار البرنامج واكتشاف الأخطاء.
- توليد الكود للبرنامج من مواصفات التحليل والتصميم.
- توليد الشاشات وقواعد البيانات.

2- أدوات هندسة البرمجيات بمساعدة الحاسب لدورة حياة التطوير المتقاطعة Cross Life Cycle Case Tools

وهي تشمل وسائل إدارة المشاريع التي تساعد المديرين في (التخطيط ، وضع الجداول الزمنية ، وإعداد التقارير ، وتوزيع الموارد).

والجدول رقم (2.6) يوضح أهم تصنيفات أدوات هندسة البرمجيات بمساعدة الحاسب وأمثلة عليها.

تصنيف أدوات هندسة البرمجيات بمساعدة الحاسب وأمثلة عليها

جدول (2.6) أهم تصنيفات أدوات هندسة البرمجيات بمساعدة الحاسب

أمثلة على الأداة Examples	نوع الأداة Tool Type
أدوات بيرت (PERT Tools) أدوات التقييم (Estimation Tools) الجدول الإلكترونية (Spreadsheets)	أدوات التخطيط (Planning Tools)
محررات النصوص (Text Editors) معالجات الكلمات (Word Processors) محررات المخططات (Diagram Editors)	أدوات التحرير (Editing Tools)
أدوات تتبع المتطلبات (Requirements Traceability Tools) نظم تغيير التحكم (Change Control Systems)	أدوات إدارة التغيير (Change Management Tools)
نظم إدارة الإصدار (Version Management Systems) أدوات بناء النظم (System Building Tools)	أدوات إدارة التكوين (Configuration Management Tools)
اللغات عالية المستوى (Very High-level Languages) مولدات واجهات المستخدم (User Interface Generators)	أدوات النمذجة الأولية (Prototyping Tools)

أمثلة على الأداة Examples	نوع الأداة Tool Type
<p>محررات التصميم (Design Editors)</p> <p>قواميس البيانات (Data Dictionaries)</p> <p>مولدات الشفرة (Code Generators)</p>	أدوات دعم الطرق (Method Support Tools)
المتجمات (Compilers) المفسرات (Interpreters)	أدوات معالجة اللغات (Language Processing Tools)
<p>مولدات التداخل (Cross reference Generators)</p> <p>المحلات الإستاتيكية (Static Analyzers)</p> <p>المحلات الديناميكية (Dynamic Analyzers)</p>	أدوات تحليل البرامج (Program Analysis Tools)
<p>مولدات اختبار البيانات (Test Data Generators)</p> <p>مقارنات الملفات (File Comparators)</p>	أدوات الاختبار (Testing Tools)
نظم التصحيح التفاعلية (Interactive Debugging Systems)	أدوات اكتشاف وتصحيح الأخطاء (Debugging Tools)
<p>برامج شكل الصفحة (Page Layout Programs)</p> <p>محررات الصور (Image Editors)</p>	أدوات التوثيق (Documentation Tools)
<p>نظم التداخل (Cross-reference Systems)</p> <p>نظم برامج إعادة الهيكلة (Program Restructuring Systems)</p>	أدوات إعادة الهندسة (Re-engineering Tools)

المصدر: Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.

المبحث الثاني : التحديات الرئيسية والمراحل التي تواجه هندسة البرمجيات :

تواجه هندسة البرامج ثلاثة تحديات كبرى في القرن الحادي والعشرين ، وهي :

1- التحدي المتعلق بالموروث :

لقد تطورت غالبية الأنظمة البرمجية الرئيسية المستخدمة حالياً منذ سنوات عديدة خلت ، ورغم ذلك لا تزال تلك الأنظمة تؤدي وظائف حساسة. إن التحدي المتعلق بالموروث هو التحدي الذي يعني أساساً بصيانة وتحديث البرامج بطريقة يمكن معها تفادي التكاليف الباهظة والاستمرار في الوقت ذاته في تقديم الخدمات الأساسية في قطاع الأعمال.

2- تحديد عدم التجانس :

تزداد الحاجة وباستمرار إلى أنظمة تعمل كأنظمة موزعة عبر شبكات تشتمل على أنواع مختلفة من الحاسبات الآلية ذات الأنظمة المساندة المختلفة وهذا التحدي يعوق تطوير أساليب إعداد برامج يمكن الاعتماد عليها بدرجة كافية للتعامل مع هذا التحدي.

3- تحد المخرجات :

4- تُعد العديد من الأساليب المتبعة في هندسة البرامج مضيعة للوقت. كما أن الزمن الذي تستغرقه يحد ويشكل كبير من فرص تطوير البرامج وضمان جودتها . على الرغم من كل ذلك ، فإنه ينبغي على قطاع الأعمال والتجارة في أيامنا هذه أن يتميز بسرعة الاستجابة والتغيير السريع، كما ينبغي أن تتغير وبنفس الوتيرة برامج المساندة الخاصة بها. إن تحدي المخرجات هذا يرتبط بالزمن اللازم للحصول على مخرجات من أنظمة كبيرة ومعقدة دون التفريط في الجودة والنوعية. وبالطبع فإن هذه المسألة ليست مسألة منفصلة فقد يلزم مثلاً إجراء تغيير سريع في نظام قديم العهد من أجل جعل النفاذ إليه من خلال الشبكة أمراً سهلاً.

وللتعامل مع هذه التحديات يلزمنا توفر أدوات وأساليب جديدة واستخدام طرق إبداعية من أجل القيام بالبرمجة واستخدام الطرق الحالية في هندسة البرمجيات جنباً إلى جنب مع تلك الطرق الإبداعية.

3-10 مهنة البرمجة:

مهنة البرمجة من المهن الهامة والمطلوبة في السوق العربية بصفة خاصة والعالمية بصفة عامة ، ولكن بشرط أن يكون المبرمج على كفاءة عالية وقدرة على استخدام معظم الأدوات البرمجية وتوظيفها بصورة مثالية، ومن يريد العمل بمهنة البرمجة يجب أن يكون مستخدماً متمرساً للحاسب الآلي وله خبرة طويلة في التعامل مع شتى أنواع برمجياته ليس فقط كمستخدم عادي ولكن كشخص قادر على فهم كيفية تصميم وإنشاء هذه البرمجيات ، مع القدرة على اكتساب المهارات الخاصة بأدوات البرمجة وتطويرها ، وكذلك يكون له عقل يجيد التعامل مع الأسس الرياضية ، حيث إن مهنة البرمجة لا تعتمد على مجرد أداء المهام فقط وإنما تتطلب فكراً خصباً وذهناً حاضراً وحباً للإبداع في العمل والمثابرة عليه.

ولتصميم وتطوير البرامج والتطبيقات بطريقة قياسية ، يجب أن يتكون فريق العمل الخاص بذلك من التالي :

محللو النظم (System Analysts): وهم الأشخاص القائمون علي دراسة وتحليل متطلبات النظام ومدخلاته ومخرجاته، وكذلك تحديد الموارد اللازمة لتنفيذه، بالإضافة إلى بيان كيفية التنفيذ وشرح ديناميكية العمل وتنظيم العلاقات المختلفة بين الكائنات الموجودة بالنظام .

مصممو النظم (System Designers) : يأتي دورهم بعد مرحلة التحليل وتحديد الاحتياجات، حيث يكون النظام بحاجة الآن إلي كيفية التطبيق من حيث الشكل العام وتصميم كائنات ونماذج النظام وبنية كل كائن علي حده. وفي هذه المرحلة يتم تصميم نماذج وأشكال الشاشات ومواضعها وطرق عرضها وربطها مع بعضها البعض.

المبرمجون (المطورون) (Programmers) (Developers) ويأتي دورهم بعد مرحلتي التحليل والتصميم حيث يتم التنفيذ الفعلي للنماذج والشاشات المصممة وكتابة الشفرات (Source Code) المسؤولة بدورها عن تشغيل النظام .

فمثلاً إذا كنا بصدد إنشاء نظام لإدارة شركة ما من الناحية المالية والتجارية، فسيقوم المحللون بدراسة الدورة المستندية لهذه الشركة، وكيفية تعاملها مع الشركات الأخرى، وديناميكية العمل من حيث المستندات المستخدمة في دورات العمل المختلفة... الخ ، وكيفية تدفق البيانات من مرحلة إلي الأخرى، وبالتالي تحليل النظام ككل بشكل متكامل ثم. بعد ذلك يأتي دور المصممين حيث يتم تصميم نماذج وأشكال الشاشات ومواضعها وطريقة عرضها وربطها ببعض والتي سيبرمجها المبرمجون، وبعد ذلك يأتي دور المبرمجين حيث يتم التنفيذ الفعلي لما تم تصميمه سابقاً حيث يتم كتابة الشفرات اللازمة لإنشاء كل النماذج وربطها ببعضها ببعض.

ومن خلال ما سبق ، يتضح أن المبرمج هو الشخص القائم على كتابة الشفرات اللازمة لبث روح الحياة في النظام وجعله وحدة واحدة مترابطة يؤدي في النهاية - عند تشغيله من قبل المستخدم - جميع المهام الذي صمم من أجلها ، وبالتالي فالمبرمج هو حلقة الوصل بين الحاسب الآلي والمستخدم ، فكلاهما لا يعرف لغة الآخر ، ولكن المبرمج يعرف لغة الاثنين.

والمبرمج ليس من تعلم لغة برمجية فحسب ، بل المبرمج هو من يعرف فن البرمجة ، أي كيف يضع الإستراتيجية المناسبة في المكان المناسب ، وهذه بحد ذاتها موهبة ربانية كالرسم والنحت ، أما ما تبقى من العمل البرمجي فلا يتعدى تطويع الذخائر والأدوات البرمجية وتشكيل الفكرة في قالب ذي طابع فني برمجي لإنتاج المنتج البرمجي.

ولكي يبدأ أي شخص بامتهان مهنة البرمجة كوظيفة يجب عليه أولاً التعرف على أنواع لغات البرمجة من حيث نقاط القوة والضعف في كل منها ، وكذلك التطبيقات الخاصة بكل منها ، وتعلم مبادئ البرمجة ومفاهيمها الأساسية والمشاركة بين جميع لغات البرمجة ، وهو الهدف الأساسي من هذا الكتاب الذي بين يديك ، فإذا كنت تريد أن تبدأ في عالم البرمجة فعليك بدراسة هذا الكتاب وبالتسلسل المذكور به (14).

(14) Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.

3-11 المسؤولية المهنية والأخلاقية لمهندسي البرمجيات :

مثلهم في ذلك مثل غيرهم من المهندسين ، فإنه يتوجب على مهندسي البرمجيات القبول بحقيقة أن مهامهم تتضمن مسؤوليات أكبر من مجرد استخدام المهارات الفنية حيث أن عملهم يتم ضمن إطار قانوني واجتماعي معين،ومن الواضح أن هندسة البرمجيات مقيدة بمجموعة من القوانين المحلية والوطنية والدولية، وتبعاً لذلك فإن عليهم أن يتصرفوا بطريقة مسئولة من الناحيتين الأخلاقية والمعنوية إذا كان لهم أن يحوزوا على الاحترام كمحترفين في مجالات عملهم، ومن البديهي كذلك أن يتمسك هؤلاء المهندسون بالقواعد والمعايير المتعارف عليها في مجال الشرف والنزاهة والأمانة والاستقامة. كما ينبغي عليهم عدم استخدام مهاراتهم وقدراتهم بطريقة غير شريفة أو تسيء إلى سمعة المهنة، ومع ذلك فإن هناك مجالات لا تكون فيها معايير السلوك المقبول ملزمة بموجب القانون بل ترتبط بمفاهيم فضفاضة للمسئولية المهنية، وهذه بعض منها :

- السرية : يجب على المهندسين في العادة احترام خصوصية وسرية المعلومات الخاصة بعملائهم وأرباب العمل بصرف النظر عما إذا تم توقيع اتفاقية خاصة بالسرية أم لا.
- الاختصاص: يتوجب على المهندسين الامتناع عن إعطاء معلومات خاطئة عن مستوى تأهيلهم واختصاصهم، كما أن عليهم عدم قبول أي عمل خارج عن نطاق اختصاصهم وهم يعلمون ذلك.
- حقوق الملكية الفكرية: يجب أن يكون المهندسين مطلعين على القوانين المحلية التي تحكم استخدام الممتلكات الفكرية مثل براءات الاختراع وحقوق النشر والطبع والتأليف وغيرها، كما يجب عليهم أن يحرصوا على التأكد من حماية الممتلكات الفكرية لأرباب عملهم وعملائهم.
- إساءة استخدام أجهزة الحاسب الآلي : ينبغي على مهندسي البرمجيات عدم استخدام مهاراتهم الفنية في الإساءة إلى أجهزة الحاسب الآلي التي تعود للآخرين وإساءة الاستخدام تتراوح بين أمور تافهة نسبياً (مثل ممارسة الألعاب على جهاز صاحب العمل) وأمور خطيرة جداً (كنشر الفيروسات مثلاً).

وفى هذا الخصوص تلعب الجمعيات والمؤسسات المهنية دورا هاما ، فالمنظمات مثل : "اتحاد أصحاب الآلات الحاسبة" ، و"معهد المهندسين الكهربائيين والإلكترونيين"، و"جمعية الحاسبات البريطانية" تقوم بنشر لوائح السلوك المهني أو قواعد أخلاق المهنة. كما يتعهد أعضاء هذه المنظمات بالتقيد بتلك اللوائح والقواعد عند التوقيع على العضوية.

وينبغي على مهندسي البرمجيات الالتزام بجعل مهنة تحليل وتصميم وتطوير واختبار وصيانة البرامج مهنة محترمة وذات فائدة وجدوى، ووفق التزامهم بصحة وسلامة ورفاهية الجمهور فإن على مهندسي البرامج الالتزام بالمبادئ الثمانية التالية :

- 1- الجمهور: يجب على مهندسي البرمجيات العمل بما يتوافق مع المصلحة العامة.
- 2- العميل وصاحب العمل: يجب على مهندسي البرمجيات العمل بطريقة تخدم مصالح عملائهم وأرباب الأعمال وتكون متوافقة مع المصلحة العامة.
- 3- المنتج :: يجب على مهندسي البرمجيات التأكد من أن منتجاتهم والتعديلات المتعلقة بها تفي بأعلى المستويات المهنية الممكنة.
- 4- النزاهة : يجب على مهندسي البرامج المحافظة على التكامل والاستقلالية في آرائهم المهنية.
- 5- الإدارة : يجب على المدراء والقياديين في مجال هندسة البرمجيات المساهمة والتعهد بالالتزام بالمنهج الأخلاقي في إدارة عملية تطوير وصيانة البرمجيات.
- 6- المهنة : يجب على مهندسين البرمجيات تحسين سمعة وصورة المهنة وبما يتوافق مع المصلحة العامة.
- 7- الزملاء : يجب على مهندسي البرامج أن يكونوا منصفين وعادلين مع زملائهم وان يقدموا الدعم والمساندة لهم.

8- الذات : يجب على مهندسي البرامج المشاركة في برامج التعليم الدائم والمتعلق بممارسة مهنتهم كما أن عليهم تعزيز الأسلوب الأخلاقي في ممارسة هذه المهنة (15).

9-3-11 عمليات البرمجيات :

هي مجموعة من الأنشطة المرتبة (الطبقية) المترابطة (Consistency Activities) والقيود (Constraints) والموارد (Resources) تهدف إلى توصيف (Specifying) وتصميم (Designing) وتنفيذ (Implementing) واختبار (Testing) وصيانة وترقية (Maintenance and Evolution) منتج برمجي يفي بجميع متطلبات العميل.

3-12 جودة البرمجية :

مصطلح يتعلق بعملية تطوير المنتج البرمجي والطريقة التي تم تطويره بها بحيث تكون وفقا لمعايير مقاييس الجودة والتي يتم وضعها من خلال مجموعة مراقبة الجودة من فريق التطوير بعد تحديد متطلبات المشروع البرمجي مباشرة وقبل البدء في التنفيذ الفعلي للمشروع.

التجزئة للبيانات :

تقسيم النظام البرمجي المعقد إلى أقسام أبسط تسمى كتل.

التجريد للبيانات :

تحديد المظاهر المهمة وتجاهل تفاصيلها أثناء دراسة مسألة معينة (فصل الشيء المهم عن الشيء غير المهم في المسألة) ، وهي نظرة نسبية مرتبطة بالهدف الذي نعمل عليه.

(15)Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley..

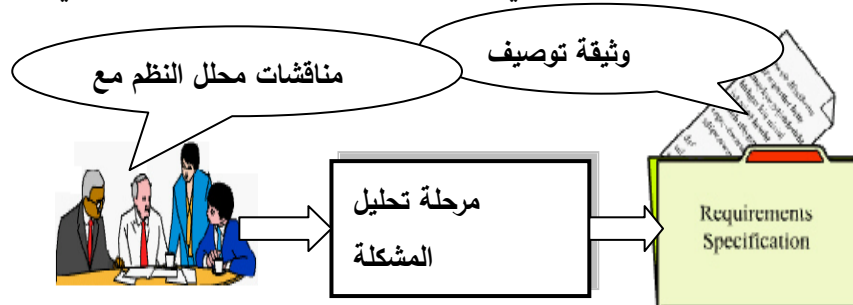
توقع التغيير:

إحصاء الأماكن التي يمكن أن يتم تعديل عليها في البرمجيات. إعادة الاستخدام :

يشير هذا المصطلح إلى تطوير برمجية معتمده على استخدام أجزاء جاهزة تم إعدادها وفقاً لمعايير الجودة حيث تم اختبارها من قبل مما يزيد من إنتاجية المشروع وتوفير المزيد من الوقت. ويمكن النظر لهذا المصطلح - أيضاً - على أنه استخدام ما هو متاح فعلياً لتحقيق ما هو مطلوب.

13-3 محللو النظم: System Analysts

وهم الأشخاص القائمون على دراسة وتحليل متطلبات النظام ومدخلاته ومخرجاته، وكذلك تحديد الموارد اللازمة لتنفيذه، بالإضافة إلى بيان كيفية التنفيذ وشرح ديناميكية العمل وتنظيم العلاقات المختلفة بين الكائنات الموجودة بالنظام . سبق وأن عرّفنا في الوحدة الأولى عمليات البرمجيات بأنها : هي مجموعة من الأنشطة المرتبة (الطبقية) المترابطة (Consistency Activities) والقيود (Constraints) والموارد (Resources) التي تهدف إلى توصيف (Specifying) وتصميم (Designing) وتنفيذ (Implementing) واختبار (Testing) وصيانة وترقية (Maintenance and Evolution) منتج برمجي يفي بجميع متطلبات العميل، وغالباً ما يقوم بهذه الأنشطة مهندسو البرمجيات. ويمكن وصف عمليات البرمجيات بوضع هيكل عام لها ، وذلك من خلال وضع عدد محدد من نشاطات الهيك والتي يمكن تطبيقها على جميع المشاريع البرمجية بغض النظر عن حجمها أو تعقيدها ، وعدد من مجموعات المهام (Task Set) اللازمة لكل نشاط هيكل ، هذا بالإضافة إلى مجموعة من نشاطات هندسة البرمجيات والتي يطلق عليها " نشاطات المظلة " ، وهي عبارة عن



الشكل (2.1): شكل توضيح لمرحلة تحليل المشكلة

المصدر: Ian Somerville , "Software Engineering", Addison Wesley, 2001.

مجموعة من الأنشطة مستقلة عن أي نشاط برمجي هيكلية وتُجرى طوال عملية بناء المنتج البرمجي (مثل : متابعة المشروع البرمجي ومراقبته ، ضمان جودة المنتج البرمجي ، إدارة تكوين البرنامج ، إدارة المخاطر ، وخلافه) ، كما سيتم توضيحه فيما بعد.

فمثلاً ، يُعد الاتصال بالعميل بنشاط هيكلية يحتاج إلى مجموعة من المهام منها على سبيل المثال : إعداد وثيقة عمل وجدول أعمال للاجتماع الرسمي مع العميل ، وإجراء أبحاث لتحديد الحلول المقترحة والطرق الموجودة لمعالجة الموضوع و تجميع متطلبات العميل ، مراجعة متطلبات العميل مع جميع المعنيين بالأمر ، وخلافه من المهام.

وتشير تلك العمليات (Software Process) . في الوقت نفسه . إلى مراحل تطوير البرمجيات. وبمفهوم علم هندسة البرمجيات (Software Engineering) يُطلق على تلك العمليات - أيضاً - اصطلاح عام وهو دورة حياة البرمجيات (Software Life Cycle) . وعلم هندسة البرمجيات هو علم يهدف إلى إنتاج برمجيات خالية من الأخطاء وذات جودة عالية في وقت محددة وبميزانية محددة ، وبطريقة اقتصادية بحيث يفي بجميع متطلبات الجهة المستفيدة.

وبغض النظر عن مجال التطبيق أو حجمه أو درجة تعقيده ، وبدون الدخول في تفاصيل النماذج (Models) التي تصف عمليات المراحل مثل⁽¹⁶⁾ Waterfall Model و Spiral Model

[1] Ian Somerville , "Software Engineering", Addison Wesley, 2001.

[2] Ronald J. Leach,, "Introduction to Software Engineering", CRC Press, 1999.

3-14 مراحل تطوير النظام :

يمكن تقسيم مراحل تطوير البرمجيات إلى المراحل الرئيسية التالية :

1- مرحلة تحليل المشكلة (Problem Analysis Phase).

2- مرحلة التطوير (Development Phase) وتشمل مرحلتي التصميم

(Design Phase) والتنفيذ (Implementation Phase).

3- مرحلة الاختبار وتشخيص الأخطاء (Testing and Debugging Phase).

ويأتي بعد كل هذه المراحل أهم مرحلة و أطولها وأكثرها تكلفة وهي :

مرحلة صيانة وترقية البرنامج (Maintenance Phase).

وتُعد نتائج كل مرحلة من هذه المراحل معطيات للمرحلة التي تليها ، فمثلاً نتائج مرحلة تحليل المشكلة هي

معطيات لمرحلة التصميم ، ونتائج مرحلة التصميم هي معطيات لمرحلة التنفيذ ، وهكذا وسوف نتناول الآن هذه

المراحل بإيجاز وبدون الدخول في التفاصيل الخاصة بكيفية إنجاز كل مرحلة ،

3-14-1. المرحلة الأولى: مرحلة تجميع المتطلبات

تركز هذه المرحلة (أو العملية) على دراسة متطلبات حل المشكلة ، حيث إنها تُعد مرحلة التوصيف الدقيق للمشروع البرمجي وتهيئته لمرحلة التصميم النهائي، ويتم ذلك عبر إعداد مجموعة من المخططات (Charts) والنماذج (Models) واللوائح (Lists) من خلال طرح مجموعة من الأسئلة المتخصصة من قبل محلل النظم للعميل لمعرفة كافة احتياجات وشروط المشروع البرمجي ، وذلك طبقاً لمنهجية محددة تسمى منهجية التحليل، وتنتهي هذه المرحلة بكتابة وثيقة رسمية (Formal Document) تشمل على المواصفات الفنية للمشروع البرمجي بناءً على متطلبات العميل مثل نطاق المعلومات الخاص بالمشروع (Information Domain) و الأداء الوظيفي (Functionality) ، الربط البيئي بين البرمجية والمستخدمين (User Interface) ، ومستويات الأمن المطلوبة (Security Levels) ، بالإضافة إلى المتطلبات الاستثنائية (Exceptional

(Requirements) .ويطلق علي هذه الوثيقة "وثيقة توصيف المتطلبات " (Requirements Specification Document) ، والتي ستكون بمثابة العقد النهائي بين مطوري المشروع البرمجي والعميل. والشكل رقم 2.1 يمثل شكل توضيحي لهذه المرحلة ، حيث إن معطيات هذه المرحلة تتمثل في متطلبات العميل ونتائجها تتمثل في كتابة "وثيقة توصيف المتطلبات"

وبصفة عامة تتضمن مرحلة (أو عملية) تحليل المشكلة التعريف بالمشكلة من خلال العديد من النواحي والتي من أهمها :

- المدخلات (Inputs): يجب على المحلل معرفة الطرق التي يفضلها العميل في عملية إدخال البيانات إلى النظام ، فهل عملية إدخال البيانات سوف تتم عن طريق القراءة من ملف ، أم سوف تكون تفاعلية بين المستخدم والنظام عن طريق قوائم اختيار بواسطة الفأرة ، أو تكون تفاعلية ولكن بطريقة مفتوحة عن طريق لوحة المفاتيح ، أو غيرها من طرق إدخال البيانات.
- المخرجات (outputs): يجب - أيضاً - على المحلل الاهتمام بمعرفة متطلبات العميل من حيث شكل وهيئة نماذج المخرجات (تقارير النظام) وصيغها المختلفة ، هل تكون على شكل أرقام فقط أو أرقام وحروف ، أو خليط من الأرقام والحروف والرسومات ، أو ما شابه ذلك.
- المعالجة (Processing) : يجب أن يدقق المحلل في كيفية عملية المعالجة من حيث نوعية البيانات والحسابات والقرارات التي سوف تعمل على هذه البيانات لإنجاز المخرجات المطلوبة ، وكذلك من حيث بيئة التشغيل التي سوف يتم تشغيل النظام من خلالها (نظام التشغيل (Operating System) ، وهل عملية التشغيل سوف تتم على حاسب آلي شخصي (Personal Computer) ، أو من خلال استخدام الشبكات المحلية. (LAN)⁽¹⁷⁾

[1]Bernstein, P., and Goodman, N, Concurrency Control in Distributed Database system, VLDB, 2000

[2]Bernstein, P., Hadzilacos, V., and Goodman.N, Concurrency Control and Recovery in Database Systems, Addison Wesley, 2008.

3-14-2 المرحلة الثانية مرحلة التطوير :

تنقسم مرحلة التطوير إلى مرحلتين هما :

مرحلة التصميم :

تُعد وثيقة توصيف متطلبات العميل هي معطيات هذه المرحلة (أو العملية) ، حيث يتم توزيع المهام على المختصين من فريق العمل من المصممين ، الذين يقومون في هذه المرحلة بتحديد الخوارزميات التي سوف يستخدمونها ، وكذلك رسم مخططات التصميم التي تتناسب مع المتطلبات المتفق عليها سابقا مع العميل ، فهناك العديد من القوالب والنماذج يتم التصميم على أساسها، فتصنف بعضها حسب تحليل البيانات وعرضها، والبعض حسب التسلسل الزمني أو الفترة الزمنية المحددة، وأخرى على حسب بيئة التصميم وغيرها، وتُعد خريطة التدفق للبيانات (Data Flowchart) إحدى أهم الطرق المستخدمة في مرحلة التصميم ، حيث تستخدم لتوضيح المدخلات والمخرجات وتسلسل العمليات والمعدات المستخدمة لحل المشكلة، وإذا كانت المشكلة معقدة وتتعلق بأقسام عديدة تخص العميل فإنه يتم إعداد ما يسمى بخريطة التدفق للنظام (System Flowchart) والتي توضح العلاقات المتشعبة التي تربط بين هذه الأقسام المختلفة ونظام تدفق البيانات بين هذه الأقسام بعضها البعض ، وعلى هذه الخريطة يتم إيضاح محطات العمل والأفراد العاملين والمعدات وشكل الوثائق والأقسام المؤثرة في هذه الخريطة يمكن للمبرمج استخدام خريطة التدفق المختصرة أو خريطة التدفق المفصلة و أيضا من الأدوات التي يمكن للمصمم استخدامها لشرح التعليمات المكونة للخوارزمية قائمة القرارات (Decision Table) التي يتم فيها بيان العلاقات المختلفة وأسباب ومكان التفريع للعمليات هذا وبعد إتمام عملية التصميم يتم تسجيل كل ما يتعلق بهذه المرحلة في " وثيقة توصيف التصميم" (Design Specification Document) ، مثل تحديد مكونات البرمجية والبناء الهيكلي للبرمجية و الربط البيني مع المستخدمين ، هياكل وجداول البيانات ، والخوارزميات ، وذلك بحيث يضمن إنجاز الوظائف الأساسية لتحقيق متطلبات العميل . والشكل رقم 2.2 يوضح هذه المرحلة.

• مرحلة التنفيذ :

تُعد وثيقة توصيف متطلبات التصميم هي معطيات هذه المرحلة (أو العملية) ، حيث يتم ترجمتها إلى منتج برمجي متكامل باستخدام إحدى لغات البرمجة المناسبة أو باستخدام أدوات البرمجيات المعتمدة على الحاسب (CASE Tools) على أيدي فريق من المبرمجين ، وتتم هذه المرحلة على خطوتين :

الخطوة الأولى : وهي عملية البرمجة (Programming) ، حيث يتم خلالها تجزئة المشروع البرمجي إلى أجزاء تركيبية صغيرة (Modules) للتغلب على التعقيدات (Complexities) التي يمكن أن تنشأ في حالة التعامل معه ككتلة برمجية واحدة، وبعد ذلك توزع هذه الأجزاء على فريق المبرمجين ، بحيث يكون كل مبرمج مسؤولاً عن برمجة واختبار وتوثيق جزئية معينة من هذه الأجزاء ، بطريقة تسمح بالتكامل مع الأجزاء الأخرى.

الخطوة الثانية : وهي عملية التكامل (Integration) ، حيث يتم خلالها تجميع جميع أجزاء البرنامج المنفصلة (Individual Modules) التي تم برمجتها في الخطوة السابقة لتصبح منتجاً برمجياً متكاملًا

3-14-3 . المرحلة الثالثة : مرحلة الاختبار وتشخيص الأخطاء:

مرحلة (أو عملية) الاختبار ليست مرحلة منفصلة تنفذ بعد اكتمال المشروع البرمجي أو بعد انتهاء كل مرحلة فقط ، بل يجب أن تنفذ باستمرار خلال جميع مراحل تطوير المشروع البرمجي، وبصفة عامة ، تشمل مرحلة الاختبار على خطوتين :

• الخطوة الأولى : وهي خطوة التحقق :

وهي تتم في كل مرحلة من مراحل تطوير البرنامج، ويقصد بالتحقق في مرحلة معينة - هنا - تحديد إن كان البرنامج قد تم تحويله من المرحلة السابقة إلى هذه المرحلة بكفاءة وبدقة عالية وبدون أخطاء وإنه يحقق متطلبات المرحلة السابقة أم لا، فعلى سبيل المثال فإن مرحلة التنفيذ تُحول وثيقة توصيف التصميم إلى برنامج متكامل قابل للتنفيذ (Executable Integrated Software) ، لذا يقصد بالتحقق في هذه المرحلة تحديد إن كان البرنامج قد

تم بناؤه بالشكل الذي يحقق جميع المتطلبات الموصَّفة في وثيقة توصيف التصميم وبدون أي نوع من الأخطاء أم لا، ويُعرف العاملون في مجال هندسة البرمجيات التحقق (Verification) بأنه إجراء جميع الاختبارات اللازمة لكل مرحلة للإجابة على السؤال التالي : هل نحن نبني البرنامج بصورة صحيحة . (Are we building the software right) وخطوة التحقق هنا تتم من خلال وجهة نظر المطوّر نفسه من البرنامج.

- الخطوة الثانية وهي خطوة المصادقة :

وهي تتم بعد تطوير البرنامج ، ويقصد بالمصادقة - هنا - تقويم البرنامج للتأكد من أنه قد تم تصميمه بالطريقة التي يتوقعها ويرضى بها العميل وبدون أي أخطاء من أي نوع. ويُعرّف العاملون في مجال هندسة البرمجيات التحقق (Validation) بأنه إجراء جميع الاختبارات اللازمة للإجابة على السؤال التالي : هل نحن نبني البرنامج الصحيح ؟ (Are we building the right software?) ، وخطوة المصادقة هنا تتم من خلال وجهة نظر العميل من البرنامج.

وعملية تشخيص الأخطاء (Debugging) تختلف عن عملية اختبار العيوب (Defect Testing) ، حيث إن عملية اختبار العيوب تهتم فقط بإثبات وجود عيوب بالبرنامج من عدمه بدون تحديد موضع أو كيفية إصلاح هذا العيب في حالة وجود عيوب. أما عملية تشخيص الأخطاء فهي تهتم بتحديد مواضع هذه العيوب وإصلاحها في نفس الوقت.

وسوف نتناول الآن أهم أنواع الأخطاء التي يمكن أن تنشأ أثناء مرحلة الاختبار وتشخيص الأخطاء ، حيث يمكن تصنيفها إلى الأنواع التالية :

1- أخطاء قواعد ومعاني اللغة (Syntax and Semantic Errors): وتنتج هذه الأخطاء عن كتابة تعليمات

لا يتم فيها مراعاة قواعد ومعاني لغة البرمجة المكتوب بها البرنامج ، ويتم اكتشافها أثناء عملية ترجمة

البرنامج المصدر (Source Code) لإنتاج البرنامج الهدف (Object Code) ، وذلك باستخدام مترجم اللغة (Compiler).

2- أخطاء وقت التنفيذ (Run Time Errors) : تظهر هذه الأخطاء أثناء وقت التنفيذ من خلال نظام التشغيل ، رغم من ترجمة البرنامج بنجاح بدون أخطاء في قواعد ومعاني اللغة، فالبرنامج قد يكون صحيحاً من ناحية قواعد ومعاني اللغة ويحتوي على جمل صحيحة ، ولكنها تسبب أخطاء عند تنفيذ البرنامج، من هذه الأخطاء مثلا القسمة علي الصفر تعطي قيمة لانهائية أو القسمة على قيمة كبيرة جداً تعطي قيمة صغيرة جداً ولا يمكن في أي لغة برمجة تمثيل هذه القيم (Over-Flows and Under-Flows Errors) ، أو قد يكون البرنامج يحاول فتح ملف غير معرف مسبقاً ، أو أنه قد دخل في حلقة تكرارية غير منتهية (Open)
Loops Errors. وبمجرد اكتشاف هذه الأخطاء يقوم نظام التشغيل بوقف تنفيذ البرنامج وإعطاء رسالة تفيد بوجود الخطأ ففي هذه الحالة يجب تشغيل مشخص ومصحح الأخطاء (Debugger) الخاص بنفس اللغة المستخدمة لتشخيص مواضع هذه الأخطاء وتصحيحها.

3- الأخطاء المنطقية (Logical Errors) : هي للأسف ليست أخطاء لغوية يمكن للمترجم اكتشافها أو أخطاء في التنفيذ يمكن اكتشافها أثناء التنفيذ ، ولكنها أخطاء في تراكيب المدخلات من بيانات أو في المعادلات الرياضية ، وبالتالي تكون المخرجات غير متوقعة أو غير منطقية وغير مقبولة والسبب في هذه الأخطاء هو المبرمج الذي لا يتأكد من المدخلات أو لا يتأكد من وضع المعادلات الرياضية التي سوف تتم المعالجة علي أساسها ، أو أن يضع علامة القسمة بدلا من الضرب أو علامة الطرح بدلا من الجمع ، وخلافه.

وتعد الأخطاء المنطقية من أصعب الأخطاء في اكتشافها ، حيث إنه في حالة وجود مثل هذه الأخطاء يجب فحص البرنامج ومطابقته مع مستندات التصميم حتى يمكن معرفة مواقع هذه الأخطاء وتصحيحها.

3-14-4 المرحلة الرابعة: مرحلة الصيانة والارتقاء

كل المنتجات البرمجية الناجحة هي التي تُطَوَّر مع الوقت للتوافق مع التغييرات التي يريدها العميل ، لذا فإن عملية تطوير البرنامج لا تنتهي بتسليم المنتج النهائي للعميل ، بل تبدأ مرحلة من أهم المراحل في عمر المنتج وهي مرحلة الصيانة ومن أهم الأعمال التي قد تشملها مرحلة الصيانة ما يلي :

1. معالجة الأخطاء التي قد تنشأ مع التشغيل (Bugs Fixing).
2. إضافة عمل وظيفي جديد للبرنامج (Add New Functionality).
3. إضافة تقارير خرج جديدة (Add New Output Reports).
4. تهيئة البرنامج للعمل على أنظمة تشغيل جديد⁽¹⁸⁾ (Adapt the software to new Platforms).

[18][1] Boehm, B. (1988), "A Spiral Model of Software Development and Enhancement", IEEE Computer 21, 5, 61-72.

[2] Royce, W. (2011), "Managing the Development of Large Software Systems : Concept and Techniques", Proc. WESCON, August 2011.

[3] Ian Somerville , "Software Engineering", Addison Wesley, 2001.

3-15 . نشاطات المظلة:

يجرى تكامل المراحل السابقة وعملياتها المختلفة في إطار هندسة البرمجيات بعدد من نشاطات المظلة ، والتي

تُجرى طوال عملية تطوير المنتج البرمجي ، ومن أهم هذه النشاطات :

3-15-1 إعداد الوثائق وإنتاجها :

ويطلق عليها مرحلة التوثيق ، وهي ليست مرحلة منفصلة تنفذ بعد اكتمال المشروع البرمجي ، بل تبدأ هذه

المرحلة مع بداية المرحلة الأولى من مراحل تطوير البرنامج إلى أن يتم تسليم كامل المنتج البرمجي للعميل

(Delivery Phase) ، والتي يمكن أن تُعد آخر مرحلة في تطوير البرنامج والتي يبدأ بعدها مرحلة الصيانة.

إن التوثيق السليم للبرنامج يساعد علي الوقوف علي آخر ما تم من تعديلات وأسبابها ، وبالتالي تكون الرؤية

واضحة للتعديل الجديد ؛ لذلك يجب أن يتضمن التوثيق السليم للبرنامج ما يلي :

1. تعريف المشكلة أو تحديد الهدف من البرنامج (وثيقة توصيف المتطلبات).

2. وثائق ومخططات تصميم البرنامج (وثيقة توصيف التصميم).

3. تسجيل الملاحظات عن الخطوات والإجراءات الهامة في شفرة البرنامج ، فهذه الملاحظات تساعد

المبرمج وقارئ البرنامج على فهم البرنامج وسهولة تتبعه واكتشاف الأخطاء وتصحيحها.

4. متطلبات تنفيذ البرنامج من بيئة التشغيل وحدات الإدخال والإخراج المختلفة.

5. نسخة من برنامج المصدر.

6. عينة من نتائج البرنامج .

3-15-2 متابعة المشروع البرمجي ومراقبته :

تُعد إدارة المشاريع البرمجية من نشاطات المظلة ضمن هندسة البرمجيات ، حيث إنها تبدأ قبل الشروع بأية

نشاط تقني وتستمر طوال تعريف البرنامج وتطويره وصيانته.

3-15-3 ضمان جودة المنتج البرمجي :

وذلك من خلال أخذ القياسات الضرورية طوال بناء المشروع البرمجي للمساعدة في تقييم جودة المنتج البرمجي واتخاذ القرارات التصحيحية أثناء تقدم المشروع ومن أهم القياسات المباشرة لعملية هندسة البرمجيات قياس الكلفة والجهد المطبقين ، و عدد أسطر الشيفرة البرمجية المنتجة وسرعة التنفيذ و حجم الذاكرة المطلوب لعملية التنفيذ و عدد الأعطال المعلن عنها خلال فترة زمنية معينة ، هذا بالإضافة إلى قياسات المنتج غير المباشرة مثل درجة التعقيد ، والفاعلية ، والموثوقية ، وقابلية الصيانة ، وخلافه من صفات وخصائص المنتج البرمجي الجيد المذكورة في الوحدة السابقة.

3-15-4 إدارة تكوين البرنامج :

هي نشاط مظلة يبدأ مع بداية المشروع البرمجي وينتهي فقط عند التوقف عن استخدام البرنامج ، والغرض منها هو إدارة التعديلات الحاصلة على البرنامج الذي يبنيه فريق البرمجة والتحكم فيها خلال كامل دورة حياة البرنامج بغرض تحسين السهولة التي تجرى بها التغييرات وتخفيض مقدار الجهد المبذول عندما يجب إجراء تلك التغييرات ، وذلك لزيادة الإنتاجية إلى حدها الأقصى وتقليل الأخطاء إلى حدها الأدنى، وهي أيضاً مسئولة عن تعيين هوية كل بند من بنود تكوين البرنامج (البرنامج المصدر (Source Code)) ، والبرنامج التنفيذي (Executable Code) وثائق تصف البرنامج موجهة للمستخدمين والممارسين التقنيين ، وخلافه من الوثائق والبيانات الأخرى المتعلقة بالبرنامج).

5.5 إدارة المخاطر :

وذلك من خلال تحديد المهام اللازمة لتقييم المخاطر التقنية والإدارية وفهمها والقيام بالإجراءات المناسبة لها في كل مرحلة من مراحل تطوير المنتج البرمجي.

3-15 مراحل تطوير النظام البرمجي:

في هندسة البرمجيات، بناء النظام البرمجي ليس مجرد كتابة شفرة (كود Code)، وإنما هي عملية إنتاجية لها عدة مراحل أساسية وضرورية للحصول على المنتج، يُطلق على هذه المراحل اسم دورة حياة النظام البرمجي.

- إدارة المشروعات: هي تنظيم و تخطيط و جدولة مشاريع البرمجيات .

أهداف إدارة المشروعات :

- عرض إدارة المشاريع و وصف الخصائص المميزة .
- مناقشة تخطيط المشروع و إجرائية التخطيط .
- عرض كيفية استخدام مخططات الجدولة لتمثيل إدارة المشروع .
- مناقشة فكرة المخاطر و إجرائية إدارة المخاطر .
- إدارة المشروع مطلوبة لان تطوير البرامج دائما ما تخضع لقيود مالية وزمنية من قبل المؤسسة التي تقوم بتطوير البرمجيات.
- وامتيازات إدارة البرامج (19).

[1]Barker, W, Introduction to the Analysis of the Data Encryption Standard(DES), Aegean Park Press, 2001. Ian Somerville ,

"Software Engineering", Addison Wesley, 2001.

[2] Ronald J. Leach,, "Introduction to Software Engineering", CRC Press, 1999.

3-15-1 الفرق بين إدارة المشاريع الهندسية و إدارة مشاريع البرمجيات هي :

جدول (2.7) الفرق بين إدارة المشاريع الهندسية وإدارة مشاريع البرمجيات

المشاريع الهندسية	مشاريع البرمجيات
The product is غير ملموس intangible	هندسة البرمجيات غير منضبطة كالهندسة الكهربائية و الميكانيكية
المنتج البرمجي مرن	عملية تطوير البرامج غير موحدة لان مشاريع البرامج تعتبر مشاريع وحيدة

المصدر : Ian Somerville , Aegean Park Press, 2001. Barker, W, Introduction to the Analysis of the Data Encryption Standard(DES),

"Software Engineering", Addison Wesley, 2001.

3-16 نشاطات الإدارة:

- كتابة الاقتراح Proposal writing
- تخطيط المشروع و جدولته Project planning and scheduling
- كلفة المشروع Project costing
- مراقبة المشروع و مراجعته Project monitoring and reviews
- انتقاء الكادر و تقييمه Personnel selection and evaluation
- كتابة التقارير و العروض التقديمية Report writing and presentations

3-16-1 عموميات الإدارة:

1- هذه النشاطات ليست خاصة بإدارة البرمجيات فالكثير من المشاريع التقنية الهندسية تتساوي مع إدارة البرمجيات في هذه النشاطات .

2- الأنظمة الهندسية المعقدة تميل إلى المعاناة من نفس المشاكل كأنظمة البرامج

3-16-2 أفراد (فريق عمل) المشروع:

مدير المشروع يختار الأشخاص الأكفاء لانجاز العمل ، ولكنه قد يضطر إلي التعامل مع فريق تطوير ذو خبرة متواضعة وذلك للأسباب التالية:

1- قد لا تغطي ميزانية المشروع رواتب الفريق الماهر .

2- قد لا يتواجد الفريق الماهر أصلا .

3- قد يكون من أهداف الشركة تطوير مهارات و خبرات أفرادها .

المدرء يجب أن يعملوا ضمن هذه القيود خصوصا (في الوضع الحالي) عندما يكون هناك نقص دولي في خبراء تقنية المعلومات .

3-16-3 تخطيط المشروع:

1- من المحتمل أن أكثر نشاطات إدارة المشاريع التي تضيع الوقت من المفاهيم الأولية إلي تسليم النظام هي تخطيط المشروع .

2- الخطط يجب أن تراجع مع توافر معلومات جديدة .

4- الأنواع المختلفة للخطط قد تطور لدعم الخطة الرئيسية للمشروع البرمجي و التي تكون مهمة بالجدول الزمني و الميزانية .

4-16-3 إجرائية تخطيط المشروع:

- 1- حدد قيود المشروع .
- 2- قدر معاملات (بارمترات) المشروع بشكل أولي .
- 3- حدد المحطات الأساسية و المخرجات التي يجب أن تكون جاهزة للتسليم .
- 4- ما دام (المشروع لم ينتهي أو لم يلغى) نفذ الحلقة التالية .

- ضع الجدول الزمني للمشروع
- أبدأ النشاطات وفقاً لهذا الجدول .
- انتظر (فترة من الزمن) .
- راجع سير العمل في المشروع .
- نقح التقديرات الخاصة ببرامترات المشروع .
- قم بتحديث الجدول الزمني .
- اعد المفاوضات بشأن القيود و المخرجات الجاهزة .
- إذا (حدثت مشكلة) عندئذ .
- أبدأ مراجعة تقنية للمشروع ، و أي احتمال للتعديل .
- نهاية إذا .
- نهاية الحلقة .

3-16-4 هيكل خطة المشروع:

تحتوي أي خطة علي الأقسام التالية :

1. المقدمة Introduction .
2. تنظيم المشروع Project organization .
3. تحليل المخاطر Risk analysis .
4. متطلبات المشروع من مكونات صلبة و برمجيات Hardware and software resource requirements .
5. تقسيم العمل Work breakdown .
6. الجدول الزمني للمشروع Project schedule .
7. آليات المراقبة و إصدار التقارير Monitoring and reporting mechanisms .

3-16-5 تنظيم النشاطات:

- نشاطات المشروع يجب أن تنظم لإنتاج نواتج ملموسة تقدم للإدارة لتقييم تقدم العمل
- المحطة الأساسية (Milestone) هي عبارة عن نهاية نشاط معينة في إجرائية البرمجيات
- المنتج الجاهز (Deliverable) هو عبارة عن منتج جاهز للتسليم إلي الزبون

3-16-6 الجدولة الزمنية للمشروع:

- تقسيم المشروع إلي مهام (نشاطات) ثم نقدر المصادر و الوقت اللازم لإكمال كل مهمة .
- نسق المهام المتوازية مع بعضها البعض بما يضمن استخدام القوة العاملة بشكل أمثل .

- قلة من تبعيات المهام لتجنب التأخير الذي قد يكون سببه أن مهمة تنتظر مهمة أخرى للكمال (الانتهاء) .

- تعتمد الجدولة على حدس وخبرة مدراء المشروع .

7-16-3 مشاكل الجدولة:

- تقدير صعوبة المشكلة و لذلك تقدير تكلفة الحل صعبة .
- معدل الإنتاج ليس نسبي لعدد الأفراد الذين يعملون في المهمة .
- إضافة الأشخاص إلي مشروع متأخر يجعله متأخرا بسبب نفقات الاتصال الزائدة .
- الأشياء الغير متوقعة دائما ما تحدث .
- دائما يسمح للطوارئ في التخطيط .

8-16-3 المخططات البيانية و شبكات النشاطات:

- هي عبارة عن مخططات اصطلاحية تستخدم لتمثيل الجدول الزمني للمشروع .
- تعرض تقسيم المشروع إلي عدة مهام .
- النشاطات (المهام) يجب أن لا تكون صغيرة . يجب أن تأخذ أسبوع أو أسبوعين .
- مخطط النشاطات (شبكة النشاطات Activity charts) تظهر العلاقات بين مختلف نشاطات العمل في المشروع و المسار الحرج (the critical path) .
- المخططات البيانية (Bar charts) تعرض الجدولة خلال تقويم الوقت .

9-16-3 إجرائية إدارة المخاطر:

تشتمل على المراحل:

- 1- تحديد المخاطر Risk identification .

- 1- تحديد مخاطر المشروع و المنتج و العمل .
- 2- تحليل المخاطر Risk analysis .
- 3- تحديد احتمالية وقوع كل خطر و آثاره أن وقع
- 4- تخطيط المخاطر Risk planning .
- 5- وضع خطط للتعامل مع المخاطر و ذلك لتجنبه أو بالتخفيف من حدته .
- 6- مراقبة المخاطر Risk monitoring .
- 7- مراقبة المخاطر أثناء المشروع . (20)

(20) [1] مهندس عبد الحميد بسيوني ، "أساسيات هندسة البرمجيات" ، دار الكتب العلمية للنشر والتوزيع ، القاهرة ، 2005 م.

[2] أحمد شعبان دسوقي وآخرون ، "أساسيات الحاسب الآلي وتطبيقاته في التعليم" ، مكتبة الرشد ، الرياض ، الطبعة الأولى ، 1427هـ - 2006م.

المبحث الثالث

الدراسات السابقة

1- DATABASE SECURITY – ATTACKS AND CONTROL METHODS[Emil-BURTESCU,2009]

Ensuring the security of databases is a complex issue for companies. The more complex the databases are the more complex the security measures that are to be applied are.

2- Network and Internet connections to databases may complicate things even further. Also, each and every additional internal user that would be added to user base can create further serious security problems. This purpose of this paper is to highlight and identify the main methods and facets of attack on a database, as well as ways to deflect attacks, through focusing on the delicate issue of data inference. Database security presents features that must be seriously taken into account. The first option, for a secure database is represented by its optimal protection. Ensuring database security must be done from outside ton inside, this involving ensuring security starting from the physical level and ending with the data level (physical, network, host, applications and data). Databases are a favourite target for attackers because of the data these are containing and also because of their volume. Data warehouse is the ultimate goal. Efforts to ensure database security are considerably higher than for the other types of data. It is easier to implement an access list for a great number of files than an access list for the elements of a database. Database security mechanisms should not irritate their users ⁽⁽²¹⁾⁾.

(21)DATABASE SECURITY – ATTACKS AND CONTROL METHODS[Emil- BURTESCU,2009]

2- Oracle Database Vault: Segregation of Duties and Privileged User Controls[Oracle Corp. ,2009]

Oracle Database Vault is the industry's leading database security solution for addressing regulatory compliance and concerns over the insider threat. Oracle Database Vault helps address access control requirements associated with regulations such as PCI and Sarbanes Oxley. Oracle Database Vault is available for Oracle Database 9i Release, Oracle Database 10g Release 2 and Oracle Database 11g Release 1. Oracle Database Vault has been validated with Oracle PeopleSoft Applications. Validation with additional applications including Oracle E Business Suite and Siebel is currently underway. Using Oracle Database Vault highly privileged database users can be prevented from accessing application data. In addition, access to applications, databases and data can be tightly controlled based on such variables as time of day IP address or subnet. In summary, Oracle Database Vault provides the flexible transparent and highly adaptable security controls required in today's global economy ⁽²²⁾.

(22)Oracle Database Vault: Segregation of Duties and Privileged User Controls[Oracle Corp. ,2009]

3- How the database security controls adapted to threats over the last 30 years[Paul Lesov,2006]

It is important to also note that many problems with securing data stored in the database is not due to the lack of research but lacking security in implementation of the database product or an application front ending the database. The shift from full trust to partial trust was driven in part by natural tendency to not provide full trust to anyone single individual based on dual control principle but also due to the inability of the users to keep their own PC computers secure and database frontend not being able to detect malicious attacks such as SQL injections.

While intense focus on database security in the last 30 years created some welcome improvements it was still outpaced by the threat growth. Data security concerns are evolving with new requirements emerging year to year. The amount to data and the complexity and modularity of the databases is grown fast over time. The recent decentralization process of outsourcing data processing creates a difficult problem for database security. Intellectual Property Rights create a need for specific techniques for storing and marking this data and appropriate providing access to it. New data types used in spatial and sensor network database require new ways to provide access control and encryption. The semantic web outlook suggests the outsourcing trend to continue with related requirement for maintaining confidentiality and integrity of these query communications.⁽²³⁾

(23)How the database security controls adapted to threats over the last 30 years[Paul Lesov,2006]

4-Web and Database Security[Zhejiang Normal University,2015]

In recent years, with the frequent occurrence of security incidents, enterprises and organizations have now realized the importance of designing a safety information system. Today, information systems are heavily relied on web and database technologies, thus the risks and threats those technologies faced will also affect the security of information systems.

Web and database security technologies can ensure the confidentiality, integrity and usability of data in information system, and can effectively protect the security and reliability of information system.

Therefore, in order to better secure the information systems, we need to learn Web and database security-related knowledge.

This chapter covers extensively practical and useful knowledge of web and database security.

This chapter can be divided into three parts: advanced security threats, the principles of safety design and safety audit; Advanced security threats section contains cross-site scripting (XSS) attacks, AJAX and SQL injection attacks and other security threats, which will be presented in detail; the principles of safe design section describe the general safety design principles to help design information systems security; last section describes the manual and automatically audit methods, and general security audit framework to help readers to understand more clearly.

Database security threats

- SQL inject
- Process and Methods of SQL Attack
- Discovery of SQL Inject Bugs

Security design principle

- Web security design principle
- Database design principles

Security audit

Security audit is based on certain security policy, improving system's performance and safety by recording and analyzing historical events and data. Security audit includes all actions and instruments, e.g. testing, assessing and analyzing all of the weak links in the network information system to find the best ways to let the business run normally, based on the maximum guarantee of safety.

It is to ensure the safe operation of network systems and prevent confidentiality integrity and availability of the data from being damaged, prevent intentional or unintentional human error and detect criminal activity on the network.

The network status and processes can be targeted to recorded, tracked and reviewed by using the audit mechanism, and find safety problems.

In addition, the audit can provide the basis of making filtering rules for online information, if the harmful information is found in the website, it will be added into the list of route filtering, to reject all information of IP addresses on the filtering list through information filtering mechanism. ⁽²⁴⁾

(24)Web and Database Security[Zhejiang Normal niversity,2015]

5- Database Security: What Students Need to Know [JITE,2010]

Database security is a growing concern evidenced by an increase in the number of reported incidents of loss of or unauthorized exposure to sensitive data. As the amount of data collected, retained and shared electronically expands, so does the need to understand database security. The Defense Information Systems Agency of the US Department of Defense (2004), in its Database Security Technical Implementation Guide, states that database security should provide controlled, protected access to the contents of a database as well as preserve the integrity, consistency, and overall quality of the data. Students in the computing disciplines must develop an understanding of the issues and challenges related to database security and must be able to identify possible Solutions.

At its core, database security strives to insure that only authenticated users perform authorized activities at authorized times. While database security incorporates a wide array of security topics, notwithstanding, physical security, network security, encryption and authentication, this paper focuses on the concepts and mechanisms particular to securing data. Within that context,

Database security encompasses three constructs: confidentiality or protection of data from unauthorized disclosure, integrity or prevention from unauthorized data access, and availability or the identification of and recovery from hardware and software errors or malicious activity resulting in the denial of data availability.

In the computing discipline curricula, database security is often included as a topic in an introductory database or introductory computer security course. This paper presents a set of sub-topics that might be included in a database security component of such a course. Mapping to the three constructs of data security, these topics include access control, application access, vulnerability,

inference, and auditing mechanisms. Access control is the process by which rights and privileges are assigned to users and database objects. Application access addresses the need to assign appropriate access rights to external applications requiring a database connection. Vulnerability refers to weaknesses that allow malicious users to exploit resources. Inference refers to the use of legitimate data to infer unknown information without having rights to directly retrieve that information. Database auditing tracks database access and user activity providing a way to identify breaches that have occurred so that corrective action might be taken. ⁽²⁵⁾

(25)Database Security: What Students Need to Know [JITE,2010]

6- HYBRID ENCRYPTION FOR CLOUD DATABASE SECURITY [IJESAT, 2015]

In cloud computing environment the new data management model is in use now a days that enables data integration and access on a large scale cloud computing as a service termed as Database-as-a-service (DAAS). Through which service provider offers customer management functionalities as well as the expensive hardware. Data privacy is the major security determinant in DAAS because data will be shared with a third party; an un-trusted server is dangerous and unsafe for the user. This paper shows a concern on the security element in cloud environment. It suggests a technique to enhance the security of cloud database. This technique provides the flexible multilevel and hybrid security. It uses RSA, Triple DES and Random Number generator algorithms as an encrypting tool.

Taking into account the services provided by cloud computing is numerous. But still there are some security concerns that are to be redressed. Especially because cloud users have no choice but to rely on the service provider. Amongst the possible solutions one can keep a local copy of its data which is not feasible as we are taking the benefit of the services of the CSP (cloud service provider) [1]. Another factor of concern is that the cloud is still under development process and there are no set standards for the data storage and application communication. So one couldn't move his data by changing service provider though some organizations are working towards this direction and will soon come out with a solution but till that time, we must have some mechanism to provide security to the critical and private data stored in the cloud like credit card information and passwords. Keeping in view this fact, some application must be developed that will implement multi-level hybrid encryption mechanism by using some strong cryptographic algorithms viz. RSA, Random Number Generator and 3DES. From the implementation of code the security has been enhanced. One can have hierarchy of application of algorithms. The Encryption Algorithm applicability provides

the flexibility in range and sequence to the user's choice because from the three of the Encryption Methods a user can apply all or omit any in any order. Even if the user does not select any encryption technique, then random number algorithm will be implemented by default thus providing at least a single level security. The opted sequence will also be stored in the database so that the decryption may be possible. The negative effect of this scheme is that it creates an overhead on the query performance due to multilevel of encryption and decryption but for the sake of security the performance issue can be over looked as we are concerned with only a small amount of data like that of passwords and not the large files. In this way we can conclude that multilevel hybrid encryption enhances security. ⁽²⁶⁾

(26)HYBRID ENCRYPTION FOR CLOUD DATABASE SECURITY [IJESAT, 2015]

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

الفصل الرابع

المبحث الأول : الإطار التطبيقي :

4-1 نماذج تحليل النظام :

مراحل تطوير البرمجيات إلى المراحل الرئيسية التالية :

1- مرحلة تحليل المشكلة (Problem Analysis Phase).

2- مرحلة التطوير (Development Phase) وتشمل مرحلتي التصميم (Design Phase) والتنفيذ

(Implementation Phase).

3- مرحلة الاختبار وتشخيص الأخطاء (Testing and Debugging Phase).

ويأتي بعد كل هذه المراحل أهم مرحلة و أطولها وأكثرها تكلفة وهي :

4- مرحلة صيانة وترقية البرنامج (Maintenance Phase).

وتُعد نتائج كل مرحلة من هذه المراحل معطيات للمرحلة التي تليها ، فمثلاً نتائج مرحلة تحليل المشكلة هي معطيات

لمرحلة التصميم ، ونتائج مرحلة التصميم هي معطيات لمرحلة التنفيذ ، وهكذا وسوف نتناول الآن هذه المراحل

بإيجاز وبدون الدخول في التفاصيل الخاصة بكيفية إنجاز كل مرحلة.

4-1-1 المرحلة الأولى: مرحلة تجميع المتطلبات :

تركز هذه المرحلة (أو العملية) على دراسة متطلبات حل المشكلة ، حيث إنها تُعد مرحلة التوصيف الدقيق للمشروع

البرمجي وتهيئته لمرحلة التصميم النهائي، ويتم ذلك عبر إعداد مجموعة من المخططات (Charts) والنماذج

(Models) واللوائح (Lists) من خلال طرح مجموعة من الأسئلة المتخصصة من قبل محلل النظم للعميل لمعرفة

كافة احتياجات وشروط المشروع البرمجي ، وذلك طبقاً لمنهجية محددة تسمى منهجية التحليل. وتنتهي هذا لمرحلة بكتابة وثيقة رسمية (Formal Document) تشتمل على المواصفات الفنية للمشروع البرمجي بناءً على متطلبات العميل مثل نطاق المعلومات الخاص بالمشروع (Information Domain) و الأداء الوظيفي (Functionality) ، الربط البيني بين البرمجية والمستخدمين (User Interface) ، و مستويات الأمن المطلوبة (Security Levels) ، بالإضافة إلى المتطلبات الاستثنائية (Exceptional Requirements). ويطلق علي هذه الوثيقة "وثيقة توصيف المتطلبات" (Requirements Specification Document) ، والتي ستكون بمثابة العقد النهائي بين مطوري المشروع البرمجي والعميل. والشكل رقم 2.1 يمثل شكل توضيحي لهذه المرحلة ، حيث إن معطيات هذه المرحلة تتمثل في متطلبات العميل ونتائجها تتمثل في كتابة "وثيقة توصيف المتطلبات"

وبصفة عامة تتضمن مرحلة (أو عملية) تحليل المشكلة التعريف بالمشكلة من خلال العديد من النواحي والتي من أهمها :

- المدخلات (Inputs): يجب على المحلل معرفة الطرق التي يفضلها العميل في عملية إدخال البيانات إلى النظام ، فهل عملية إدخال البيانات سوف تتم عن طريق القراءة من ملف ، أم سوف تكون تفاعلية بين المستخدم والنظام عن طريق قوائم اختيار بواسطة الفأرة ، أو تكون تفاعلية ولكن بطريقة مفتوحة عن طريق لوحة المفاتيح ، أو غيرها من طرق إدخال البيانات.
- المخرجات (outputs): يجب - أيضاً - على المحلل الاهتمام بمعرفة متطلبات العميل من حيث شكل وهيئة نماذج المخرجات (تقارير النظام) وصيغها المختلفة ، هل تكون على شكل أرقام فقط أو أرقام وحروف ، أو خليط من الأرقام والحروف والرسومات ، أو ما شابه ذلك.
- المعالجة (Processing) : يجب أن يدقق المحلل في كيفية عملية المعالجة من حيث نوعية البيانات والحسابات والقرارات التي سوف تعمل على هذه البيانات لإنجاز المخرجات المطلوبة ، وكذلك من حيث بيئة التشغيل التي سوف يتم تشغيل النظام من خلالها (نظام التشغيل (Operating System) ، وهل

عملية التشغيل سوف تتم على حاسب آلي شخصي (Personal Computer) ، أو من خلال استخدام

الشبكات المحلية (LANs).

4-1-2 المرحلة الثانية مرحلة التطوير:

تتقسم مرحلة التطوير إلى مرحلتين هما :

1- مرحلة التصميم :

تُعد وثيقة توصيف متطلبات العميل هي معطيات هذه المرحلة (أو العملية) ، حيث يتم توزيع المهام على المختصين من فريق العمل من المصممين ، الذين يقومون في هذه المرحلة بتحديد الخوارزميات التي سوف يستخدمونها ، وكذلك رسم مخططات التصميم التي تتناسب مع المتطلبات المتفق عليها سابقا مع العميل ، فهناك العديد من القوالب والنماذج يتم التصميم على أساسها، فتصنف بعضها حسب تحليل البيانات وعرضها، والبعض حسب التسلسل الزمني أو الفترة الزمنية المحددة، وأخرى على حسب بيئة التصميم وغيرها، وتُعد خريطة التدفق للبيانات (Data Flowchart) إحدى أهم الطرق المستخدمة في مرحلة التصميم ، حيث تستخدم لتوضيح المدخلات والمخرجات وتسلسل العمليات والمعدات المستخدمة لحل المشكلة، وإذا كانت المشكلة معقدة وتتعلق بأقسام عديدة تخص العميل فإنه يتم إعداد ما يسمى بخريطة التدفق للنظام (System Flowchart) والتي توضح العلاقات المتشعبة التي تربط بين هذه الأقسام المختلفة ونظام تدفق البيانات بين هذه الأقسام بعضها البعض ، وعلى هذه الخريطة يتم إيضاح محطات العمل والأفراد العاملين والمعدات وشكل الوثائق والأقسام المؤثرة في هذه الخريطة و يمكن للمبرمج استخدام خريطة التدفق المختصرة أو خريطة التدفق المفصلة و أيضا من الأدوات التي يمكن للمصمم استخدامها لشرح التعليمات المكونة للخوارزمية قائمة القرارات (Decision Table) التي يتم فيها بيان العلاقات المختلفة وأسباب ومكان التفريع للعمليات هذا وبعد إتمام عملية التصميم يتم تسجيل كل ما يتعلق بهذه المرحلة في " وثيقة توصيف التصميم (Design Specification Document) ، مثل تحديد مكونات البرمجية والبناء الهيكلي

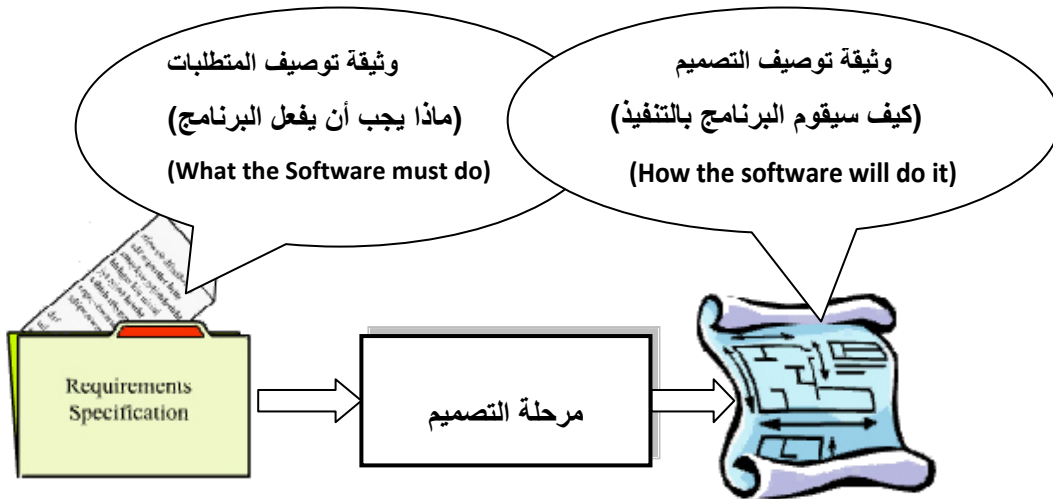
للبرمجية و الربط البيني مع المستخدمين ، هياكل وجداول البيانات ، والخوارزميات ، وذلك بحيث يضمن إنجاز الوظائف الأساسية لتحقيق متطلبات العميل . والشكل رقم (2.2) يوضح هذه المرحلة.

2- مرحلة التنفيذ :

تُعد وثيقة توصيف متطلبات التصميم هي معطيات هذه المرحلة (أو العملية) ، حيث يتم ترجمتها إلى منتج برمجي متكامل باستخدام إحدى لغات البرمجة المناسبة أو باستخدام أدوات البرمجيات المعتمدة على الحاسب (CASE Tools) على أيدي فريق من المبرمجين ، وتتم هذه المرحلة على خطوتين :

- الخطوة الأولى : وهي عملية البرمجة (Programming) ، حيث يتم خلالها تجزئة المشروع البرمجي إلى أجزاء تركيبية صغيرة (Modules) للتغلب على التعقيدات (Complexities) التي يمكن أن تنشأ في حالة التعامل معه ككتلة برمجية واحدة، وبعد ذلك توزع هذه الأجزاء على فريق المبرمجين ، بحيث يكون كل مبرمج مسؤولاً عن برمجة واختبار وتوثيق جزئية معينة من هذه الأجزاء ، بطريقة تسمح بالتكامل مع الأجزاء الأخرى .

- الخطوة الثانية :وهي عملية التكامل (Integration) ، حيث يتم خلالها تجميع جميع أجزاء البرنامج المنفصلة (Individual Modules) التي تم برمجتها في الخطوة السابقة لتصبح منتجاً برمجياً متكاملًا.



(شكل 2.2) : شكل توضيحي لمرحلة التصميم

3-1-4 المرحلة الثالثة : مرحلة الاختبار وتشخيص الأخطاء

مرحلة (أو عملية) الاختبار ليست مرحلة منفصلة تنفذ بعد اكتمال المشروع البرمجي أو بعد انتهاء كل مرحلة فقط ، بل يجب أن تنفذ باستمرار خلال جميع مراحل تطوير المشروع البرمجي، وبصفة عامة ، تشمل مرحلة الاختبار على خطوتين:

الخطوة الأولى : وهي خطوة التحقق:

وهي تتم في كل مرحلة من مراحل تطوير البرنامج، ويقصد بالتحقق في مرحلة معينة - هنا - تحديد إن كان البرنامج قد تم تحويله من المرحلة السابقة إلى هذه المرحلة بكفاءة وبدقة عالية وبدون أخطاء وإنه يحقق متطلبات المرحلة السابقة أم لا، فعلى سبيل المثال فإن مرحلة التنفيذ تُحول وثيقة توصيف التصميم إلى برنامج متكامل قابل للتنفيذ (Executable Integrated Software) ، لذا يقصد بالتحقق في هذه المرحلة تحديد إن كان البرنامج قد تم بناؤه بالشكل الذي يحقق جميع المتطلبات الموصَّفة في وثيقة توصيف التصميم وبدون أي نوع من الأخطاء أم لا، ويُعرف العاملون في مجال هندسة البرمجيات التحقق (Verification) بأنه إجراء جميع الاختبارات اللازمة لكل مرحلة للإجابة على السؤال التالي : هل نحن نبني البرنامج بصورة صحيحة ؟ (Are we building the software right?) وخطوة التحقق هنا تتم من خلال وجهة نظر المطوّر نفسه من البرنامج.

الخطوة الثانية وهي خطوة المصادقة :

وهي تتم بعد تطوير البرنامج ، ويقصد بالمصادقة - هنا - تقويم البرنامج للتأكد من أنه قد تم تصميمه بالطريقة التي يتوقعها ويرضى بها العميل وبدون أي أخطاء من أي نوع، ويُعرّف العاملون في مجال هندسة البرمجيات التحقق (Validation) بأنه إجراء جميع الاختبارات اللازمة للإجابة على السؤال التالي : هل نحن نبني البرنامج الصحيح ؟ (Are we building the right software?) ، وخطوة المصادقة هنا تتم من خلال وجهة نظر العميل من البرنامج.

وعملية تشخيص الأخطاء (Debugging) تختلف عن عملية اختبار العيوب (Defect Testing) ، حيث إن عملية اختبار العيوب تهتم فقط بإثبات وجود عيوب بالبرنامج من عدمه بدون تحديد موضع أو كيفية إصلاح هذا العيب في حالة وجود عيوب. أما عملية تشخيص الأخطاء فهي تهتم بتحديد مواضع هذه العيوب وإصلاحها في نفس الوقت.

وسوف نتناول الآن أهم أنواع الأخطاء التي يمكن أن تنشأ أثناء مرحلة الاختبار وتشخيص الأخطاء ، حيث يمكن تصنيفها إلى الأنواع التالية :

1- أخطاء قواعد ومعاني اللغة (Syntax and Semantic Errors): وتنتج هذه الأخطاء عن كتابة تعليمات لا يتم فيها مراعاة قواعد ومعاني لغة البرمجة المكتوب بها البرنامج ، ويتم اكتشافها أثناء عملية ترجمة البرنامج المصدر (Source Code) لإنتاج البرنامج الهدف (Object Code) ، وذلك باستخدام مترجم اللغة (Compiler).

2- أخطاء وقت التنفيذ (Run Time Errors) : تظهر هذه الأخطاء أثناء وقت التنفيذ من خلال نظام التشغيل ، رغم من ترجمة البرنامج بنجاح بدون أخطاء في قواعد ومعاني اللغة، فالبرنامج قد يكون صحيحاً من ناحية قواعد ومعاني اللغة ويحتوي على جمل صحيحة ، ولكنها تسبب أخطاء عند تنفيذ البرنامج، من هذه الأخطاء مثلا القسمة علي الصفر تعطي قيمة لانهائية أو القسمة على قيمة كبيرة جداً تعطي قيمة صغيرة جداً ولا يمكن في أي لغة برمجة تمثيل هذه القيم (Over-Flows and Under-Flows Errors) ، أو قد يكون البرنامج يحاول فتح ملف غير معرف مسبقاً ، أو أنه قد دخل في حلقة تكرارية غير منتهية (Open)
Loops Errors. وبمجرد اكتشاف هذه الأخطاء يقوم نظام التشغيل بوقف تنفيذ البرنامج وإعطاء رسالة تفيد بوجود الخطأ ففي هذه الحالة يجب تشغيل مشخص ومصصح الأخطاء (Debugger) الخاص بنفس اللغة المستخدمة لتشخيص مواضع هذه الأخطاء وتصحيحها.

3- الأخطاء المنطقية(Logical Errors) :هي للأسف ليست أخطاء لغوية يمكن للمترجم اكتشافها أو أخطاء في التنفيذ يمكن اكتشافها أثناء التنفيذ ، ولكنها أخطاء في تراكيب المدخلات من بيانات أو في المعادلات الرياضية ، وبالتالي تكون المخرجات غير متوقعة أو غير منطقية وغير مقبولة والسبب في هذه الأخطاء هو المبرمج الذي لا يتأكد من المدخلات أو لا يتأكد من وضع المعادلات الرياضية التي سوف تتم المعالجة علي أساسها ، أو أن يضع علامة القسمة بدلا من الضرب أو علامة الطرح بدلا من الجمع ، وخلافه.

وُعد الأخطاء المنطقية من أصعب الأخطاء في اكتشافها ، حيث إنه في حالة وجود مثل هذه الأخطاء يجب فحص البرنامج ومطابقته مع مستندات التصميم حتى يمكن معرفة مواقع هذه الأخطاء وتصحيحها (27).

4-1-4 المرحلة الرابعة: مرحلة الصيانة والارتقاء :

كل المنتجات البرمجية الناجحة هي التي تُطَوَّر مع الوقت للتوافق مع التغيرات التي يريدها العميل ، لذا فإن عملية تطوير البرنامج لا تنتهي بتسليم المنتج النهائي للعميل ، بل تبدأ مرحلة من أهم المراحل في عمر المنتج وهي مرحلة الصيانة ومن أهم الأعمال التي قد تشملها مرحلة الصيانة ما يلي :

- معالجة الأخطاء التي قد تنشأ مع التشغيل (Bugs Fixing).
- إضافة عمل وظيفي جديد للبرنامج (Add New Functionality).
- إضافة تقارير خرج جديدة (Add New Output Reports).
- تهيئة البرنامج للعمل على أنظمة تشغيل جديد (Adapt the software to new Platforms).

[27] [1] Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

[2] Barkley, J, Comparing Simple Role-Based Access Control Models and Access Control Lists, Proceedings of the second ACM

Workshop on Role-Based Access Control, 2009.

2-4 أنواع المتطلبات:

- بصفة عامة يمكن تقسيم المتطلبات إلى نوعين رئيسيين هما :
- متطلبات وظيفية (Functional Requirements) .
- متطلبات غير وظيفية (Non-Functional Requirements) .
- [1] Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.
- [2] Bernstein, P., and Goodman, N, An algorithm for Concurrency Control and Recovery in Replicated Distributed Database, TODS, 2004

1-2-4 المتطلبات الوظيفية:

هي عبارة عن وصف للخدمات التي يجب أن تقدمها البرمجية بالتفصيل ، وكيفية استجابتها لمدخلات محددة ، وكيفية سلوكها في حالات محددة ، وتعتمد هذه المتطلبات . في المقام الأول . على نوع البرمجية نفسها وعلى المستخدمين النهائيين لها ، وكذلك على نوع النظام الداخلة في تكوينه والمتطلبات الوظيفية - في بعض الحالات - ربما تتضمن ماذا يجب ألا يفعله النظام ، مثل : المتطلبات الخاصة بالأمان (Security Requirements).

وتُعد المتطلبات الوظيفية هي الجوهر الأساسي لمواصفات المتطلبات أنها تبين ماذا يجب أن يفعله النظام حيث إنها تتميز بالأفعال التي تؤدي عملاً ما ، والأمثلة هي :

- النظام يجب أن يُظهر عناوين الكتب التي كتبها مؤلف معين.
- النظام يجب أن يُظهر باستمرار درجة حرارة كل المكينات.
- يجب أن يكون المستخدم قادراً على البحث في جميع قاعدة البيانات الفعلية الأولية أو اختيار مجموعة فرعية منها.

كل مُدخلة يجب أن يخصص لها معرف مفرد (ORDER_ID) والتي تمكن المستخدم من التعامل معه بسهولة من حيث البحث والتخزين وخلافه.

4-2-2 المتطلبات غير الوظيفية:

هي عبارة عن وصف للقيود المفروضة على الخدمات التي يجب أن يقدمها النظام ، مثل التكلفة ، و تاريخ التسليم ، و كمية الذاكرة المتوفرة ، و كمية التخزين المدعومة المتوفرة ، و وقت الاستجابة، و لغة البرمجة وطريقة التطوير المستخدمة (Particular CASE Development System) ، و أحجام البيانات ، و اعتمادية النظام ، و قدرات وحدات الإدخال والإخراج، و خلافه. وقد تكون المتطلبات غير الوظيفية أكثر أهمية من المتطلبات الوظيفية لأنه إذا لم يتم تحقيق المتطلبات غير الوظيفية فلا فائدة للنظام، و عادة تهتم المتطلبات غير الوظيفية بالخصائص النوعية (Qualitative Features) الخاصة بالمنتج البرمجي والتي يمكن قياسها مثل الاعتمادية (Reliability) ، و قابلية التنقل (Portability) ، و المتانة (Robustness) (قدرته على متابعة العمل في جميع الظروف حتى تلك غير المتوقعة) ، والأمان (Security) ، و سهولة الاستخدام (Ease of Use) ، و سهولة الصيانة (Maintainability) ، و السرعة (Speed) ، و الحجم (Size)، و خلافه. و الجدول رقم (3.1) يبين أهم تلك الخصائص وطرق قياسها.

جدول (3.1) أهم خصائص وطرق قياس المتطلبات غير الوظيفية

الخاصية	القياس
الاعتمادية	تقاس بـ : متوسط الوقت الذي يحدث من بعده عطل أو فشل في النظام ، و احتمالية عدم جاهزية النظام ، و معدل حدوث عطل أو فشل في النظام ، و جاهزية النظام للعمل.
المتانة	تقاس بـ : الوقت اللازم لتشغيل النظام بعد حدوث عطل أو فشل ، النسبة المئوية للخسارة نتيجة فشل النظام ، و احتمالية تخريب البيانات (Data Corruption) بسبب فشل النظام.
الحجم	ويقاس بسعة رقاقة الذاكرة العشوائية المطلوبة لتشغيل النظام (كيلو بايت). وكذلك يمكن أن يقاس بعدد سطور التعليمات البرمجية التي يتكون منها النظام.
السرعة	تقاس بـ : عدد الإجراءات التي تعالج في الثانية ، وقت الاستجابة للمستخدم أو لحدث ما ، وقت تجديد الشاشة.
سهولة الاستخدام	تقاس بـ : وقت التدريب اللازم لإتقان العمل على النظام ، وعدد أشكال المساعدة الفورية أثناء العمل على النظام.
قابلية النقل	تقاس بعدد أنظمة التشغيل التي يمكن تشغيل النظام من خلالها بدون أي تعديلات تذكر في التعليمات البرمجية الخاصة بالنظام.

المصدر: Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

ويمكن تصنيف المتطلبات غير الوظيفية ، كما هو موضح بالشكل (4.2) ، على النحو التالي :

متطلبات المنتج (Product Requirements) : وهي المتطلبات التي تحدد سلوك المنتج البرمجي النهائي

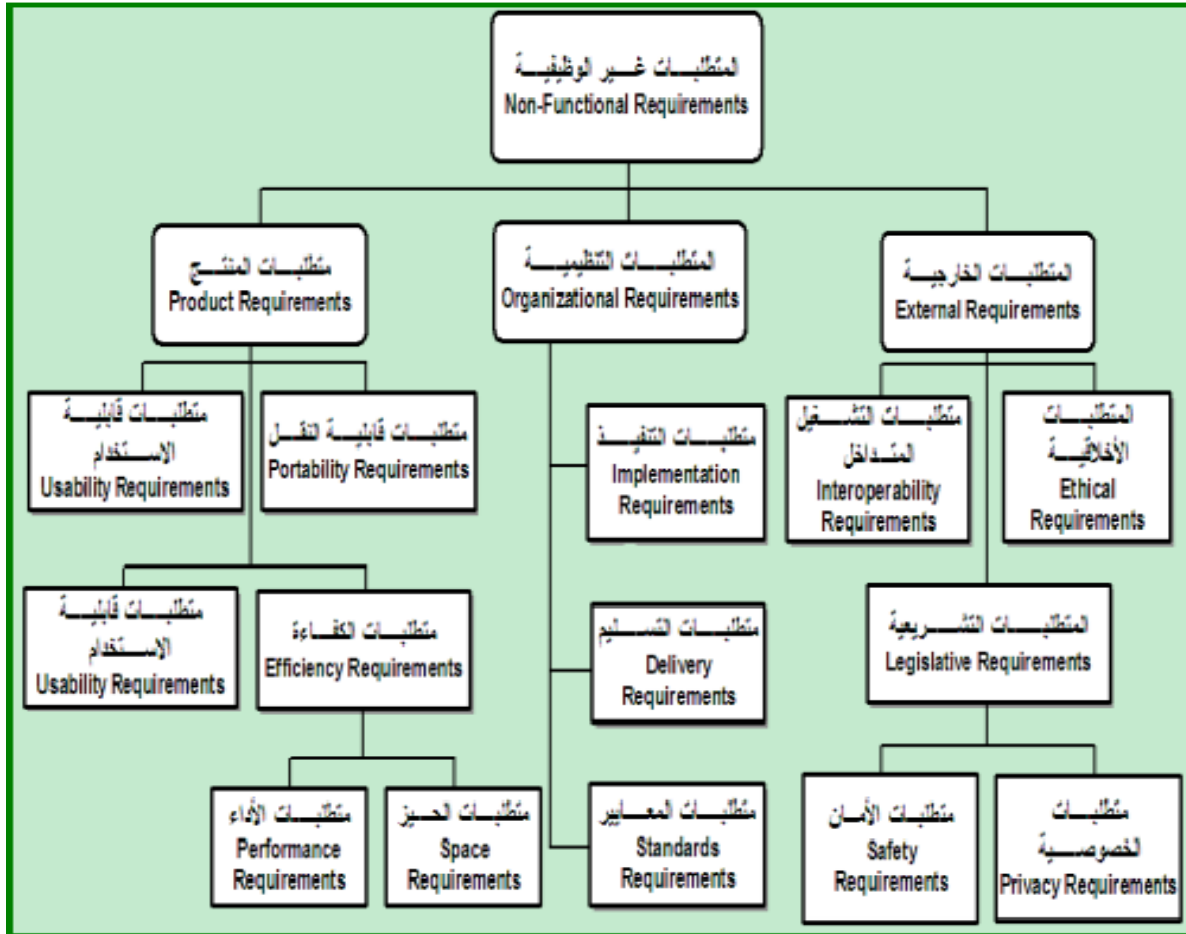
مثل : سرعة التنفيذ (Execution Speed) ، الاعتمادية (Reliability) المتطلبات التنظيمية

(Organizational Requirements): وهي المتطلبات الناتجة عن السياسات والإجراءات التنظيمية

(Organizational Policies and Procedures) الخاصة بالشركة أو المؤسسة التي يطورها لها المنتج

البرمجي ، مثل : المعايير القياسية المستخدمة في عملية التطوير (Process Standards) ، (28)

1- ومتطلبات التنفيذ (Implementation Requirements).



شكل (2.2) توضيحي لأهم أنواع المتطلبات غير الوظيفية.

المصدر: Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

(28) Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

2- المتطلبات الخارجية : (External Requirements) : وهي المتطلبات التي تنشأ من عوامل خارجية

للنظام ولعملية التطوير الخاص بها ، مثل المتطلبات التشريعية فمثلاً يجب أن لا يفشي النظام أية معلومات شخصية عن العملاء بغض النظر عن أسمائهم وأرقام هواتفهم) إلى مشغلي النظام.

وقد تكون المتطلبات غير الوظيفية صعبة جداً بحيث لا يمكن وصفها بدقة ، ولكن المتطلبات غير الدقيقة (الغامضة) من الصعب التأكد من صحتها ، فعلى سبيل المثال عند تحديد الهدف من النظام بالقول " يجب أن يكون النظام سهل الاستعمال من قبل مراقبين ذوي خبرات ، وكذلك يجب أن يكون منظماً بحيث يقلل من أخطاء المستخدمين " بالتدقيق في منطوق هذا الهدف نجد أنه غامض وغير قابل للقياس والتحقق من تنفيذه ، وحيث إن تحديد أهداف النظام مفيدة للمطورين لأنها تظهر الرغبات والاهتمامات الحقيقية لمستخدمي النظام ، لذا يجب أن يكون منطوق الهدف واضحاً وقابلاً للقياس ، فمثلاً يمكن تعديل المنطوق السابق إلي منطوق واضح وقابل للقياس بالقول : " يجب أن يكون المراقبين من ذوي الخبرة قادرين على استخدام جميع وظائف النظام بعد تدريب حوالي ساعتين وبعد هذا التدريب فإن متوسط عدد الأخطاء التي يرتكبها المستخدمين من ذوي الخبرة يجب أن لا تزيد عن خطأين في اليوم " .

4-3 وثيقة مواصفات المتطلبات:

المنتج النهائي لعملية هندسة المتطلبات هو وثيقة مواصفات المتطلبات ، وهي ضرورية لنجاح وتطوير أي مشروع تطوير برمجية، فإذا لم نحدد بوضوح ما يستطيع أن يفعله النظام فكيف يمكننا أن نطور البرمجية وبأي ثقة. وكيف نأمل أن يتوافق المنتج النهائي مع احتياجاتنا؟ . ووثيقة المواصفات هي الوثيقة التي يمكن الرجوع إليها عند تقويم أي تطوير ، وهي عنصر رئيسي لأي عقد ارتباط شرعي، ومن أهم العوامل التي يجب مراعاتها عند كتابة وثيقة مواصفات المتطلبات هو الأخذ بعين الاعتبار مستخدم هذه الوثيقة (أنظر شكل 4.3) ، لكي

يفهمها كل من له صلة في تطوير النظام باختلاف مستوياتهم من حيث الخلفيات العلمية والخبرات العملية ،
وخصوصاً المستخدمين والمطورين .

وبرغم أن كلاً من هؤلاء يشتركون في الهدف المشترك وهو الوصف الواضح لمتطلبات النظام ، فإنهم
سيضطرون لاستعمال لغات مختلفة ، حيث إن خلفياتهم العلمية وخبراتهم العملية مختلفة، فمثلاً سيفضل
المستخدمون وصفات غير تقنية بلغة طبيعية ، وبالرغم من أن تلك اللغات تكون ممتازة في القوائد وكتابة
الخطابات ولكنها فقيرة وضعيفة في عرض المواصفات الدقيقة والمتكاملة والواضحة ، من الناحية الأخرى فإن
المحللين لكونهم أصحاب خبرة تقنية سيفضلون استخدام رموز دقيقة (ربما رياضية) لكي يصفوا النظام. (29)

[1] (Cormen, T, Leiserson, C, Rivest, R, and Stein, C, Introduction to Algorithms, Cambridge, MA, MIT Press, 2001.

[2] Campbell, K., and Wiener. M, Proof that DES Is Not a Group, Spring, 2006.



شكل (2.4) يوضح مستخدمو وثيقة مواصفات المتطلبات

المصدر: Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

هناك رموز متعددة لكتابة المواصفات:

- الكتابة غير الرسمية (Informal) : وهي تتم باللغة الطبيعية حيث ستستخدم بوضوح وبعناية قدر الإمكان.

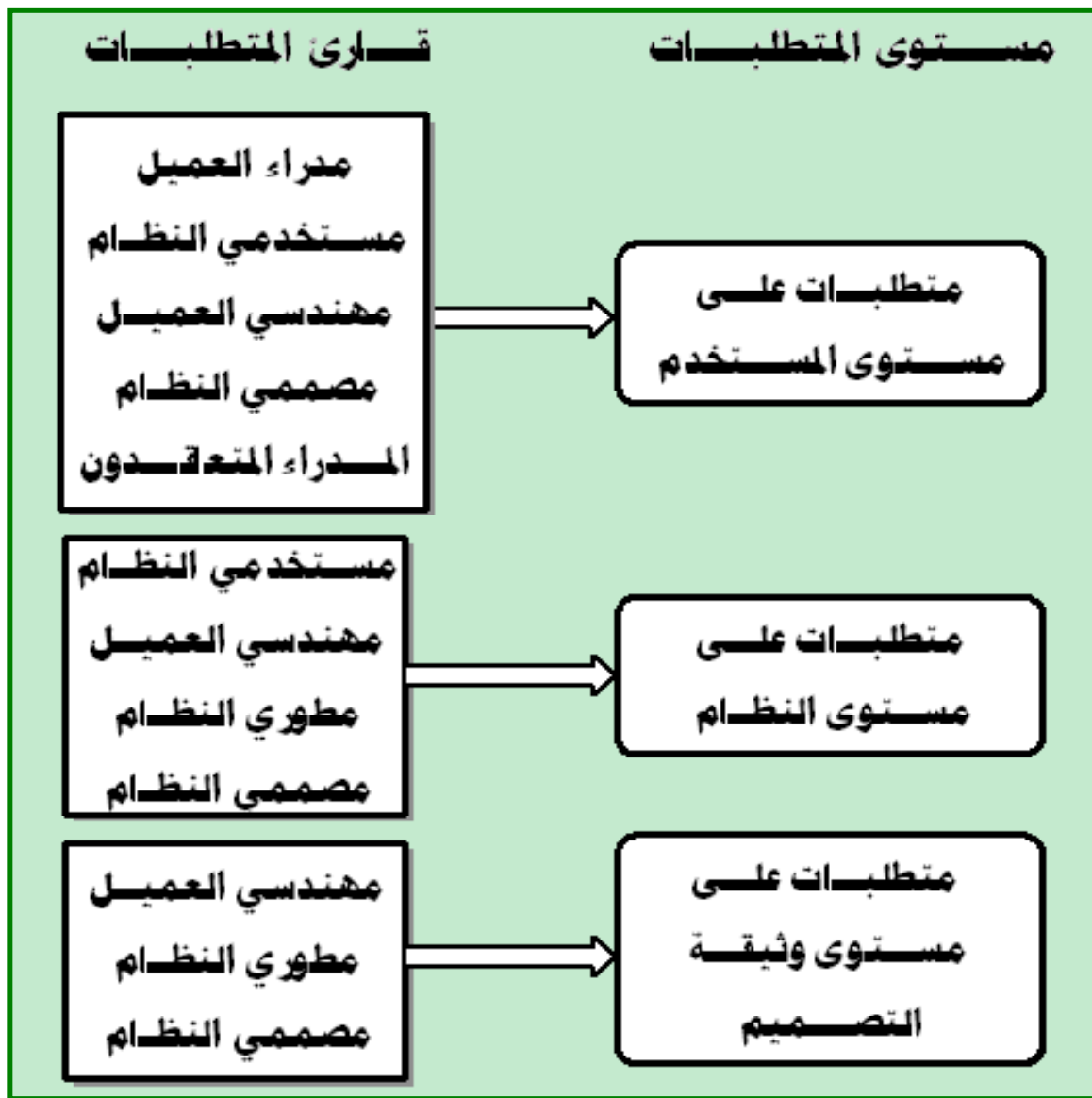
- الكتابة الرسمية (Formal) : حيث تستعمل الرموز الرياضية بكل دقة وباختصار.

- الكتابة شبه رسمية (Semiformal) : حيث تتم باستعمال مزيج من اللغة الطبيعية مع رموز وجداول ورسوم مختلفة.

ومعظم هذه الرموز لديها أصول في الطرق لتصميم برمجية حيث تشمل الشفرة الكاذبة (Pseudo Code) ، ورسوم تدفق البيانات (Flow Charts) ، وخلافه. وفي الوقت الحالي فإن معظم مواصفات البيانات تكتب بلغة طبيعية مع رموز شبه رسمية مثل رسومات تدفق البيانات المشروحة حيث تستخدم أحياناً لتوضيح نص اللغة الطبيعية.

4-4 مستويات المتطلبات:

لتوصيف متطلبات النظام بصورة جيدة يفهمها كل من يستخدم هذه الوثيقة بمختلف مستوياتهم الوظيفية وخبراتهم العملية يجب تقسيم وثيقة مواصفات المتطلبات إلى مستويات مختلفة من التفاصيل لكي تتناسب مع الاهتمامات الخاصة لكل من يقرأها ، مثل المستخدم النهائي و المدير التنفيذي و المطور ، وخلافه ، كما هو موضح بالشكل رقم (2.5) .



شكل (2.5) توضيح لأهم مستويات المتطلبات وقارئها.

المصدر: Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

المستويات ما يلي :

4-5 متطلبات المستخدم:

متطلبات المستخدم للنظام ينبغي أن تصف المتطلبات الوظيفية وغير الوظيفية لكي تكون مفهومة لمستخدمي النظام بدون المعرفة التقنية المفصلة له، ويجب أن تشمل سلوك النظام الخارجي وتجنب بقدر الإمكان خواص تصميم النظام، ويراعى عند كتابة متطلبات المستخدم أن لا تستعمل المصطلحات البرمجية (نظم الترميز)

الرسمية أو وصف المتطلبات بواسطة وصف تنفيذ النظام و ينبغي أن تكتب متطلبات المستخدم باللغة البسيطة ، وجداول وأشكال بسيطة ورسوم بيانية واضحة.

وقد تظهر العديد من المشاكل المختلفة عندما تكتب المتطلبات بجملة لغة طبيعية في نصّ المستند للأسباب التالية :

1- نقص الوضوح (Lack of Clarity) : أحيانا يكون من الصعب استخدام اللغة الطبيعية بطريقة دقيقة وواضحة بدون جعل الوثيقة مفصلة وصعبة القراءة.

2- التشويش بين المتطلبات (Requirements Confusion) : حيث أنه يكون من الصعب التمييز بوضوح بين المتطلبات الوظيفية وغير الوظيفية ، وكذلك يبين أهداف النظام ومعلومات التصميم.

3- دمج المتطلبات (Requirements Amalgamation) : حيث إنه يتم التعبير عن عدة متطلبات مختلفة كمتطلب واحد.

ولتقليل سوء الفهم عند كتابة متطلبات المستخدم ، يوصي البعض بإتباع التوجيهات البسيطة التالية :

1- تحديد شكل قياسي لكتابة المتطلبات والتأكد من أنّ كلّ تعريف للمتطلب ينتسب إلى ذلك الشكل و توحيد الشكل يجعل المحذوفات أقلّ احتمالاً والمتطلبات أسهل للفحص, فعلى سبيل المثال يعرض الشكل المتطلب الأولي بخطّ كبير, والتفاصيل بخطوط مختلفة متضمّنة أسباب كل متطلب ومعلومات عن مقترح المتطلب (مصدر المتطلب) ، لكي تعرف من تستشير في حال تغير المتطلبات.

2- استعمل لغة على نفس النهج ، و ينبغي أن تميّز دائماً بين المتطلبات الإجبارية والمفضّلة و المتطلبات الإجبارية هي المتطلبات التي يجب على النظام أن يدعمها ، أما المتطلبات المفضّلة هي المتطلبات المرغوب فيها ولكنها غير أساسية.

3- استخدم إبراز النصّ (أسود عريض , مائل أو لون) لتلنقط الأجزاء الأساسية من المتطلبات.

④تجنب بقدر الإمكان , استعمال المصطلحات العلمية التقنية لعلوم الحاسب الآلي ، وفي حالة استخدامها يجب تعريفها بدقة ووضوح.

4-6 متطلبات النظام:

تُعد متطلبات النظام امتداد لمتطلبات المستخدم والتي يستخدمها مهندسو البرمجيات كنقطة انطلاق لتصميم النظام حيث تضيف تفاصيلاً وشرحاً لكيفية تناول متطلبات المستخدم ، ويمكن استخدامها كجزء من العقد لاستخدام النظام ، لذا يجب أن تكون كاملة ومتوافقة مع النظام الكلي.

وعموماً فإن متطلبات النظام تصف ببساطة سلوك النظام الخارجي والقيود الموضوعية عليه ولا يجب أن تتطرق إطلاقاً إلى كيف يتم تصميم النظام. وتستخدم اللغة الطبيعية لكتابة كل من مواصفات متطلبات النظام وكذلك متطلبات المستخدم حيث تُعد مواصفات النظام أكثر تفصيلاً عن متطلبات المستخدم.⁽³⁰⁾

[1]Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

[2]Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

[3]Bernstein, P., and Goodman, N, An algorithm for Concurrency Control and Recovery in Replicated Distributed Database, TODS, 2004

[4]Bray, O., Computer Integrated Manufacturing -The Data Management Strategy, Digital Press, 2008.

[5]Bernstein, P., and Goodman, N, Concurrency Control in Distributed Database system, VLDB, 2000

المبحث الثاني : دوال السرية و حماية و تأمين البيانات في اوراقل :

1-5 ما هي اوراقل :

هي شركة متخصصة في مجال قواعد البيانات تقدم مجموعة هائلة من المنتجات البرمجية في هذا المجال و تشمل

ثلاث فئات أساسية :

- نظام إدارة قواعد البيانات
- نظم تطوير تطبيقات لاستخلاص البيانات من نظام الإدارة
- نظم جاهزة تضمن تلبية حاجة السوق البرمجي

2-5 أنواع مستخدمي قواعد البيانات :

تتنوع مهام وظائف المعنيين بإدارة قواعد البيانات اوراقل طبقا لحجم العمل و حجم قواعد البيانات و أهمية قاعدة البيانات, و بناء على تلك العوامل يمكن تخصيص أو دمج المهام المتعلقة بإدارة و تشغيل قاعدة البيانات, و يمكننا أن نذكر التخصصات الموصى بها من قبل شركة اوراقل في توزيع المهام على حسب الوظائف التالية :

- مدير قاعدة البيانات Database Administrator .
- مسئول أمن البيانات Security offeser .
- مدير الشبكة Network Administrator .
- مطور التطبيقات Application Developer .
- مدير التطبيقات Application Administrator .
- مستخدم قاعدة البيانات Database User .

3-5 إدارة المستخدمين :

ما هي الحاجة لوجود مستخدمين ؟

تكمّن الحاجة في حتمية وجود خصوصية للبيانات بحيث يستطيع كل شخص أن يحتفظ ببياناته و قدرته على

تعديلها و منح الحق لغيره بالاطلاع و إجراء العمليات المختلفة عليها . (31)

(31) () 1- على كمال، نظم إدارة قواعد البيانات، القاهرة ، الدار المصرية ،السنة 2005

2- جمال بطيخ ، قواعد البيانات، سوريا ، دار شعاع ،السنة 2003

1- دوال السرية و حماية و تأمين البيانات في اوراكل :

1- إنشاء المستخدمين Database User :

يمكن منح الصلاحية للمستخدم أثناء الإنشاء أو بعد ذلك و تكون تعليمة الإنشاء كلمة User كما يلي:

```
CREATE USER user (user name)
IDENTIFIED {BY password } [ DEFAULT TABLESPACE tablespace ] TEMPORARY
TABLESPACE tablespace [ [ QUOTA {integer [K | M ] | UNLIMITED } ON tablespace] [
PASSWORD EXPIRE ] [ ACCOUNT { LOCK | UNLOCK } ] [ PROFILE { profile |
DEFAULT } ]
```

و يكون الإنشاء كالتالي

```
create user Mustafa
identified by jazeera_pas;
```

و إذا أردنا أن نتوسع أكثر و نجعل للمستخدم صلاحية على User Tabale space و هو فضاء جدولي

افتراضي تم إنشائه عند تنصيب اوراكل يكون ذلك كالتالي :

```
create user mustafa identified by mustafa_pas
default tablespace users
quota 50 m on users;
```

و يمكن تغيير كلمة المرور للمستخدم في أي وقت كالتالي :

```
create user mustafa
identified by mustafa_pass
default tablespace users
quota 50m on users
password Expire;
```

أيضا يمكن التعديل على كلمة المرور للمستخدم كما يلي :

```
ALTER USER user  
[ DEFAULT TABLESPACE tablespace]  
[ TEMPORARY TABLESPACE tablespace]  
[ QUOTA {integer [K | M] | UNLIMITED } ON  
tablespace  
...]
```

و يمكن للمستخدم نفسه أن يقوم بتغيير في كلمة المرور الخاصة به كما يلي :

```
SQL> alter user Mustafa  
2 quota 50m on users;  
User altered
```

يكون فتح حساب المستخدم و إغلاقه كما هو موضح أدناه

- إغلاق الحساب .

```
SQL> alter user mustafa account lock;
```

- فتح الحساب .

```
SQL> alter user mustafa account unlock;
```

4-5 تسجيل الدخول بالمستخدم :

```
SQL> create user Mustafa  
2 identified by jazeera_pass;  
User created.  
SQL> connect mustafa / jazeera_pass  
ERROR:  
ORA-01045: user jazeera lacks CREATE SESSION privilege; logon denied  
Warning: You are no longer connected to ORACLE.
```

5-5 الصلاحيات:

الصلاحيات هي قدرة المستخدم على عمل شي ما. و تعتبر اوراكل من اقوي نظم الإدارة التي تحوي على صلاحيات متنوعة تعاد تشمل كل شي حتى الصلاحيات الصغيرة و هي تعتبر نقطة قوة في اوراكل. و تنقسم الصلاحيات الى

نوعين :

1- صلاحيات على مستوى النظام System Level .

و هذه الصلاحية تسمح بتنفيذ على العمليات على القاعدة بشكل عام كإنشاء الجداول و حذفها.

```
GRANT {system_privilege|role}
[, {system_privilege|role} ]...
TO {user|role|PUBLIC}
[, {user|role|PUBLIC} ]...
[WITH ADMIN OPTION]
```

2- صلاحيات على مستوى الغرض Object .

و هذه الصلاحية تسمح بتنفيذ العمليات على شكل محدد, مثلا حذف سجلات من الجدول

5-6 أنواع صلاحيات على مستوى الغرض : Object

جدول (3.2) أنواع الصلاحيات

الصلاحية	الفئة
INDEX	CREATE ANY INDEX ALTER ANY INDEX DROP ANY INDEX
TABLE	CREATE TABLE CREATE ANY TABLE ALTER ANY TABLE DROP ANY TABLE SELECT ANY TABLE UPDATE ANY TABLE DELETE ANY TABLE
SESSION	CREATE SESSION ALTER SESSION RESTRICTED SESSION
TABLESPACE	CREATE TABLESPACE ALTER TABLESPACE DROP TABLESPACE UNLIMITED TABLESPACE

المصدر: Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

الصيغة العامة كالتالي:

```
GRANT { object_privilege [(column_list)]
[, object_privilege [(column_list)] ]...
|ALL [PRIVILEGES]} ON [schema.]object
```

**TO {user|role|PUBLIC}[, {user|role|PUBLIC}]...
[WITH GRANT OPTION]**

5-7 دوال السرية و حماية و تأمين البيانات في اوراكل :

في الاوراكل 11g كل الذي تطرقنا له في 10g نجده في الإصدار 11g مع وجود بعد الاختلافات و التغييرات في شفرات المستخدم و الصلاحيات و سوف يكون ذلك واضحا في مرحلة التحليل لاحقا من الأشياء التي يمكن ذكرها في 11g كما يلي:

• الوظائف:

هي مجموعة صلاحيات تتميز بسهولة استخدامها حيث يمكن إسنادها الى المستخدم و سحبها منه و يمكن إسناد أكثر من وظيفة role لمستخدم واحد و يمكن إسنادها الى أكثر من مستخدم و يمكن إنشاء الوظيفة Role حسب الصيغة التالية:

**CREATE ROLE role [NOT IDENTIFIED |
IDENTIFIED
{BY password | GLOBALLY }]**

و يمكن إننا نستطيع إسناد role الى أخرى فتصبح الثانية تحوى صلاحيات الاولى على حسب الصيغة التالية

**GRANT role [, role]...
TO {user|role|PUBLIC}
[, {user|role|PUBLIC}]...
[WITH ADMIN OPTION]**

• التمكين و إلغاء التمكين:

يمكن للمستخدم أن يفعل أو يلغي وظيفة role معينة من قاعدة البيانات على حسب الصيغة العامة التالية :
التفعيل الصلاحية و ذلك باستخدام التعليمة set كالتالي:

Set role user name _role;

❖ سحب الصلاحية و ذلك باستخدام التعليمة User كالتالي:

SQL> alter user user name

2 default role mustafa _role ⁽³²⁾

[32][1] Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

[2] Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.

[3] Bernstein, P., and Goodman, N, An algorithm for Concurrency Control and Recovery in Replicated Distributed Database, TODS,

2004

المبحث الثالث : آلية عمل هندسة أداء البرمجيات :

بالزامو (Balsamo 2004) شرح آلية عمل أسلوب هندسة الأداء للبرمجيات. حيث أوضح إنها تقوم أساسا على نمودجين اثنين يساعدان على دمج عملية تقييم الأداء داخل مراحل تطوير النظام: النموذج الأول هو نماذج تنفيذ البرمجيات و الثاني نماذج تنفيذ النظام. نماذج تنفيذ البرمجيات تستخدم مخططات التنفيذ لتمثيل سلوك البرمجيات و فى نفس الوقت تعتبر احد مدخلات نموذج تنفيذ النظام. نماذج تمثيل النظام تمثل منصة النظام و تقوم باستقبال مخرجات بيانات النموذج السابق يضاف لها المعلومات المتعلقة بالأجزاء الصلبة. عليه فان نموذج تنفيذ النظام يمثل كل النظام (برمجيات، و مكونات صلبة أو أجهزة). يعتبر نموذج تنفيذ النظام الأكثر تعقيدا و يتم بناؤه على أساس قائمة أنظار الشبكة. و عليه فان النموذج الناتج قابل للتحليل و يتم مقارنته بمتطلبات الأداء الموضوعة سلفا. برناردو (Bernardo, Hillston et al. 2007) يوفر تفاصيل شاملة حول خطوات و آلية تنفيذ أسلوب هندسة أداء البرمجيات فى تقييم كفاءة الأداء فى دوال السرية فى لغة برمجة قواعد البيانات اوراكل.

خلصت الدراسة أعلاه أن هذا الأسلوب عند إتباعه يحسن من التوقع للأداء .يتم استخدام النماذج للتحكم فى برمجة و إدارة دوال السرية المذكورة أعلاه بصورة فاعلة. لا بد من التحقق من توافق الأداء مع متطلبات النظام فى تطبيق دوال السرية خلال مراحل بناء النظام. و بعدها يتم التحكم فى بدائل التصميم المتاحة و اتخاذ القرار بخصوصها بناء على متطلبات الأداء المطلوبة. و التى تساعد فى اتخاذ قرارات تقييم دوال سرية اوراكل و تطويرها للأكثر صوابا

6-2-1 تطوير دوال التأمين والسرية فى نموذج نظم قواعد البيانات النشطة :

برز فى الآونة الأخيرة و بشكل واسع جدا أسلوب (التطوير المدفوعة بالنماذج) لبناء البرمجيات و التركيز على بناء دوال السرية فى نظم قواعد البيانات (قواعد البيانات النشطة) موضوع البحث. هذا الأسلوب فى بناء النظم يبنى عملية التطوير و المراد هنا تطوير دوال التأمين و السرية حول النموذج، على عكس الأسلوب التقليدي (التطوير حول الشيفرة). الأسلوب يعتمد لغة معيارية لتعريف النماذج مثل (لغة النمذجة الموحدة) ماخذًا فى

الاعتبار أن لغة اوراكل تعتبر لغة برمجة لقواعد البيانات و لغة برمجة كائنية المنحى. كما يتيح تحويل النماذج من شكل الى آخر فضلا عن إمكانية توليد شيفرات البرامج من المستوى الأعلى للتجريد (A. W. Brown (2006). و يمكن أن يتم توليد الشيفرة بصورة آلية كاملة أو شبه آلية (Giese. Holger., Fritzsche. (2008) Mathias. et al. في قواعد البيانات النشطة و من أمثلتها الزنادات. يتم فى المرحلة الاولى تمثيل النظام بلغة النمذجة الموحدة و يكون هذا التمثيل بعيدا عن التفاصيل التقنية و يعرف بالنموذج المستقل عن الحاسوب و بواسطة عملية تحويلية يتم الوصول بهذه النماذج الى المستوى الثانى و هو مستوى النماذج المستقلة عن منصة العمل و بإجراء عملية تحويل جديدة يتم الحصول على النماذج المعتمدة على منصة عمل محددة. أن أسلوب التطوير المدفوع بالنموذج يعتمد النموذج كأداة أولية فى بناء النظم. هذه النماذج يمكن تحويلها الى نوعية أخرى من النماذج بصورة آلية. و هذا التحويل يتم بناءا على معايير متوافقا لتي يتم بها تقييم دوال السرية.

6-2-2-2 توظيف طريقة هندسة البرمجيات لتوقع أداء دوال التأمين و السرية فى نموذج نظم قواعد البيانات النشطة :

تقييم الأداء حتى مرحلة الانتشار، اقترحت عدة طرق لاختبار الأداء و التحقق من مطابقته للمتطلبات فى وقت مبكر . مما يتيح فرصة أفضل للتقييم و اتخاذ الإجراءات التصحيحية اللازمة و من ناحية أخرى إتاحة دراسة بدائل التصميم أثناء فترة بناء النظام و ليس بعد ذلك. لهذا الغرض يتم توظيف طريقة تطوير دوال التأمين و السرية لنظم, و لتحقيق إنتاج برمجيات بجودة و أداء سرية أعلى. تركز الطريقة على كيفية تخصيص الموارد اللازمة للنظام بصورة مثالية مع مراعاة كيفية تأثير هذه الموارد على أداء النظام (Tawhid 2008). حيث يتم بناء نماذج للأداء مشتقة من نماذج تصميم البرمجية نفسها ، و يتم ذلك بشكل مستمر ، وبعدها يتم مقارنة النماذج مع المتطلبات المحددة مسبقا لمعرفة مدى استيفائها لوحدة المستخدم من عدمه، و عليه يمكن القيام بالتغييرات اللازمة فى وقت مبكر. وعلاوة على ذلك، يمكن أن يتم التكيف فى نموذج بديل و المفاضلة بين التصاميم المختلفة المقترحة باختيار أفضل تصميم يحقق أعلى جودة أداء للنظام (Street 2007). فعلاوة

على أنها تسهل من تنفيذ البرمجيات المعقدة فان استخدام هذه الطريقة يعتبر فعالا من حيث التكلفة و يتيح تطوير النظم فترة زمنية معقولة، وكذلك بأقل جهد . و بعض الباحثين استخدموا هذه الطريقة بفاعلية لتصميم ما يعرف بالنماذج المضادة و مهمتها الذهاب ابعدها من اكتشاف وجود المشكلة الى تحديد وصفة جاهزة مضادة تتناسب نوعية المشكلة المحددة , كمثل يمكن الرجوع الى (Cortellessa, Di Marco et al. 2010).

في الفقرة الحالية عرضنا بإيجاز ماهية و أهمية أسلوب تطوير النظم المدفوع بالتمازج و دوره في إنتاج برمجيات عالية الأداء في سرية و تأمين قواعد البيانات بشكل عام و قواعد البيانات النشطة بشكل خاص. فالمرونة التي توفرها الطريقة تساعد في تجاوز تلك النظم المعقدة التي يتم تطويرها بواسطة النظم القائمة على المكونات . الفقرة التالية و هي محور هذه البحث في استخدام هندسة البرمجيات في تقييم كفاءة الأداء في دوال التأمين و السرية في تطبيقات لغة اوراكل العناصر المكونة لمعمارياتها التي لها تأثير في أداء البرمجيات ثم استعراض لبعض أوجه القصور فيها.

3-2-6 التوقع للأداء في نظم هندسة البرمجيات لدوال التأمين و السرية في نموذج نظم قواعد البيانات النشطة :

يستعرض هذا الفصل بعض المشاكل و التعقيدات التي يجب أخذها في الاعتبار عند تطوير طريقة أو أسلوب جديد لتوقع أداء في دوال التأمين و السرية في البرمجيات. ،معظم هذه المشاكل تتبع من طبيعة الأسلوب نفسه، و إضافة الى ذلك نستعرض أهم أنواع الطرق التي أتيج لنا الاطلاع عليها عبر تطبيق الخوارزميات التالية .

4-2-6 المشاكل المتعلقة بعملية التوقع للأداء :

نعرض في هذه الفقرة بعض الجوانب المهمة التي تؤثر سلبا في دقة التوقع لأداء دوال التأمين و السرية في البرمجيات ، و نشير هنا الى أن هذه الجوانب ذات علاقة وثيقة بخصوصية طريقة التطوير، و المشاكل نلخصها فيما يلي :

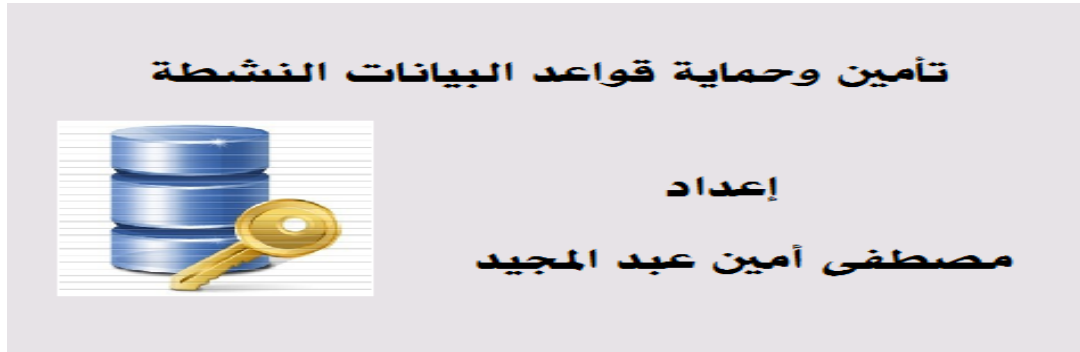
1- فى نظم قواعد البيانات يتربك النظام من مجموعة مكونات يتم تأليفها مع بعضها البعض بحيث تحقق أهداف النظام. ملاحظ أن المكونات هذه عند بناءها تراعى فقط الظروف و الافتراضات عند التصميم، بينما يجب أن تكون معدة للتعامل فى العالم المفتوح، بمعنى أن تكون قادرة على التكيف مع ظروف محيطية أخرى من المحتمل أن تعمل فيها لاحقاً و فى الغالب تكون فى وقت التنفيذ. (Ghezzi 2009) يتسبب عدم مراعاة هذا التباين فى الحصول على دقة اقل فى توقع الأداء و بالتالى جودة برمجية ضعيفة.

2- وثمة مشكلة أخرى تقلل من فعالية معظم الأساليب المتبعة للتوقع. فبالرغم عن توفر العديد من الأدوات التى تساعد فى تحليل التغذية الراجعة من نموذج التحليل إلا انه ما تزال هنالك حاجة الى تواجد خبراء أداء البرمجيات للقيام بتفسير المخرجات الناتجة عن نموذج الأداء. (Mirandola, Gorton et al. 2009)، ذلك أن النتيجة الراجعة تكون غامضة أحياناً و غير مفهومة للمطور العادي، و هذا يزيد من التكلفة و يحد من فعالية تعامل المطورين غير المتخصصين فى أداء البرمجيات من اتخاذ قرارات بدائل التصميم السليمة.

3- معلوم بان الهدف الرئيسى من التوقع لأداء البرمجيات هو تحسين أداء المنتج البرمجي (دوال التأمين و السرية). و لما كانت نظم قواعد البيانات تقوم على فكرة تجميع و ربط المكونات الضرورية من مصادرها المختلفة لتكوين النظام. فالبدهى أن زمن استجابة النظام ككل يمكن حسابها بتجميع زمن استجابة المكونات التى يتألف منها النظام. هذه الفرضية تستندم بواقع مغاير و هو أن زمن الاستجابة للمكون الواحد ربما يعتمد على انتظار مدخلات من خدمات أو مكونات أخرى أو ينتظر إكمال إرسال مخرجات الى خدمات أو مكونات أخرى و بالتالى فان عدم اخذ هذه الحالة فى الاعتبار كما هو حادث فى كثير من أساليب التوقع إنما يقود الى عدم دقة زمن الاستجابة الكلى للنظام و بالتالى يؤدي الى عدم مطابقة الأداء للمتطلبات. و فى حالات أخرى يكون زمن الاستجابة للمكون مبنى على أساس منصة تشغيل محددة بتغيرها تظهر الحاجة لإعادة الإعدادات المتعلقة بالأداء و بالتالى بذل جهد اضافى لتحقيق الأداء المطلوب لذا قدمت فى هذا البحث بعمل تطبيقي لخوارزميات تمثل فى ثلاث خوارزميات أساسية لحماية و تأمين البيانات و هي :

- خوارزمية الوصول للبيانات من خلال اسم المستخدم و كلمة المرور .
- خوارزمية تغيير كلمة المرور . وخوارزمية النسخ الاحتياطي.

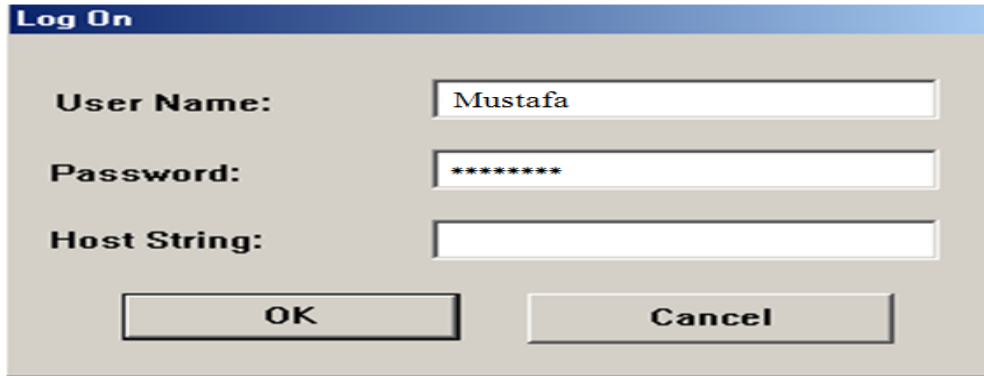
الشاشة الرئيسية لتطبيق خوارزميات تأمين و حماية قواعد البيانات النشطة



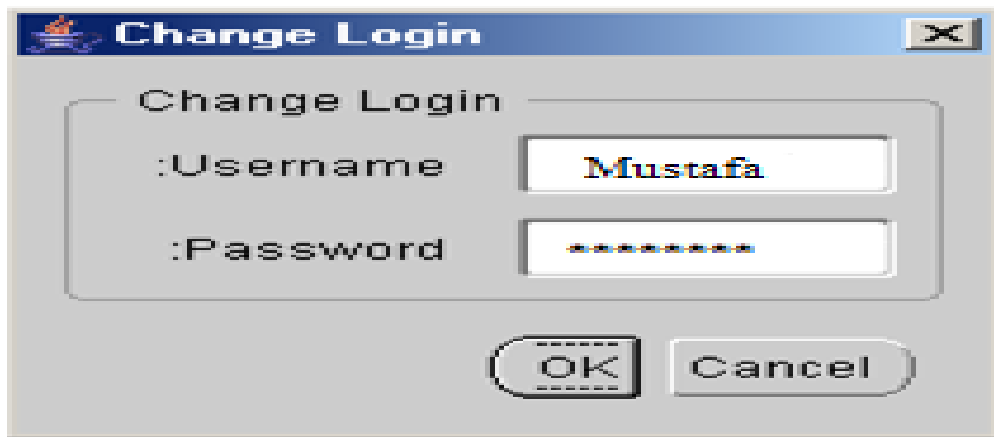
شكل (3.1) الشاشة الرئيسية للنظام

المصدر: إعداد الباحث

شاشة خوارزمية الوصول للبيانات باستخدام اسم المستخدم و كلمة المرور



شكل (3.2) شاشة الدخول للنظام



شكل (3.3) شاشة تغيير كلمة المرور

المصدر: إعداد الباحث



شكل (3.4) شاشة النسخ الاحتياطي

المصدر: إعداد الباحث

4-6 المؤشرات :

- 1-الدخول الى النظام .
- اسم المستخدم .
- كلمة المرور .
- 2-التحكم في الدخول الى النظام .
- 3-النسخ الاحتياطية للبيانات .

1-4-6 تحليل المؤشرات :

الدخول الى النظام :

البيانات المطلوبة لإحتساب مؤشر الدخول الى النظام :

- نسبة الموظفين بدوام كامل و الذين يستخدمون النظام (Employees to FT Ratio).
- نسبة الموظفين بدوام جزئي و الذين يستخدمون النظام (Employees to FTE Ratio).

جدول (3.3) الموظفين بدوام كامل (FT)

م	الاسم	الوظيفة	عدد ساعات العمل على النظام
1	المستخدم رقم 1	مدير إدارة	6
2	المستخدم رقم 2	مدير عام	2
3	المستخدم رقم 3	موظف عمليات	9
4	المستخدم رقم 4	موظف عمليات	9
5	المستخدم رقم 5	موظف عمليات	9
6	المستخدم رقم 6	موظف عمليات	9
7	المستخدم رقم 7	مدير إدارة	6
8	المستخدم رقم 8	مستخدم	2
9			
10			
11			
12			
	متوسط عدد ساعات العمل على النظام (Average Workload FT)		7

المصدر: إعداد الباحث

Calculation of the Full Time Equivalent Employees Number of registered customer

$$\text{Customers} / \text{Number of FT Employees} = 80/8 = 10$$

جدول (3.4) الموظفون بدوام جزئي (PT)

م	الاسم	الدرجة	العبء التدريسي للفصل الحالي
1	المستخدم رقم 1	تعاقد جزئي	4
2			
3			
4			
5			
6			
7			
8			
إجمالي عدد الساعات (Total) (Workload PT)			4

المصدر: إعداد الباحث

Calculation of the Full Time Equivalent Employees (FTE)

$$FTE = FT \text{ number} + \frac{\text{Total Workload PT}}{\text{Average Workload FT}}$$

$$FTE = 10 + \frac{4}{7} = 10$$

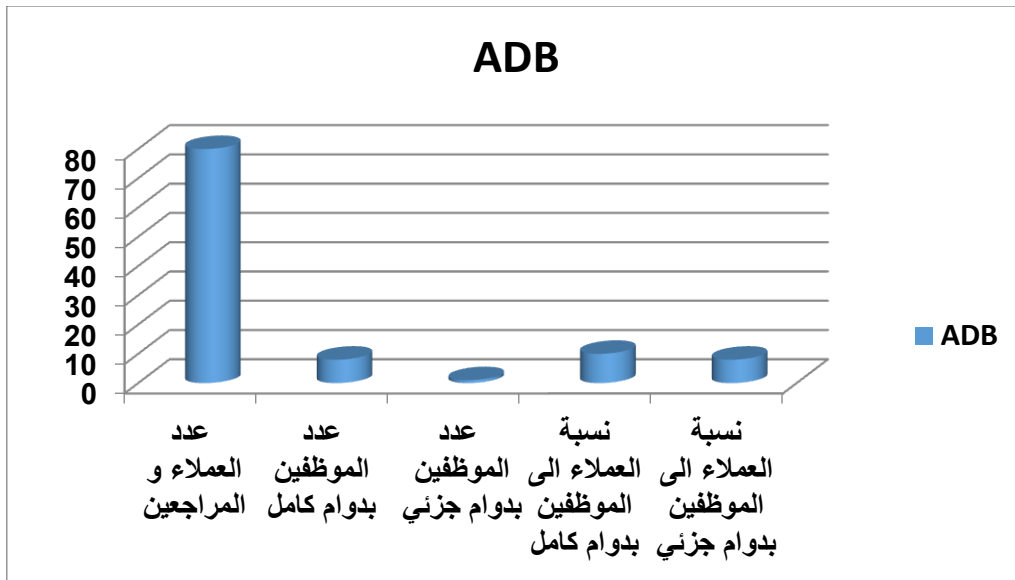
Ratio Customers to FTE faculties:

$$\text{Number of registered Customers} / \text{Number of FTE Employees} = \frac{80}{10} = 8$$

جدول (3.5) دخول الى النظام

البرنامج	عدد العملاء و المراجعين	عدد الموظفين بدوام كامل	عدد الموظفين بدوام جزئي	نسبة العملاء الى الموظفين بدوام كامل	نسبة العملاء الى الموظفين بدوام جزئي
ADB	80	8	1	10	8

المصدر: إعداد الباحث



شكل (3.5) رسم بياني عدد الدخول للبرنامج

المصدر: إعداد الباحث

التعريف بالمؤشر :

نسبة العملاء الى الموظفين بدوام كامل :

ع هو نسبة أعداد العملاء و المراجعين إلى عدد الموظفين بدوام كامل .

هو نسبة أعداد العملاء و المراجعين إلى عدد الموظفين بدوام جزئي و ما يعادله

الهدف :

استخدام هذا المقياس في رسم السياسات العامة على تامين و حماية البيانات و الوصول إليها من خلال اسم المستخدم و كلمة المرور الخاصة بالموظفين و يعتبر هذا المقياس مهم جدا في عملية تقييم جودة البرنامج .

إجراءات القياس و النتائج :

(1) نسبة العملاء و المراجعين الى الموظفين بدوام كامل .

يتم تطبيق هذا المؤشر بعد فترة كافية من العمل عليه بعد حدوث الثبات النسبي لكل من العملاء الموظفين .

• يتم حساب أعداد العملاء على فترات متتالية .

• يتم حساب أعداد الموظفين الذين على رأس العمل وبدوام كامل .

• يتم حساب نسبة العملاء الي الموظفين .

نسبة العملاء الى الموظفين بدوام كامل = اجمالي عدد العملاء / اجمالي الموظفين بدوام كامل

(2) نسبة العملاء الى الموظفين بدوام جزئي و ما يعادله .

و كانت النتائج كما يلي :

1- نسبة الموظفين بدوام كامل و الذين يستخدمون النظام = 10.

2- نسبة الموظفين بدوام جزئي و الذين يستخدمون النظام = 8.

التحكم في الدخول الى النظام :

من سياسات التأمين و الحماية التي تم تطبيقها على النظام هي إعادة تغيير كلمة المرور بعد كل ثلاثة أشهر و ذلك لضمان سرية و تأمين و حماية البيانات في نظام قواعد البيانات كما هو موضح في الشكل (3.3) و ذلك يعطي جودة ضمان تأمين عالية لإتباع سياسة من سياسات التأمين العالمية لقواعد البيانات

النسخ الاحتياطية للبيانات :

تعتبر التغذية الراجعة من المستخدمين للبرنامج مهمة لتحسين جودة البرامج. و من خلال البيانات و المعلومات الواردة في إاستبانه و التي تم جمع المعلومات من خلالها بالإجابة على كل التساؤلات الواردة فيها و بعد تحليلها كانت النتائج كالآتي :

تعريف المؤشر :

تقدير جودة خوارزمية النسخ الاحتياطي في تأمين و حماية بيانات النظام من خلال استبانه تقييم الكفاءة في الأداء . حيث أن لا تقل نسبة الرضا عن 70% من التقييم .

أهداف قياس المؤشر :

1- قياس جودة الخوارزمية التي إنشاءها في النظام.

2- وجود مقياس لجودة عملية النسخ الاحتياطي على مستوى النظام. (33)

(33) [1]Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.

[1]Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.

نتائج استبانة تقييم خوارزمية النسخ الاحتياطي :

جدول (3.6) نتائج استبانة خوارزمية النسخ الاحتياطي

الوسط الحسابي	الإجابات					الأسئلة
	لا أوافق بشدة	لا أوافق	صحيح لحد ما	أوافق	أوافق بشدة	
3.67	1	2	5	20	2	1
4.00	0	1	3	21	5	2
3.90	1	1	3	20	5	3
4.00	0	2	5	19	5	4
3.87	1	1	3	21	4	5
3.93	1	1	2	21	5	6
4.13	0	1	0	23	6	7
3.47	2	3	5	19	1	8
3.77	1	1	4	22	2	9
3.93	1	2	2	18	7	10
3.57	1	3	5	15	5	11
3.83	2	1	2	20	5	12
3.70	1	1	6	20	2	13
4.30	2	1	1	18	10	14
4.73	1	1	3	20	10	15
3.93	1	1	2	21	5	16
3.93	1	2	1	20	6	17
4.20	0	0	2	20	8	18
3.77	1	2	2	23	2	19
3.80	1	2	4	18	5	20
3.83	1	1	3	22	3	21
3.73	1	2	3	22	2	22
	21	32	66	443	105	المجموع

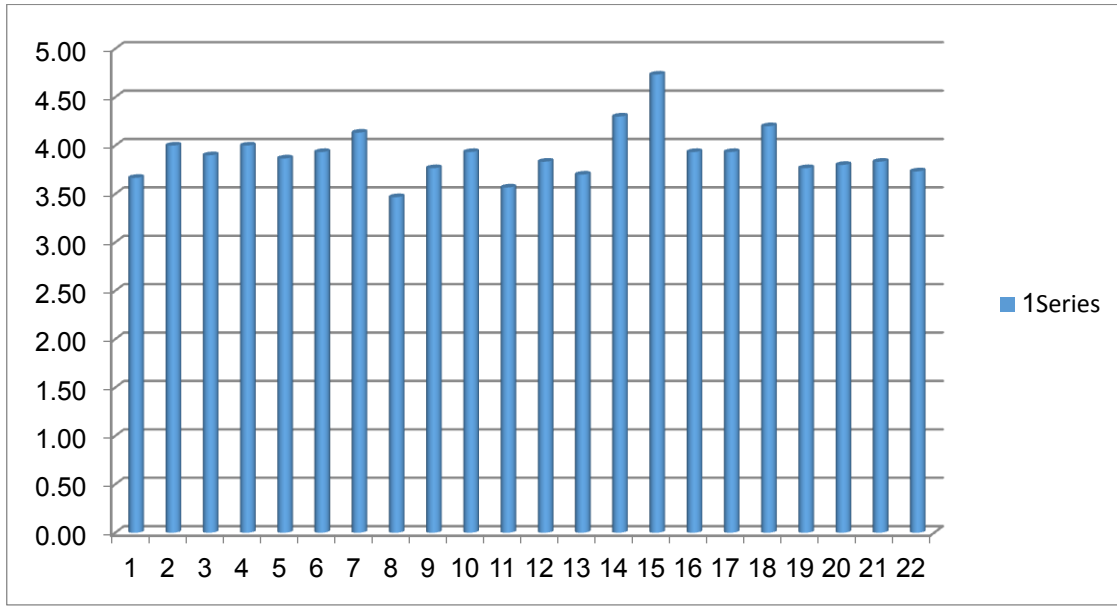
المصدر: إعداد الباحث 2016

جدول (3.7) نسبة الرضا بتقييم النسخ الاحتياطي

15.74	105	أوافق بشدة
66.42	443	أوافق
9.90	66	صحيح لحد ما
4.80	32	لا أوافق
3.15	21	لا أوافق بشدة
	667	المجموع

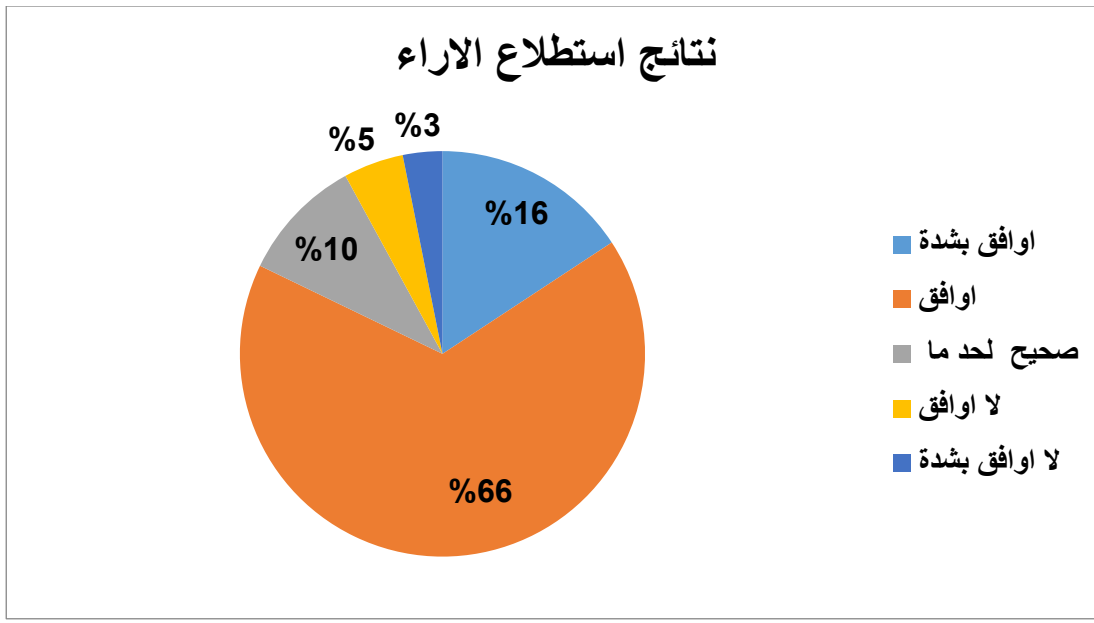
667	مجموع الوسط الحسابي
-----	---------------------

المصدر: إعداد الباحث 2016



شكل (3.6) رسم بياني لتحليل النسخ الاحتياطي

المصدر: إعداد الباحث 2016

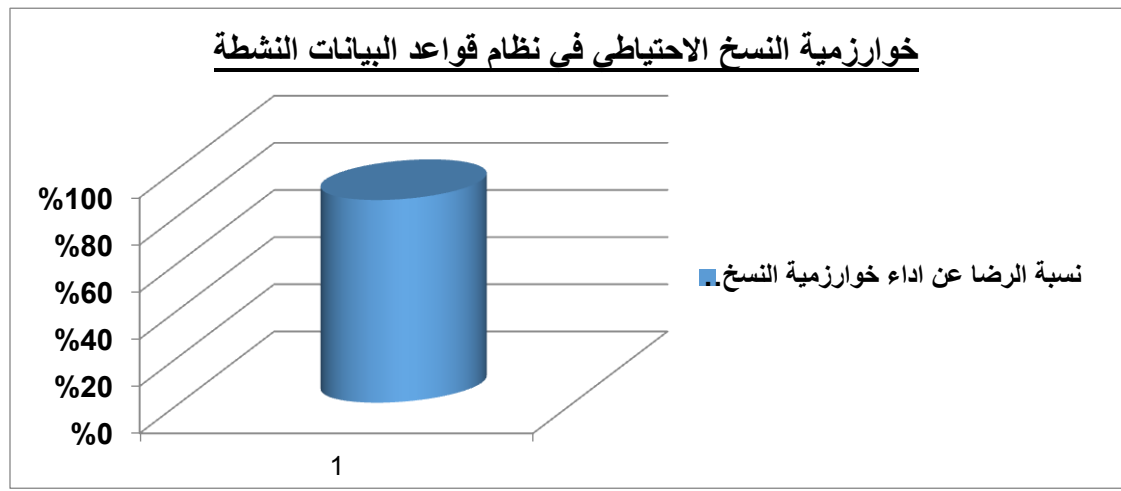


شكل (3.7) رسم بياني يوضح نتائج استطلاعات الآراء للنسخ الاحتياطي

المصدر: إعداد الباحث 2016

نتيجة تطبيق المؤشر :

نسبة الرضا عن أداء خوارزمية النسخ الاحتياطي كانت 92.6% وذلك بمجموع (أوافق بشدة وأوافق وصحيح لحد ما) كما في الشكل (3.7).



شكل (3.8) يوضح نسبة الرضا بخوارزمية النسخ الاحتياطي

المصدر: إعداد الباحث 2016

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

المستخلص

هدفت الدراسة الى التعرف على موضوع استخدام هندسة البرمجيات في تقييم كفاء الاداء لدوال التأمين و السرية في قواعد البيانات النشطة وقد استخدمت الدراسة المنهج الوصفي والتطبيقي في تحقيق الجودة من خلال ثلاث مؤشرات لقياس وتقييم جودة خوارزميات حماية وتأمين البيانات والمؤشرات هي مؤشر الدخول الى النظام باسم المستخدم وكلمة المرور, مؤشر تغيير كلمة المرور ومؤشر النسخ الاحتياطي للبيانات ومن ثم القيام بتحليلها باستخدام آلية قياس هندسة البرمجيات والاستعانة باستبانة لجمع البيانات لتقييم خوارزمية النسخ الاحتياطي للبيانات للمؤشر الثالث والتوافق النسبي مع السياسات العالمية لحماية وتأمين البيانات ثم جمع وتحليل البيانات واختبار الفرضيات لتحقيق أهداف الدراسة وتمثلت أهم النتائج التي توصلت إليها الدراسة في الآتي :

- تحقق طريق القياس نسبة عالية لمعيار الدقة في التأمين وسرية وحماية البيانات .
- طريقة القياس تحقق ثبات نسبي لكل من العملاء والموظفين بدوام كامل ودوام جزئي في المؤشر الأول .
- تم تطبيق سياسة تغيير المستخدم وكلمة المرور المذكورة في المؤشر الثاني موافقة بما جاء في السياسات العالمية لتأمين وحماية البيانات .
- نتيجة تقييم القياس لجودة خوارزمية النسخ الاحتياطي مرضية للغاية وكانت النتيجة %92.6 في المؤشر الثالث .

Abstract

The study aimed to identify the use of software engineering in the evaluation of the performance efficiency of the functions of security and confidentiality in active databases, study used a descriptive method and applied method in achieving the quality through three indicators to measure and assesses the quality of protection and secure data algorithms. These indicators are entering the system through username and password index, password index and data backup the index, then analyzing it using software engineering measurement mechanism, and then use a questionnaire to collect data for assessing the backup data algorithm for the third index, and relative compatibility with global policies to protect and secure data, then collecting and analyzing data and testing hypotheses to achieve the objectives of the study.

The most important results of the study are the following:

Measurement method achieves a high rate of accuracy standard in the security, confidentiality and data protection.

Measurement method achieves proportional stability for customers, full-time employees and part-time employees in the first indicator.

The policy of change the user name and password mentioned in the second indicator had been applied, in accordance with the global policy to secure and protect data.

The result of assessing the measurement of the algorithm quality for the backup is very satisfactory evaluation, which was 92.6% in the third indicator.

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

قائمة المحتويات

الصفحة	الموضوع
أ	الاستهلال
ب	الإهداء
ج	الشكر و التقدير
د	مستخلص البحث باللغة العربية
هـ	مستخلص البحث باللغة الانجليزية
و	قائمة الموضوعات
ك	قائمة الجداول
ل	قائمة الأشكال
الفصل الأول	
1	1-1 تمهيد
3	2-1 مشكلة الدراسة
3	3-1 أهمية الدراسة
3	4-1 أهداف الدراسة
4	5-1 منهج الدراسة
4	6-1 فرضيات الدراسة
4	7-1 حدود الدراسة
الفصل الثاني: الإطار النظري	
5	المبحث الأول : نظم إدارة قواعد البيانات
7	1-2 أنواع نظم إدارة قواعد البيانات
8	2-3-1 نظم إدارة قواعد البيانات الشبكة
9	2-3-2 نظم إدارة قواعد البيانات العلائقية
12	4-2 تطبيقات قواعد البيانات
17	6-2 عمليات نظام إدارة قاعدة البيانات

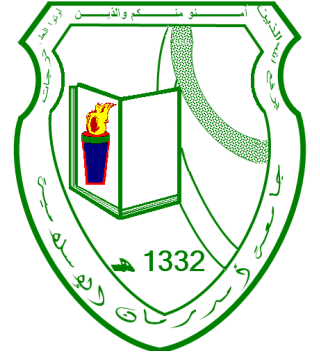
17	7-2 تعريف الزنادات
18	المبحث الثاني: مفهوم ومكونات ومراحل تنفيذ الزنادات
18	1-7-2 مكونات الزنادات
19	2-7-2 أنواع الزنادات
20	3-7-2 مراحل تنفيذ الزنادات
الفصل الثالث: تقنية هندسة البرمجيات	
21	المبحث الأول : مفهوم ومبادئ وأدوات هندسة البرمجيات
23	1-3 المهام النموذجية لهندسة البرمجيات
25	2-3 مبادئ هندسة البرمجيات
28	3-3 أنواع البرمجيات
33	4-3 المقاييس التي تستخدم لقياس تعقيد المنتج البرمجي
36	5-3 المعايير الدولية لجودة البرمجيات
39	6-3 أهداف التقييس وفوائده
41	7-3 جودة البرمجيات
44	8-3 ميزات أدوات هندسة البرمجيات بمساعدة الحاسوب
46	1-8-3 قصور أدوات هندسة البرمجيات بمساعدة الحاسوب
47	2-8-3 التصنيف العام لأدوات هندسة البرمجيات المساعدة
50	المبحث الثاني : التحديات الرئيسية التي تواجه هندسة البرمجيات
51	10-3 مهنة البرمجيات
53	11-3 المسؤولية المهنية والأخلاقية لمهندسي البرمجيات
55	12-3-9 عمليات البرمجيات
55	12-3 جودة البرمجية
56	13-3 محللو النظم
58	14-3 مراحل تطوير النظام
58	1-14-3 مرحلة تجميع المتطلبات
60	2-14-3 مرحلة التطوير

61	3-14-3 مرحلة الاختبار وتشخيص الأخطاء
64	4-14-3 مرحلة الصيانة والارتقاء
65	3-15-3 نشاطات هندسة البرمجيات
65	3-15-1 إعداد الوثائق وإنتاجها
65	3-15-2 متابعة المشروع البرمجي ومراقبته
66	3-15-3 ضمان جودة المنتج البرمجي
66	3-15-4 إدارة تكوين البرنامج
66	5.5 إدارة المخاطر
67	3-15-3 مراحل تطوير النظام البرمجي
68	3-15-1 الفرق بين إدارة المشاريع الهندسية و إدارة مشاريع البرمجيات
68	3-16-3 نشاطات الإدارة
68	3-16-1 عموميات الإدارة
69	3-16-2 أفراد (فريق عمل) المشروع
69	3-16-3 تخطيط المشروع
70	3-16-4 إجرائية تخطيط المشروع
71	3-16-4 هيكل خطة المشروع
71	3-16-5 تنظيم النشاطات
71	3-16-6 الجدولة الزمنية للمشروع
72	3-16-7 مشاكل الجدولة
72	3-16-8 المخططات البيانية و شبكات النشاطات
72	3-16-9 إجرائية إدارة المخاطر
74	المبحث الثالث : الدراسات السابقة
الفصل الرابع : الإطار التطبيقي	
83	المبحث الأول : الإطار التطبيقي
83	4-1 نماذج تحليل النظام
83	4-1-1 المرحلة الأولى مرحلة تجميع المتطلبات

85	4-1-2 المرحلة الثانية مرحلة التطوير
87	4-1-3 المرحلة الثالثة مرحلة الاختبار وتشخيص الأخطاء
89	4-1-4 المرحلة الرابعة مرحلة الصيانة والارتقاء
90	4-2 أنواع المطلوبات
90	4-2-1 المطلوبات الوظيفية
91	4-2-2 المطلوبات غير الوظيفية
94	4-3 وثيقة مواصفات المتطلبات
97	4-4 مستويات المتطلبات
98	4-5 مطلوبات المستخدم
100	4-6 مطلوبات النظام
101	المبحث الثاني : دوال السرية و الحماية لتأمين البيانات في اوراكل
101	5-1 ما هي اوراكل
101	5-2 أنواع مستخدمي قواعد البيانات
102	5-3 إدارة المستخدمين
104	5-7 تسجيل الدخول بالمستخدم
104	5-8 الصلاحيات
105	5-9 أنواع صلاحيات على مستوى الغرض
106	5-10 دوال السرية و حماية و تأمين البيانات في اوراكل
108	المبحث الثالث : آلية عمل هندسة أداء البرمجيات
108	6-1 تطوير دوال التأمين والسرية في نموذج نظم قواعد البيانات النشطة
109	6-2-2 توظيف طريقة هندسة البرمجيات لتوقع أداء دوال التأمين والسرية في نموذج نظم قواعد البيانات النشطة
110	6-2-3 التوقع للأداء في نظم هندسة البرمجيات لدوال التأمين و السرية في نموذج نظم قواعد البيانات النشطة
110	6-2-4 المشاكل المتعلقة بعملية التوقع للأداء

112	3-6 تطبيق الخوارزميات
113	4-6 المؤشرات
113	1-4-6 تحليل المؤشرات
الخاتمة	
121	4--1 النتائج
121	4-2 التوصيات
123	3-4 المراجع
130	4-4 الملحقات

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:



جمهورية السودان

جامعة أم درمان الإسلامية

معهد بحوث ودراسات العالم الإسلامي

قسم الدراسات التطبيقية

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات

بحث مقدم لنيل درجة الدكتوراه في علوم الحاسوب

إشراف :

د. عبد الرحمن الشريف كرار

إعداد :

مصطفى أمين عبد المجيد

2016-1437م

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

الاستهلال

قال الله تعالى :

﴿ وَقُلِ اعْمَلُوا فَسَيَرَى اللَّهُ عَمَلَكُمْ وَرَسُولُهُ وَالْمُؤْمِنُونَ ﴾

(سورة التوبة: 105)

الإهداء

إلى من أسير بفضل الله ثم بفضل دعائها ... الى أمي...

الي روح أبي الطاهرة....أسأل الله له الرحمة و المغفرة..

إلى من تبعث فيّ الأمل كلما أحسست بالملل ، إلى أم أولادي ..

إلى إخواني و أخواتي الى كل أهلي...

الشكر والتقدير

فالحمد لله رب العالمين والشكر خالص لوجهه الكريم أن وفقني لإعداد وإخراج هذا البحث.

الشكر أجزله للمشرف على هذا البحث د. عبد الرحمن الشريف لإشرافه المتميز وملاحظاته البناءة

والشكر لكل الإخوة الأستاذة الزملاء لما قدموه من توجيه وملاحظات والشكر أولاً وأخيراً لله الذي ألهم ويسر و وفق .

المستخلص

هدفت الدراسة الى التعرف على موضوع استخدام هندسة البرمجيات في تقييم كفاء الاداء لدوال التأمين و السرية في قواعد البيانات النشطة وقد استخدمت الدراسة المنهج الوصفي والتطبيقي في تحقيق الجودة من خلال ثلاث مؤشرات لقياس وتقييم جودة خوارزميات حماية وتأمين البيانات والمؤشرات هي مؤشر الدخول الى النظام باسم المستخدم وكلمة المرور, مؤشر تغيير كلمة المرور ومؤشر النسخ الاحتياطي للبيانات ومن ثم القيام بتحليلها باستخدام آلية قياس هندسة البرمجيات والاستعانة باستبانة لجمع البيانات لتقييم خوارزمية النسخ الاحتياطي للبيانات للمؤشر الثالث والتوافق النسبي مع السياسات العالمية لحماية وتأمين البيانات ثم جمع وتحليل البيانات واختبار الفرضيات لتحقيق أهداف الدراسة وتمثلت أهم النتائج التي توصلت إليها الدراسة في الآتي :

- تحقق طريق القياس نسبة عالية لمعيار الدقة في التأمين وسرية وحماية البيانات .
- طريقة القياس تحقق ثبات نسبي لكل من العملاء والموظفين بدوام كامل ودوام جزئي في المؤشر الأول .
- تم تطبيق سياسة تغيير المستخدم وكلمة المرور المذكورة في المؤشر الثاني موافقة بما جاء في السياسات العالمية لتأمين وحماية البيانات .
- نتيجة تقييم القياس لجودة خوارزمية النسخ الاحتياطي مرضية للغاية وكانت النتيجة %92.6 في المؤشر الثالث .

Abstract

The study aimed to identify the use of software engineering in the evaluation of the performance efficiency of the functions of security and confidentiality in active databases, study used a descriptive method and applied method in achieving the quality through three indicators to measure and assesses the quality of protection and secure data algorithms. These indicators are entering the system through username and password index, password index and data backup the index, then analyzing it using software engineering measurement mechanism, and then use a questionnaire to collect data for assessing the backup data algorithm for the third index, and relative compatibility with global policies to protect and secure data, then collecting and analyzing data and testing hypotheses to achieve the objectives of the study.

The most important results of the study are the following:

Measurement method achieves a high rate of accuracy standard in the security, confidentiality and data protection.

Measurement method achieves proportional stability for customers, full-time employees and part-time employees in the first indicator.

The policy of change the user name and password mentioned in the second indicator had been applied, in accordance with the global policy to secure and protect data.

The result of assessing the measurement of the algorithm quality for the backup is very satisfactory evaluation, which was 92.6% in the third indicator.

قائمة المحتويات

الصفحة	الموضوع
أ	الاستهلال
ب	الإهداء
ج	الشكر و التقدير
د	مستخلص البحث باللغة العربية
هـ	مستخلص البحث باللغة الانجليزية
و	قائمة الموضوعات
ك	قائمة الجداول
ل	قائمة الأشكال
الفصل الأول	
1	1-1 تمهيد
3	2-1 مشكلة الدراسة
3	3-1 أهمية الدراسة
3	4-1 أهداف الدراسة
4	5-1 منهج الدراسة
4	6-1 فرضيات الدراسة
4	7-1 حدود الدراسة
الفصل الثاني: الإطار النظري	
5	المبحث الأول : نظم إدارة قواعد البيانات
7	1-2 أنواع نظم إدارة قواعد البيانات
8	2-3-1 نظم إدارة قواعد البيانات الشبكة
9	2-3-2 نظم إدارة قواعد البيانات العلائقية
12	4-2 تطبيقات قواعد البيانات
17	6-2 عمليات نظام إدارة قاعدة البيانات

17	7-2 تعريف الزنادات
18	المبحث الثاني: مفهوم ومكونات ومراحل تنفيذ الزنادات
18	1-7-2 مكونات الزنادات
19	2-7-2 أنواع الزنادات
20	3-7-2 مراحل تنفيذ الزنادات
الفصل الثالث: تقنية هندسة البرمجيات	
21	المبحث الأول : مفهوم ومبادئ وأدوات هندسة البرمجيات
23	1-3 المهام النموذجية لهندسة البرمجيات
25	2-3 مبادئ هندسة البرمجيات
28	3-3 أنواع البرمجيات
33	4-3 المقاييس التي تستخدم لقياس تعقيد المنتج البرمجي
36	5-3 المعايير الدولية لجودة البرمجيات
39	6-3 أهداف التقييس وفوائده
41	7-3 جودة البرمجيات
44	8-3 ميزات أدوات هندسة البرمجيات بمساعدة الحاسوب
46	1-8-3 قصور أدوات هندسة البرمجيات بمساعدة الحاسوب
47	2-8-3 التصنيف العام لأدوات هندسة البرمجيات المساعدة
50	المبحث الثاني : التحديات الرئيسية التي تواجه هندسة البرمجيات
51	10-3 مهنة البرمجيات
53	11-3 المسؤولية المهنية والأخلاقية لمهندسي البرمجيات
55	12-3-9 عمليات البرمجيات
55	12-3 جودة البرمجية
56	13-3 محللو النظم
58	14-3 مراحل تطوير النظام
58	1-14-3 مرحلة تجميع المتطلبات
60	2-14-3 مرحلة التطوير

61	3-14-3 مرحلة الاختبار وتشخيص الأخطاء
64	4-14-3 مرحلة الصيانة والارتقاء
65	3-15-3 نشاطات هندسة البرمجيات
65	3-15-1 إعداد الوثائق وإنتاجها
65	3-15-2 متابعة المشروع البرمجي ومراقبته
66	3-15-3 ضمان جودة المنتج البرمجي
66	3-15-4 إدارة تكوين البرنامج
66	5.5 إدارة المخاطر
67	3-15-3 مراحل تطوير النظام البرمجي
68	3-15-1 الفرق بين إدارة المشاريع الهندسية و إدارة مشاريع البرمجيات
68	3-16-3 نشاطات الإدارة
68	3-16-1 عموميات الإدارة
69	3-16-2 أفراد (فريق عمل) المشروع
69	3-16-3 تخطيط المشروع
70	3-16-4 إجرائية تخطيط المشروع
71	3-16-4 هيكل خطة المشروع
71	3-16-5 تنظيم النشاطات
71	3-16-6 الجدولة الزمنية للمشروع
72	3-16-7 مشاكل الجدولة
72	3-16-8 المخططات البيانية و شبكات النشاطات
72	3-16-9 إجرائية إدارة المخاطر
74	المبحث الثالث : الدراسات السابقة
الفصل الرابع : الإطار التطبيقي	
83	المبحث الأول : الإطار التطبيقي
83	4-1 نماذج تحليل النظام
83	4-1-1 المرحلة الأولى مرحلة تجميع المتطلبات

85	4-1-2 المرحلة الثانية مرحلة التطوير
87	4-1-3 المرحلة الثالثة مرحلة الاختبار وتشخيص الأخطاء
89	4-1-4 المرحلة الرابعة مرحلة الصيانة والارتقاء
90	4-2 أنواع المطلوبات
90	4-2-1 المطلوبات الوظيفية
91	4-2-2 المطلوبات غير الوظيفية
94	4-3 وثيقة مواصفات المتطلبات
97	4-4 مستويات المتطلبات
98	4-5 مطلوبات المستخدم
100	4-6 مطلوبات النظام
101	المبحث الثاني : دوال السرية و الحماية لتأمين البيانات في اوراكل
101	5-1 ما هي اوراكل
101	5-2 أنواع مستخدمي قواعد البيانات
102	5-3 إدارة المستخدمين
104	5-7 تسجيل الدخول بالمستخدم
104	5-8 الصلاحيات
105	5-9 أنواع صلاحيات على مستوى الغرض
106	5-10 دوال السرية و حماية و تأمين البيانات في اوراكل
108	المبحث الثالث : آلية عمل هندسة أداء البرمجيات
108	6-1 تطوير دوال التأمين والسرية في نموذج نظم قواعد البيانات النشطة
109	6-2-2 توظيف طريقة هندسة البرمجيات لتوقع أداء دوال التأمين والسرية في نموذج نظم قواعد البيانات النشطة
110	6-2-3 التوقع للأداء في نظم هندسة البرمجيات لدوال التأمين و السرية في نموذج نظم قواعد البيانات النشطة
110	6-2-4 المشاكل المتعلقة بعملية التوقع للأداء

112	3-6 تطبيق الخوارزميات
113	4-6 المؤشرات
113	1-4-6 تحليل المؤشرات
الخاتمة	
121	4--1 النتائج
121	4-2 التوصيات
123	3-4 المراجع
130	4-4 الملحقات

قائمة الجداول

الصفحة	الجدول
18	جدول (1.2) مكونات الزنادات
19	جدول (2.2) مستويات مدى الزنادات
19	جدول (2.3) أنواع الزنادات
20	جدول (2.4) مراحل تنفيذ الزنادات
23	جدول (2.5) تطوير المشاريع البرمجية
48	جدول (2.6) أهم تصنيفات أدوات هندسة البرمجيات بمساعدة الحاسب
68	جدول (2.7) الفرق بين إدارة المشاريع الهندسية و إدارة مشاريع البرمجيات
92	جدول (3.1) أهم خصائص وطرق قياس المتطلبات غير الوظيفية
105	جدول (3.2) أنواع الصلاحيات
114	جدول (3.3) الموظفين بدوام كامل
115	جدول (3.4) الموظفين بدوام جزئي
115	جدول (3.5) الدخول الى النظام
118	جدول (3.6) نتائج استنباط خوارزمية النسخ الاحتياطي
119	جدول (3.7) نسبة الرضا بتقييم النسخ الاحتياطي

قائمة الأشكال

الصفحة	الشكل
56	الشكل (2.1) توضيح لمرحلة تحليل المشكلة.
86	الشكل (2.2) توضيحي لمرحلة التصميم
93	الشكل (2.3) توضيحي لأهم أنواع المتطلبات غير الوظيفية.
96	الشكل (2.4) يوضح مستخدمو وثيقة مواصفات المتطلبات
98	الشكل (2.5) توضيح لأهم مستويات المتطلبات وقارئها.
112	الشكل (3.1) الشاشة الرئيسية للنظام
112	الشكل (3.2) شاشة الدخول للنظام
112	الشكل (3.3) شاشة تغيير كلمة المرور
113	الشكل (3.4) شاشة النسخ الاحتياطي
116	الشكل (3.5) رسم بياني عدد الدخول للبرنامج
119	الشكل (3.6) رسم بياني لتحليل المسخ الاحتياطي
120	الشكل (3.7) رسم بياني يوضح نتائج استطلاعات الاراء للنسخ الاحتياطي
120	الشكل (3.8) يوضح نسبة الرضا بخوارزمية النسخ الاحتياطي

الفصل الأول

المقدمة

1-1 تمهيد :

لقد دخل الحاسب الآلي في مجالات الحياة وظهر أثره في حل العديد من المشاكل التي تعاني منها الشعوب والأفراد. ومن تلك المشاكل القدرة على تخزين كم هائل من البيانات وما يترتب عليه من أسلوب حفظ واسترجاع وفهرسة تلك البيانات والتي تتطلب جهد وتكلفة تحتاج إلي وقت طويل وعلى سبيل المثال أسلوب حفظ بيانات , ولقد كان الاعتماد حتى الآن على العنصر البشري فهو الذي يقع عليه العبء كله تقريباً وتتعدد مسؤوليته ابتداء من وضع استقبال البيانات ثم فهرستها لحفظها ناهيك عن العوامل الطبيعية التي تتعرض لها أوساط الحفظ الورقية وكما نرى نجد أن هذه العملية الروتينية تستغرق وقت طويل ويعتبر الوقت هو العامل الأساسي والحاسم لجميع الأعمال في هذا العصر.

من ذلك خسارة جزء من السوق بفقدان عدد من الزبائن. فالأداء معنى بتحديد مدى فعالية و إنتاجية النظام للمستخدم. يعرف الأداء كما يلي " إلى أي مدى يستطيع النظام موافقة الوظائف المحددة في متطلبات التصميم مثل السرعة و الدقة و استخدام الذاكرة" (IEEE 1991) و نخلص إلى إن للأداء اتجاهين اثنين هامين هما : التوسعية و تعنى إمكانية مواكبة التغيرات المستقبلية في النظام، و الاستجابة . استجابة النظام يعبر عنها بزمن الاستجابة أو بإنتاجية الاستجابة و يتم قياسهما بمدى و سرعة استجابة النظام للأحداث أو بعدد الأحداث التي يمكن للنظام معالجتها في إطار زمني محدد. هناك تقارير تؤكد إن 25% من مشاكل الأداء في أنظمة الحاسوب مرتبطة بعمارية و تصميم النظام يمكن اكتشافها أثناء بنا النظام (أثناء دورة تصميم النظام) برناردو و هيلستون و آخرون (Bernardo, Hillston et al. 2007) يعرف هندسة أداء البرمجيات بأنها "طريقة قياس كمي لأداء البرمجيات تتسم بالمنهجية و التنظيم مما يساعد على بناء نظام يلبي متطلبات الأداء الموضوعة مسبقا ". و يقوم أسلوب هندسة أداء البرمجيات على إجراءات وموجهات. الإجراءات يتم عن طريقها تجميع مواصفات الأداء ، و الموجهات

تساعد على تحديد نوع التقييم الواجب تطبيقه في كل مرحلة من مراحل بناء النظام. و بما أن هذه الطريقة تقوم على بناء نماذج الأداء من نماذج البرنامج ،فان النماذج الناتجة يمكن تقييمها و مقارنتها مع متطلبات الأداء المحددة مسبقا.

في منحى اخر و من اجل تجاوز صعوبات بناء الأنظمة المعقدة انتشر و بصورة واسعة أسلوب البرمجة أو التطوير المدفوع بالنموذج(OMG 2007). هذا الأسلوب لعب دورا مهما في استخدام المخرجات الكمية بصورة فاعلة في إعادة ضبط معمارية و تصميم النظام ليلبي متطلبات الأداء. و جدير بالذكر إن هذا الأسلوب في التطوير يتمحور حول النموذج (التطوير المتمحور حول النموذج) كعنصر أولى يستخدم في تطوير الأداء و بالتالي يتجنب إخفاقات الأسلوب الأخر المتمحور حول الشفرة (التطوير المتمحور حول الشفرة). عليه، يمكن الاستفادة من التطوير المتمحور حول النموذج في التوقع أو التحقق من استيفاء المتطلبات و ذلك ببناء نماذج للأداء من نماذج تطوير النظام و مقارنة النموذج الناتج بتلك المتطلبات.

مما سبق يتضح لنا جليا أهمية التوقع لأداء البرمجيات أثناء مراحل تطوير النظام لتجنب أوجه القصور التي تظهر عند التنفيذ و بالتالي ضرورة إتباع الأسلوب المنهجي و المنظم لضمان اعلي مستوى من الدقة ،كما تعرضنا إلى لمحة عامة حول التطوير المتمحور حول النموذج و أهميته في جعل مهمة التطوير أكثر سهولة و يسر و بالتالي إمكانية التوقع بالأداء بأقل جهد. بقية الورقة مرتبة كما يلي : الفقرة 2 تتناول موضوع هندسة أداء البرمجيات،مميزاتها،و آلية عملها.الفقرة 3 تتعرض لموضوع تطوير النظم المدفوع بالنماذج و كيفية توظيفه في تحليل و توقع أداء النظم. الفقرة الأخيرة تتناول توقع الأداء في نظم هندسة البرمجيات المؤسسة على المكونات ، المشاكل ، الطرق ، المقارنة و التحليل للمقارنة.

1-2 مشكلة الدراسة:

تتمثل مشكلة الدراسة في استخدام هندسة البرمجيات في تقييم الكفاءة و الأداء لدوال قواعد البيانات النشطة Active Database, في الوصول للبيانات في لغة أوراكل . في منحى آخر و من اجل تجاوز صعوبات بناء الأنظمة المعقدة انتشر و بصورة واسعة أسلوب البرمجة أو التطوير المدفوع بالنموذج . هذا الأسلوب لعب دورا مهما في استخدام المخرجات الكمية بصورة فاعلة في إعادة ضبط معمارية و تصميم النظام ليلبي متطلبات الأداء. و جدير بالذكر إن هذا الأسلوب في التطوير يتمحور حول النموذج (التطوير المتمحور حول النموذج) كعنصر أولى يستخدم في تطوير الأداء و بالتالي يتجنب إخفاقات الأسلوب الأخر المتمحور حول الشفرة (التطوير المتمحور حول الشفرة). عليه، يمكن الاستفادة من التطوير المتمحور حول النموذج في التوقع أو التحقق من استيفاء المتطلبات و ذلك ببناء نماذج للأداء من نماذج تطوير النظام و مقارنة النموذج الناتج بتلك المتطلبات.

يعتبر أداء البرمجيات احد أهم العوامل في تحديد مدى جودة المنتج البرمجي ، و ابعد من ذلك، يعتبر نجاح أو فشل المنظمة رهين بالوفاء بالأداء المطلوب للبرمجيات(Williams L.G. 2002). و الفشل في هذا يتمثل في الإهدار المالي بزيادة الصرف على المعدات الصلبة و على تطوير و إعادة تطوير النظام ، و تجاوز الوقت المخطط لتنفيذ النظام.

1-3 أهمية الدراسة :

تستمد الدراسة أهميتها من أهمية المتغيرات التي طرأت على عمل دوال قواعد البيانات النشطة الموجودة في لغة أوراكل و استخدام الطرق المختلفة في هندسة البرمجيات لقياس و تقييم الأداء و كفاءة المخرجات.

1-4 أهداف الدراسة :

تسعى الدراسة إلى تحقيق الأهداف الآتية :

1- المساهمة الجادة في كيفية استخدام هندسة البرمجيات في تقييم الأداء في العمل و قياس المخرجات في تطبيقات دوال قواعد البيانات النشطة في لغة أوراكل .

2- بيان مفهوم الفرق في عمل الدوال و قياس المخرجات باعتبارها المعطيات الأساسية لتقييم الأداء و أثرها في تقدير درجة الكفاءة.

1- 5 منهج الدراسة :

اعتمدت الدراسة المنهج الوصفي التحليلي و التطبيقي إذ مثل التحليل المنطقي أساس الجانب النظري من الدراسة في حين كانت خوارزميات تأمين حماية البيانات أساس الجانب التطبيقي منها .

1- 6 فرضيات الدراسة :

- 1- تأمين و حماية البيانات باسم المستخدم و كلمة المرور يعتبر جزء من جودة سرية النظام.
- 2- نظام قواعد البيانات النشطة يتبع السياسات العالمية (NIST , ISO) في تأمين و حماية البيانات.
- 3- النسخ الاحتياطي من أهم مؤشرات جودة تأمين و حماية البيانات و الوصول إليها عند الحاجة .

1- 7 حدود الدراسة:

أ- التطبيق على قواعد البيانات النشطة .

1- 8 المراجع و المصادر:

تتنوع المصادر و المراجع في هذا البحث ما بين الكتب العلمية و الدراسات السابقة و المنشورات

و الدوريات العلمية و المواقع العلمية على شبكة الانترنت و التلقي الشفوي من ذوي العلم

و المعرفة ، والتي تدور جميعها حول موضوع البحث.

الفصل الثاني

الإطار النظري

المبحث الأول : نظم إدارة قواعد البيانات :

بدأت معظم الشركات التجارية خاصة في البلدان المتقدمة تخزين وحفظ ملفاتها على الكمبيوتر منذ عام 1960 وقام علماء وخبراء الحاسبات في تطوير نظريات وأساليب لتطوير كيفية إعادة استخدام وزيادة كفاءة استخدام هذه الملفات المخزنة داخل الحاسب و التي تسمى ملفات آلية وبالتالي ظهرت واستحدثت مصطلحات كمبيوترية تعبر عن استخدامات هذه الملفات وأيضاً استخدمت طرق لمعالجة هذه الملفات آلية. و تخزن الملفات الكبيرة في قاعدة كبيرة وتحتوي على جميع البيانات المسجلة و التي يمكن استخدامها في زمن لاحق هذه القاعدة تسمى قاعدة بيانات Database . ولأن قواعد البيانات مهمة ومؤثرة جداً في جميع المجالات و الأنشطة الرئيسية . لذلك يلزم وجود نظم معينة لتنظيم وإدارة البيانات المخزنة . وهو ما يطلق عليه نظم إدارة قواعد البيانات Database Management Systems.

وتعرف نظم إدارة قواعد البيانات : بأنها هي البرامج التي تساعد على إنشاء قواعد البيانات و التعامل معها وتشغيل البيانات المخزنة بها . فمثلاً بعد إضافة عملاء جدد لدليل التليفون فإنك تحتاج إلى ترتيب الأسماء من جديد أبجدياً أو ترتيب عناوينهم . بمعنى آخر تتيح للمستخدم إضافة بيانات جديدة وتحديث البيانات وطباعة التقارير على الشكل التي تريده مثل القوائم و الجداول و النماذج و التقارير .

وقد تكون قاعدة البيانات كبيرة جداً وتحتوي على آلاف من البلايين من الكلمات وهي أكبر من الذاكرة الموجودة ونتيجة لذلك كانت لـ DBMS أن تعالج وتدير البيانات في الذاكرة الثانوية ومن البرامج التي صممت لهذه الشأن كثيرة منها التي تعمل على الحاسبات الكبيرة Mainframes أو التي تعمل على الحاسبات الشخصية. PCs ومثل

هذه البرامج DBASE IV : و Clipper و Paradox و Oracle و FoxBASE و FoxPro و SQL

و DMS و IDMS و برنامجنا الجميل MS Access و الكثير من هذه البرامج بمختلف الإصدارات .

ويتكون نظام إدارة البيانات من مجموعة من الملفات بالإضافة إلى البرنامج أو مجموعة البرامج التي تتضافر لحل مشكلة أو لتحويل نظام يدوي إلى نظام يعمل بالحاسب مثل تحويل نظام حسابات العملاء أو حسابات المخازن من نظام الدفاتر اليدوية إلى نظام وملفات تستخدم بواسطة الحاسب فإن هذه البرامج مع ملفات النظام يطلق عليه نظام إدارة قاعدة البيانات أو قد يشتمل على مجموعه من البرامج بالإضافة إلى ملفات النظام وفي هذه الحالة فإن البرامج مجتمعه يطلق عليها نظام إدارة قاعدة البيانات⁽¹⁾.

وبعيدا عن التعقيدات ودون الدخول في تفاصيل وفرت نظم إدارة قواعد البيانات المرنة المطلوبة عن نظم إدارة الملفات التي كانت تستخدم من قبل.

ولكن كل ما نود أن نعرفه أن الملفات المسلسلة و الملفات الثنائية و العشوائية لها الدور الأكبر في الانتقال من نظام الملفات إلى نظام قواعد البيانات .

(1) Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.

Royce, W. (2011), "Managing the Development of Large Software Systems : Concept and Techniques", Proc. WESCON, August 2011.

2-1 أنواع نظم إدارة قواعد البيانات :

ولهذا فإن نماذج البيانات هي تمثيل بيانات العالم الحقيقي بصورة يسهل استخدامها بواسطة الحاسب وهناك أنواع من نماذج البيانات تتوقف على نظام إدارة قواعد البيانات المستخدم وكذلك على طبيعة البيانات وتبعاً لأنواع نماذج البيانات فهناك ثلاثة أنواع شائعة من نظم إدارة قواعد البيانات وهي.

1- نظم إدارة قواعد البيانات الهرمية .

2- نظم إدارة قواعد البيانات الشبكية .

3- نظم إدارة قواعد البيانات العلائقية .

قواعد البيانات الهرمية أو النظم الهرمية Hierarchical DBMS تقوم بتنظيم البيانات على شكل هرمي أو علي شكل شجرة مقلوبة أي جذرها في القمة وتخرج منها الفروع . شأن هذه التركيبة شأن شجرة الأسرة فلها جد واحد و الجد له عدة أبناء و الأبناء هم آباء الأحفاد ويستحيل وجود حفيد له أكثر من أب . وهذا شكل توضيحي ليوضح النظم الهرمية وتفرعاتها .

والملفات الهرمية هي ملفات لها نفس البناء الشجري ولها نفس العلاقات بين السجلات مثلاً لبعض أنواع السجلات التي يمكن أن تتواجد في تكوين هرمي فهناك سجلات مبيعات متعددة لكل بائع حيث يوجد سجل إحصائيات واحد لكل عملية جارية كما يوجد أيضاً سجلات عديدة للعملاء لكل بائع حيث أن كل بائع له عملاء محددين ويمكن أن يكون لكل عميل عدة سجلات حسابات مدينين سجل واحد لكل عملية شراء لم يتم تسديد ثمنها.

ومن المهم أن نفهم أنه ليس من الضروري أن تتصل كل الملفات الموجودة في قاعدة البيانات مع بعضها . وكل ما هو مطلوب أن تتصل الملفات التي تستخدم كمجموعة مع بعضها في التطبيقات .

وسجلات المبيعات السابقة لها مثل هذه العلاقة المنطقية تسمى فئة . و الفئة Set عبارة عن مجموعة من السجلات متصلة مع بعضها منطقياً.

وعلى هذا تصبح قاعدة البيانات الهرمية عبارة عن تجميع لملفات وفئات ملفات متصلة مع بعضها منطقيا.

ويستخدم نظام إدارة المعلومات IMS الذي أعدته شركة IBM التكوين الهرمي وهو من اكبر نظم إدارة قواعد البيانات dbms الموجودة حاليا واعقدها . ولهذا السبب فأنه يتطلب مستوى رفيع من الخبرة لإمكانية بنائه وعلى أي حال فهو قوي واثبت كفاءة كبيرة في معاملة قواعد بيانات كبيرة جدا كما انه يقدم إجراءات استرجاع و أمن جيدة هذا بالإضافة إلى إمكانية استخدامه في نظام الاتصال النشط من خلال شبكة الاتصالات.

2-3-1 نظم إدارة قواعد البيانات الشبكية:

رغم أن كلمة الشبكة استخدمت كثيرا في شبكات الحاسب ومعالجة البيانات فقد وجد من الأفضل استخدام مسمى قواعد البيانات الضفيرة Plex رغم أن مسمى قواعد البيانات الشبكية لازال شائع الاستخدام.

ويتغلب هيكل بيانات التركيب الشبكي على معوقات التكوين الهرمي الذي لا يسمح للابن أن يكون له أكثر من أب واحد ويظهر ذلك في الشكل التوضيحي للتكوين الشبكي حيث نلاحظ أن للسجل رقم (4) عائلان هما السجل رقم (2) و السجل رقم 3 .

ومثل هذا النوع من قواعد البيانات حل كثيرا من مشاكل العلاقات فإذا فرضنا أن هناك أكثر من مورد يورد قطع غيار فإن كل مورد قادر على توريد أكثر من نوعية قطعة غيار وبالتالي فإن كل قطعة غيار يوردها أكثر من مورد مما يحتم لفهم المثال عرض العلاقة بين قطعة الغيار و الموردون على النحو الموضح في الشكل التالي .

ومن هذا الشكل يتضح لنا بما لا يقبل الشك أن تبسيط العلاقة الشبكية إلى علاقة هرمية أوجد تعقيدات أكثر حيث حولها إلى نوعين من شجرة العلاقات وفي هذا جهد إضافي في التنفيذ.

إن ما عرضنا حول العلاقات الشجرية (الهرمية) وقواعد البيانات الشبكية يؤكد أن كلاهما يمكن تحقيقه وان كانت بعض حزم إدارة قواعد البيانات يمكنها التعامل فقط من الشكل الشجري كما أن البعض الآخر يمكنه التعامل مع

النوع الشبكي كما أن هناك تنوع من برامج إدارة قواعد البيانات فبعض برامج إدارة قواعد البيانات الهرمية لا تتعامل مع العلاقات البسيطة و البعض يمكنه التعامل مع العلاقات المعقدة.

و أوجه التشابه بين نظم قواعد البيانات الشبكية و نظم قواعد البيانات الهرمية إنها تتطلب إلى ذاكرات ذات أحجام كبيرة وعادة تحتاج إلى لغات راقية لبرمجتها وهي صعبة التعلم ولها مزايا كثيرة فهي بالطبع أكثر كفاءة من قواعد البيانات العلائقية ويتعاملان مع كم كبير جدا من البيانات و المعلومات بالإضافة إلى إنها توفر بناء على طريقة تنظيم البيانات التي تتبعها مساحات كبيرة من وسائط تخزين البيانات.

2-3-2 نظم إدارة قواعد البيانات العلائقية:

أثبتت الأيام صحة القول الشائع أن الأبسط هو الأجل والأكفأ . فكلما كان سبك بسيط وكلما عشت في بساطة و بعدت عنك المشاكل وكلما كانت الآلة بسيطة سهلت إدارتها وصيانتها . وهذا ما أكدته التعامل مع قواعد البيانات الهرمية و الشبكية التي تعقدت ملفاتها وأساليب إدارتها لدرجة كادت تؤدي بها كلما أضيفت تطبيقات جديدة أو متطلبات جديدة تحتاج مؤشرات جديدة مما ضخم منها وعقدها.

وهذه المشاكل كانت المنطلق للبحث عن حلول تحقق جملة أهداف منها:

1- يمكن فهم قاعدة البيانات لمن لم يدرسوا علوم الحاسب .

2- يمكن تعديل وإضافة وحذف بيانات دون تغيير المخطط المنطقي للقاعدة .

3- 3 تتيح للمستخدم اعلي درجة من المرونة في التعامل مع البيانات .

في عام 1970 أستحدث E.E.Codd أسلوبا لتنظيم وفرز بيانات قواعد البيانات . وهي قواعد البيانات العلائقية . وقد وجد العالم الأمريكي E.E.Codd أن هذا لا يتحقق إلا برص البيانات على هيئة جداول لان الإنسان تعود على الجداول منذ طفولته بداية من جدول الحصص إلى جدول الضرب إلى كشف الأسماء و الدرجات. و هذه النظم تتعامل مع أكثر من ملف في نفس الوقت وتعامل البيانات داخل الملف كما لو كانت جدولا مكونا من صفوف و

أعمدة ويسمى علاقة Relation وتمثل أعمدة الجدول حقول قاعدة البيانات Fields وتسمى أيضا Attributes بينما تمثل صفوفها سجلات قاعدة البيانات وتسمى Tuples و النظام العلائقي Relation يقوم بربط البيانات بين العلاقات بناء على حقل مشترك بينهما .

والنظم العلائقية قامت أساسا علي النظريات العلائقية في الرياضيات وقد بدأ تطبيقها على الحاسبات الكبيرة أولا مثل

DBaseII . ORACLE . SQL ثم ظهرت عدة نظم علائقية على الحاسبات الشخصية PCs مثل برامج DBaseII

. FoxPro . FoxBase . DBaseIV ويمكن القول عن هذا النوع من قواعد البيانات مايلي :

1- تنظيم البيانات في قواعد البيانات العلائقية في جداول ذات بعدين ويمكن اعتبار كل جدول ملف ويستخدم

مصطلح ملف مسطح Flat File لان محتويات الملف مرتبة على محورين س , ص فقط.

2- نشأت مجموعة جديدة من المصطلحات تستخدم في وصف قواعد البيانات العلائقية هذه المصطلحات التي

تستخدم في وصف قواعد البيانات الهرمية أو الشبكية ففي النموذج العلائقي يستخدم مصطلح نموذج بيانات

علائقي جزئي أو رؤية لبيانات علائقية Relational Data Submodel or View بدلا من المخطط

الجزئي Subschema ومصطلح رؤية View مناسب فهو لجزء المستفيد من قاعدة البيانات.

3- كما استخدمت بالإضافة إلى ذلك أسماء لوصف مكونات الملفات المسطحة ويوضح الجدول التالي عينة

لملف ويشار إلى أعمدة الملف بأنها مسطح رأسي (نطاق) والى الصفوف بأنها مسطح أفقي و الجدول

عبارة عن تجميع من المسطحات الأفقية خاصة بموضوع معين و الجدول خاص بالبائع ويمكن استخدامه

في توفير أسمائهم ومبيعاتهم منذ بداية العام.

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

الخاتمة

النتائج و التوصيات

4-1 النتائج :

- 1- طريقة القياس تحقق نسبة عالية لمعيار الدقة. في التأمين و سرية و حماية البيانات .
- 2- طريقة القياس تحقق ثبات نسبي لكل من العملاء و الموظفين بدوام كامل و دوام جزئي.
- 3- تم تطبيق سياسة تغيير المستخدم و كلمة المرور المذكورة في المؤشر الثاني موافقة بما جاء في السياسات العالمية لتأمين و حماية البيانات.
- 4- نتيجة تقييم القياس لجودة خوارزمية النسخ الاحتياطي مرضية للغاية و كانت النتيجة %92.6 .
- 5- تطبيق و تنفيذ خوارزمية النسخ الاحتياطي أعطى نتيجة ايجابية في حماية البيانات و الحفاظ عليها.
- 6- طلب إدخال المستخدم (اسم المستخدم و كلمة المرور) أعطى موثقيه لدخول المستخدم الحقيقي للنظام و ذلك يجعل قواعد البيانات أكثر أمانا.

4.2- التوصيات:

لتطوير نظام التأمين و الحماية لقواعد البيانات النشطة في المستقبل:

1. عمل خوارزمية للتحقق من الهوية كإضافة داعمة لخوارزمية الدخول إلى النظام ليزيد من جودة الحماية و الأمان للبيانات .
2. رسم السياسات العامة لرفع جودة تأمين و حماية البيانات في النظام و تطويره.
3. تطوير دوال تأمين و حماية البيانات لتكون متناسبة مع تطور برامج نظم إدارة قواعد البيانات المختلفة .

4. أيضا نوصي باستخدام لغة أوراكل في برمجة نظم قواعد البيانات و ذلك لما تحتويه من خصائص و دوال

قوية في تأمين و حماية و سرية البيانات.

5. زيادة حجم المساحة التخزينية للنسخ الاحتياطي و ذلك لزيادة حجم البيانات المراد حفظها.

6. استخدام الرسم البياني في تطبيق تقارير نظام التأمين و الحماية للبيانات و ذلك لمعرفة درجة و نسبة

الحماية من خلال الرسم البياني.

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

3-4 المراجع

(أ) روجر بريسمان ، "هندسة البرمجيات" ، الدار العربية للعلوم ، مركز التعريب والبرمجة ، الطبعة الأولى ، 1425هـ - 2004م.

(ب) مهندس إياد هلالى ، "هندسة البرمجيات" ، المركز الألماني السوري لأعمال الانترنت (gsibc.net)

(ج) أسماء المنقوش ، "دورة هندسة البرمجيات" ، جامعة القاهرة ، مصر ، 2009.

(د) مهندس عبد الحميد بسيوني ، "أساسيات هندسة البرمجيات" ، دار الكتب العلمية للنشر والتوزيع ، القاهرة ، 2005 م.

(هـ) أحمد شعبان دسوقي وآخرون ، "أساسيات الحاسب الآلي وتطبيقاته في التعليم" ، مكتبة الرشد ، الرياض ، الطبعة الأولى ، 1427هـ - 2006م.

(و) علي كمال شاكر ، نظم إدارة قواعد البيانات ، القاهرة ، الدار المصرية ، 2005

(ك) جمال بطيخ ، قواعد البيانات ، سورية ، شعاع ، 2003

(ل) مراد شلبياية ، مفاهيم أساسية خ قواعد البيانات ، عمان ، د.المنيرة ، 2002

- [1] Abiteboul, S, Hull, R, and Vianu, V., Foundations of Database, Addison Wesley 2005.
- [2] Adams, C, Simple and Effective Key Scheduling for Symmetric Cipher, SAC, 2004.
- [3] After resection in patients with hepatic cellular carcinoma treated in the carcinoma. Cancer Chemo her Pharmacia, 2004.
- [4] Alexander, S, Password Protection for Modern Applications, login, 2004.
- [5] Alvare, A, How Crakers Crack Passwords to Avoid, Proceedings, Security Workshop II, August 2000.
- [6] Anderson, J, Computer Security Threat Monitoring and Surveillance, P. Anderson Co, 2002.
- [7] Andrews, M., and Whittaker, J, Computer Security IEEE security and Privacy, 2004.
- Ante, S, and Grow, B, Meet the Hackers, 2006.
- [8] B Bhargava, B., and Helal, Efficient Reliability Mechanisms in Distributed Systems, CIKM 2003.
- [9] Barker, W, Introduction to the Analysis of the Data Encryption Standard (DES), Aegean Park Press, 2001.
- [10] Barkley, J, Comparing Simple Role-Based Access Control Models and Access Control Lists, Proceedings of the second ACM Workshop on Role-Based Access Control, 2009.
- [11] Beck, H., Gala, S., and Navathe, S, Classification as a Query Processing Technique, 2009.
- [12] Bellare, M, and Rogaway, P, Optimal Asymmetric Encryption, 2004.
- [13] Bellare, M, Canetti, R Kilian, and Rogaway, The Security of the Cipher Block Chaining Message Authentication Code, 2002.
- [14] Bernstein, P., and Goodman, N, An algorithm for Concurrency Control and Recovery in Replicated Distributed Database, TODS, 2004
- [15] Bernstein, P., and Goodman, N, Concurrency Control in Distributed Database system, VLDB, 2000.

- [16] Bernstein, P., Hadzilacos, V., and Goodman, N., Concurrency Control and Recovery in Database Systems, Addison Wesley, 2008.
- [17] Bertino, E, Data Hiding and Security on Object-Oriented Databases, ICDE, 2002.
- [18] Bertino, E., Negri, M., Pelagatti, G., and Sbattella, L., Object-Oriented Query Language, The Notion and the Issues, TKDE, 1999.
- [19] Bhargava, B., and ed, Concurrency and Reliability in Distributed Systems, Van Nostrand-Reinhold, 2007.
- [20] Bhatti, R, Bertino, E, and Ghafoor, An Integrated Approach to Federated Identity and Privilege Management in Open Systems, ACM, 2007.
- [21] Biller, A, The Performance of three Database Strong Structures for Managing Large Objects, SIGMOD, 2002.
- [22] Bishop, M, Computer Security, Boston, Addison-Wesley, 2005.
- [23] Blaha, M., Premerlani, W, Object-Oriented Modeling and Design for Database Application, Prentic-Hall, 2003.
- [24] Bloom, B, Trade-offs in Hash Coding with Allowable Errors, ACM, 2001.
- [25] Boehm, B. (1988), "A Spiral Model of Software Development and Enhancement", IEEE Computer 21, 5, 61-72.
- [26] Boneh, D, Twenty years of Attacks on the RSA, Spring, 2002.
- [27] Bosch FX, Ribes J, Diaz M, Celeries R. Primary liver cancer: worldwide
- [28] Bray, O., Computer Integrated Manufacturing -The Data Management Strategy, Digital Press, 2008.
- [29] Campbell, K., and Wiener. M, Proof that DES Is Not a Group, Spring, 2006.
- [30] Cheng JC, Chuang VP, Cheng SH, Huang AT, Lin YM, Cheng TI,
- [31] Chokhani, S, Trused Products Evaluation, ACM, 2006.
- [32] Collett, S, Encrypting Data at Rest, Computerworld, 2006.
- [33] Colombo M. Hepatocellular carcinoma. J Hepatol 2002.

- [34] Cormen, T, Leiserson, C, Rivest, R, and Stein, C, Introduction to Algorithms, Cambridge, MA, MIT Press, 2001.
- [35] CRC Ressler, the biomedical engineering hard book, second edition volume 1,2000.
- [36] David Jraham,Diagnosis cancer,us diagnosis center,2004..
- [37] Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.
- [38] Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.
- [39] Douglas Bell , "Software Engineering A Programming Approach", 3rd Edition, Addison Wesley.
- [40] Duchange.Zheug,Wany lua-min, luchong "A new medical diagnosis system" pacific Asian conference on expert system,hungsham,chino,2004.
- [41] Esnaola NF, Mirza N, Lauwers GY, Ikai I, Regimbeau JM, Belghiti J, et al. Local radiotherapy with or without transcatheter arterial,
- [42] Ian Somerville , "Software Engineering", Addison Wesley, 2001.
- [43] incidence and trends. Gastroenterology 2004.
- [44] Jepraeel kopel,Gary rily,Expert system principles and programming,2007.
- [45] Kiyosawa K, Umemura T, Ichijo T, Matsumoto A, Makuuchi M, Sano K. The surgical approach to HCC: our progress and Matsuura M, Nakajima N, Arai K, Ito K. The usefulness of radiation.
- [46] Ronald J. Leach,, "Introduction to Software Engineering", CRC Press, 1999.
- [47] Royce, W. (2011), "Managing the Development of Large Software Systems Concept and Techniques", Proc. WESCON, August 2011.
- [48] Seong J, Keum KC, Han KH, Lee DY, Lee JT, Chon CY, et al.

[49] Shari Pfleeger, "Software Engineering - Theory and Practice", 2nd Edition.

[50] therapy for hepatocellular carcinoma. Hepatogastroenterology,2008.

[51] United States, France, and Japan. Ann Surg 2003. Uno T, Itami J, Shiina T, Toita T, Mikuriya S, Hatano K, et al.

A- Oracle Database Vault: Segregation of Duties and Privileged User Controls[Oracle Corp. ,2009]

B- How the database security controls adapted to threats over the last 30 years[Paul Lesov,2006]

D-Web and Database Security[Zhejiang Normal University,2015]

E- Database Security: What Students Need to Know [JITE,2010]

F- HYBRID ENCRYPTION FOR CLOUD DATABASE SECURITY [IJESAT, 2015]

G-DATABASE SECURITY – ATTACKS AND CONTROL METHODS[Emil- BURTESCU,2009]

6-4 محكمي الإستبانة:

- 1- د.خالد عدنان - الكلية الأردنية السودانية .
- 2- د.حذيفة احمد - جامعة السودان .
- 3- د.سلام فقيري - جامعة الزعيم الازهرى .
- 4- د.أكرام محمد - كلية الخرطوم للعلوم والتقانة .
- 5- أ.أمنه الشيخ - جامعة المستقبل .
- 6- أ.مجاهد احمد - جامعة الخرطوم كلية العلوم الرياضية وعلوم الحاسوب .
- 7- أ.أيمن سعيد - كلية الخرطوم للعلوم والتقانة .

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:

لا وافق بشدة	لا وافق	صحيح لحد ما	وافق	وافق بشدة	
					<ul style="list-style-type: none"> ▪ (أوافق بشدة) تعني أن العبارة صحيحة دائماً أو في كل الأحيان تقريباً، و أن المطلوب تمت تأديته على أكمل وجه. ▪ (أوافق) تعني أن العبارة صحيحة غالباً أو في أغلب الأحيان، و أن المطلوب تمت تأديته بشكل جيد تقريباً. ▪ (صحيح لحد ما) تعني أن المطلوب تمت تأديته بشكل متوسط. ▪ (لا أوافق) تعني أن المطلوب تمت تأديته بشكل ضعيف أو لم يؤد في معظم الأحيان. ▪ (لا أوافق بشدة) تعني أن المطلوب تمت تأديته بشكل سيء جداً، أو لم يؤد أصلاً، أو نادراً ما تمت تأديته.
					المساعدة والدعم اللذان قدما لتعليمي
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	1 أتيح لي التعرف على النظام و العمل عليه
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	2 كانت جميع الدوال الخاصة بالسرية و تأمين البيانات واضحة و سهلة
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	3 تعرفت من خلال العمل على النظام على أن يمكن أن تكون هناك نسخ احتياطية للبيانات
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	4 سهولة عمل النسخ الاحتياطي من خلال النظام
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	5 الذي صمم هذا النظام له خبرة كافية في المجال
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	6 تعرفت على أهمية النسخ الاحتياطية للبيانات
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	7 سهولة الوصول الى النسخ الاحتياطية من النظام

المصادر الخاصة بدعم تعليمي	
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	8 كانت خطوات عمل النسخ الاحتياطي من النظام سهلة و واضحة
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	9 خاصية النسخ الاحتياطي أعطتني ثقة في النظام
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	10 اتسمت تجهيزات النظام بالجودة
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	11 كانت تجهيزات النظام بما فيها النسخ الاحتياطي كافية لتأمين و حماية البيانات
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	12 توفرت التجهيزات المناسبة للأنشطة اللامنهجية (بما في ذلك التجهيزات الخاصة بالرياضة والترفيه).
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	13 هناك مرافق مناسبة لأداء الشعائر الدينية.
	14 دوال التأمين و النسخ الاحتياطي واضحة للمتدربين
تقويم التعليم الذي حصلت عليه	
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	15 ما تعلمته في هذا النظام هو كيف أداوم على عمل النسخ الاحتياطي .
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	16 لقد ساعدني النظام في تطوير الاهتمام الكافي لدي للسعي في الاستمرار في تحديث معلوماتي و الشعور بالأمان.
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	17 لقد طور النظام قدرتي على استقصاء وحل المشكلات عند ضياع أي جزء من البيانات بالرجوع الى النسخة الاحتياطية
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	18 لقد طور البرنامج قدرتي على العمل بفاعلية و موثوقيه.
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	19 لقد حسن البرنامج مهاراتي في الاتصال .
	20 لقد ساعدني البرنامج في تطوير مهاراتي الأساسية في استخدام التقنية لدراسة القضايا والتعبير عن النتائج
	21 لقد طورت المعارف والمهارات اللازمة لمهنتي التي اخترتها.
التقويم العام	
<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>	22 أشعر بالرضا بشكل عام عن مستوى جودة النظام في عمل النسخ الاحتياطي
أسئلة مفتوحة	
	23 هل أعجبت بعملية التأمين و الحماية للبيانات من خلال النسخ الاحتياطي؟

<p>-----</p> <p>-----</p> <p>-----</p>	
<p>ما أكثر شيء لم يعجبك فيما يخص تأمين البيانات في النظام؟</p> <p>-----</p> <p>-----</p> <p>-----</p>	24
<p>ما الاقتراحات التي لديك لتحسين النظام من حيث الأمان و السرية للبيانات؟</p> <p>-----</p> <p>-----</p> <p>-----</p>	25

Login trigger code : User name and password

In the first form of the app, use the ON-LOGON trigger to call another form to get name and password:

```
BEGIN
  Call_form ('LOGON_APP', NO_HIDE);
  IF :GLOBAL.logged_on != 'TRUE' THEN
    RAISE Form_Trigger_Failure;
  END IF;
END;
```

That form contains a LOGON_BLOCK WITH:

- USERNAME
- PASSWORD
- CONNECT (connect string: the name of the database)
- OK button
- CANCEL button

The OK button contains:

```
IF :logon_block.username IS NULL THEN
  BELL;
  RETURN FALSE;
END IF;
```

```
Set_Application_Property(Cursor_Style, 'BUSY');
```

```
LOGON (:logon_block.username,
```

```
:logon_block.password||'@'||:logon_block.connect,  
FALSE);
```

```
IF FORM_SUCCESS THEN  
  Set_Application_Property(Cursor_Style, 'DEFAULT');  
  RETURN TRUE;  
ELSE  
  Set_Application_Property(Cursor_Style, 'DEFAULT');  
  res := Show_Alert ('probleme');  
  RETURN FALSE;  
END IF;
```

Change Login trigger code

password user_name or passw user_name

```
SQL> passw scott
```

Changing password for scott

New password:

Retype new password:

Password changed

Backup code

```
PROCEDURE DO_BACKUP IS  
;(GET_APPLICATION_PROPERTY(USERNAME =: (USERID VARCHAR2(50  
;(GET_APPLICATION_PROPERTY(PASSWORD =: (VARCHAR2(50 PSW  
;(GET_APPLICATION_PROPERTY(CONNECT_STRING =: (CSTRING VARCHAR2(50  
;(fPathName varchar2(200  
;(BDIR VARCHAR2(1000  
;(DMPFILENAME VARCHAR2(100  
  BEGIN  
;('...MESSAGE( 'Doing Backup  
;SYNCHRONIZE  
if bdir is null then  
  into bdir select BKPDIR  
  URparam from  
;'Pcode = 'BACKUP where  
;end if  
if Substr(bdir,length(bdir),1) != '\' then  
;'\' || (bdir := rtrim(bdir  
  ;end if  
;((bdir := ltrim(rtrim(bdir  
;(to_char(sysdate, 'ddMONyy')||'_'||dbms_random.string('x', 5 =: dmpfilename  
;(USERID||'/'||PSW||'@'||cstRING||' '||bdir||dmpfilename)|| host('backup.bat  
exception  
when others then  
;('check parameters for the backup option message('Please  
;END
```

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات	العنوان:
عبدالمجيد، مصطفى أمين	المؤلف الرئيسي:
كرار، عبدالرحمن الشريف محمد(مشرف)	مؤلفين آخرين:
2016	التاريخ الميلادي:
ام درمان	موقع:
1 - 133	الصفحات:
912244	رقم MD:
رسائل جامعية	نوع المحتوى:
Arabic	اللغة:
رسالة دكتوراه	الدرجة العلمية:
جامعة أم درمان الاسلامية	الجامعة:
معهد بحوث ودراسات العالم الإسلامي	الكلية:
السودان	الدولة:
Dissertations	قواعد المعلومات:
نظم إدارة قواعد البيانات، هندس البرمجيات، جودة البرمجيات، مهنة البرمجيات، قواعد البيانات النشطة	مواضيع:
https://search.mandumah.com/Record/912244	رابط:



جمهورية السودان

جامعة أم درمان الإسلامية

معهد بحوث ودراسات العالم الإسلامي

قسم الدراسات التطبيقية

قياس أداء دوال تأمين قواعد البيانات النشطة باستخدام هندسة البرمجيات

بحث مقدم لنيل درجة الدكتوراه في علوم الحاسوب

إشراف :

د. عبد الرحمن الشريف كرار

إعداد :

مصطفى أمين عبد المجيد

2016-1437م