للعلوم

ا

العليا

ب وتقانة

المعلومات

مدى ملاءمة استخدام نموذج COCOMO2 لمشاريع البرمجة في الأسواق السودانية

(بحث تكميلي لنيل درجة الماجستير في علوم الحاسوب)

إعداد الطالب: مجتبى أحمد إبراهيم آدم

إشراف الدكتوره: نسرين بشير عثمان بشير

2015م

الاستهلال

ﭽ ﭑ ﭒ ﭓ ﭔ ﭕ ﭖ ﭗ ﭘ ﭙ ﭚ ﭛ ﭜ ﭝ ﭞ ﭟ ﭠ ﭡ ﭢ ﭣ ﭤ ﭥ ﭦ ﭧ ﭨ ﭩ ﭪ ﭫ ﭬ ﭭ ﭮ ﭯ ﭰ ﭱ ﭲ ﭳ ﭼ

سورة الإسراء، الآية: (85).

إهـــداء

إلى الوالدة العزيزة متعها الله بالصحة والعافية..
وإلى روحي والدي الحبيب تغمده الله بواسع رحمته..
وإلى أسرتي الصغيرة..
زوجتي ...
وابني.. محمد..
وإلى أختي العزيزتين..
وإلى كل حادب على العلم..

شكر وتقدير

الشكر من قبل ومن بعد لله عز وجلَّ على نعمة الصحة والعافية وأن منَّ على بإكمال هذا البحث بصورته هذه..

ثم الشكر لجامعة السودان للعلوم والتكنولوجيا ممثلة في كلية الدراسات العليا وأسرة كلية علوم الحاسوب وتقانة المعلومات.

والشكر للدكتورة/ نسرين بشير التي قامت بالإشراف على هذا البحث ولما قدمته من نصح ومشورة وسعة صدر.. والدكتور/ محمد الحافظ الذي قدم لي النصح بأفكاره ورأيه السديد في موضوع هذا البحث..

ولأسرتي الكريمة التي صبرت عليَّ وكانت خير معين في فترة دراستي وبحثي..

ولزملائي في دراسة الماجستير وأخص منهم مصعب وعبد الله وأحمد..

وللزملاء في العمل سناجق ومجتبى ومحمد النوراني.

مستخلص

نموذج تكلفة البنـاء (COCOMO) هـو نمـوذج تقـدير تكلفـة البرمجيـات. ونحتاج للتحقق من مدى ملاءمته مـع الأسـواق الصـغيرة. هـذه الدراسـة الـتي قدمت في هذه الأطروحة تتناول بعض القضايا المتعلقة باسـتخدام COCOMO 2 نموذج لتقدير التكلفة في سوق البرمجيـات صـغير مثـل السـوق السـوداني. وكان الهدف من هذه الدراسة هو تقييم فعاليـة النمـوذج وأهميتـه فـي السـوق السـوداني. اسـتخدمت الدراسـة المنهـج الوصـفي التحليلـي حيـث اسـتخدمت الاستبيانات لجمع البيانات من (31) مشروعا تم اختيارها عشوائيا من إجمـالي (100) مشروعا. تم تحليل البيانات وأظهرت نتائج التحليل أن هناك فرق كـبير بين تقدير التكـاليف الـتي ينتجهـا النمـوذج والتكلفـة الفعليـة لجميـع المشـاريع البرمجية في هذه المنظمات. وهذا يدلل بأن اسـتخدام هـذا النمـوذج للأسـواق الصغيرة لابد من تعديله حتى يوائم هذا النوع من الاسواق . وقـد أجريـت هـذه الدراسة في ولاية الخرطوم فقط لذلك فمن المستحسن لتمديد الدراسة إلـى منطقة جغرافية أخرى في البلاد ومنـاطق أخـرى مـن سـوق البرمجيـات علـى نطاق صغير.

Abstract

The Constructive Cost Model (COCOMO) is a software cost estimation model and is one of the mostly used commercially. However, the effectiveness of the model for estimating the cost of software production in small scale companies and market need to be investigated. This study presented in this thesis addresses some of the issues related to the use of the COCOMO2 model for cost estimation in in small software market such as the Sudanese market. The aim of the study was to evaluate the effectiveness of the model and its relevance to the Sudanese market. The study used descriptive analytical method where questionnaires were used to collect data from (31) projects randomly selected from a total of (100) projects. The data was analyzed and the results of the analysis showed that there is a great difference between the cost estimation produced by the model and the actual cost of all software projects in these organizations. This suggests that for the model to be used for cost estimation in small markets it needs to be adjusted. The study was carried in Khartoum state only so it is recommended to extend the study to other geographical area in the country and other regions of small scale software market.

فهرس المحتويات

الفصل الأول
الإطار العام للبحث

الفصل الأول
الإطار العام للبحث

1-1 المقدمة:

يعتبر التخطيط لعمليات تطوير البرمجيات أحد أهم متطلبات نجاح إنتاج المنتج البرمجي وتسليمه للزبون ضمن حدود الوقت، والتكلفة المتفق عليها ونظراً لاختلاف المنتج البرمجي عن بقية المنتجات بسبب كونه غير محسوس، فإن ذلك يؤدي إلى وجود صعوبة في تحديد الجهد، والتكلفة المطلوبة لتطوير هذا المنتج.

وفي عصرنا الحالي أصبحت تكلفة المشروع البرمجي هي العنصر الأساسي في تكلفة معظم الأنظمة المعتمدة على الحاسب الآلي، وتُعرف بأنها عملية التنبؤ بالمجهودات، والموارد المطلوبة لتطوير المشروع البرمجي، والتحقق، والمصادقة على صحته، وإدارة النشطات المختلفة الخاصة بذلك؛ وكذلك تُعد من أهم التحديات، والنشاطات المهمة في دورة حياة تطويره؛ حيث إنه بدون تقدير هذه التكلفة بصورة موثوق بها لا يمكن تخطيط المشروع البرمجي، والتحكم في تنفيذه بطريقة فاعلة، وصحيحة.

يجب تنفيذ حسابات كلفة البرمجيات بشكل موضوعي والهدف هو التنبؤ الدقيق بكلفة تطوير البرمجية. قد يكون الهدف من تقدير كلفة المشروع هو تحديد السعر الذي يجب أن يدفعه الزبون مقابل هذا النظام (عندئذٍ يكون السعر جزءاً من العقد المقدم للزبون). إنَّ السعر (Price) بشكله التقليدي هو عبارة عن الكلفة (Cost) زائد الربح (Profit) المنشود.

ويجب أن نراعي عند تسعير البرمجية العديد من الاعتبارات المؤسساتية والاقتصادية والسياسية والتجارية، ولهذا فإن العلاقة بين السعر الذي سيدفعه الزبون وكلفة التطوير ليست بسيطة ومباشرة، لذا تشارك الإدارة العليا للشركة (الإدارة المسؤولة عن وضع الأهداف الاستراتيجية) وكذلك مديرو المشاريع في تسعير البرمجية.

1-2 مشكلة البحث:

للحصول على أفضل التقديرات التي تمكننا من أداء العمل بالصورة الكاملة يجب علينا تحسين نموذج COCOMO2 ليواكب السوق السوداني وذلك بدراسة

الانحراف بين التقديرات المستخرجة من نموذج COCOMO2 ومقارنتها بالسوق السودانية. وذلك من خلال الإجابة عن التساؤلات التالية:

1/ هل يصح تطبيق نموذج COCOMO2 في السودان؟

2/ ما هو مقدار الانحراف بين النموذج والواقع في السوق السودانية؟

1-3 أهمية البحث:

تنبع أهمية هذا البحث من أنه يعمل على إدخال أحد الأدوات العالمية لتقدير تكلفة البرمجيات لتتناسب مع الواقع السوداني، ويطور موقع إلكتروني لتقدير التكلفة وفقاً لنموذج COCOMO2 بمراعاة الانحراف في السوق السودانية.

1-4 أهداف البحث:

يهدف هذا البحث:

أ/ دراسة ملائمة نموذج COCOMO2 مع الأسواق الصغيرة ودراسة السوق السودانية.

1-5 فروض البحث:

الفرضية الأولى: يمكن تنفيذ حسابات كلفة البرمجيات بشكل موضوعي للتنبؤ الدقيق بكلفة تطوير البرمجيات.

الفرضية الثانية: يمكن تقدير تكلفة البرمجيات بالقياس والخبرة المتراكمة للبرامج المشابهة.

1-6 هيكل البحث:

يشتمل هذا البحث على ثلاثة فصول وخاتمة ونتائج وتوصيات على النحو التالي:

الفصل الأول: الإطار العام للبحث، وفيه:

(مقدمة البحث، مشكلة البحث، أهمية البحث، أهداف البحث، فروض البحث، منهج البحث، أدوات البحث، طريقة جمع المعلومات، الدراسات السابقة، هيكل البحث).

الفصل الثاني: الإطار النظري، وفيه مبحثين على النحو التالي:

المبحث الأول: تقدير تكلفة البرمجيات

المبحث الثاني: نموذج الـ COCOMO

الفصل الثالث: منهجية البحث

الفصل الرابع: الدراسة الميدانية.

وذيل البحث بخاتمة ونتائج وتوصيات، تلتها المصادر والمراجع.

الفصل الثاني
الإطار النظري

2-1 تقدير تكلفة البرمجيات

2-2 نموذج الـ COCOMO

2-1 تقدير تكلفة البرمجيات

2-1-1 تمهيد:

بصفة عامة، يتضمن تقدير المشروع البرمجي تقدير المعالم الأساسية للمشروع(Project Key Milestones) والحجم (Size) والموارد (Resources)، وفريق التطوير (Staffing) والجداول الزمنية (Schedules)، التكلفة (Cost)، ويتم ذلك من خلال تنفيذ عدد من الخطوات التكرارية (Iterative Steps)

2-1-2 أهمية التقدير الدقيق لتكلفة المشاريع البرمجية: Importance of Accurate Cost Estimation of Software Projects

يتحكم في تطوير المشاريع البرمجية أربعة عوامل رئيسة بصورة نموذجية هي: الوقت (Time)، والمتطلبات (Requirements) والموارد (Resources) (فريق العمل، البنية التحتية، والاعتمادات المالية)، والمخاطر (Risks)، وأي تغير غير متوقع في هذه العوامل سوف يؤثر تأثيراً مباشراً على خطة تطوير المشروع.

لذا فعملية التقدير الجيدة والموثوق بها لكل من هذه العوامل تُعد حرجة (Crucial) بالنسبة لتطوير المشروع. فالتقدير الأقل من التكلفة الفعلية (Under-Est imating Cost) يمكن أن يقود إلى:

- تكوين فريق عمل قليل العدد مقارناً بالمجهود المطلوب لتطوير المشروع، مما يؤدي إلى إجهاد هذا الفريق، وبالتالي عدم الالتزام بالجدول الزمني الخاص بتطويره.
- تحقيق مستوى أقل من أفق ضمان الجودة المطلوب، مما يؤدي إلى تسليم المنتج البرمجي وهو يعمل ضمن منطقة مخاطر الجودة المنخفضة.
- تحديد جدول زمني قصير المدى لإنجاز المشروع، مما يؤدي إلى عدم تسليمه في المواعيد المحددة، وبالتالي فقد الثقة في مطور المشروع.

أما التقدير الأعلى من التكلفة الفعلية (Over-Estimating Cost) فيمكن أن يؤدي إلى إضافة موارد غير ضرورية وعدم تحكم كافٍ لأفق المشروع، مما يزيد من مدة تسليم المشروع، وبالتالي عدم الاستفادة منه في الوقت المناسب، وربما يؤدي إلى تأجيل تنفيذ المشروع لعدم وجود الموارد المالية اللازمة، وكذلك ربما يؤدي إلى عدم فوز الشركة أو المؤسسة المطورة بالعقد الخاص بتطوير المشروع مما يؤدي إلى فقدان بعض الوظائف بها.

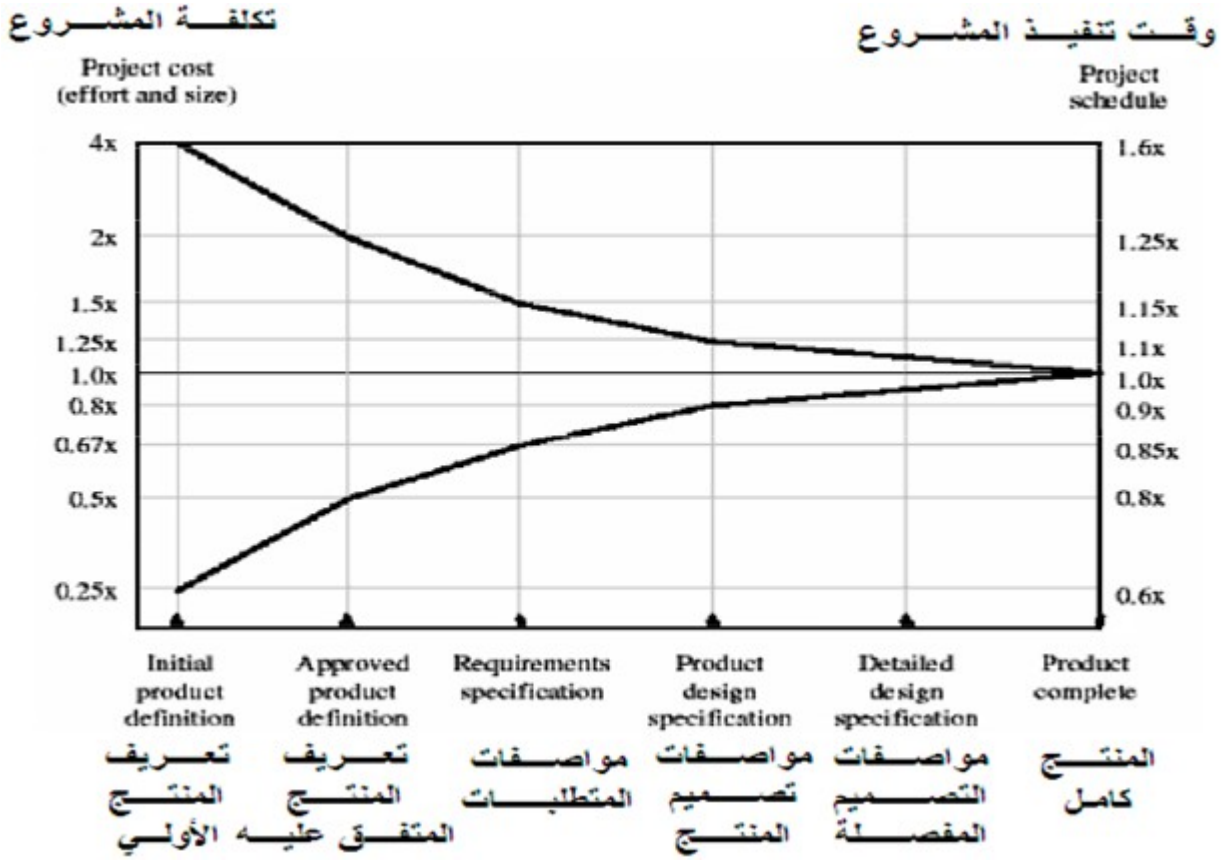
2-1-3 العوامل التي تؤثر علي دقة تقدير تكلفة المشاريع البرمجية

Factors That Affect the Accuracy of Software Cost Estimation:

تتوقف دقة تقدير تكلفة المشاريع البرمجية على العديد من العوامل من أهمها:

أ/ درجة دقة تقدير حجم وأفق المشروع.

ب/ وقت عملية التقدير؛ حيث وجد بوهيم في كتابه الشهير "اقتصاديات هندسة البرمجيات" (بوهيم، 1981: 230) أن عملية تقدير تكلفة المشاريع التي تتم في المراحل الأولى من دورة حياة المشروع تكون بعيدة عن التقدير الدقيق، وتصبح أكثر دقة كلما تقدمت عملية التطوير، ويمكن معرفتها على وجه الدقة عندما ينتهي العمل به، وذلك كما هو موضح بالشكل التالي:



شكل رقم (2/1/1) شكل توضيحي لدقة التقدير في المراحل المختلفة لتطوير المنتج البرمجي

وكما هو واضح من الشكل أعلاه إنه في الوقت المفترض أن يقدر فيه تكلفة المشروع البرمجي، وهي مرحلة التعريف الأولى للمنتج (Initial Product Definition Stage)، يمكن أن يقدر الفرق بين أعلى تقدير وأقل تقدير بـ 16 مرة، وحتى بعد الانتهاء من مرحلة إعداد مواصفات المنتج (Requirements Specification) يكون الجهد المقدر في حدود 50%، ويقل الفرق بين أعلى تقدير وأقل تقدير حتى يتساوى تقريباً مع تسليم المنتج النهائي.

جدول (2/1/1) عوامل ضرب التقدير للمراحل المختلفة لتطوير المشروع(ستيف، 1996: 232)

(Estimation Multiplier by Project Phase)

المرحلة (Phase)	عامل ضرب التكلفة (الحجم والمجهود) Cost (SizeandEffort) Multiplier		عامل ضرب فترة التطوير ScheduleMultiplier	
	Optimistic تفاؤلي	Pessimistic تشاؤمي	Optimistic تفاؤلي	Pessimistic تشاؤمي
التعريف الأولي للمنتج	0.25	4.0	0.60	1.6
التعريف المتفق عليه للمنتج	0.50	2.0	0.80	1.25
مواصفات المتطلبات	0.67	1.5	0.85	1.15
مواصفات تصميم المنتج	0.8	1.25	0.90	1.10
مواصفات تصميم المنتج المفصلة	0.9	1.10	0.95	1.05

الجدول أعلاه يوضح مضمون الشكل (2/1/1) بصورة جدولية، حيث يبين عوامل الضرب (Multipliers) للحصول على: التقديرات المتفائلة(Optimistic Estimates)، والتقديرات المتشائمة(PessimisticEstimates) لكل من الجهد (Effort) والحجم (Size)وفترة الجدولة الزمنية (Schedule)، وذلك عبر دورة حياة المنتج.

1- دقة تحديد متطلبات المشروع ومقدار ثباتها أثناء عملية التطوير.
2- مقدار ثبات ظروف بيئة التطوير التي على أساسها تم تحديد متطلبات المشروع.
3- دقة ترجمة هذا الحجم وهذه المتطلبات إلى جهد بشري، وخطة زمنية، ونفقات مالية.
4- قدرات وكفاءات وإنتاجية فريق العمل ومدى توافقها مع خطة المشروع.
5- حجم البرمجيات الجاهزة التي يمكن إعادة استخدامها في المشروع الجديد.
6- مدى دراية وخبرة مديري المشاريع وطاقم التطوير بالتغير الهائل في التقنيات

المستخدمة في المشاريع، حيث إن هناك طرق تطوير جديدة تـم اسـتحداثها فـي الآونة الأخيرة ومن أمثلتها:

- اسـتخدام البرمجـة الكينونيـة (Object-Oriented) فـي عمليـة التطوير بدلاً من التطوير المبني على الدوال (Function-Oriented).
- استخدام تطبيقات خـادم العميـل (Client-ServerApplications)، والتطبيقـات المعتمـدة علـى الـويب (Web-BasedApplications) بـدلاً مـن التطبيقـات المبنيـة علـى نظـم الحاسـبات المركزيـة (MainframeSystem –BasedApplications)).
- اسـتخدام المكونـات المتناولـة (OffShelfComponents) والمُعـدة سابقاً لبعض أجزاء المشروع بدلا من كتابتها من جديد.
- تطوير المشاريع البرمجية بطريقة هندسة البرمجيات المبنيـة علـى المكونـات (Components-BasedSoftwareEngineering)، بغـرض إعـادة الاستخدام لبعض المكونات في مشاريع أخرى.

استخدام أدوات هندسة البرمجيـات بمسـاعدة الكمـبيوتر ومولـدات البرامـج (CASE Tools and Programs Generators) بدلاً من الوسائل القديمة.

2-1-4 من هو المسئول عن تقدير تكلفة المشـروعات البرمجيـة؟ Who is Responsible for Software Cost Estimation?

المسئولون عن عملية تقدير تكلفة المشروعات البرمجية يمكن أن يختلفوا من منشأة إلى أخـرى حسـب تنظيـم وظـروف كـل منهـا، ولكـن فـي معظـم المنشـآت البرمجية تتم عملية التقدير كالتالي:

أ/ تكليف فريق تطوير المشروع البرمجي بإنجاز عملية التقدير.

ب/ تكليف مدير المشروع بتقديم تقـدير واقعـي لتكلفـة المشـروع الـبرمجي، حيـث يمكنه القيام بهذه المهمة بصـورة منفـردة أو بمشـورة المـبرمجين المسـئولين عـن المشروع.

ج/ تكليف المبرمجين أنفسهم بالقيام بعملية التقدير، حيث أثبتـت معظـم الدراسـات أن اشتراك المبرمجين في عملية التقدير تجعلها أكثر دقة.

د/ تكليف فريق مستقل عن فريق تطوير المشروع لإنجاز عملية التقدير.

هـ/ تكليف خبراء مستقلين من خارج المنشأة للقيام بعملية التقدير، ومن ثم التشاور

مع فريق مشكل من المنشأة للوصول إلى التقدير النهائي.

لذا، ولمحاولة ضمان الدقة في تقدير هذه التكلفة فإنه يجب على مديري المشاريع البحث عن إجابات للأسئلة التالية:

- ما مقدار الجهد **(**Effort**)** اللازم لاستكمال نشاط بحجم **(**Size**)** معين (فرعية معينة)؟.
- ما هي فترة الجدولة الزمنية **(**Schedule**)** اللازمة لاستكمال هذا النشاط؟.
- ما هي التكاليف الكلية **(**Total Cost**)** الخاصة بهذا النشاط؟.
- ما هي النشاطات الإدارية المتداخلة في تقدير كامل للمشروع؟.
- ما هي إنتاجية كل فرد من فريق العمل في عملية التطوير؟.

2-1-5 منهجيات تقدير الجهد Effort Estimation Approaches

يُعرف المصطلح "الجهد" **(**Effort**)** بأنه كمية العمل الإنساني **(**Human Work Amount**)** المبذولة لتطوير المشروع، ويمكن أن يعبر عنها باستخدام العديد من الوحدات، مثل: شخص- ساعات**(**Person - Hours**)**، شخص- أيام **(**Person- day**)**، شخص-أسابيع **(**Person-Weeks**)**، شخص-شهور **(**Person - Months**)**، شخص-سنين **(**Person- Years**)**. ويجب أخذ نظام العمل (عدد أيام العمل في الأسبوع، عدد ساعات العمل في اليوم، عدد أيام الإجازات الاعتيادية والمرضية في السنة) في الاعتبار عند حساب المجهود، فمثلاً في الولايات المتحدة الأمريكية متوسط عدد أيام العمل في السنة حوالي 220 يوم، 5 أيام عمل في الأسبوع، 8 ساعات عمل في اليوم (تحسب عادة 6 ساعات بعد استقطاع أوقات الراحة والاجتماعات).

وبصفة عامة يقدر المجهود من خلال المعادلة العامة:= Staff Effort Size of Work / Production Rate

المجهود= حجم العمل/ معدل الإنتاجية، فمثلاً:

عدد أوراق المواصفات/ معدل إنتاجية المحلل شهرياً = محلل- شهور

عدد سطور الشفرة/ معدل إنتاجية المبرمج شهرياً = مبرمج- شهور

ويمكن القول بأنه ليس هناك منهجية بسيطة لعملية تقدير الجهد اللازم لتطوير مشروع برمجي، حيث يوجد العديد من الصعوبات التي تواجها.

وعموماً هناك عدة منهجيات تُستخدم لتقدير الجهد والتكاليف، والتي يمكن تلخيصها كما أوردها Boehm (1981) في التالي:

1- **نمذجة التكلفة الخوارزمية** (Algorithmic Cost Modeling): وهي منهجية تعتمد على المعادلات (Formulaic Approach)، حيث إنه يتم تطوير نموذج

مبنى على معلومات التكاليف التاريخية (Historical Information Cost Model) والتي تربط بعض سمات المشروع البرمجي ـ غالبا ما تكون حجم المشروع- بتكاليف المشروع، ومن ثم عمل تقدير لهذه السمة بحيث يستطيع النموذج المطور التنبؤ بالجهد المطلوب. تمتاز هذه المنهجية بأنها موضوعية (Objective) وذات علاقة مباشرة بعملية التكلفة، وقابلة للتكرار (Repeatable) والتحليل (Analyzable)، وتمتاز بالكفاءة في تحليل الحساسية (Sensitivity Analysis)، ومعايرة بطريق موضوعية للخبرات (Objectively Calibrated to Experience)، ولكن يعيبها الحساسية الزائدة للمدخلات، حيث إنها ترتبط بمدخلات غير موضوعية(Subjective Inputs)، ومخرجاتها تتأثر بالظروف المحيطة بعملية التطوير.

2- **رأي الخبير**(Expert Judgment): وفيه يتم الاستفادة من الخبراء المتخصصين في مجال تطوير المشاريع البرمجية حيث يقوم كل منهم بعمل تقدير لتكلفة المشروع حيث يتم مناقشتها والوصول إلى رأي نهائي متفق عليه لهذه التكاليف، وتعتمد دقة تقدير هذه التكلفة على مدى دراية هؤلاء الخبراء بمشاريع مماثلة.

3- **التقدير المبني على التشابه** (Estimation by Analogy): وفيه يتم تقدير تكلفة المشروع بناءً على تكلفة مشاريع مماثلة سابقة في مجال التطبيق نفسه، حيث يتم تقدير تكلفة المشروع قيد الدراسة بالتشابه، وتمتاز هذه الطريقة بالدقة إذا توافرت بيانات لمشاريع مماثلة للمشروع تحت التقدير، ويعيبها أنها مستحيلة الحدوث إذا لم يكن هناك مشاريع مماثلة.

4- **قانون باركنسون** (Parkinson's Law): ينص على: تمديد المشروع لينفذ خلال الفترة المتاحة، وهذا يعني أن تقدير التكاليف يتم بناءً على المدة المحددة لتنفيذ المشروع والموارد المتاحة، فعلى سبيل المثال إذا ما طلب تسليم مشروع برمجي خلال 12 شهراً ولدينا خمس مبرمجين فإن الجهد اللازم يتم تقديره بـ 60 شخص ـ شهور (60 person-months).

5- **التقدير للفوز**(Pricing to Win) : وفيه يتم تقدير تكلفة المشروع البرمجي بناءً على مقدرة العميل على الإنفاق عليه، أي أن الجهد يقدر بناءً على ميزانية العميل وليس على أداء البرمجية. وقد تبدو هذه الطريقة غير أخلاقية، ولكن يلجأ إليها بعض الشركات والمؤسسات كثيراً في حالة عدم توافر مواصفات تفصيلية

للمشروع المطلوب تنفيذه، وتُقدر التكاليف بحيث تتوافق مع أساسيات العرض العام المقدم بدون النظر إلى جودة أداء المشروع التي يجب أن يتصف بها، حيث تتقيد عملية التطوير بالتكلفة، وربما يتم التفاوض بشأن تفاصيل المواصفات أو منهجية الارتقاء بالنظام فيما بعد.

وعموماً فإن سريان تقدير الجهد في الطرق السابقة يمكن أن يتم عن طريق اتجاهين إما من "أعلى إلى أسفل" (Top-Down Approach) أو من "أسفل إلى أعلى" (Bottom-Up Approach). والاتجاه الأول "من أعلى إلى أسفل" يبدأ من مستوى النظام ككل حيث يقوم المثمن بفحص الأداء الكلي للنظام وتفريع هذا الأداء إلى فرعيات أقل حيث يؤخذ في الاعتبار تكلفة أنشطة النظام مثل (التكاملية، الأداء، التوثيق....)، وهو قابل للاستخدام بدون معرفة معمارية النظام ومكوناته. والاتجاه الثاني "من أسفل إلى أعلى" على العكس يبدأ من أسفل من مستوى المكونات البسيطة حيث يتم تفكيك النظام إلى مركبات ويتم تقدير الجهد اللازم لكل مركبة وتجميع هذه التكاليف لتقدير التكاليف الكلية، وهو قابل للاستخدام عند معرفة معمارية النظام وتعريف مكوناته. والميزات للاتجاه "من أعلى إلى أسفل" تُعد عيوب في الاتجاه "من أسفل إلى أعلى" والعكس. فعلى سبيل المثال الاتجاه الأول "من أعلى إلى أسفل" يمكن استخدامه في تقدير تكلفة المسائل المعقدة المصاحبة للنظام كواجهات الاستخدام، ولكن قد يصعب تقدير تكلفة مشاكل التقنيات الدنيا الصعبة (Difficult Low Level Technical Problems) بالدقة المطلوبة، حيث إن ذلك يتطلب تصميم النظام بالتفصيل، في حين أن الاتجاه "من أسفل إلى أعلى" يأخذ في الاعتبار تكلفة هذه المشاكل إلا أنها غالية التكاليف حيث إنه لا بد من وضع تصميم بدائي للنظام، ولكن في الوقت نفسه قد يكون من الصعب تقدير تكلفة أنشطة النظام مثل (التكاملية، الأداء، التوثيق....).

ولكل طريقة تقدير نقاط قوه ونقاط ضعف، ويحبذ للمشاريع الكبيرة أن يُستخدم العديد من الطرق لتقدير كلفتها، ومقارنة نتائجها، فإذا أعطت نتائج متنافرة وغير متفقة فهذا يعني عدم توافر المعلومات الكافية عن تقدير التكلفة، وبالتالي فإنه لابد من النظر في إمكانية الحصول على معلومات أكثر وإعادة تقدير التكلفة بالطرق المختلفة حتى تتقارب تقديراتها.

2-1-6 تقدير التكلفة Cost Estimation:

هناك ثلاثة متغيرات أساسية تدخل في تقدير التكاليف الكلية لمشروع تطوير برمجية:

1- تكلفة العتاد والبرمجيات وهي تشمل الصيانة.
2- تكلفة السفر للعاملين والتدريب.
3- التكاليف المدفوعة للمطورين (يطلق عليها تكلفة المجهود).

ويمكننا القول إن التكاليف السائدة في معظم المشاريع هي "تكلفة الجهد"؛ حيث تكون تكلفة العتاد اللازمة بسيطة جداً، وكذلك فإن تكلفة السفر إلى الأماكن المختلفة التي ينفذ بها المشروع بسيطة نسبياً، حيث إن وجود وسائل الاتصال المختلفة (فاكس، البريد الإلكتروني، وسائل الاتصال من بعد) يمكن أن تقلل وإلى حد كبير من تكلفة السفر.

وتكاليف الجهد ليست فقط تشمل مرتبات فريق تطوير المشروع البرمجي (مهندسي هندسة نظم البرمجيات ومهندسي هندسة البرمجيات، ومهندسي اختبار البرمجيات فريق إدارة ودعم المشروع) حيث تحسب الجهات المطورة تكلفة الجهد بدلالة التكاليف الكلية حيث يتم تقدير التكاليف الكلية وقسمتها على عدد المطورين ولذا فإنها فرعية من الفرعيات التالية:

- تكلفة الإعاشة وتشمل الإنارة والتدفئة للمكاتب.
- تكلفة العمالة المساعدة (المحاسبين، أفراد السكرتارية، عمال النظافة، والعمال الفنيين).
- تكلفة الشبكات والاتصالات.
- تكلفة المشاركة في الأماكن الجماعية للعاملين (المطعم للعاملين، المكتبة).
- تكلفة التأمين الاجتماعي للعاملين، التأمين الصحي ونحوه.

وبالإضافة إلى العوامل الأساسية السابقة يوجد عوامل أخرى يمكن أن تؤثر في تقدير تكلفة المشروع البرمجي من قبل الجهة المطورة من أهمها:

اكتساب سابقة أعمال: ربما تتجه الجهة المطورة إلى تحديد سعر رخيص لأنها تريد أن تتطلع إلى مرحلة مستقبلية في سوق البرمجيات وقبولها لهذا الربح البسيط في مشروع ربما يكون سبباً في تحقيق أرباح طائلة في مشاريع أخرى مستقبلية وهذه الخبرة المكتسبة ربما تسمح بتطوير منتجات جديدة.

تقدير التكاليف غير المؤكد: في حالة عدم تأكد الجهة صاحبة المشروع من تقدير التكاليف ربما تزيد من السعر وتحقق ربح أعلى من الربح المعتاد.

بنود تعاقدية: ربما يرغب العميل من المطور استخدام مصدر البرنامج في مشـاريع أخرى وفي هذه الحالـة تكـون التكـاليف أقـل بكـثير عمـا لـو أمتلـك العميـل مصـدر البرنامج.

صلاحية المتطلبات: إذا مـا تغيـرت المتطلبـات تخفـض المؤسسـة السـعر حـتى تحصل على العقد وبعد حصولها على العقد تزداد التكاليف بتغيير المتطلبات.

الحالة المالية: المطورون الذين لديهم اعتمادات مالية ضـعيفة غالبـاً مـا يقومـون بتخفيض السعر للحصول على العقد وفي هذه الحالة من المستحسن الحصول على ربح ولو بسيط بدلاً من عدم وجود عمل.

2-2 نموذج الـ COCOMO

2-2-1 تمهيد:

نماذج التقدير الخوارزمية (Algorithmic Estimation Models)هـي: عبـارة عـن دوال رياضـية مسـتنبطة علـى أسـاس نظـري (Theory–Based Models) (مثـل: Putnam SLIM Model)ـ أو علـى أسـاس تجريـبي (Empirical-Based Model) باسـتخدام التحليـل الانحـداري (Regression Analysis) علـى بيانـات تاريخيـة (Historical Data) مجمعة من مشاريع برمجية سابقة (مثـل:COCOMO Models)، هذا بالإضافة إلى البديهة(Intuition) والخـبرات (Experiences)، أي أن معظـم هـذه النماذج في النهاية تعتمد على مجموعة من البيانات التاريخية المـأخوذة مـن عينـات مشاريع محددة، لذلك يجب معايرتها قبل اسـتخدامها لتلائـم مـع بيئـة التطـوير الـتي سوف تستخدم بها والتعامل مع نتائجها بحذر، حيث إنه لا يوجد نمـوذج ملائـم لشـتى أنواع البرمجيات وجميع بيئات التطوير البرمجي.

والشكل العام لمعظم هذه النماذج يأخذ الصيغة الشائعة الموضحة بالمعادلـة رقـم (1):

E = A* (Size)B * EMF.....(1)

حيث "E" تمثل الجهد المقـدر بشـخص- شـهور(Effort in PM)، و"A"، و"B" فهي ثوابت تُشتق تجريبياً، حيث الثابت "A" يعتمد على هيكلية وخـبرة المؤسسـة أو الشركة المطورة (Organization Dependent)، وكذلك على نوع المشروع الذي تحت التطوير، والثابت "B" يعكس تأثير بعـض العوامـل المتعلقـة بأسـلوب التطـوير مثل: مرونـة التطـوير (Development Flexibility)، ونضـج عمليـات التطـوير (Development Process Maturity)، ومـدى تنـاغم وتفـاهم فريـق التطـوير (Team Cohesion)، ومنهجيـــات إدارة المخـــاطر (Risk Management Techniques)، ويتناسب طردياً مع زيادة حجـم المشـروع، ويعكـس الحقيقـة الـتي تقول إن التكلفة تزداد بزيادة الحجم ولكن بطريقة غير خطية، حيث إنه بزيادة حجـم المشروع يزداد حجم فريق التطـوير وتقـل الإنتاجيـة نتيجـة الفاقـد الناتـج مـن كـثرة الاتصـالات بيـن الفريـق، وزيـادة المجهـود الـذي يبـذل فـي إدارة الفريـق، أمـا " Size"فيمثـل حجـم البرمجيـة، و"EMF" ثـابت يمثـل عامـل ضـبط الجهـد (Effort Adjustment Factor) لضـبط الجهـد ليتضـمن تـأثير موجهـات التكلفـة (Cost

. (Drivers

ويمكن استخدام النماذج الخوارزمية في تقدير تكلفة المشاريع البرمجية كـأداة أساسية أو كأداة ثانوية للتحقق من النتائج بناءً على حالة المشروع، فمثلاً فـي حالـة نضج متطلبات المشروع وفهم مواصـفات المتطلبـات والتصـميم فهمـاً جيـداً يفضـل استخدام منهجية التناظر فـي عمليـة التقـدير كمنهجيـة أساسـية واسـتخدام منهجيـة النماذج الخوارزمية كمنهجية ثانوية للتحقق من التقدير، أمـا فـي حالـة تقـدير تكلفـة المشروع في بداية دورة حياته مع متطلبات غير واضـحة فيمكـن اسـتخدام النمـاذج الخوارزمية كأداة أساسية بجانب التناظر مع مشاريع مماثلة لمعـايرة هـذه النمـاذج. وأيضاً يمكـن اسـتخدام النمـاذج الخوارزميـة للتفاضـل بيـن مجموعـة مـن منهجيـات التطوير من خلال تحليل تأثيراتها النسبية على عملية التطوير.

وإجمالاً تتصف معظم هذه النماذج بما يلي:

- استخدام معادلات رياضية لإجراء عملية التقدير.
- المعادلات الرياضية معتمدة على النظريات الرياضية أو البيانات التاريخية.
- المدخلات المستخدمة (Used Inputs) في عمليات التقدير مثل: عدد سطور الشـيفرة (LOC)، عدد نقاط الدالة (FP)، مسببات أخرى للتكلفة (Other Cost Drivers).
- دقة هذه النماذج يمكن أن تُحسن عن طريـق معايرتهـا فـي نفـس بيئـة التطـوير التي سوف تستخدمها في عملية التقدير.

ولسوء الحظ تعاني معظم هذه النماذج من الصعوبات الأساسية التالية:

- غالباً ما يكون من الصعب تقدير حجم البرمجية في مرحلة متقدمة من دورة حياة المنتج البرمجي، وذلك لأن حجم البرمجية يعتمد على قرارات التصميم (Design Decisions) والتي غالباً لا تحدد في بداية دورة حياة المنتج البرمجي، حيث تشمل هذه القرارات مثلاً: لغة التطوير والتي تؤثر تأثيراً مباشراً في تحديد حجم البرمجية، نوع قاعدة البيانات التي سوف تستخدم، كمية الشفرة التي سوف يعاد استخدامها، وخلاف ذلك. ورغم أنه في هذه الحالة يمكن اللجوء إلى نقاط الدالة أو نقاط الكائن كمقياس للحجم بدلاً من عدد سطور الشفرة، إلا أنها غالباً ما تكون غير دقيقة.
- تقدير تأثير العوامل التي تساهم في تحديد قيم الثوابت "EMF"،"B" غير موضوعية (Subjective)، حيث إنه لا يوجد تعريفات ملائمة لهذه العوامل مما يجعلها شبه مبهمة، وبالتالي يعتمد تقديرها على الشخص المكلف بتحديدها وعلى مدى خبرته بنوعية المنتج تحت التطوير، لذا فهذه الثوابت يمكن أن تتغير من شخص إلى آخر وفي بيئة التطوير نفسها وللمنتج نفسه، وبالتالي تختلف نتيجة التقديرات من شخص إلى آخر.

معظم هذه النماذج تعاني من عدم دقة تمثيل تأثير الإنتاجية (Productivity) على عملية التقدير، وخصوصاً عند استخدام لغات الجيل الرابع، وبالتالي فإن قيم التقديرات تكون محل شك، ويجب إجراء دراسات إضافية في هذا المجال.

2-2-2 مقياس الصناعة لدقة تنبؤ نموذج التكلفة Industry Measure for Cost Model's Predictive Accuracy:

يمكن قياس مدى دقة النموذج التجريبي في تقدير المشاريع البرمجية من خلال حساب قيمة الخطأ النسبي (Magnitude of Relative Error "MRE") باستخدام المعادلة التالية:

$$MRE = \left(\frac{\left(\left| E_{pred} - E_{act} \right| \right)}{E_{act}} \right) \ldots (2)$$

أو يمكن من خلال حساب متوسط قيمة الخطأ النسبي (Mean Magnitude of Relative Error "MMRE") الأكثر دقة باستخدام المعادلة التالية:

$$MMRE = \frac{1}{n} . \sum_{i=1}^{i=n} \left(\frac{\left(\left| E_{pred} - E_{act} \right| \right)}{E_{act}} \right) i \ldots (3)$$

حيث "n" تمثل عدد عينة المشاريع/التقديرات تحت الاختبار (Number of Projects/Estimates Under Test)، و E_{pred} قيمة الجهد التي تنبأ بها النموذج (Predicted Value)، و E_{act} هي القيمة الحقيقة (Actual Value) للجهد، ويجب ألا تزيد قيمة "MMRE" عن 25% في حالة كون نتائج النموذج مرضية.

هذا بالإضافة إلى وجود العديد من مقاييس اختبار الدقة في صناعة تطوير البرمجيات لتقويم أداء نماذج التقدير التجريبية لتكلفة المشاريع البرمجية، أحد هذه المقاييس ينظر لقدرة النموذج على التنبؤ بالجهد على مستوي عينة من التطبيقات غير المترابطة عند مستوى معين من الدقة (L)، ويُعرف بالمعادلة رقم (4):

PRED (L) = K / N.....(4)

حيث "K" يمثل عدد مجموعة (Subset) من عينة المشاريع /التقديرات تحت الاختبار ذات الدقة الواقعة ضمن المدى المقبول من قيمة الخطأ النسبي (L=MRE) و"N" عدد عينة المشاريع/التقديرات تحت الاختبار،"L" مستوى دقة التنبؤ ويمثل متوسط قيمة الخطأ النسبي (MRE)، ويجب ألا يقل هذا المقياس عن 75%. لذا فقد اقترحت بعض الدراسات التجريبية [16] استخدام هذا المقياس للدلالة على دقة النموذج عن قيمة خطأ نسبي يساوي 0.25(L=MRE=0.25)، ويجب ألا يقل PRED(0.25) عن 0.75.

2-2-3 تقويم نماذج تقدير التكلفة Evaluating of Cost Estimation Models:

تستخدم المعايير التالية لتقييم نماذج تقدير تكلفة المشاريع البرمجية [7]:

أ- التعريف(Definition): هل عرف النموذج أنواع التكلفة المقدرة وتلك المستبعدة من عملية التقدير؟.

ب-الدقة(Fidelity): هل التكاليف المقدرة قريبة من التكاليف الحقيقية، وما نسبة هذا التقارب؟.

ج-الموضوعية (Objectivity): هل يتجنب النموذج تخصيص معظم متغيرات تكاليف البرمجية (مثل التعقيد "Complexity") لعوامل غير موضوعية (Subjective) تؤدي إلى معايرة النموذج على نحو رديء؟ وهل من الصعب ضبط هذه العوامل للحصول على النتائج التي يريدها المستخدم؟.

د-البناء الاستدلالي (Constructiveness): هل يمكن للمستخدم أن يفسر الأسباب التي أدت إلى نتائج التقدير الناتجة من استخدام النموذج؟ . وهل يساعد النموذج المستخدم على فهم الخطوات التي يجب أن تتم للحصول على نتائج

جيدة؟.

هـ-التفاصيل (Details): هل النموذج ملائم لتقدير نظم برمجية تتكون من العديد من الوحدات والنظم الفرعية؟ وما مدى هذه الملائمة من حيث سهوله تكيفه مع هذه النظم وإعطاء تقسيم دقيق للنشاطات والمراحل المختلفة الخاصة بتطوير البرمجية؟.

و-الاستقرار(Stability): هل التغيرات الصغيرة في العوامل الداخلة في عملية تقدير التكلفة تؤدي إلى تغييرات صغيرة في نتائج تقدير التكلفة؟.

ز-الأفق(Scope): هل يمكن للنموذج أن يستخدم في تقدير تكلفة معظم أصناف المشاريع البرمجية التي يحتاجها المستخدم؟.

ح-سهولة الاستخدام (Ease of Use): هل من السهل فهم وتعيين مدخلات واختيارات النموذج والتعامل معها بيسر؟.

ي-التوقع المستقبلي (Prospectiveness): هل يتجنب النموذج استخدام المعلومات التي لا يمكن معرفتها بصورة جيدة إلا مع اكتمال المشروع؟.

ك- التدبير(Parsimony): هل يتجنب النموذج استخدام عوامل زائدة عن الحاجة، أو تلك التي لا تؤثر تأثيراً فاعلاً يمكن إدراكه في نتائج التقدير؟.

2-2-4 أنواع النماذج الخوارزمية Types of Algorithmic Models :

وقد تم اقتراح العديد من النماذج الخوارزمية *(*Algorithmic Models*)* لتقدير الجهد وفترة الجدولة الزمنية وتكلفة المشاريع البرمجية، منها القليل ممّا هو مستنتج على أساس نظري *(*Theory–Based Models*)*، أما الكثير منها فهو مستنتج على أساس تجريبي *(*Empirical-Based Models*)* وهي جمعياً متشابهه من ناحية الشكل والمفهوم، ولكنهم يستخدمون قيم مختلفة للعوامل *(*Parameters*)* الداخلة في تكوينها، وسوف نتناول هنا أهم نموذجين وهما: نموذج Putnam(SLIM Model*)* ونموذج Boehm (COCOMO Model*)*.

2-2-5 نموذج بوهم (نموذج التكلفة الاستنتاجي)[1, 3, 20 , 23, 24, 25, 26] :COCOMO " Boehm Model (COnstructive COst MOdel ")

نموذج كوكومو نموذج تجريبي مستنتج من التحليل الانحداري لبيانات مشاريع عديدة وخبرات تطوير سابقة، وقد قام بتطويره Boehm عام 1981م، وصدر الإصدار الأول منه في نفس العام وأطلق علية نموذج COCOMO ا، أو COCOMO

81، وتم بعد ذلك تحسينه حتى صدر منه الإصدار الثاني المحسن COCOMO II في عام 2000م، حيث يأخذ فـي الاعتبـار المنهجيـات المختلفـة فـي تطـوير البرمجيـات وإعادة الاستخدام وغيرها من العوامل التي تؤثر على تكلفة المنتج البرمجي. لذا يُعد نموذج Boehm بنية هرمية لنماذج تقدير تكلفة المنتج البرمجي تحمـل اسـم نمـوذج كوكومو.

ونموذج كوكومو مفتوح وموثق جيداً ومتاح للجميع، ولكنه معقـد بعـض الشـيء، ويُعد أشهر نموذج لتقدير تكلفة المشاريع البرمجية في الوقت الحالي.

2-2-6 نموذج كوكومو الأصلي (COCOMO I Model):

يتضمن نموذج كوكومو الأصلي COCOMO) (امجموعة من النماذج الفرعية تستخدم المعادلتين التاليتين (معادلة رقم (5)، والمعادلة رقم (6)):

E = a * Sizeb .....(5)

TDEV = 2.5 * Ec... .....(6)

حيث:

"E": يمثل جهد التطوير مقيساً بشخص ـ شهور.

"Size": يمثل حجم البرمجية مقيساً بعدد سطور الشيفرة بالكيلو (KLOC).

" TDEV ": يمثل الوقت اللازم لتطوير البرمجية بالشهور.

"a"،"b"،"c": عوامل (Coefficients) تعتمد على شكل التطوير (DevelopmentMode)، وكذلك على مستوى النموذج (ModelLevel).

ويوجد ثلاثة أشكال من التطوير (ThreeDevelopmentModes) هما:

أ-عضوي أو متناسق الأجزاء أو مترابط(Organic): يعني أن المشروع البرمجي تحت التطوير صغير الحجم نسبياً، وبسيط، ومفهوم جيداً، ويطور بواسطة فريق صغير ذي خبرة جيدة في مجال تطبيق المشروع، ويحتاج قليلاً من الابتكار أو الفكر الجديد (Little Innovation)، ولا توجد قيود صارمة على وقت التسليم (No Tight Deadline)، وكذلك يطور في بيئة تطوير مستقرة (Environment Stable Development).

ب- شبه مترابط(Semi-Detached):يعني أن المشروع البرمجي تحت التطوير: متوسط من ناحية الحجم والتعقيد، ويطور بواسطة فريق ذي خبرة متوسطة في مجال تطبيق المشروع، ويحتاج شياً متوسطاً من الابتكار أو الفكر الجديد (Medium Innovation)، وتوجد قيود متوسطة على وقت التسليم (Medium Constrain for Deadline)، وكذلك يطور في بيئة تطوير متوسطة التعقيد والإمكانيات (Medium Development Environment) .

ج- مضَّمن (Embedded): يعني أن المشروع البرمجي تحت التطوير: كبير الحجم ومعقد، حيث تكون البرمجيات جزءاً من ارتباط شديد التعقيد مع العتاد وإجراءات التشغيل، ويطور بواسطة فريق ذي خبرة محدودة في مجال تطبيق المشروع،

ويحتاج إلى ابتكار أو فكـر جديـد (High Innovation)، وتوجـد قيـود صـارمة علـى وقت التسليم (Tight Deadline)، وكذلك يطور في بيئة تطوير معقـدة مـن حيـث العتاد وواجهات المستخدم.

وكذلك يوجد ثلاثة مستويات للنموذج (Three Model Levels) هي:

أ- **أساسي**(Basic):حيث يتم حساب الجهد وتكلفـة المشـروع الـبرمجي كدالـة فـي حجـم المشروع معبراً عنـه بعـدد أسـطر الشـفرة المقـدرة، والجـدول التـالي (2/2/1) يوضـح قيـم العوامل "a"،"b"،"c" لهذا المستوى لمختلف أشكال التطوير.

جدول (2/2/1) يوضح قيم العوامل (a, b, c)

Basic COCOMO	a	b	c
Organic	2.4	1.05	0.38
Semi-detached	3.0	1.12	0.35
Embedded	3.6	1.20	0.32

ب- متوسط (Intermediate):حيث يتم حساب الجهد وتكلفة المشروع البرمجي كدالة في حجم المشروع معبراً عنه بعدد أسطر الشفرة المقدرة، ومجموعة من موجهات التكلفة (Cost Drivers) الممكن تجميعها وتوزيعها على أربع فئات أساسية سمات المنتج، سـمات العتاد، سمات فريق التطوير، وأخيراً سمات المشروع، حيث تقدر هذه السمات بـ(15) سـمة تُقدّر كل منها من سلم ذي ست علامات تقع بين "منخفض جداً" و"عال جداً جداً" في الأهمية والقيمة، كما هو موضح بالجدول رقـم (2/2/2). وبنـاءً علـى تقـديرات هـذه السـمات يجـري تحديد عامل ضبط للجهد ("Effort Adjustment Factor "EAF) بضرب عوامل الضرب السابقة، ويأخذ نموذج كوكومو المتوسط الصيغة التالية للجهد (المعادلة رقم 13):

E = a * Sizeb * EAF.....(13)

جدول رقم (2/2/2) يوضح قيم العوامل "a"،"b"،"c" لهذا المستوى لمختلف أشكال التطوير

Intermediate COCOMO	a	b	c
Organic	3.2	1.05	0.38
Semi-detached	3.0	1.12	0.35
Embedded	2.8	1.20	0.32

تلاحظ اختلاف قيم العامل "a" للمستوى الأساسي عنه للمستوى المتوسط، مع احتفاظ باقي العوامل بنفس القيم.

جدول رقم (2/2/3) قيم موجهات التكلفة في نموذج كوكومو 81]21، 22، 23[

Cost drivers used in the standard COCOMO model and the Effort Multipliers associated with different ratings

Category الفئة	CostDriver موجه التكلفة	VeryLow منخفض جداً	Low منخفض	Nominal عادي	High عالٍ	VeryHigh عالٍ جداً	ExtraHigh عالٍ جداً جداً
ProductAttributes سمات المنتج	RELY اعتمادية النظام المطلوب	0.75	0.88	1.00	1.15	1.40	-
	DATA حجم قاعدة البيانات المستخدمة	-	0.94	1.00	1.08	1.16	-
	CPLX مدى تعقيد المنتج البرمجي	0.70	0.85	1.00	1.15	1.30	1.65
ComputerAttributes سمات الكمبيوتر	TIME القيود المفروضة على وقت التنفيذ	-	-	1.00	1.11	1.30	1.66
	STOR القيود المفروضة على سعة الذاكرة	-	-	1.00	1.06	1.21	1.56
	VIRT مدى تغيير بيئات التشغيل الخاصة بالعتاد المستخدم	-	0.87	1.00	1.15	1.30	-
	TURN المدى الزمني الذي يدور حوله استخدام العتاد	-	0.87	1.00	1.07	1.15	-
Personnel Attributes السمات الشخصية	ACAP قدرة محلل المشروع	1.46	1019	1.00	0.86	0.71	-
	AEXP مدى الخبرة بالتطبيقات	1.29	1.13	1.00	0.91	0.82	-
	PCAP قدرة المبرمج	1.42	1.17	1.00	0.86	0.70	-
	VEXP الخبرة بالعتاد وخصوصاً نظم التشغيل	1.21	1.10	1.00	0.90	-	-
	LEXP الخبرة بلغات البرمجة	1.14	1.07	1.00	0.95	-	-

Category الفئة	CostDriver موجه التكلفة	VeryLow منخفض جداً	Low منخفض	Nominal عادي	High عالٍ	VeryHigh عالٍ جداً	ExtraHigh عالٍ جداً جداً
ProjectAttributes سمات المشروع	MODP مدى استخدام خبرات البرمجة الحديثة	1.24	1.10	1.00	0.91	0.82	-
	TOOL استخدام أدوات البرمجيات	1.24	1.10	1.00	0.91	0.83	-
	SCED ضغط جدول مواعيد التطوير	1.23	1.08	1.00	1.04	1.10	-

ج- متقدم (Advanced): حيث يتم حساب الجهد وتكلفة المشروع البرمجي كدالة في حجم المشروع معبراً عنه بعدد أسطر الشفرة المقدرة، ومجموعة من موجهات التكلفة (Cost Drivers) مثل النموذج المتوسط، إضافة إلى السماح بضبط موجهات التكلفة لكل مرحلة من مراحل التطوير، وكذلك إجراء بعض التعديلات على مستوى وحدة البرمجة المركبة (Module)، وعلى مستوى النظام (System Level).

2-2-7 نموذج كوكومو الثاني (COCOMO II Model):

يفترض نموذج COCOMO 81 أن البرمجية سوف تطور طبقاً لنموذج الشلال مستخدماً اللغات الأمرية العادية مثل: لغة C، ولغة FORTRAN. ولكن في الوقت الراهن يوجد تغيير جوهري في تقنيات ومنهجيات ولغات تطوير البرمجيات مقارنة بما كان عليه الحال في وقت ظهور هذا النموذج، حيث إنه حالياً تطور البرمجيات غالباً باستخدام لغات برمجة قواعد البيانات مثل SQL، وكذلك نظم قواعد البيانات التجارية (Commercial Database Management System)، مستخدماً في كثير من الأحيان المكونات المعادة للاستخدام (Reusable Components) . هذا بالإضافة إلى استخدام الهندسة العكسية (Re-Engineering) لتطوير برمجيات جديدة من برمجيات موجودة بالفعل مع وجود الدعم الكامل لجميع مراحل تطوير البرمجيات من خلال استخدام الـ CASE Tools . لذلك تم تطوير هذا النموذج إلى نموذج COCOMOII ليشمل هذه التغييرات في عملية تقدير تكلفة المشاريع البرمجية، حيث إنه يدعم منهجيات مختلفة لتطوير البرمجيات مثل: نموذج Prototyping، والنموذج الحلزوني (Spiral Model)، وكذلك يدعم استخدام الهندسة العكسية في تطوير البرمجيات وإعادة استخدام المكونات البرمجية، وبصفة عامة ويتميز هذا الموديل عن نظيره السابق بالأتي:

- الموديل السابق يتطلب معرفة مبدئية لحجم البرمجية كأساس في حين أن هذا الموديل يأخذ جهود تخمين مختلفة معتمداً على مرحلة التطوير للمشروع.
- الموديل السابق يعطي تقدير للجهد والوقت في حين أن هذا الموديل يعطي مدى للتقدير الذي يقع الجهد المطلوب في حدوده.
- الموديل الحالي يتكيف مع متطلبات إعادة الاستخدام والهندسة العكسية (Reengineering) حيث تستخدم العديد من الوسائل لترجمة البرمجية الموجودة في حين أن الموديل السابق لا يأخذ في الاعتبار هذه العوامل.
- الموديل الحالي يأخذ في الاعتبار التغير السريع في المتطلبات.
- مركبة الحجم في الموديل الحالي تستخدم 5 عوامل لتعميم واستبدال تأثيرات نموذج التطوير بدلاً من استخدام ثلاثة أشكال من التطوير (Three Development Modes) في النموذج السابق، كما سيتم توضيحه فيما بعد.
- تم إضافة (7) موجهات تكلفة جديدة (7 New Cost Drivers) هي:,DOCU RUSE, PVOL,PLEX, LTEX, PCON, SITE، وتم حذف (5) موجهات تكلفة في

النمـوذج السـابق هـي: MODP ,LEXP ,VEXP ,TURN ,VIRT، كمـا سـيتم توضيحه فيما بعد.

يتضمن نموذج COCOMO II سلسلة مـن أربعـة نمـاذج فرعيـة متتاليـة تسـمح بتقدير تكلفة المشاريع البرمجية بنـاءً علـى التفاصـيل المتاحـة لحسـاب التقـديرات، وهي كما يلي:

أ- **نموذج تكوين التطبيق** (The Application Composition Model):

وهـو يـدعم تقـدير تكلفـة المشـاريع المطـورة باسـتخدام نمـوذج العمليـات Prototyping، وكذلك المشاريع المطورة عن طريق تجميع وتهيئة مكونات برمجيـة موجودة، والتي تستخدم تقنية إعادة الاستخدام بكثافة. ويستخدم هذا النموذج نقـاط الكائن (Object Points) لتقـدير الجهـد اللازم لعمليـة التطـوير باسـتخدام معادلـة بسيطة (المعادلة رقم 7)، وهي:

E = NOP * (1 – Reuse/100) / PROD.....(7)

حيث:"E": الجهد (Effort) مقيـس بالشـخص ــ شـهور (Person-Month"،"NOP": عدد نقاط الكائن (Object Points)، و "PROD": الإنتاجية (Productivity)، وتقدر كما هو موضح بالجدول رقم 7.5، و"Reuse": النسبة المئويـة لنقـاط الكـائن الـتي سيعاد استخدامها من مشاريع سابقة.

جدول رقم (2/2/4) يوضح قيم معامل إنتاجية نقاط الكائن كدالة في إمكانات وخبرة المطور وأدوات وإمكانات]ICASE]23

Developer's Experience and Capability إمكانات وخبرة المطور	Very Low	Low	Nominal	High	Very High
ICASE Maturity and Capability نضج وإمكانات ICASE	Very Low	Low	Nominal	High	Very High
معامل الإنتاجية "PROD"	4	7	13	25	50

ويتـم تحديـد الحجـم الابتـدائي للتطـبيق الـبرمجي بإحصـاء عـدد الشاشـات (Screens)، والتقارير (Reports)، والمكونات البرمجيـة المكتوبـة بلغـات الجيـل الثـالث(

Third-Generation Components) والتي سوف تستخدم في التطبيق، ومن ثم يتـم تحديـد مـدى تعقيـد الكـائن (Object Complexity) مسـتخدماً الإرشـادات الموضحة بالجداول التالية: (2/2/5)، (2/2/6)، (2/2/7)([23]).

جدول رقم (2/2/5) يوضح مستويات تعقيد نقاط الكائن للشاشات [23]

Object Points Complexity Level for Screen

Number of Views Contained	Number and Source of Data Tables		
	Total <4	Total <8	Total <+8
3>	Simple	Simple	Medium
3-7	Simple	Medium	Difficult
8+	Medium	Difficult	Difficult

جدول رقم (2/2/6) مستويات تعقيد نقاط الكائن للتقارير (كيم جونسون، 1998)

Object Points Complexity Level for Reports

Number of Views Contained	Number and Source of Data Tables		
	Total <4	Total <8	Total <+8
3>	Simple	Simple	Medium
3-7	Simple	Medium	Difficult
8+	Medium	Difficult	Difficult

جدول رقم (2/2/7) يوضح وزن مستويات تعقيد نقاط الكائن (كيم جونسون، 1998)

Complexity Levels Weights for Object Points

نوع الكائن	Simple	Medium	Difficult
Screen	1	2	3
Report	2	5	8

3GL Component	-	-	10

ومن ثم حساب الحجم النهائي للتطبيق البرمجي (NOP) بعد أخـذ مـدى تعقيـد الكائن في الاعتبار، ومن ثـم حسـاب الجهـد المطلـوب مـن المعادلـة رقـم (8) بعـد تحديد نسبة إعادة الاستخدام (Reuse)، وإنتاجية المطـور والـتي تعتمـد علـى خـبرة وإمكانيات المطور (Developer's Experience and Capability)، وكذلك تعتمد على إمكانيات الـ CASE Tools المستخدمة لدعم عملية التطوير.

2-2-8 نموذج التصميم المبكر The Early Design Model

يستخدم هذا النموذج، بعد الموافقة على متطلبات المشروع البرمجي تحت التطوير ومع بداية العمل في المراحل الأولية من تصميمه، لعمل تقدير مبدئي للجهد والوقت اللازمين لتطويره، وفيه يستخدم مجموعة جديدة من موجهات التكلفة (Cost Drivers) ومعادلات تقدير تعتمد على نقاط الدالة (FP) أو عدد أسطر الشفرة مقدراً بالكيلو (KLOC).

وتعتمد عملية التقدير في هذه المرحلة على المعادلة القياسية للنماذج الخوارزمية (المعادلة رقم (8))، وهي كما يلي:

$$E = a * Size^b * EAF \quad .....(8)$$

حيث:

- "a": ثابت ويساوي (2.94) بناءً على ما اقترحه مطور هذا النموذج (Boehm) من خلال الدراسات التي قام بها على العديد من البيانات التاريخية لمشاريع سابقة، ويستحسن أن يتم معايرته طبقاً لبيانات تاريخية خاصة بمشاريع سبق وأن تم تطويرها في نفس بيئة التطوير.
- "Size": هو حجم البرمجية ويعبر عنه بـ "KSLOC"(Thousands of Lines of Source Code)، ويتم حسابه من خلال تقدير عدد نقاط الدالة (FP) في البرمجية ومن ثم تحويلها إلى "KSLOC" باستخدام الجداول المعيارية التي تربط حجم البرنامج(KSLOC) بنقاط الدالة (FP) لعدة لغات مختلفة، والتي تم تناولها في الفصل الأول من هذه الوحدة.
- "b": هو معامل الأس (Exponent Term) يعكس مدى الزيادة المطلوبة في الجهد لمقابلة الزيادة في حجم البرمجية، وهو ليس ثابتاً كما هو الحال في نموذج COCOMO81 بل يختلف من نظام برمجي إلى آخر، وتحسب طبقاً للمعادلة التالية:

$$b = 0.91 + 0.01\sum_{i=1}^{i=5} SF_i \quad ...(9)$$

ويقدر وزن هذه المعاملات الخمسة (SF_i) على مقياس من: عالية جدا جداً إلى منخفضة جداً، مروراً بعالية جداً، وعالية، وعادية، ومنخفضة وهذه المعاملات كالتالي:

1. سابقة الأعمال (Precedentedness "PREC"): وهو مقياس يعكس الخبرة السابقة للمنشأة المنفذة للمشروع في التعامل مع مشاريع مماثلة للمشروع القائم، فإذا كانت سابقة الأعمال قليلة جداً فهذا يعني عدم وجود خبرة سابقة، وفي هذه الحالة يقدر وزن هذا المقياس بالعدد (6.2)، وإذا كانت سابقة الأعمال عالية جداً جداً فهذا يعني أن للمؤسسة باعاً طويلاً في تطوير مثل هذه المشاريع، وفي هذه الحالة يقدر وزن هذا المقياس بالعدد (0). كما هو موضح بالجدول رقم 7.9، وفي حالة وجود سابقة أعمال عالية جداً جداً فهذا يعني أن المنشأة على قدر كبير جداً من فهم وظائف ومتطلبات المنتج البرمجي، بالإضافة إلى عدم الحاجة أو حاجتها بقدر ضئيل لابتكار هياكل وخوارزميات جديدة لمعالجة البيانات.
2. مرونة التطوير (Development Flexibility "FLEX"): وهو مقياس يعكس درجة المرونة في عملية التطوير، حيث توصف عملية التطوير بعدم المرونة في حالة الحاجة إلى الالتزام الكامل من قبل الشركة المنفذة للمشروع بتنفيذ جميع مواصفات ومتطلبات المنتج البرمجي بدون أي تغيير (FLEX = 5.07)، وتوصف بالمرونة الكاملة في حالة قيام العميل بوضع الأهداف العامة للمنتج البرمجي فقط وترك التفاصيل بالكامل للشركة المنفذة (FLEX= 0).
3. دقة التصميم المعماري/تحليل المخاطر(Architecture/Risk Resolution "RESL"):وهو مقياس يعكس درجة تحليل المخاطر التي تمت أثناء التصميم المعماري للمنتج البرمجي، ويتم وضع RESL= 0 في حالة التحليل الكامل للمخاطر، و RESL = 5.65 في حالة التحليل المنخفض للمخاطر، وهكذا....، أما في حالة عدم تحليل المخاطر على الإطلاق أو المنخفض جداً فيتم وضع RESL = 7.07.
4. تماسك وتناغم الفريق (Team Cohesion "TEAM"): وهو مقياس يعكس درجة تماسك الفريق من حيث معرفتهم لبعضهم، ومدى خبرتهم في العمل كفريق متكامل، ومدى توافق أهدافهم وثقافتهم ولغاتهم، ومدى القدرة والرغبة في تعاونهم وتفاعلهم مع بعضهم الى غير ذلك. ويُعد تماسك وتناغم الفريق عالياً جداً جداً (TEAM= 0) في حالة وجود تكامل وفعالية بين فريق العمل بدون مشاكل اتصالات، ومنخفض جداً (TEAM = 5.48) في حالة عدم وجود تكامل وتفاعل وصعوبة في الاتصالات بين أعضاء الفريق، كما هو موضح بالجدول رقم (2/2/8).

5. كمال العملية Process Maturity "PMAT": وهو مقياس يعكس مدى نضج واكتمال عمليات المؤسسة، ويمكن التعامل معه طبقاً للقيم الواردة بالجدول رقم (2/2/8).

جدول رقم (2/2/8) قيم وزن المعاملات الخمسة (SF)

Scale Factors عوامل الوزن	قيم وزن المعاملات ("Factors Weight Values "SF					
	Very Low	Low	Nominal	High	Very High	Extra High
PREC	6.2	4.96	3.72	2.48	1.24	0.00
FLEX	5.07	4.05	3.04	3.03	1.01	0.00
RESL	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	7.80	6.24	4.68	3.12	1.56	0.00

- "EAF": عامل ضبط الجهد (Effort Adjustment Factor) لضبط الجهد ليتضمن تأثير موجهات التكلفة (Cost Drivers)، ويشمل هذا النموذج عدد (7) موجهات للتكلفة موضحة بالجدول 7.10 مع نظائرها من النموذج المعماري اللاحق (Post-Architecture Model)، والموضحة في الجدول أعلاه. وتقدر قيمة "EAF" من المعاداة التالية:

$$EAF = \prod_{i=1}^{i=n} EM_i .....(10)$$

حيث "EM_i" هو قيمة مضروب الجهد لموجهات التكلفة، و "n" هو عدد موجهات التكلفة وتساوي (7) لنموذج التصميم المبكر وللنموذج المعماري اللاحق.

جدول رقم (2/2/9) يوضح العلاقة بين موجهات التكلفة لنموذج كوكومو

	Early Design cost drivers	Post-Architecture cost drivers (Counterpart combined)
Product reliability and complexity	RCPX	RELY, DATA, CPLX, DOCU
Required reuse	RUSE	RUSE
Platform difficulty	PDIF	TIME, STOR, PVOL
Personnel capability	PERS	ACAP, PCAP, PCON
Personnel experience	PREX	AEXP, PEXP, LTEX
Facilities	FCIL	TOOL, SITE
Required Development Schedule	SCED	SCED

جدول رقم (2/2/10) يوضح موجهات التكلفة لنموذج كوكومو (النموذج المعماري اللاحق)

Cost Driver	Description	Rating					
		Very Low	Low	Nominal	High	Very High	Extra High
Product سمات المنتج							
RELY	Required Software Reliability	0.82	0.92	1.00	1.10	1.26	-
DATA	Database Size	-	0.90	1.00	1.14	1.28	-
CPLX	Product Complexity	0.73	0.87	1.00	1.17	1.34	1.74
RUSE	Required Reusability	-	0.95	1.00	1.07	1.15	1.24
DOCU	Documentation	0.81	0.91	1.00	1.11	1.23	-
Platform سمات منصة التشغيل							
TIME	Execution Time Constraint	-	-	1.00	1.11	1.29	1.63

Cost Driver	Description	Rating					
		Very Low	Low	Nominal	High	Very High	Extra High
STOR	Main Storage Constraint	-	-	1.00	1.05	1.17	1.46
PVOL	Platform Volatility	-	0.87	1.00	1.15	1.30	-
Personnel السمات الشخصية							
ACAP	Analyst Capability	1.50	1.22	1.00	0.83	0.67	-
PCAP	Programmer Capability	1.34	1.15	1.00	0.88	0.76	-
PCON	Personnel Continuity	1.29	1.12	1.00	0.90	0.81	-
AEXP	Applications Experience	1.22	1.10	1.00	0.88	0.81	-
PLEX (PEXP)	Platform Experience	1.19	1.09	1.00	0.91	0.85	-
LTEX	Language and Tool Experience	1.20	1.09	1.00	0.91	0.84	
Project سمات المشروع							
TOOL	Software Tools	1.17	1.09	1.00	0.90	0.78	-
SITE	Multisite Development	1.22	1.09	1.00	0.93	0.86	0.80
SCED	Development	1.4	1.14	1.00	1.00	1.00	-

Cost Driver	Description	Rating					
		Very Low	Low	Nominal	High	Very High	Extra High
	Schedule	3					

ولكن السؤال الآن كيف يتم حساب مضروب الجهـد (Effort Multiplier) لكـل موجة من موجهات التكلفة السبعة من خلال التناظر مـع موجهـات التكلفـة الخاصـة بالنموذج المعماري اللاحق: للإجابة على هذا السؤال اتبع الخطوات التالية:

1. قم بإسناد قيمة رقمية على المقياس من 1 إلى 6 إلى موجهات التكلفة الخاصـة بالنموذج المعماري اللاحق المقابلة لموجه التكلفة المراد حساب مضروب الجهد الخـاص بـه طبقـاً للتقـديرات الخاصـة بهـا: بمعنـى إذا التقـديرات الخاصـة بهـا "منخفضة جداً" تعطى القيمة الرقمية (1)، وإذا كـانت "منخفضـة" تعطـي القيمـة الرقمية (2)، .. وهكذا كما هو موضح بالجدول التالي.

منخفض جداً Very Low	منخفـض Low	عادي Nominal	عالٍ High	عالٍ جداً Very High	عـالٍ جـداً جداً Extra High
1	2	3	4	5	6

2. قم بجمع هذه التقديرات والتي من خلالها يتم تقـدير مـوجه التكلفـة المـراد حسـاب مضروب الجهد له وبالتالي معرفة مضروب الجهد من الجـدول الخـاص بـه، ونوضـح ذلك بالمثال التالي:

- قــم بحســاب مضــروب الجهــد الخــاص بمــوجه التكلفــة RCPX إذا كــان: RELY = Low،DATA = Low،CPLX = Nominal،DOCU = Nominal،Time = High،STOR = LOW،PVOL = Very High

الحــل: مــن الجــدول نــرى أن مــوجه التكلفــة RCPX يقابــل موجهــات التكلفــة DOCU ، CPLX،DATA، RELY، بإتباع الخطوات السابقة نجد أن:

1. RELY = 2،DATA = 2،CPLX = 3،DOCU = 3 .

2. مجموع التقديرات = 3 + 3 + 2 + 2 = 10
3. من الجدول الخاص بموجه التكلفة RCPX التالي نجد أن القيمـة 10 تـدل علـى أن RCPX = Low، ومضروب الجهد الخاص به هو:0.83.

الجداول الخاصة بقيم مضروب الجهد لموجهات التكلفة السبع

RCPX Cost Driver

RCPX Descriptors:							
• Sum of RELY, DATA, CPLX, DOCU Ratings	5, 6	7, 8	9 - 11	12	13 - 15	16 - 18	19 - 21
Rating Levels	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.49	0.60	0.83	1.00	1.33	1.91	2.72

PDIF Cost Driver

PDIF Descriptors:					
• Sum of TIME, STOR, and PVOL ratings	8	9	10 - 12	13 - 15	16, 17
Rating Levels	Low	Nominal	High	Very High	Extra High
Effort Multipliers	0.87	1.00	1.29	1.81	2.61

PERS Cost Driver

PERS Descriptors:							
• Sum of ACAP, PCAP, PCON Ratings	3, 4	5, 6	7, 8	9	10, 11	12, 13	14, 15
Rating Levels	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	2.12	1.62	1.26	1.00	0.83	0.63	0.50

PREX Cost Driver

PREX Descriptors:							
• Sum of APEX, PLEX, and LTEX ratings	3, 4	5, 6	7, 8	9	10, 11	12, 13	14, 15
Rating Levels	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.59	1.33	1.22	1.00	0.87	0.74	0.62

FCIL Cost Driver

FCIL Descriptors:							
• Sum of TOOL and SITE ratings	2	3	4, 5	6	7, 8	9, 10	11
Rating Levels	Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Effort Multipliers	1.43	1.30	1.10	1.0	0.87	0.73	0.62

2-2-9 نموذج إعادة الاستخدام (Reuse Model)

تُعرف الشفرة الجديدة (New Code) في نموذج COCOMOII: بأنها الشـفرة المكتوبة بطريقة يدوية بدون استخدام أي من الأدوات البرمجية (CASE Tools)، ما

عدا ذلك يُعد استخدام أي شفرة نوعاً من أنواع إعادة الاستخدام، وفـي هـذه الحالـة يستخدم نموذج إعادة الاستخدام لتقـدير الجهـد اللازم لدمـج (Integrate) الشـفرة المعادة للاستخدام (Reused Code) أو المولدة من الأدوات البرمجية المساعدة (Generated Code) مـع الشـفرة الجديـدة، ويمكـن حصـر أنـواع أو صـور إعـادة الاستخدام فيما يلي:

- الشفرة معادة الاستخدام (Reused Code): وهي شفرة موجودة مسبقاً (Pre-Existed Code) وتم استخدامها كما هي بـدون أي تغييـر وبـدون محاولـة لفهمهـا، ويطلق عليها أيضاً شفرة الصندوق الأسود (Black Box Code)، والجهد المطلوب في هذه الحالة يساوي الصفر.
- الشفرة المعاد تهيئتهـا (Adapted Code): وهـي شـفرة موجـودة مسـبقاً وتـم تعديلها لتدمج مع بـاقي المكونـات البرمجيـة للمنتـج الـبرمجي، ويطلـق عليهـا أيضـاً شفرة الصندوق الأبيض (White Box Code)، وفي هذه الحالـة يوجـد جهـد لفهـم الشفرة قبل تعديلها وجهد لتعديلها حتى تعمل بنجاح مع باقي المكونات البرمجية.
- المكونـــات التجاريـــة الجـــاهزة ("COTS" Commercial off-the-shelf Components): وهي شفرة تستخدم عن طريق التـأجير أو عـن طريـق الـترخيص باستخدامها (Leased or Licensed Source Code).
- الشفرة المعاد تحويلها آلياً (Automatically Translated Code): وهي شفرة موجودة مسـبقاً ويتـم تحويلهـا بطريقـة آليـة باسـتخدام طـرق الهندسـة العكسـية (Reengineering) باستخدام الأدوات البرمجية المؤتمتة (Automated Software Tools). وتُعد تضمين الجهد المطلوب لتنفيذ ذلك في الجهد الكلي المطلوب لتطوير البرمجية من التحسينات الإضافية في موديل COCOMOII، وهذا الجهد الإضافي يتم حسابه من المعادلة رقم (11):

$$E_{Auto} = ASLOC * (AT/100) / ATPROD \quad .....(11)$$

حيث "ATPROD": إنتاجية المطورين في دمج الشفرة المعاد تحويلها آلياً مـع بـاقي مكونات المنتج البرمجي وتساوي 2400 كما اقترحها مطـور النمـوذج (COCOMOII Manual)، و"ASLOC": حجم المكونات البرمجية (الشفرة) التي تـم تهيئتهـا (Size of Adapted Code)، و "AT" النسبة المئوية لحجم الشفرة المعاد تحويلها آلياً.

- أسطر الشفرة المكافئة (Equivalent Source Line of Code "ESLOC"): وهي أسطر الشفرة التي يتم إضافتها إلى حجم البرمجية الأصلي لتضمين الجهد المبـذول في تعديل وتضمين الشفرات المعاد استخدامها (Reused Code) أو تلك التي يعاد تهيئتها (Adapted Code)، ويمكن تقديرها باستخدام المعادلة رقم (12):

KESLOC = KASLOC * (1 – AT/100) * AAM.....(12)

حيث:"KESLOC": حجم الشـفرة المكافئـة مقاسـة بـالكيلو، وتلاحـظ نقصـان حجـم الشفرة المكافئة كلما زاد حجم الشفرة المولدة آلياً. و"AAM" يمثـل معامـل ضـبط لعملية تهيئة المكونات المعادة للاستخدام للاندماج بصورة صحيحة مع باقي مكونـات المنتج البرمجي، وتحسب من خلال المعادلة رقم (13):

AAM = (AA + SU + 0.4DM +0.3CM + 0.3IM) / 100 ...(13)

حيث إن:"AA": معامل يعكس الجهد المطلوب لاختيار وتحليل المكونـات المفـترض إعادة استخدامها، حيث الجهد المطلوب لإعـادة اسـتخدام شـفرة معينـة لا يبـدأ مـن الصفر حتى إن لم يتم إعـادة اسـتخدامها، حيـث إنـه تـم بـذل مجهـود لتقريـر مـدى صلاحية استخدامها من عدمه، وتتراوح قيمة هذا المعامـل مـن (0) إلـى (8) حسـب الجهد المطلوب في عمليتي الاختيار والتحليل.

"SU": معامل يعكس الجهد المطلوب لفهم المكونات البرمجية المعـادة للاسـتخدام وتعتمد قيمتـه علـى مـدى تعقيـد الشـفرة مـن حيـث الهيكـل (Structure)، ودرجـة وضوح التطبيق (Application Clarity)، ومدى الشرح والوصف الذاتي للشـفرة (Self-Descriptiveness)، وكذلك تعتمد علـى مـدى درايـة وألفـة المطـورين بهـذه المكونات، وتتراوح قيمة هذا المعامل من (50) للشـفرات غيـر المهيكلـة والمعقـدة إلى (10) للشفرات المكتوبة بصورة هيكلية منظمة وعلـى صـورة مكونـات برمجيـة مركبة (Strong Modularity) وموثقة توثيقاً جيداً. مـع ملاحظـة اسـتخدامها فقـط في حالة وجود مكونات قد تم إعادة استخدامها بالفعل.

"IM"،"CM"،"DM": وهي على التوالي: النسبة المئوية للتصميم المعـدل (Percent Design Modified)، النســبة المئويــة للشــفرة المعدلــة (Percent Code Modified)، والنسبة المئوية للمجهود المطلوب لدمج مكونـات إعـادة الاسـتخدام (Percent of Integration Required for Adapted Software).

2-2-10 النمــوذج المعمــاري اللاحــق: ((Post - Architecture Model COCOMOII))

يستخدم هذا النموذج نفـس معـادلات التقـديرات فـي نمـوذج التصـميم المبكـر بالإضافة إلى معادلات إعادة الاستخدام الواردة فـي نمـوذج إعـادة الاسـتخدام، هـذا بالإضافة إلى عامل جديد (REVL) لضبط الحجم الفعـال (Effective Size) للمنتـج البرمجي نتيجة تطور وعـدم ثبـات المتطلبـات Requirements Evolution and Volatility)) أثناء مراحل التطوير المختلفة، وذلك باستخدام المعادلة رقم (14):

Effective Size = Original Size (1 + REVL/100) .... (14)

حيث REVL: هو عامل لضبط الحجم نتيجة تطور وعدم ثبات المتطلبات، وهو عبـارة عن نسبة مئوية من الحجم الأصلي للمنتج البرمجي.

المعادلات التالية تمثل نموذج COCOMOII في صورته النهائية:

$$E = a * Size^b * EAF + E_{Auto} \ldots (15)$$

a = 2.94 (يجب معايرته بيانات مشاريع سابقة مطورة في نفس بيئة التطوير)

$$b = 0.91 + 0.01\sum_{i=1}^{i=5} SF_i \ldots (16)$$

$$EAF = \prod_{i=1}^{i=n} EM_i \ldots (17)$$

حيث "n = 7" لنموذج التصميم المبكر، 17 للنموذج المعماري اللاحق

E_{Auto} = ASLOC * (AT/100) / ATPROD ... (18)

Size = (KSLOC + KASLOC * (1 – AT/100) * AAM)(1+REVL/100)... (19)

2-2-11 تقدير فترة تطوير المشروع في نموذج COCOMO II:

(Schedule Estimation in COCOMO II Model):

يتم تقدير فترة تطوير المشروع البرمجي في نموذج COCOMO II في صورته المبسطة من خلال المعادلة رقم (20):

T_{DEV}= [C * ENS (D + 0.2*(B – 0.91))] * SCED% / 100...(20)

حيث:0.28 = D،3.67 = C، ولكن يجب معايرتهما طبقاً لبيانات مشاريع سبق وأن تم تطويرها فـي نفـس بيئـة التطـوير الحاليـة. أمـا عامـل الضـرب"SCED" فيمثـل النسـبة المئويـة لضـغط أو زيـادة فـترة التطـوير، و "E_{NS}" فهـو الجهـد الكلـي اللازم لعملية التطوير مـع وضـع عامـل ضـرب الجهـد "SCED" يسـاوي واحـد (Nominal Value) أثناء تقديره .

الفصل الثالث
منهجية البحث

3-1-1 المنهج المتبع في البحث:

اتبع الباحث المنهج الوصفي التحليلي وهو يتبع في البحـوث الـتي تعتمـد علـى المنهج الوصفي في تفسير الوضع القائم للظـاهرة أو المشـكلة، مـن خلال ظروفهـا وأبعادها وتوصيف العلاقـات بينهـا بهـدف الانتهـاء إلـى وصـف علمـي دقيـق متكامـل للظاهرة أو المشكلة بالاعتماد على الحقائق المرتبطة بها[12].

3-1-2 عينة البحث:

عينة عشوائية من بيوتات البرمجيـات ومراكـز المعلومـات العاملـة علـى بنـاء البرمجيات داخل ولاية الخرطوم تتكون من (31) مشروع و تم تقدير هذا بنـاء علـى تقدير عدد المشاريع بالولاية بجملـة 100 مشـروع. وقـد اخترنـا أسـواق البرمجيـات الصغيرة لعدة أسباب

1- ان السوق السودانية تعد واحدة من هذه الأسواق
2- ان قاعدة بيانات النموذج في الغالب مبنية على مشـاريع قادمـة مـن الأسـواق الاوربية والامريكية وهذه الأسواق تعمل في ظروف ومستوى جودة مختلف

وقد اخترنا نموذج الـ COCOMO2 لأسباب الاتية

1- نموذج واسع الانتشار في مجال هندسة البرمجيات
2- نموذج يعتمد على تمثيل الخبرات والمعلومات السابقة

3-1-4 الأدوات:

الاستبيان وهو: مجموعـة مـن الأسـئلة المتنوعـة والـتي ترتبـط بعضـها ببعـض بشكل يحقق الهدف الذي يسعى إليه الباحث من خلال المشكلة التي يطرحها بحثـه، ويرسـل الاسـتبيان بالبريـد أو بـأي طريقـة أخـرى إلـى مجموعـة مـن الأفـراد أو المؤسسات التي اختارها الباحث لبحثـه لكـي يتـم تعبئتهـا ثـم إعادتهـا للبـاحث.[11] وتحتوي الاستبانة الـتي تـم اسـتعمالها فـي البحـث علـى (26) سـؤال موزعـة علـى خمسة محاور وتم توزيعها حسب الإطار المبين. واعتمد الاستبيان المصمم للدراسـة على استبيان صمم من قبل (مركز هندسة البرمجيـات بجامعـة جنـوب كاليفورنيـا) [14] والذي يستخدم لجمع البيانات التي تغذي قاعـدة معلومـات نمـوذج COCOMO وقد اختار الباحث من الاستبيان ما يتوافق مـع الأسـواق الصـغيرة بالسـودان. وذلـك على النحو التالي:

1/ حجـم البرنامـج مقـدراً بالـ Function Points وذلـك حـتى يتناسـب مـع المطلوب ليعطي درجة أعلى من الدقة في القياس ويتناسـب مـع مسـتوى التوثيـق المتاح في العينة.

2/ تصـنيف الشـركة (المؤسسـة) المعتمـدة للبرنامـج علـى حسـب: (خـبرة المؤسسة مع البرامج المشابهة، المعمارية والمخاطر، مـدى نضـج عمليـة التطـوير، مرونة التطوير، تماسك فريق العمل).

3/ المنهج البرمجي.

4/ فريق العمل.

5/ منصة العمل.

6/ المشروع.

7/ الصيانة وهنا نشير إلى فترة الضـمان مـا بعـد الـتركيب لأنهـا تكـون ضـمن تكلفة المنتج.

8/ تكلفة المشروع.

والملاحظة وهي: المشاهدة والمراقبة الدقيقة لسلوك ما أو ظاهرة معينة في ظل ظروف وعوامل بيئة معينة بغرض الحصول على معلومات دقيقة لتشخيص هـذا السلوك أو هذه الظاهرة [13].

3-1-6 حدود البحث:

حدود زمانية: الفترة ما بين 2009-2014م التي نفذت فيها المشروعات.

حدود مكانية: بيوتات البرمجيات ومراكز المعلومات العاملة في بناء البرمجيات داخل ولاية الخرطوم

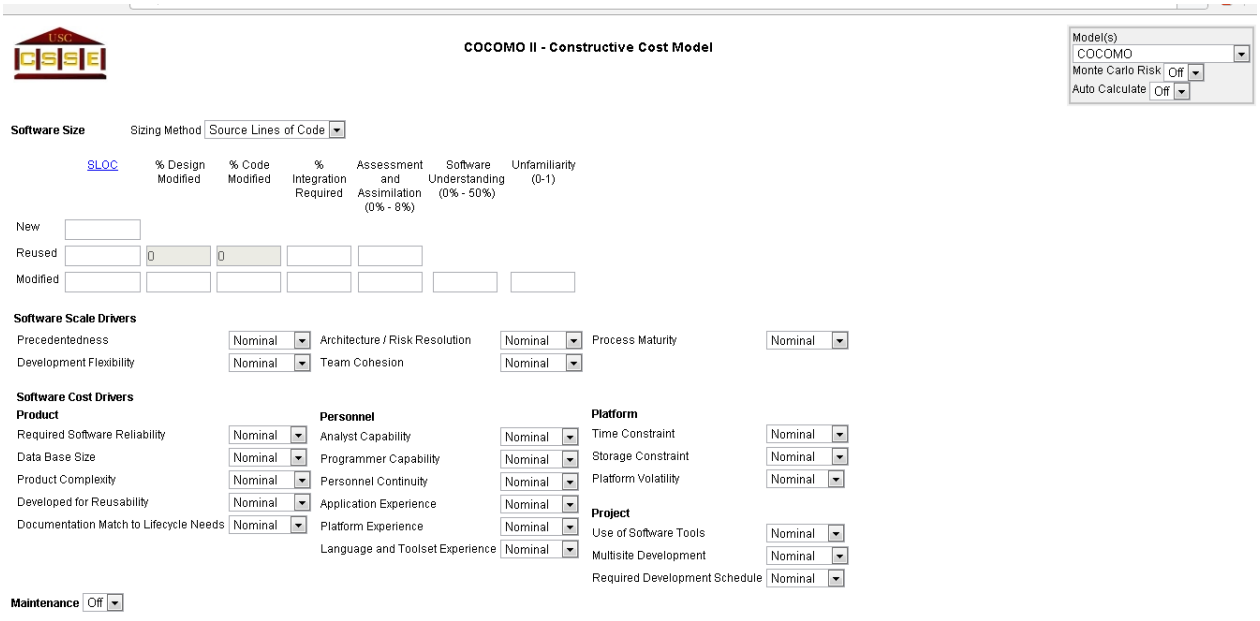
3-1-7 طريقة جمع المعلومات:

البيانات الأولية: استخدم الباحث الاستبانة لجمع البيانات الأولية من المبحوثين.

البيانات الثانوية: من الكتب والمراجع العلمية والدوريات المحكمة ذات الصلة بالموضوع.

3-1-8 طريقة تحليل البيانات:

حسبت تقديرات المشاريع بواسطة نموذج COCOMO2 بواسطة تطبيق المصمم من قبل جامعة جنوب كاليفورنيا مركز النظم وهندسة البرمجيات وهي الجهة المتبنية النموذج .[10] [9] استخدم الباحث برنامج الحزمة الإحصائية (SPSS) لتحليل البيانات، وكذلك معدلات النموذج للحصول على النتائج وتقييم الانحراف بين النموذج والسوق السودانية.



شكل رقم (3/1) شكل توضيحي لتطبيق حساب تقدير التكلفة وفق نموذج COCOMO2

الفصل الرابع

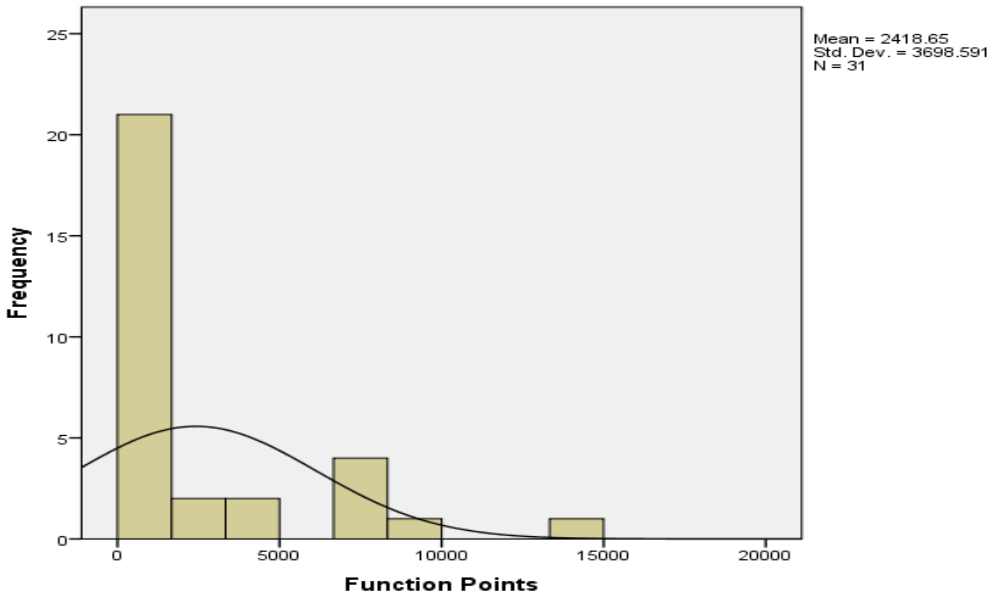
دارسة الحالة السوق السودانية

نتائج دراسة استبانة نموذج COCOMO2

جدول رقم(4/1): يوضح توزيع حجم البرنامج Function Points

Mean	StdDev	Min	Max
2419	**3698.591**	**25**	**15000**

يتضح من النتائج أعلاه أن متوسط حجـم البرنامـج بلـغ 2419 نقطـة وظيفيـة مما يعني أن هذه البرامج في عينة الدراسة ذات وظائف متوسطة مما يعكس حجـم هذه البرمجيات ومدي دورها في أعمال الشركة، حيث نجـد أن أقـل برنامـج بـه 25 وظيفـة وأعلـى برنامـج بـه 15000 وظيفـة ممـا يعكـس التبـاين العـالي للبرامـج والوظائف التي به.



شكل رقم(4/1): يوضح توزيع حجم البرنامج Function Points

جدول رقم(4/2): يوضح توزيع لغة التطوير المستخدمة

اللغة	العدد	النسبة
Basic	1	3%
C	5	17%
Java	1	3%
3rd G	4	45%
اخري	10	32%

تشير النتائج أن أكثر اللغات المستخدمة في التطوير اللغة بنسبة بلغـت 45% وهي أعلى نسبة بينما نجد 32% يستخدمون لغات أخرى (وهي تشـير الـى مشـاريع هجين تضمنت أكثر من لغة برمجة) و17% يسـتخدمون لغـة C ممـا يعكـس اختلاف نوعية لغة التطوير في عينة الدراسة.

شكل رقم(4/2): يوضح توزيع لغة التطوير المستخدمة

جدول رقم(4/3): يوضح وصف تصنيف الشركة المنفذة للبرنامج

Software Scale

رقم العبارة	العبارة	درجة الموافقة							المتوسط الحسابي	الانحراف المعياري	الرتبة
		النسبة	منخفض جدا	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع			
1	Precedentedness	ك	4	8	8	7	2	2	3.65	1.518	2
		%	13%	26%	26 %	23 %	6%	6%			
2	المعمارية /المخاطر	ك	2	8	3	7	8	3	3.10	1.399	5
		%	6%	26%	10 %	23 %	26 %	10%			
3	مدي نضج عمليات التطوير	ك	4	5	15	5	2	2	3.81	1.376	1
		%	13%	16%	42 %	16 %	6%	6%			
4	مرونة التطوير	ك	1	4	8	10	4	4	3.13	1.500	4
		%	3%	13%	26 %	32 %	13 %	13%			
5	تماسك فريق العمل	ك	8	2	5	10	6	-	3.61	955.	3
		%	26%	6%	16 %	32 %	19 %	-			
المتوسط العام									3.46	1.35	

من خلال النتائج الموضحة في الجدول رقم (3) يتضح أن أفراد عينة الدراسة موافقون بدرجة عادي على تصنيف الشركة المنفذة للبرنامج وذلك بمتوسط (3.46 من 6) وهو متوسط يقع في الفئة الثالثة من فئات المقياس السداسي من (من 2.66 وحتى 4.44) وهي الفئة التي تشير إلى خيار (عادي) على أداة الدراسة. كما يتضح أيضاً أن هناك تجانس في موافقة أفراد عينة الدراسة على أن تصنيف الشركة المنفذة للبرنامج بأنه عادي حيث تراوحت متوسطات موافقتهم ما بين (3,10 إلى 3,81)، وهي متوسطات تقع في الفئة الثالثة من فئات المقياس السداسي والتي تشير إلى (عادي) مما يوضح التجانس في موافقة أفراد عينة الدراسة ومن أهم تلك العبارات وتتمثل في العبارات:

- جاءت العبارة رقم (3) وهي "**مدي نضج عمليات التطوير**" بالمرتبة الأولى من حيث موافقة أفراد عينة الدراسة عليها بمتوسط (3,81 من 6,00).

- جاءت العبـارة رقـم (1) وهـي "Precedentedness" بالمرتبـة الثانيـة مـن حيث موافقة أفراد عينة الدراسة عليها بمتوسط (3,65 من 6,00).
- جاءت العبارة رقم (5) وهي **"تماسك فريق العمل"** بالمرتبـة الثالثـة مـن حيث موافقة أفراد عينة الدراسة عليها بمتوسط (3.61 من 6.00).

كـل هـذه النتائـج توضـح ان تصـنيف الشـركات الـتي قـامت ببرمجـة هـذه البرمجيات في عينـة الدراسـة تصـنيف عـادي مـن حيـث المعماريـة ونضـج البرامـج وتماسك فريق العمل وغيرها من تصنيفات

شكل رقم(4/3): يوضح وصف تصنيف الشركة المنفذة للبرنامج

Software Scale

ويعزو الباحث تلك النتائج الي ان الظروف العامة المحيطة بـبيوت البرمجيـات قريبة من بعضها البعض و تقاربها في المفاهيم و تشاركهم في نفس الكادر البشري في كثير من الاحيان

جدول رقم(4/4): يوضح وصف المنتج Product

رقم العبارة	العبارة		درجة الموافقة							المتوسط الحسابي	الانحراف المعياري	الرتبة
			النسبة	منخفض جدا	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع			
1	مدي الموثوقية المطلوبة في البرمجية المراد تطويرها	ك	-	3	13	8	7	-	2.81	1.25	4
		%	-	10%	42 %	26 %	23 %	-			
2	حجم قاعدة البيانات	ك	5	9	7	7	3	-	2.87	1.33	3
		%	16%	29%	23 %	23 %	10 %	-			
3	ما هو مدي تعقيد البرمجة المطلوبة	ك	6	7	7	7	4	-	2.90	1.37	2
		%	19%	23%	23 %	23 %	13 %	-			
4	حجم البرمجيات المعدة لا عادة الاستخدام	ك	5	9	5	10	-	2	2.52	1.50	5
		%	16%	29%	16 %	32 %	-	6%			
5	حجم التوثيق المطلوب	ك	12	5	4	6	4	-	3.42	1.17	1
		%	38%	16%	13 %	19 %	13 %	-			
المتوسط العام									2.90	1.33	

من خلال النتائج الموضحة في الجدول رقم (3) يتضح أن أفراد عينة الدراسة موافقون بدرجة عادي على عناصر وصف المنتج وذلك بمتوسط (2,90 من 6) وهو

متوسط يقع في الفئة الثالثة من فئات المقياس السداسي من (من 2,66 وحتى 4,44) وهي الفئة التي تشير إلى خيار (عادي) على أداة الدراسة.

كما يتضح أيضاً أن هناك تفاوت في موافقة أفراد عينة الدراسة على أن عناصر وصف المنتج بأنه عادي،حيث تراوحت متوسطات موافقتهم ما بين (2,52 إلى 3,42)، وهي متوسطات تقع في الفئات الثانية والثالثة من فئات المقياس السداسي والتي تشيران إلى (منخفض وعادي) على التوالي مما يوضح التفاوت في موافقة أفراد عينة الدراسة ومن أهم تلك العبارات وتتمثل في العبارات:

- جاءت العبارة رقم (5) وهي **"حجم التوثيق المطلوب"** بالمرتبة الأولى من حيث موافقة أفراد عينة الدراسة عليها بمتوسط (3.42 من 6.00).
- جاءت العبارة رقم (1) وهي "**ما هو مدي تعقيد البرمجة المطلوبة**" بالمرتبة الثانية من حيث موافقة أفراد عينة الدراسة عليها بمتوسط (2.90 من 6.00).

شكل رقم(4/4): يوضح وصف المنتج Product

جدول رقم(4/5): يوضح الموظفين Personnel

رقم العبارة	العبارة	درجة الموافقة							المتوسط الحسابي	الانحراف المعياري	الرتبة
		النسبة	منخفض جدا	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع			
1	مستوي قدرة محللي النظم	ك	3	3	8	12	5	-	3.42	1.17	2
		%	10%	10%	26 %	39 %	16 %	-			
2	مستوي قدرة المبرمجين	ك	-	1	14	11	5	-	3.65	798.	1
		%	-	3%	45 %	36 %	16 %	-			
3	ما احتمال استمرارية الموظفين	ك	11	10	3	6	1	-	2.23	1.23	5
		%	36%	32%	10 %	19 %	3%	-			
4	مستوي خبرة فريق العمل مع نوع التطبيق	ك	2	6	13	7	3	-	3.10	1.04	4
		%	6%	19%	42 %	23 %	10 %	-			
5	خبرة فريق العمل مع منصة التطبيق	ك	1	3	11	14	2	-	3.42	886.	3
		%	3%	10%	36 %	45 %	6%	-			
المتوسط العام									3.16	1.03	

من خلال النتائج الموضحة في الجدول رقم (5) يتضح أن أفراد عينة الدراسة موافقون بدرجة عادي على عناصر وصف الموظفين وذلك بمتوسط (3,16 من 6) وهو متوسط يقع في الفئة الثالثة من فئات المقياس السداسي من (من 2.66 وحتى 4.44) وهي الفئة التي تشير إلى خيار (عادي) على أداة الدراسة.

كما يتضح أيضاً أن هناك تفاوت في موافقة أفراد عينة الدراسة على أن عناصر وصف المنتج بأنه عادي حيث تراوحت متوسطات موافقتهم ما بين (2,23

إلى 3,65)، وهي متوسطات تقـع فـي الفئـات الثانيـة والثالثـة مـن فئـات المقيـاس السداسي والتي تشيران إلى (منخفض وعادي) على التوالي مما يوضح التفاوت في موافقة أفراد عينة الدراسة ومن أهم تلك العبارات وتتمثل في العبارات:

- جاءت العبارة رقم (5) وهي **"مستوي قدرة المبرمجين"** بالمرتبة الأولى من حيث موافقة أفراد عينة الدراسة عليها بمتوسط (3,65 من 6,00).
- جاءت العبارة رقم (1) وهي **"مستوي قدرة محللي النظم"** بالمرتبـة الثانيـة من حيث موافقة أفراد عينة الدراسة عليها بمتوسط (3,42 من 6,00).

شكل رقم(4/5): يوضح الموظفين Personnel

جدول رقم(4/6): يوضح منصة العمل Platform

رقم العبارة	العبارة	درجة الموافقة							المتوسط الحسابي	الانحراف المعياري	الرتبة
		النسبة	منخفض جدا	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع			
1	مدي ملاءمة الوقت المشروع مع المشروع	ك	15	8	5	3	-	-	1.87	1.02	3
		%	48%	26%	16%	10%	-	-			
2	مستوي الشروط المفروضة على تخزين البيانات	ك	15	7	6	3	-	-	1.90	1.04	2
		%	48%	23%	19%	10%	-	-			
3	مستوي استقرار منصة العمل	ك	15	5	9	2	-	-	1.94	1.03	1
		%	48%	16%	29%	6%	-	-			
المتوسط العام									1.90	1.03	

من خلال النتائج الموضحة في الجدول رقم (6) يتضح أن أفراد عينة الدراسة موافقون بدرجة منخفض على عناصر وصف منصة العمل وذلك بمتوسط (1.90 من 6) وهو متوسط يقع في الفئة الثانية من فئات المقياس السداسي من (من 1.84 وحتى 2.68) وهي الفئة التي تشير إلى خيار (منخفض) على أداة الدراسة.

كما يتضح أيضاً أن هناك تجانس في موافقة أفراد عينة الدراسة على ان على عناصر وصف منصة العمل بانة منخفض ،حيث تراوحت متوسطات موافقتهم ما بين (1.87 إلى 1.94)، وهي متوسطات تقع في الفيئة الثانية من فئات المقياس

السداسي والتي تشير إلى (منخفض) مما يوضح التجانس في موافقة أفـراد عينـة الدراسة .

جدول رقم(4/7): يوضح بيانات المشروع Project

رقم العبارة	العبارة	درجة الموافقة							المتوسط الحسابي	الانحراف المعياري	الرتبة
		النسبة	منخفض جدا	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع			
1	مدي استخدام وجودة أدوات بناء البرمجيات	ك	3	4	10	10	4	-	3.26	1.15	1
		%	10%	13%	32 %	32 %	13 %	-			
2	العمل علي موقع متعددة	ك	7	3	10	6	5	-	2.97	1.37	2
		%	23%	10%	32 %	19 %	16 %	-			
3	هل جدول العمل مطلوب وماهوا مستوي الالتزام به	ك	10	7	6	4	4	-	2.52	1.41	3
		%	32%	23%	19 %	13 %	13 %	-			
المتوسط العام									2.91	1.31	

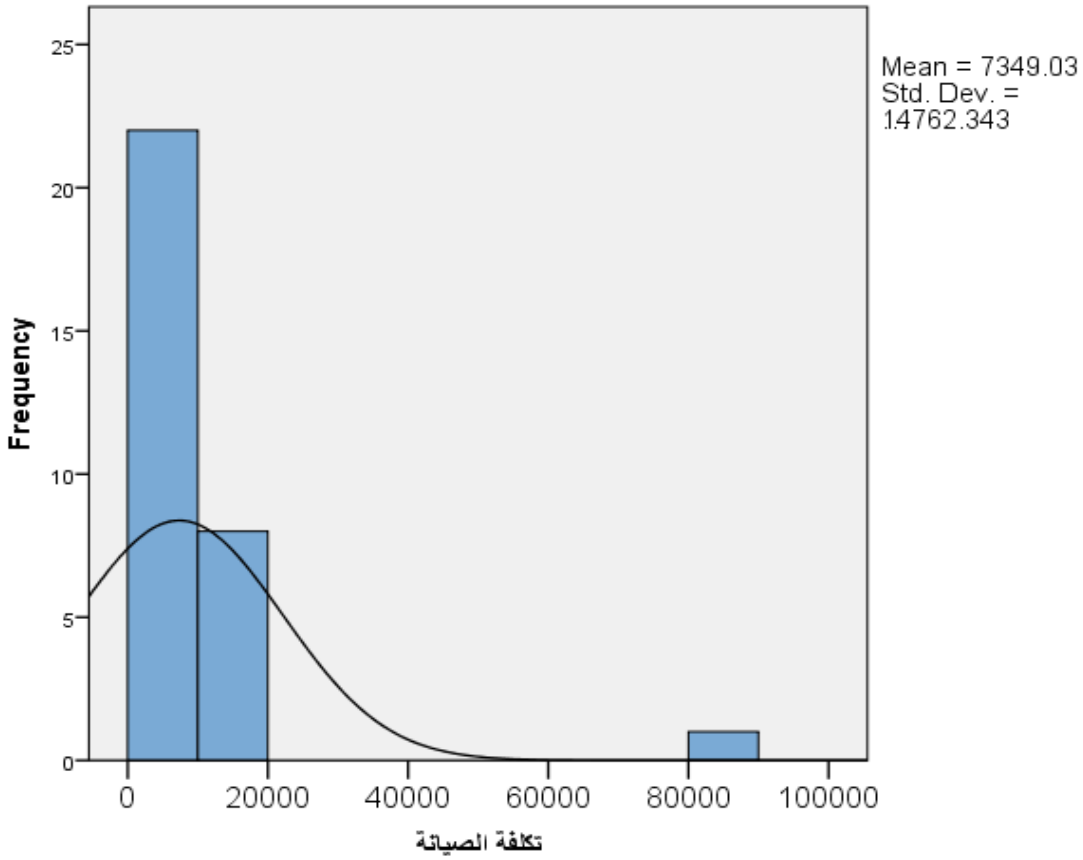
من خلال النتائج الموضحة في الجدول رقم (7) يتضح أن أفراد عينة الدراسة موافقون بدرجة عادي على عناصر وصف بيانات المشروع وذلك بمتوسط (2.91 من 6) وهو متوسط يقع في الفئة الثالثة من فئات المقياس السداسي من (من 2.66 وحتى 4.44) وهي الفئة التي تشير إلى خيار (عادي) على أداة الدراسة.

كما يتضح أيضاً أن هناك تجانس في موافقة أفراد عينة الدراسة على ان عناصر بيانات المشروع بانه عادي ،حيث تراوحت متوسطات موافقتهم ما بين (2.52 إلى 3.26)، وهي متوسطات تقع في الفئة الثالثة من فئات المقياس السداسي والتي تشير إلى (عادي) مما يوضح التجانس في موافقة أفراد عينة الدراسة

جدول رقم(4/8): يوضح توزيع تكلفة الصيانة Maintenance

Std-dev	Mean	Max	Min
14762.3	**7349**	**82000**	**0**

تشير نتائج التحليل في الجدول أعلاه إلى أن متوسط تكلفة الصيانة في عينـة الدراسة مقاسـة بشـخص واحـد بلغـت 7349 بـالجنيه السـوداني بـانحراف معيـاري **14762.3** وبهذا يظهر تشتت عالي في متوسط التكلفة حيث نجد أقل تكلفة بلغت صفر وأعلى تكلفة 82000.



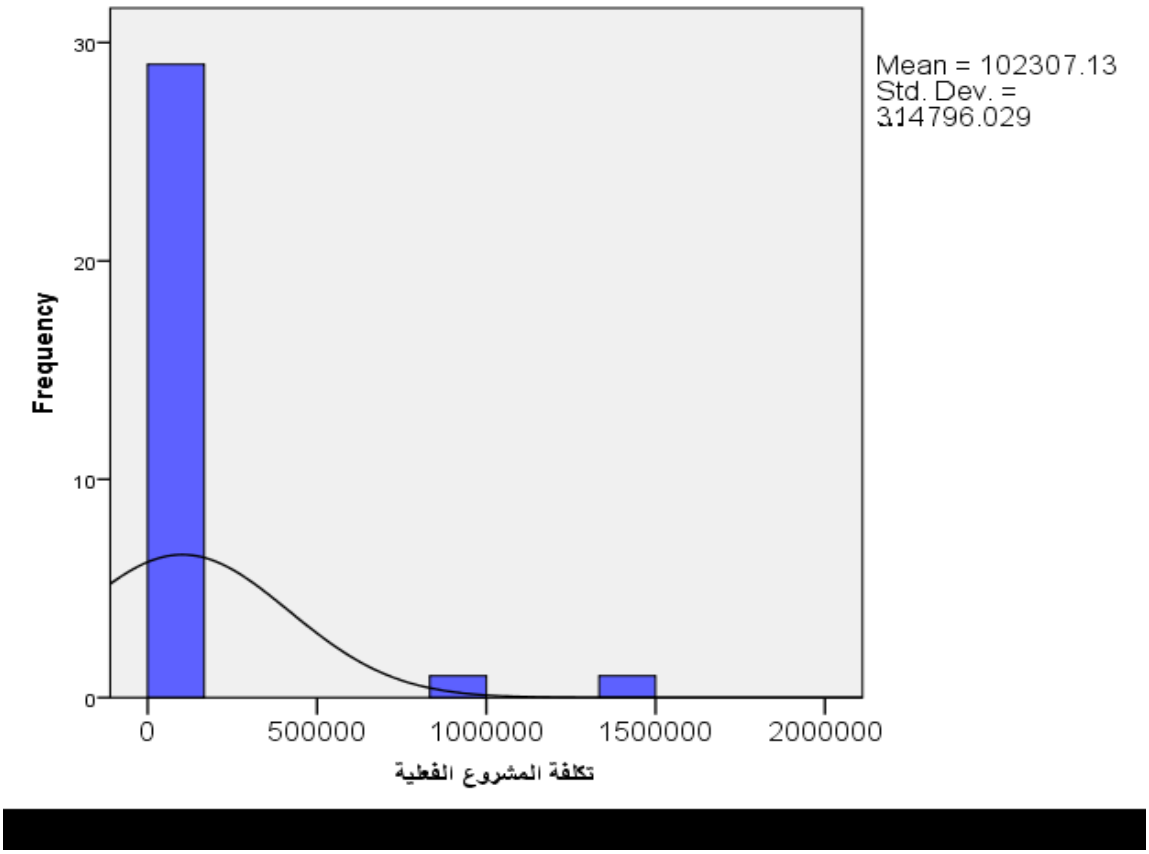
شكل رقم(4/6): يوضح توزيع تكلفة الصيانة Maintenance

جدول رقم(4/9): يوضح تكلفة المشروع الفعلية (بالدولار)

Std-dev	Mean	Max	Min

314,666$	102,747$	1,500,000$	200$

تشير نتائج التحليل فـي الجـدول اعلاه ان متوسـط تكلفـة المشـروع الفعليـة بلغت 102,747$ وبانحراف معياري 314,666$ مما يعكس تباين تكلفة المشاريع في عينـة الدراسـة حيـث نجـد اقـل مشـروع بلغـت تكلفتـه 200$ وأعلـى مشـروع 1,500,000$.



شكل رقم(4/7): يوضح تكلفة المشروع الفعلية (بالدولار)

جدول رقم(4/10): يوضح إحصائية تكلفة المشروع في نموذج COCOM2والتكلفة الفعلية بالدولار في عينة الدراسة والانحراف بين النموذجين

	Min	Max	Mean	Std-dev
Cost Model	425$	$ 32,270,964	2,167,539$	$ 6,419,622
التكلفة الفعلية	200$	1,500,000$	102,747$	314,666$
الانحراف	-20,252$	30,770,964 $	2,064,792$	6,197,159$

تشـير نتائـج التحليـل فـي الجـدول اعلاه ان متوسـط التكلفـة فـي النمـوذج COCOM2 بلغت 2,167,539، بينما نجـد متوسـط التكلفـة الفعليـة للمشـروع فـي عينة الدراسة بلغت 102,747 وهذا يعني وجود انحراف عالي جدا حيث بلغ متوسط الانحراف 2,064,792

وبهـذا تكـون نسـبة الانحـراف بلغـت 95% بيـن عينـة الدراسـة والنمـوذج COCOM2

شكل رقم(4/8): يوضح إحصائية تكلفة المشروع في نموذج COCOM2والتكلفة الفعلية بالدولار في عينة الدراسة والانحراف بين النموذجين

النتائج والنقاش

والختام خرج البحث بنتائج على النحو التالي:

1/ وجدت الدراسة أن نموذج COCOMO2 ورغم الثقة الكبيرة التي يحظى بها عالمياً إلى أنه لا يصلح لتقدير تكلفة بناء وتطوير البرمجيات للسوق السودانية والتي يعتمد تقدير التكلفة فيها على أسلوب التقدير للفوز بالعطاء متجاهلين التكلفة الفعلية للبرمجيات.

2/ ونجد أن أغلب مؤشرات الدارسة تشير إلى الوسط أو دون الوسط في هذا يعطي مؤشر عن جودة صناعة البرمجيات في السودان.

3/ الانحراف الكبير بين تكلفة البرمجية المنتجة بالسودان والتقدير العالمي يوضح المسافة بينهما.

الخاتمة والتوصيات

1/ نوصي بدارسة أكبر من حيث العينة والرقعة الجغرافية لتشمل ولايـات السـودان المختلفة او المدن الكبرى

2/ ان تشمل الدراسة بقية جوانب نموذج COCOMO2 التي لـم تتنـاول فـي هـذه الدراسة

3/راعية مثل هذه الدراسات و تنفيذها من خلال فريق عمل

مسرد المصطلحات

المصطلح بالإنجليزية	معناه بالعربية
4GLs	**لغات الجيل الرابع**
Advanced	**متقدم**
Algorithmic Estimations Models	**نماذج التقدير الخوارزمية**
Analog-Based	**معتمد على التناظر**
Application Composition Model	**نموذج تكوين التطبيق**
Architecture/Risk Resolution " RESL "	**دقة التصميم المعماري/تحليل المخاطر**
Automated Software Tools	**الأدوات البرمجية المؤتمتة**
Automatically Translated Code	**الشيفرة المعاد تحويلها آلياً**
AVC	**المتوسط الحسابي لعدد السطور للغة معينة**
Average Stuffing	**متوسط عدد فريق العمل**
Backup	**نسخ احتياطي**
Black Box Code	**شيفرة الصندوق الأسود**
Blank Lines	**السطور الخالية**
CASE Tools and Programs Generators	**هندسة البرمجيات بمساعدة الكمبيوتر ومولدات البرامج**
Client-server Applications	**تطبيقات خادم العميل**
COCOMO Model	**ونموذج Boehm**
Cohesion Team	**تفاهم فريق التطوير**
Comments Lines	**سطور التعليقات**
Commercial Cost Models	**نماذج تقدير التكلفة التجارية**
Commercial off-the-shelf "COTS" Components	**المكونات التجارية الجاهزة**

Components-Based Software Engineering	**هندسة البرمجيات المبنية على المكونات**
Constraints and Priorities	**القيود والأوليات**
Constructiveness	**البناء الاستدلالي**
Control Signals	**إشارات التحكم**
Conversion	**التحويل**
Convertible Function Points to	**يمكن تحويله إلى مقياس نقاط الوظيفة**
Cost	**التكلفة**
Crucial	**حرجه**
Data Import / Export Capability	**إمكانية تصدير واستيراد البيانات**
Data Sharing Capability	**إمكانية مشاركة البيانات**
Demo Copy	**نسخة تجريبية**
Design-Oriented	**وفقاً للتصميم**
Developer's Experience and Capability	**خبرة وإمكانيات المطور**
Development Flexibility "FLEX"	**مرونة التطوير**
Development Process Maturity	**نضج عمليات التطوير**
Early Design Model	**نموذج التصميم المبكر**
Ease of Use	**سهولة الاستخدام**
Easy of Use and Documentation	**سهولة الاستخدام والتوثيق**
Effort	**الجهد**
Effort Adjustment Factor	**عامل ضبط الجهد**
Effort Adjustment Factor "EAF"	**عامل ضبط للجهد**
Embedded	**مضَّمن**
Empirical-Based Model	**نماذج على أساس تجريبي**

Equivalent Source Line of Code "ESLOC"	**أسطر الشيفرة المكافئة**
Estimation Models Supported	**نماذج التقدير المدعومة**
Evaluating of Cost Estimation Models	**تقويم نماذج تقدير التكلفة**
Excellent Rated Program	**حالة التخطيط الممتاز**
Expandable to Source	**يمكن تحويله إلى مقياس سطور شيفرة المصدر**
Exponent Term	**معامل الأس**
External Interface Files	**ملفات المواجهة الخارجية**
Feature Points	**نقاط الميزة**
Fidelity	**الدقة**
Fine Tune Calibration	**معايرة دقيقة**
Forms	**النماذج**
Function Points "FP"	**نقاط الوظيفة**
Function-oriented	**المبني على الدوال**
Good Rated Program	**حالة التخطيط الجيد**
Hard Line Break	**بداية سطر جديد**
Hardcopy	**المطبوعة على الأوراق**
Historical Data	**بيانات تاريخية**
Historical data analysis	**التحليل التاريخي للبيانات**
Immediate Input	**دخل فوري**
Immediate Outputs	**خرج فوري**
Influence Multiplier	**مضروب التأثير**
Initial Product Definition Stage)	**مرحلة التعريف الأولى للمنتج**
Inquiries	**الاستفسارات**
Installation	**التثبيت**

Intermediate	**متوسط**
Intuition	**البديهة**
Iterative Steps	**الخطوات التكرارية**
Language Dependent	**معتمد على لغة التطوير**
Language Independent	**غير معتمد على لغة التطوير**
Linear Programming	**البرمجة الخطية**
Little Innovation	**قليل من الابتكار**
Logical Internal File	**الملفات الداخلية المنطقية**
Logical Lines	**عدد السطور المنطقية**
Magnitude of Relative Error	**قيمة الخطأ النسبي**
Mainframe System –Based Applications	**التطبيقات المبنية على نظم الحاسبات المركزية**
Management Information Systems "MISs"	**نظم إدارة المعلومات**
Mean Magnitude of Relative Error "MMRE"	**متوسط قيمة الخطأ النسبي**
Medium Constrain for Deadline	**قيود متوسطة على وقت التسليم**
Medium Innovation	**ابتكار متوسط**
Methods of Software Size Estimation	**طرق تقدير حجم البرمجية**
Module	**وحدة البرمجة المركبة**
New Code	**الشيفرة الجديدة**
No Tight Deadline	**قيود صارمة على وقت التسليم**
Object Complexity	**تعقيد الكائن**
Object Points	**نقاط الكائن**
Objectivity	**الموضوعية**
Object-oriented	**البرمجة الكينونية**

Off Shelf Components	**المكونات المتناولة**
OOPL	**اللغات الكائنية المنحى**
Optimistic Estimates	**التقديرات المتفائلة**
Organic	**مترابط**
Other Cost Drivers	**مسببات أخرى للتكلفة**
Overall Staff Distribution over Development Time	**توزيع عدد فريق العمل في أي وقت خلال تطوير المشروع**
Over-Estimating Cost	**التقدير الأعلى من التكلفة الفعلية**
Parsimony	**التدبير**
Percent of Integration Required for Adapted Software	**النسبة المئوية للمجهود المطلوب لدمج مكونات إعادة الاستخدام**
Person-Years	**الجهد الكلي مقيس بشخص ـ سنة**
Pessimistic Estimates	**والتقديرات المتشائمة**
Physical Format	**الشكل الفيزيائي**
Physical Lines	**عدد السطور الفيزيائية**
Platform and Performance	**منصة التشغيل والأداء**
Poor Rated Program	**حالة التخطيط الردئ**
Post-Architecture Model	**النموذج المعماري اللاحق**
Precedentedness "PREC"	**سابقة الأعمال**
Privileges	**الصلاحيات**
Process Maturity "PMAT"	**كمال العملية**
Productivity	**الإنتاجية**
Project Key Milestones	**المعالم الأساسية للمشروع**
Prospectiveness	**التوقع المستقبلي**
Putnam (SLIM) Model	**نموذج Putnam**
Recovery	**استعاده دوريه يمكن الوثوق بها**
Re-Engineering	**الهندسة العكسية**

Regression Analysis	**التحليل الانحداري**
Requirements Evolution and Volatility	**تطور وعدم ثبات المتطلبات**
Requirements Specification	**مرحلة إعداد مواصفات المنتج**
Resources	**الموارد**
Reusable Components	**المكونات المعادة للاستخدام**
Reuse Model	**نموذج إعادة الاستخدام**
Risk Management Techniques	**منهجيات إدارة المخاطر**
Schedules	**الجداول الزمنية**
Scope	**الأفق**
Self-Descriptiveness	**الوصف الذاتي**
Semicolons	**الفصلات المنقوطة**
Semi-Detached	**شبه مترابط**
Sets of Open-Closed Braces	**مجموعات أزواج الأقواس**
Single Flat File	**ملف مستقل**
Softcopy	**نسخ على وسائط التخزين المختلفة**
Software Functionality	**وظائفية البرمجية**
Software Instructions	**عدد التعليمات البرمجية**
Software Projects Cost Estimation	**تقدير تكلفة المشروعات البرمجية**
Software Projects Cost Estimation Tools	**أدوات تقدير تكلفة المشروعات البرمجية**
Source Lines-of-Code "SLOC"	**عدد سطور شيفرة المصدر**
Specification Based	**معتمد على المواصفات**
Stability	**الاستقرار**

Stable Development Environment	**بيئة تطوير مستقرة**
Staffing	**فريق التطوير**
Standard Deviation	**انحراف معياري**
Statistical Simulation	**المحاكاة الإحصائية**
Subjective	**موضوعية**
Synthetic Measure	**مقياس تركيبي**
System Level	**مستوى النظام**
Team Cohesion "TEAM"	**تماسك وتناغم الفريق**
Technical Support	**الدعم الفني**
Theory–Based Models	**نماذج على أساس نظري**
Total Cost	**التكاليف الكلية**
Types of Algorithmic Models	**أنواع النماذج الخوارزمية**
UFC (Unadjusted Function-Point Count)	**ونقاط الدالة غير المعدلة**
Under-Estimating Cost	**التقدير الأقل من التكلفة الفعلية**
Upgrade Acceptability	**قابلية التحديث**
User-Oriented	**طبقاً للمستخدم**
Variations as a Function of Counting Conventions	**يتغير كدالة في اصطلاحات العد**
Variations as a Function of Languages	**يتغير كدالة في اللغات**
Web-Based Applications	**التطبيقات المعتمدة على الويب**
Weight	**وزن**
What-if Analysis	**ماذا إذا؟**
White Box Code	**شيفرة الصندوق الأبيض**

المراجع

المراجع العربية:

1/ روجر بريسمان، "هندسة البرمجيات"، ترجمة مركز التعريب والترجمة بالدار العربية للعلوم، الطبعة الأولى، 2004م.

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

المراجع الأجنبية:

3. B. W. Boehm, Software Engineering Economics, Englewood Cliffs, NJ: Prentice-Hall, 1981.
4. Ian Somerville , "Software Engineering", Addison Wesley, 2001.
5. Ronald J. Leach, "Introduction to Software Engineering", CRC Press, 1999.Douglas Bell , "Software Engineering: A Programming Approach", 3rd Edition, Addison Wesley.
6. Steven C. McConnel, "Estimation", Chapter 8, Rapid Development, Microsoft Press, 1996 ,www.construx.com/stevemcc.
7. Kim Johnso, "Software Cost Estimation: Metrics and Models" , Department of Computer Science , University of Calgary , Alberta , CANADA T2N 1N4. http://sern.ucalgary.ca/courses/seng/621/W98/johnsonk/cost.htm
8. COCOMO II Model Definition Manual , Version 2.1 , 1995-2000 Center For Software Engineering , USC.

مواقع الانترنت

9. http://csse.usc.edu
10. http://csse.usc.edu/tools/COCOMOII.php
11. http://en.wikipedia.org/wiki/Questionnaire
12. http://ar.wikipedia.org/wiki/بحث_وصفي
13. **http://sunset.usc.edu/publications/TECHRPTS/2000/usccse2000-505/usccse2000-505.pdf ,**

"Software Development Cost Estimation Approaches - A Survey"

14. **www.comp.lancs.ac.uk/computing/resources/IanS/SE7/SampleChapters/ch26.pdf ,**

" **Software cost estimation**"

www.classes.cecs.ucf.edu/eel6883/ 15
berrios/slides2/CH7-art-3-4-5.pp ,

" **Software Cost Estimation** "

- **COCOMO II web page can be found at :**

  sunset.usc.edu/research/COCOMOII/index.html

- **USC COCOMO II.1999.0 Software (Implementations of Post-architecture and Early Design models) :**

  http://sunset.usc.edu/research/COCOMOII/index.html

الملاحق

استبانة

نموذج COCOMOII للسوق السودانية

حجم البرنامج : مقدراً بالـ function Points .....

ما هي لغة التطوير المستخدمة

Basic	C	DBMS	java	PERL	3rd G	أخرى

تصنيف الشركة المنفذة للبرنامج Software Scale Drivers

وهو مقياس يعكس الخبرة السابقة للمنشأة المنفذة للمشروع في التعامل مع مشاريع مماثلة للمشروع القائم Precedentedness

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

المعمارية: المخاطر Architecture / Risk Resolution

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

مدى نضج عمليات التطوير Process Maturity

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

مرونة التطوير Development Flexibility

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

تماسك فريق العمل Team Cohesion

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

المنتج Product

مدى الموثوقية المطلوبة في البرمجية المراد تطويرها Required Software Reliabilit

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً

حجم قاعدة البيانات Data Base Size

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً

ما هو مدى تعقيد البرمجية المطلوبة Product Complexity

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

حجم البرمجيات المعدة لإعادة الاستخدام Developed for Reusability

منخفض جداُ	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

حجم التوثيق المطلوب Documentation Match to Lifecycle Needs

منخفض	منخفض	عادي	مرتفع	مرتفع جداً

جداً				

الموظفين Personnel

مستوى قدرة محللي النظم Analyst Capability

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

مستوى قدرة المبرمجين Programmer Capability

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

ما احتمال استمرارية الموظفين Personnel Continuity

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

مستوى خبرة فريق العمل مع نوع التطبيق Application Experience

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

خبرة فريق العمل مع منصة التطبيق Platform Experience

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

منصة العمل Platform

مدى ملاءمة الوقت المتاح للمشروع مع المشروع Time Constraint

عادي	مرتفع	مرتفع جداً	شديد الارتفاع

مستوى الشروط المفروضة على تخزين البينات Storage Constraint

عادي	مرتفع	مرتفع جداً	شديد الارتفاع

مستوى استقرار منصة العمل Platform Volatility

منخفض	عادي	مرتفع	مرتفع جداً

المشروع Project

مدى استخدام وجودة أدوات بناء البرمجيات Use of Software Tools

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

العمل على موقع متعددة Multisite Development

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً	شديد الارتفاع

هل جدول العمل مطلوب وما هو مستوى الالتزام به Required Development Schedule

منخفض جداً	منخفض	عادي	مرتفع	مرتفع جداً

الصيانة Maintenance

تكلفة الجهد المطلوب مقاسة بشخص – شهر (بالدولار)Cost per Person-Month (Dollars)) .....
....

تكلفة المشروع الفعلية (بالدولار).....