مرجع المحاضرة : كتاب هندسة البرمجيات – تأليف د.أحمد شعبان دسوقي – أ.د.السيد محمود الربيعي

محتويات الوحدة

الموضوع	الصفحة
مقدمة	75
تمهيد	75
أهداف الوحدة	76
1. نموذج الشلال (Waterfall Model)	77
2. نموذج النمذجة الأولية (Prototyping Model)	80
3. نموذج التطوير السريع للتطبيقات (Rapid Application Development "RAD" Model)	86
4. النموذج التزايدي (Incremental Model)	89
5. النموذج الحلزوني (Spiral Model)	91
6. نموذج الطرق المنهجية (Formal Methods Model)	98
الخلاصة	104
لمحة مسبقة عن الوحدة التالية	105
إجابات التدريبات	106
مسرد المصطلحات	109
المراجع	111

المقدمة

تمهيد

عزيزي الدارس، أهلاً بك إلى الوحدة الثالثة من مقرر " هندسة البرمجيات(1)" وهذه الوحدة تتناول نماذج عمليات تطوير البرمجيات.

في هذه الوحدة سنتعرف على عدد من النماذج وهي:

نموذج الشلال أو "التتابع الخطي أو "دورة الحياة التقليدية" وسوف نتابع مع هذا النموذج المراحل التي مرت بنا في الوحدة السابقة ، ثم نوضح أهم مميزات هذا النموذج وكذلك العيوب التي يتصف بها.

نموذج النمذجة الأولية : وهنا سنتعرف على فكرة هذا النموذج ومتى يتم اللجوء إليه ، وسنتعرف على كيفية التغلب على بعض المشاكل التي تجعل هذا النموذج منهجاً فعالاً في هندسة البرمجيات ، وسنحدد أهم فوائد هذا النموذج.

نموذج التطوير السريع للتطبيقات: سنتعرف على الفكرة التي يعتمد عليها هذا النموذج وأهم العقبات التي تقف أمام استخدام هذا النموذج ، كما سنوضح في هذا القسم من الوحدة أهم نشاطات هذا النموذج.

النموذج التزايدي: يجمع هذا النموذج عناصر نموذج الشلال التتابعية بصورة تكرارية أي أنه يقوم بتنفيذ عناصر نموذج الشلال ، وسوف نتعرف على فوائد هذا النموذج.

النموذج الحلزوني:وهو من النماذج الارتقائية ، وسنعرض هنا بعض الأمثلة لكوارث خلال مشاريع تطوير البرمجيات ، وسنتعرف على أهم مميزات هذا النموذج ، كما سنتعرف أيضاً على الوثيقة التي تصف بشكل عام غير مفصل كيف يجب أن يعمل النظام "وثيقة مفهوم العمليات" .

نموذج الطرق المنهجية: ويشمل هذا النموذج مجموعة الأنشطة التي تمكِّن فريق التطوير من وصف متطلبات النظام والتحقق من صحة عملة عن طريق تدوين رياضي دقيق، وسوف نتناول أهم مميزات هذا النموذج وسلبياته.

تجد عزيزي الدارس فى ثنايا الوحدة بعض التدريبات التي يساعدك تنفيذها على فهم محتوى المادة هذا بالإضافة إلى أسئلة التقويم الذاتي التي تهدف إلى ترسيخ الفهم وتعزيزه لديك و تعميقه .

أهداف الوحدة

عزيزي الدارس،

بنهاية دراسة هذه الوحدة ينبغي أن تكون قادراً على أن :

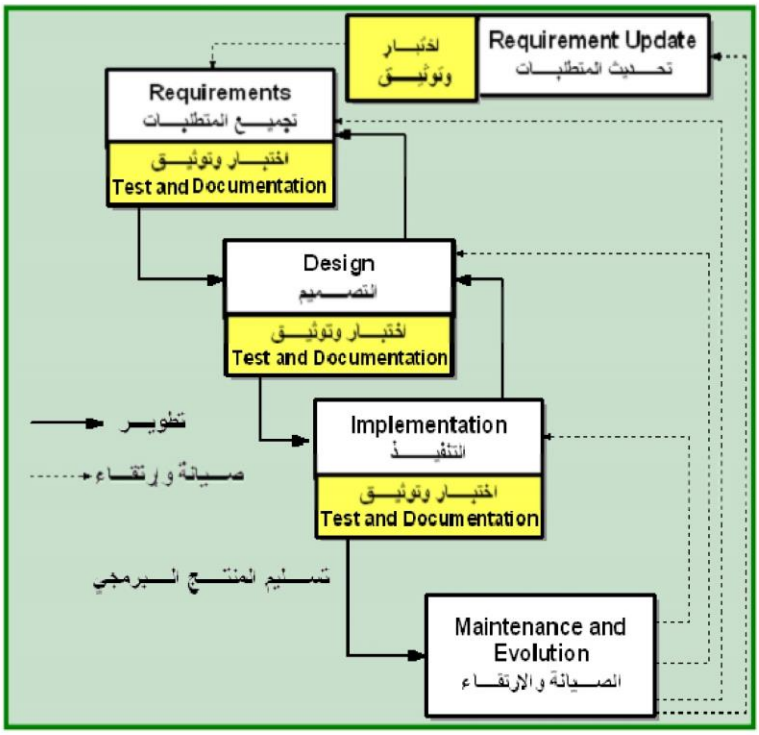
- تعبر عن أهمية استخدام نماذج تطوير عمليات البرمجيات لتطوير البرمجيات.
- تحدد الفروق المختلفة بين نماذج تطوير عمليات البرمجيات.
- تختار النموذج المناسب لتطوير منتج برمجي طبقاً لمعطيات هذا المنتج.
- تطبق المبادئ الهندسية الصحيحة على عملية تطوير البرمجيات.
- تصف الأنشطة الخاصة بكل نموذج من النماذج التي تم تناولها وكيفية تنفيذها.
- تعدد أهم مميزات وعيوبه كل نموذج من النماذج التي تم تناولها وعيوبه.
- تحدد متى وكيف يستخدم كل نموذج من النماذج التي تم تناولها .
- تشرح أهم النماذج المستخدمة في عملية تطوير البرمجيات.
- تسمي أهم التحديات الرئيسية التي تواجه عملية تطوير البرمجيات.
- تعدد المخاطر الرئيسية التي تواجه عملية تطوير البرمجيات.

توطئة

نموذج عمليات تطوير البرمجيات هو مجرد وصف أو تصور بسيط (إستراتيجية تطوير) لتمثيل أنشطة تطوير عمليات البرمجيات والتي تقدم من وجهة نظر محددة ، بحيث تؤدي في النهاية إلى إنتاج منتج برمجي كامل قابل للتنفيذ, وسوف نتناول في الأجزاء التالية أهم هذه النماذج بطريقة تساعدك عزيزي القارئ على عمل مقارنة بينها ، ومن ثم اختيار المناسب منها بناءاً على نوعية متطلبات المشروع البرمجي المطلوب تطويره ، وكذلك القيود المفروضة عليه.

1. نموذج الشلال (Waterfall Model)

يقترح **نموذج الشلال** . والذي يسمى أحياناً " بالنموذج التتابعي الخطي " ، وأحياناً أخرى " بنموذج دورة الحياة التقليدية " ـ منهجاً تتابعياً منظماً لتطوير البرمجيات ، حيث يبدأ من مرحلة " تحليل المتطلبات " وينتهي بمرحلة " الصيانة والارتقاء " مروراً بباقي المراحل التقليدية لدورة حياة البرمجيات التي ذكرناه سابقاً. وكما هو موضح بالشكل رقم 3.1 ، يوفر نموذج الشلال الأصلي ، الذي أقترحه **Winston Royce** ، حلقات تغذية رجعية ، إلا أن غالبية المؤسسات البرمجية التي تطبق هذا النموذج تعامله وكأنه تتابعي خطي تفادياً للمشاكل والعقبات التي سوف تواجههم في حالة أخذ نواتج حلقات التغذية الرجعية في الحسبان أثناء عمليات التطويرو يتكون نموذج الشلال من النشاطات العامة التي تم تتناولها سابقاً ، وهي ، كما هو موضح بالشكل رقم 3.1 ، كما يلي :



شكل 3.1 : رسم توضيحي لنموذج الشلال لتطوير البرمجيات

مرحلة تجميع المتطلبات : وكما تم توضيحه سابقاً ، فالغرض منها هو : توثيق متطلبات النظام والبرنامج ومراجعتها مع العميل قبل بداية عملية التصميم.

مرحلة التصميم : في هذه المرحلة ، وكما تم توضيحه سابقاً ، يتم تحويل وثيقة توصيف متطلبات العميل إلى وثيقة توصيف التصميم ، والتي تتضمن . في الغالب . بنية البيانات (Data Structure) ، وبنيان البرمجية (Software Architecture) ، وتفاصيل الخوارزميات (Algorithmic Detail) التي سوف تستخدم في عملية التشفير ،

وواجهة الربط البنية مع المستخدم (User Interface). ويتم تقويم وثيقة توصيف التصميم والتأكد من أنها تتطابق مع معايير الجودة المطلوبة وخالية من الأخطاء قبل البدء في مرحلة التنفيذ, وفي هذه المرحلة . أيضاً . يتم تحديد متطلبات الأجهزة التي تتناسب مع توصيف البرمجية التي سوف يتم تنفيذها.

مرحلة التنفيذ : في هذه المرحلة ، وكما تم توضيحه سابقاً ، يتم تحويل وثيقة توصيف التصميم إلى منتج برمجي متكامل قابل للتنفيذ.

مرحلة الاختبار وتشخيص الأخطاء و في هذه المرحلة ، وكما تم توضيحه سابقاً ، يتم اختبار البرمجية حال توليد الشيفرة البرمجية والتأكد من خلو البرمجية من أخطاء منطقية العلاقات الداخلية ، وكذلك الخارجية ، أي ضمان أن الدخل المعرف سيعطي نتائج فعلية تتوافق مع النتائج المطلوبة.

مرحلة الاختبار والارتقاء و في هذه المرحلة ، وكما تم توضيحه سابقاً ، يتم صيانة وتغيير بعض من أجزاء النظام البرمجي تجاوباً مع المتغيرات في متطلبات العميل.

يتميز نموذج الشلال بالبساطة في الفهم والإدارة حيث إنه من السهل على المطوّر توضيح كيفية سير العمل في المشروع البرمجي للعميل ، وكذلك يتميز بأن عمليات الاختبار **(Test)** وعملية التوثيق **(Documentation)** تتم بصفة ملازمة مع تنفيذ كل مرحلة من مراحل تطوير البرمجية, ولكن وعلى الرغم من أن نموذج الشلال هو الأكثر استخداماً في هندسة البرمجيات إلا أن لهذا النموذج العديد من العيوب منها :

- يتطلب هذا النموذج تحديد متطلبات العميل بكل دقة وأن تكون طريقة الحل واضحة ومفهومة وسبق تحليل النظام بالكامل قبل البدء في عملية التصميم ، حيث إن هذا النموذج يفتقر إلى عمليات التكرار والإعادة, ولهذا تبرز صعوبات عند استخدام هذا النموذج مع

الأنظمة الضخمة لعدم التحديد الدقيق للمتطلبات والتي تُعد ظاهرة عامة في مثل هذه الأنظمة.

- تأخر صدور النسخة العاملة من البرمجية حتى وقت متأخر من الجدول الزمني للمشروع.
- صعوبة إجراء التعديلات في المراحل الأولية في حالة تقدم المشروع إلى المراحل النهائية ، ويكون الخطأ شبه كارثة إذ لم يُكتشف إلا عند مراجعة المنتج النهائي من البرمجية.
- اعتماد المراحل (النشاطات) المختلفة على بعضها البعض يؤدي إلى حالات انتظار بين بعض من أعضاء فريق المشروع ، بمعني انتظار بعض من أعضاء فريق المشروع وجلوسهم بدون عمل حتى يُنهي البعض الآخر مهام مترابطة وضرورة لهم لبدء عملهم, مما يؤدي ذلك إلى تأخر تسليم المشروع.

أسئلة تقويم ذاتي

اذكر أهم مميزات وعيوب نموذج الشلال لتطوير البرمجيات.

2. نموذج النمذجة الأولية (Prototyping Model) :

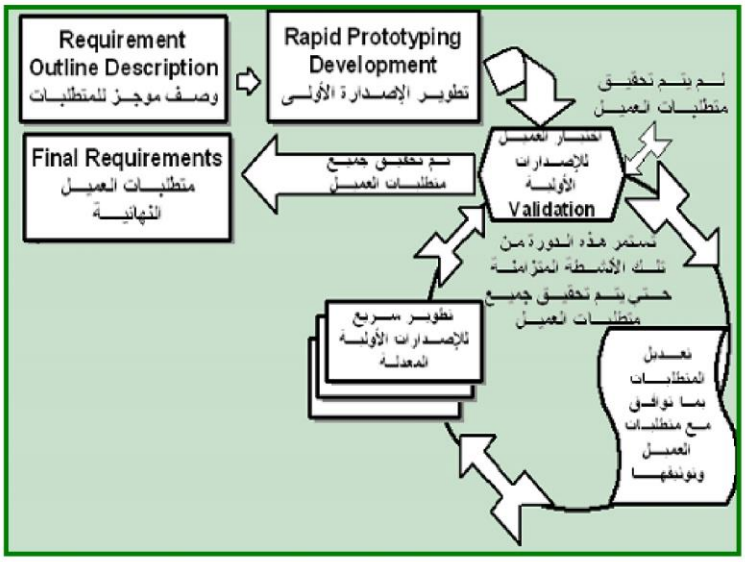
يعتمد فكرة هذا النموذج على التطوير السريع (Rapid Development) لنماذج تحاكي شكلياً البرنامج المطلوب أثناء فترة تحديد متطلبات العميل للتأكد من جدوى البرنامج وفهم متطلباته. إن الفرق الوحيد بين النماذج الأصلية والنظام المرغوب فيه هو كفاءة النماذج وكمية المعلومات التي تتعامل بها.

ويتم اللجوء إلى هذا النموذج عندما لا تتوافر متطلبات كافية ومحددة عند العميل لبناء البرنامج ولكنه يعرف مجموعة من الأهداف والخطوط العامة للبرنامج (Outline Specifications), أي أن هذا النموذج يستخدم . مثالياً . كآلية لتحديد متطلبات البرنامج قبل البدء في تطوير الإصدارة النهائية من البرنامج و في هذه الحالة يطلق على هذا

الأسلوب : "أسلوب النمذجة الأولية الملقاة جانباً " **(Throw-Away Prototyping Technique)** ، حيث إنه يتم إلقاء أخر نموذج من النماذج الأولية جانباً بعد تحقيق الهدف المنشود وهو الحصول على المتطلبات النهائية للعميل في صورة مواصفات واضحة المعالم.

كما هو واضح من الشكل 3.2 التالي ، يبدأ التطوير السريع للإصدارة الأولى الأولية **(First Prototype Version)** بناءاً على وصف موجز لمتطلبات العميل المعروفة لديه ، حيث يركز التطوير السريع على تمثيل نواحي محددة من البرنامج وخاصة تلك التي تكون مرئية للعميل ، ومن ثم يتم إعادة الإصدارة الأولى الأولية إلى العميل لتقويمها ولإضافة متطلبات إضافية ، ومن ثم تتم دورة تطوير الإصدارات الأولية المعدلة **(Intermediate Prototype Versions)** لتتواءم مع متطلبات العميل الإضافية وتقويمها من العميل وتحديث المتطلبات مرة أخرى بإضافة متطلبات العميل الجديدة إليها

ويستمر تكرار هذه العملية حتى يتم تحقيق جميع متطلبات العميل ويرضى عن الأداء الوظيفي لإحدى الإصدارات الأولية, ومن ثم يتم بناء البرنامج لتحقيق المتطلبات النهائية للعميل بناء على الدورة التقليدية للبرمجيات والتي تم شرحها سابقاً, وتُعد سرعة وتكلفة بناء النماذج الأولية, أمرا هاما في هذا الأسلوب, تحقيق السرعة العالية والتكلفة القليلة لبناء هذه النماذج يتم بتجاهل متطلبات الجودة للمنتج النهائي.



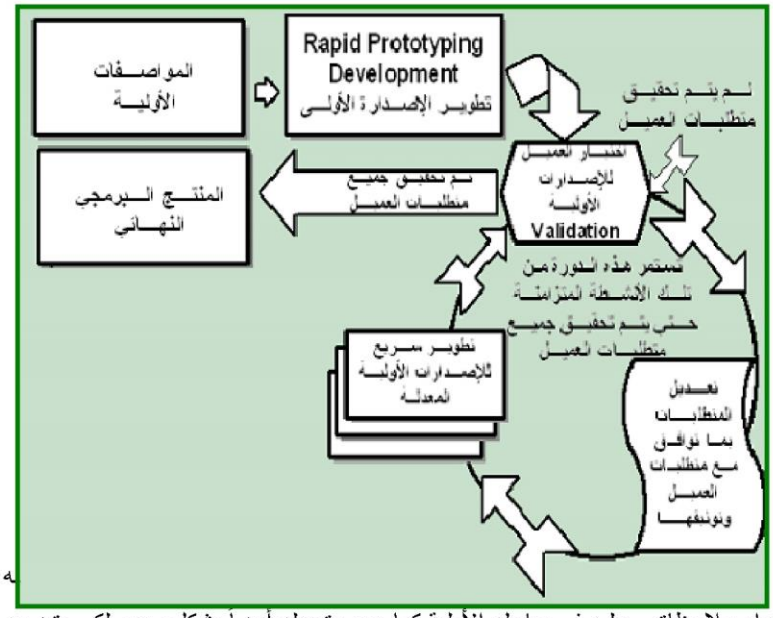
شكل 3.2 : شكل توضيحي لنموذج النمذجة الأولية لتطوير البرمجيات

تطلبات العميل النهائيه لا يتم الاستفادة من الإصدارات الاوليه التي تم تطويرها ، بل إنها ترمى جانباً ، وذلك في رأي معظم المتخصصين في هندسة البرمجيات يرجع إلى أن الإصدارات الأولية يتم تطويرها بسرعة بدون الأخذ في الاعتبار مقاييس الجودة ، حيث إن معظم المطوّرين يقوم ببعض التجاوزات أثناء عملية التنفيذ لجعل الإصدارة تعمل بسرعة ، مثل استخدام نظام تشغيل أو لغة برمجة غير مناسبين فقط لأنهما متوفران لديهم ويُحسنون استخدامهما ، وقد يعتمدون خوارزميات غير كافية مما يتسبب في أن يكون البرنامج بطيئاً جداً ، أو صعب الاستخدام ، أو يحتاج إلى ذاكرة كبيرة لتشغيله ، وخلافه, وكذلك ورغم أن استخدام نموذج النمذجة الأولية في تطوير البرمجيات يعطي للعميل فكرة عامة وجيدة عن النظام الفعلي ، لكن يمكن أن تحدث بعض التجاوزات من العميل أو من إدارة تطوير البرمجيات (المطوّرين) ، فمثلاً يمكن أن يصر العميل على إجراء بعض التعديلات

البسيطة على الإصدارات الأولية لكي يستخدمها كمنتج نهائي بالرغم من أنها لا تتوافق مع مقاييس الجودة المطلوبة ، وكذلك لا تتوافق مع معايير الصيانة على المدى البعيد ، وفي كثير من الأحيان تستجيب إدارة تطوير البرمجيات على ذلك, أما من ناحية المطوّرين فقد يقنعون أنفسهم بصلاحية الخيارات التي استخدموها في تنفيذ الإصدارات الأولية وينسى أنها غير مناسبة وبالتالي يكون المنتج البرمجي ضعيف في النواحي التي استخدمت فيها تلك الخيارات.

لذا ، وللتغلب على هذه المشاكل ولجعل نموذج النمذجة الأولية منهجاً فعالاً في هندسة البرمجيات يجب أن يتفق العميل مع مؤسسة تطوير البرمجيات على أن الإصدارات الأولية من البرنامج تستخدم فقط كآلية لتحديد المتطلبات الفعلية ، ثم ترمى جانباً وتبنى البرمجيات الفعلية مع مراعاة مقاييس الجودة ومعايير الصيانة على المدى البعيد.

ويوجد أسلوب أخر من استخدام النمذجة الأولية في تطوير البرمجيات يطلق عليه نموذج " **أسلوب النمذجة الأولية الارتقائية**" **(Evolutionary Prototyping Technique)** ، وهو يشبه في شكله الخارجي "أسلوب النمذجة الأولية الملقاة جانباً" ، ولكنه يختلف عنه في آلية التنفيذ أثناء عملية تكرار تطوير النماذج الأولية ، حيث إن عملية التطوير تتم من خلال إطار عمل نظامي **(Systematic Framework)** تبدأ بعمل تحليل مبدئي لمتطلبات العميل للحصول على مواصفات أولية **(Initial Specification)** يتم على أساسها بناء أول النماذج الأولية بطريقة نظامية يراعي فيها مقاييس الجودة والأداء ، ومن ثم تتم دورة تطوير الإصدارات الأولية المعدلة **(Intermediate Prototype Versions)** لتتواءم مع متطلبات العميل الإضافية وتقيمها من العميل وتحديث المتطلبات مرة أخرى بإضافة متطلبات العميل الجديدة إليها ، ويستمر تكرار هذه العملية حتى يتم تحقيق جميع متطلبات العميل ويرضى عن الأداء الوظيفي لإحدى الإصدارات الأولية ، والتي تُعد في هذه الحالة المنتج النهائي ، كما هو موضح بالشكل رقم 3.3.



شكل 3.3 : شكل توضيحي لأسلوب النمذجة الأولية الارتقائية في تطوير البرمجيات.

بطريقة أسرع من الطرق المعتادة ؟ ، يمكن أن يتم ذلك باستخدام عدد من الوسائل والأساليب الملائمة مثل : تكنولوجيا الجيل الرابع **(Fourth Generation Technology "4GT")** ، والأجزاء القابلة للاستخدام مرة أخرى ، حيث إنه يمكن تخفيض الوقت المطلوب لتطوير البرنامج إذا كان من الممكن إعادة استخدام أجزاء عديدة من ذلك النظام بدلاً من الأجزاء المصممة والمنفذة, كما يمكن بناء النماذج الأولية بسرعة في حالة وجود مكتبة للمكونات التي يمكن إعادة استعمالها ، وبعض الآليات الأخرى التي تساعد على دمج هذه الأجزاء في الأنظمة. وقد تستخدم

الأجزاء التي من الممكن إعادة استخدامها أيضاً في تكوين النظام النهائي ، وبهذه الكيفية يمكن تقليل تكلفة تطوير البرنامج.

- **ولكن ما هي فوائد نموذج النمذجة الأولية بصفة عامة؟** : يمكن القول إن من أهم فوائد هذا النموذج أنه أثناء تحديد متطلبات العميل يستطيع المطور أن يُطلع العميل على نظام العمل المقترح في مرحلة مبكرة جداً ، وعادةً ما يكون العملاء غير متأكدين دائماً مما يريدون من النظام أن يفعل ، ومن الصعب بالنسبة لهم توضيح المتطلبات الوظيفية بالتفصيل وبصورة واضحة قبل أن تتاح لهم الفرصة لتجربة الخيارات بشكل فعال, بالإضافة إلى أن النموذج يقدم للعميل صورة واضحة للشكل المتخيل للنظام والعمليات التي سيقوم بتنفيذها, وبواسطة فحص الخيارات بالنسبة للنسخ المختلفة من النموذج ، يشجع العملاء على اكتشاف المتطلبات التي ربما لا يكونوا قد فكروا فيها إلا بعد التنفيذ الكامل للنظام, وتكمن القيمة في الاتصال المثالي بين العميل والمحلل في استطاعة العملاء أن يخبروا المطور بآرائهم في النظام وتنفيذ التعديلات المطلوبة والتأكد من صلاحية النظام في وقت مبكر من العمر الافتراضي للمشروع, وهكذا يزيل النموذج عدداً كبيراً من أخطاء التصميم في مراحل مبكرةً جداً من المشروع, ويعزي توفير الوقت والجهد إلى تجنب العمل في تغيير النظام الذي لا ينفذ ما يطلبه المستخدم فعلياً, بالإضافة إلى إنه عندما يكون العملاء مشتركين في عملية التطوير ،غالباً ما يكسبون ثقة النظام, وقد يشاهدون أولاً المشاكل والأخطاء ، لكن يمكنهم أيضاً مشاهدة الأخطاء وهي تعالج بسرعة.

تدريب(1)

لخص أهم فوائد نموذج النمذجة الأولية بصفة عامة

أسئلة تقويم ذاتي

اذكر أهم عيوب أسلوب النمذجة الملقاة جانبا.
ما هي أهم الفروق الأساسية بين أسلوب النمذجة الأولية الملقاة جانبا وأسلوب النمذجة الأولية الارتقائية؟.
لخص أهم فوائد نموذج النمذجة الأولية بصفة عامة.

3. نموذج التطوير السريع للتطبيقات

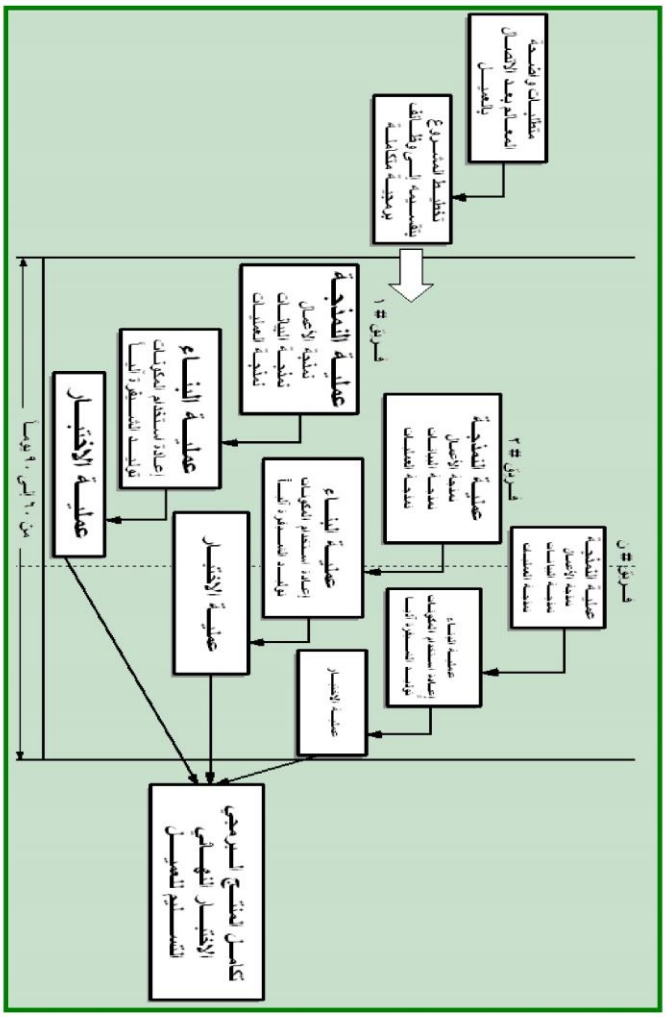
(Rapid Application Development "RAD" Model) :

يعتمد نموذج التطوير السريع للتطبيقات على تقسيم المشروع البرمجي إلى عدة وظائف برمجية رئيسية يتم تناول كل منها بواسطة فريق عمل مستقل بحيث تسمح عملية التقسيم لكل فريق بتطوير الوظيفة الخاصة به كنظام يعمل تماماً **(Fully Functional System)** خلال فترة قصيرة تتراوح من (60) إلى (90) يوماً ، ومن ثم تتكامل هذه الوظائف لتشكل كياناً واحداً ، كما موضح بالشكل رقم 3.4.

وللقيود المفروضة على زمن تطوير المشروع يُعد مبدأ إعادة استخدام المكونات البرمجية القابلة لإعادة الاستخدام ، وخصوصاً تلك التي تم اختبارها من قبل ، وكذلك تقنيات هندسة البرمجيات بمساعدة الحاسب **(CASE)** في عملية تنفيذ البرمجيات هو حجر الزاوية في نموذج التطوير السريع للتطبيقات ، هذا بالإضافة إلى عملية تقسيم المشروع إلى وظائف برمجية رئيسية كما تم ذكره.

ولاستخدام نموذج التطوير السريع للتطبيقات في تطوير الأنظمة البرمجية يجب أن تكون متطلبات العميل مفهومة جيداً ، بالإضافة إلى إمكانية تقسيم المشروع بشكل مناسب إلى وظائف برمجية رئيسية مستقلة, وفي حالة المشاريع كبيرة الحجم يتطلب نموذج التطوير السريع للتطبيقات موارد بشرية كافية لتكوين العدد المطلوب من فرق العمل, ويُعد

ما تم ذكره من أهم العقبات التي تقف أمام استخدام نموذج التطوير السريع للتطبيقات في بناء النظم البرمجية.



شكل 3.4 شكل توضيحي لنموذج التطوير السريع للتطبيقات

وتُعد تطبيقات نظم المعلومات من أهم التطبيقات التي يمكن استخدام نموذج التطوير السريع للتطبيقات في تطويرها لكونها تتميز باستقلالية بعض من وظائفها ، بالإضافة إلى إمكانية فهم وتحديد متطلبات العميل على الوجه الأمثل. والشكل رقم 3.4 يوضح أهم نشاطات نموذج التطوير السريع للتطبيقات, لتطوير تطبيقات نظم المعلومات ، وهي كمايلي :

- عملية النمذجة **(Modeling)** : وتشمل نمذجة الأعمال **(Business Modeling)** ، ونمذجة البيانات **(Data Modeling)** ، ونمذجة العمليات **(Processes Modeling)** ، وذلك لتصميم النظام من حيث تدفق الأعمال والبيانات **(Work and Data flow)** والعلاقة المترابطة بينهما لتحقيق وظائف الأعمال المطلوبة من النظام.

- عملية بناء البرمجية : وتشمل : عملية توليد التطبيق **(Application Generation)** ، باستخدام مبدأ إعادة استخدام المكونات **(Component Reuse)** ، وكذلك استخدام الأدوات المؤتمته مثل تقنيات **CASE** لتوليد الشيفرة البرمجية آلياً **(Automatic Code Generation)** مثل مطوّر ومصمم أوراكل **(Oracle Developer and Designer)** ، وذلك بدون الحاجة إلى استخدام لغات البرمجة التقليدية.

- عملية الاختبار : وتشمل : اختبار كامل البرمجية وخصوصاً المكونات الجديدة منها ، وكذلك تجربة كل الواجهات (الشاشات) تجريباً كاملاً.

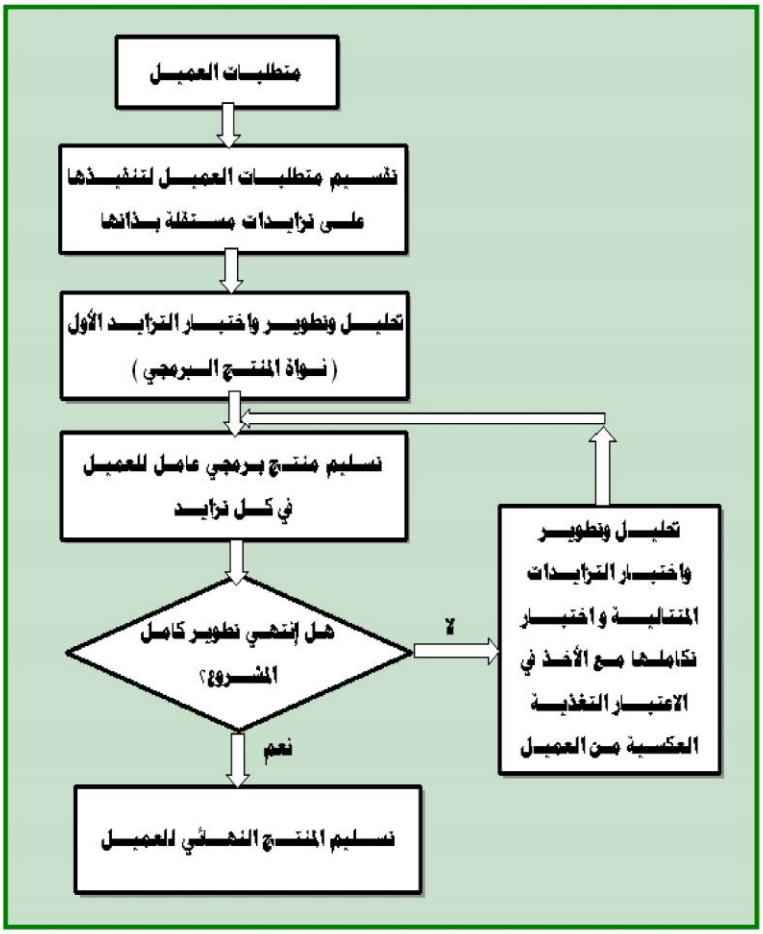
أسئلة تقويم ذاتي

ما هي أهم نشاطات نموذج التطوير السريع للتطبيقات؟.

4. النموذج التزايدي (Incremental Model) :

يُعـد هـذا النمـوذج التزايـدي مـن النمـاذج الارتقائيـة (التطوُّريـة) (Evolutionary Models) التي لها صفة التكرارية ، حيث إنه يجمع عناصر نموذج الشلال التتابعية (التحليل . التطوير . الاختبار) بصورة تكرارية ، أي إنه يقوم بتنفيذ عناصر نموذج الشلال كتتابعات خطية بأسلوب متعاقب بحيث ينتج في كل تتابع تزايداً عاملاً من البرنامج يسلم للعميل ، كما هو بالشكل رقم 3.5. وفي بداية عملية التطوير باستخدام النموذج التزايدي يتم تقسيم متطلبات العميل لتنفيذها على تزايدات بحيث يكون التزايد الأول يحقق المتطلبات الأساسية للعميل ، ويطلق على هذا التزايد : " نواة المنتج " (Product Core) . وبالرغم من أن النموذج التزايدي نموذجاً ارتقائيا مثله في ذلك مثل نموذج النمذجة الأولية إلا إنه يسلم للعميل منتجاً برمجياً عاملاً في كل تزايد بالإضافة إلى أنه يفيد المستخدم في عملية التقويم المستمر للبرنامج, ويكمن التحدي الأكبر لتطبيق النموذج التزايدي في تطوير البرمجيات في تجزئة المشروع إلى مكونات مستقلة بذاتها ومعرفة المكونات التي قد تعتمد على مكونات أخرى حتى يمكن ترتيب عملية تطوير التزايدات بطريقة متعاقبة مع تقدم زمن التطوير.

إن استخدام النموذج التزايدي في تطوير البرمجيات مفيد في حالة تطوير المشاريع العملاقة وخصوصاً عند عدم توافر الكفاءات المطلوبة في شركة واحدة للقيام الكامل بالمشروع ، ففي هذه الحالة يمكن تقسيم المشروع ليقوم بتطويره العديد من الشركات ، وأيضاً عند عدم توافر فريق كامل في الشركة الواحدة لإنجاز الأعمال المنوطة بها في الموعد المحدد للمشروع.



شكل 3.5 : شكل توضيحي للنموذج التزايدي لتطوير البرمجيات.

أسئلة تقويم ذاتي

أعد ترتيب المفردات التالية لتبين شكلاً توضيحياً للنموذج التزايدي لتطوير البرمجيات.

* هل انتهى تطوير كامل للمشروع؟.
* تسليم المنتج النهائي للعميل.
* تقسيم متطلبات العميل لتنفيذها على تزايدات مستقلة بذاتها.
* تحليل وتطوير واختبار التزايد الأول (نواة المنتج البر مجي).
* متطلبات العميل.
* تحليل وتطوير واختبار التزايدات المتتالية واختبار تكاملها مع ع الأخ في الاعتبار التغذية العكسية من العميل.

5. النموذج الحلزوني (Spiral Model)

يُعد النموذج الحلزوني الذي اقترحه **"Barry Boehm"** في عام 1988م من النماذج الارتقائية **(Evolutionary Model)**، وقد بدء التفكير فيه لتطوير البرمجيات من منطلق أنه من المجازفة الانتظار لفترة متأخرة من المشروع لاكتشاف الأخطاء والعمل على تصحيحها كما هو الحال في نموذج الشلال ؛ لذا فإن النموذج الحلزوني . في كثير من الأحيان . يدمج نموذج النمذجة الأولية **(Prototyping Model)** بطبيعته التكرارية في الإطار النظامي **(Systematic Framework)** لنموذج الشلال ، مع الاعتراف بوجود أشياء مجهولة أو مشكوك في صحتها **(Uncertainty)** في مستويات عديدة أثناء عملية تطوير البرمجيات (مخاطر **"Risk"**) ، وذلك لفهم هذه المخاطر والعمل على تلافيها في كل مستوى من مستويات تطوير المنتج البرمجي من خلال التحليل الدوري للمخاطر أثناء عملية التطوير. ويعرف المصطلح **"Risk"** على أنه كارثة أو مصيبة يمكن أن تحدث خلال تطوير

المشروعات البرمجية, وبالطبع فإن للكوارث درجات وتختلف من منطقة لأخرى ومن أمثلة هذه الكوارث خلال مشاريع تطوير البرمجيات ما يلي :

- إهمال بعض من متطلبات العميل أثناء عملية التطوير.
- قيام الشركات المنافسة بطرح منتج برمجي منافس في السوق.
- ترك شخص ما فريق التطوير.
- تأخر إحدى مهام مقومات التطوير عن الموعد النهائي المحدد.
- اكتشاف أن البرنامج يؤدي وظيفته ببطء شديد.
- اكتشاف أن البرنامج يشغل مساحة كبيرة من الذاكرة الرئيسية عند التشغيل.
- توفر آلية تطوير برمجيات جديدة وجيدة أثناء عملية التطوير.
- فهم بعض من متطلبات العميل فهماً خاطئاً.
- تعديل العميل بعض المتطلبات.
- تغيير شكل الجزء المادي المستهدف.
- عدم قدرة فريق التطوير من إنتاج البرمجية في الوقت المحدد ووفقا للميزانية الموضوعة.
- القدرة على إنتاج البرمجية ولكن بميزانية أعلى من المحددة و في الوقت المحدد.
- بعد جهد جهيد ووقت كبير واستغلال للموارد تم اكتشاف أنه لا يمكن تطوير البرمجية لتغطي المتطلبات عند أي تكاليف.
- ظهور تقنية جديدة **(Technological Breakthroughs)** تجعل المشروع من الطراز القديم ، أي لا فائدة منه بعد ظهور هذه التقنية الجديدة.

كما أن النموذج الحلزوني يحاول معالجة القصور الموجودة في نموذج الشلال والتي من أهمهما من . وجه نظر . مطور النموذج الحلزوني **(Boehm)** : أن نموذج الشلال لا يعمل بكفاءة في حالة استخدامه في تطوير العديد من التطبيقات وخصوصاً تلك التي تحتاج تفاعلاً مع المستخدم **(Interactive End-User Applications)** ، حيث إنه من الصعب على المستخدم (العميل) أن يحدد متطلبات تصميم هذه الأنواع من التطبيقات

وخصوصاً متطلبات تصميم وجهة الاستخدام **(Interface Design)** ، وبالتالي فإنه من المتوقع حدوث بعض من المشاكل حيث إن العقد بين العميل والشركة المنفذة للمشروع يتم قبل تحديد دقيق للمتطلبات, بالإضافة إلى ذلك فإن نموذج الشلال لا يدعم عملية إدارة وتحليل المخاطر أثناء المراحل المختلفة من عملية تطوير البرمجيات, وهذا في حد ذاته يُعد مجازفة وذلك لأن عملية اكتشاف الأخطاء والعمل على تصحيحها تتم في مراحل متأخرة من عمر المشروع .

لذا ، فإنه من أهم مميزات النموذج الحلزوني أنه يضع شروطاً واضحة للتعامل مع المهام التي تساورها الشكوك لتقليل المخاطر التي تواجه المشروع البرمجي ، حيث يتم التعامل مع هذه المخاطر بشكل متكرر في كل مرحلة من مراحل المشروع ، علاوة على ذلك يقدم النموذج الحلزوني إمكانية تطوير سريع لنسخ تزايدية من البرنامج ، حيث إنه حسب النموذج الحلزوني يتم تطوير البرنامج من خلال إصدار سلسلة من النسخ الأولية باستخدام النمذجة الأولية **(Prototyping)**. ولكن النموذج الحلزوني يُعد من النماذج المعقدة ويتطلب فريقاً مدرباً ذا خبرة جيدة في إدارة وتحليل المخاطر ، حيث إن نجاحه يعتمد على تلافي المخاطر في مراحل مبكرة من عمر المشروع ، وإنه في حالة عدم اكتشاف بعض من المخاطر الرئيسية التي تواجه المشروع سوف يؤدي ذلك إلى حدوث مشاكل تعوق تطويره في مراحله الأخيرة ، علاوة على أن هذا النموذج يُعد نموذجاً جديداً لم يتم استخدامه على نطاق واسع في تطوير البرمجيات ، وبالتالي فإن تحديد فعاليته لم تحدد بعد بشكل قطعي ، مما أدي إلى عزوف بعض من العملاء عن استخدامه لأنه من الصعب إقناعهم بأنه يمكن التحكم في تطوير المنتج البرمجي أثناء عملية التطوير.

يُقسم النموذج الحلزوني عملية تطوير البرمجيات إلى عدد من مناطق نشاطات الهيكل **(Framework Activities Regions)** ، بحيث يتم تفعيل كل نشاط هيكلي من خلال تنفيذ العديد من المهام في المنطقة الخاصة به. ففي النموذج الأصلي الذي اقترحه **(Boehm)** يوجد أربع مناطق يتم تكييف المهام الخاصة بكل منطقة منها طبقاً لخصائص المشروع الذي يجرى تنفيذه ، وهذه المناطق الأربع ، كما هو موضح بالشكل رقم 3.6 هي

كمايلي :

① منطقة تحديد الأهداف والبدائل والعوائق : وتشمل المهام التالية :

- تحديد أهداف المنتج في مرحلة التطوير الحالية (الأداء ، والوظيفية ، وسهولة التغيير).
- تحديد العوائق التي تؤثر على تنفيذ مهام المرحلة الحالية (التكلفة ، الموعد النهائي ، التداخل مع مكونات البرنامج الأخرى).
- تحديد الطرق البديلة لتنفيذ المرحلة الحالية (شراؤها ، إعادة استعمال شيء آخر ، تطويرها بأية كيفية).

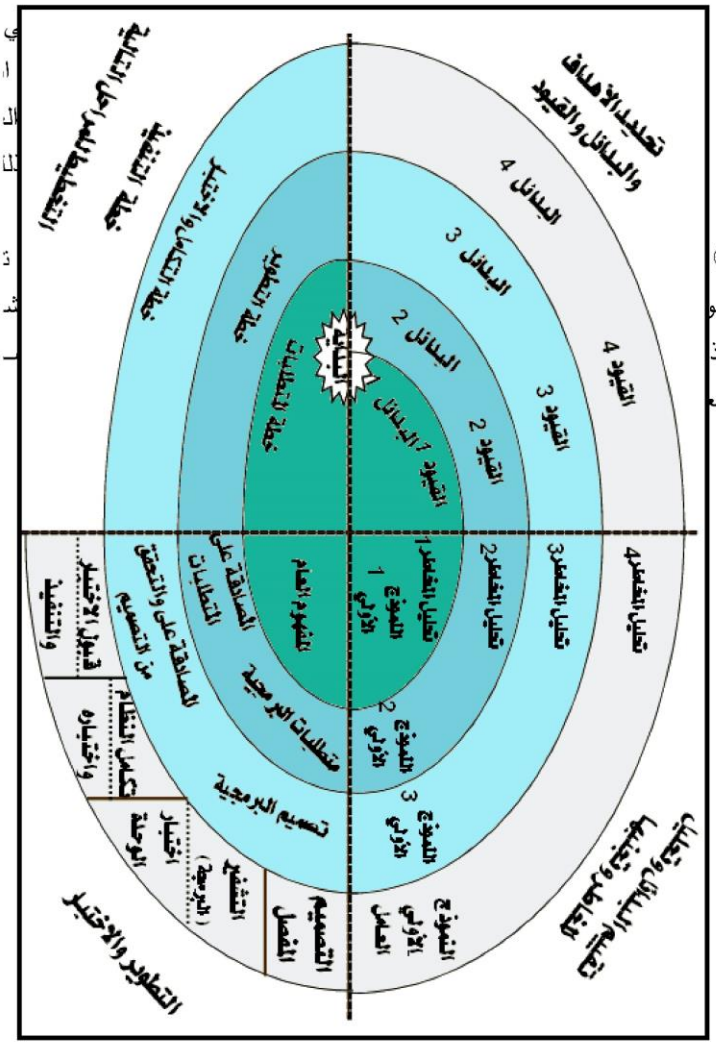
② منطقة تقويم البدائل وإدارة وتحليل المخاطر وتشمل المهام التالية :

- تقويم خطط التنفيذ البديلة مقارنةً بمجموعة المعايير بواسطة الأهداف والعوائق.
- تحديد مجالات المخاطر.
- تحديد الكيفية التي يمكن بواسطتها التغلب على المخاطر.
- تنفيذ بعض الإجراءات لتقليل المخاطر.

على سبيل المثال ، إذا أعتبر الإيفاء بمتطلبات المستخدم مشكلة محتملة ، ففي هذه الحالة قد يتخذ قرار لتنفيذ بعض النماذج الأولية لتوضيح احتياجات المستخدمين. وفي هذه المرحلة يمكن ممارسة المرونة بدرجة عالية في كل طور من المشروع.

③ منطقة التطوير والاختبار : وتشمل المهام التالية :

- تحليل المتطلبات واختيار نماذج لتطوير النظام بناء على تقويم المخاطر ، فمثلاً إذا كانت أكثر المخاطر متعلقة بتصميم وجهة المستخدم ففي هذه الحالة يمكن اختيار نموذج النمذجة الأولية الارتقائي **(Evolutionary Prototyping Model)** لتحديد متطلبات المستخدم ، أما إذا كانت المخاطر متعلقة بعملية تكامل النظم الفرعية للنظام **(Sub-System Integration)** ففي هذه الحالة يفضل استخدام نموذج الشلال **(Waterfall Model)** .
- تنفيذ المهام الخاصة بعملية الاختبار (التحقق والمصادقة **Verification and**



شكل رقم 3.6 : رسم توضيحي للنموذج الحلزوني لتطوير البرمجيات [Pfleeger 98]

وكخلاصة لهذا الموضوع : فإن النموذج الحلزوني هو شبيه بالنموذج التزايدي أو التكراري ، ولكن يتم فيه دمج فعاليات التطوير مع إدارة المخاطر من أجل التحكم فيها وتقليلها, وتبدأ عملية التطوير في النموذج الحلزوني بالدوران حول الحلزون في اتجاه عقرب الساعة من الداخل إلى الخارج ، كما هو موضح بالشكل رقم 3.6. ويبدأ النموذج الحلزوني بمتطلبات العميل مع خطة العمل المبدئية (الميزانية ،و قيود النظام ،و البدائل المتاحة) ، ثم يتقدم خطوة إلى الأمام بتقدير المخاطر وتقويم البدائل المتاحة قبل تقديم ما يُعرف "بوثيقة مفهوم العمليات" **(Concept Of Operation)** ، والتي تصف بشكل عام غير مفصل كيف يجب على النظام أن يعمل. وفي حالة عدم اكتمال تطوير هذا المفهوم ، والذي يُعد منتجاً في حد ذاته ، خلال التزايد الأول (الدورة الأولى على مسار الحلزون الداخلي) تحدث تزايدات متعددة على نفس المسار الحلزوني حتى يكتمل تطوير المفهوم, وبعد ذلك يتم تدقيق محتويات هذه الوثيقة والتأكد من أنها تامة ودقيقة إلى أقصى حد ممكن، وبذلك تكون "وثيقة مفهوم العمليات" هي المنتج من الطور الأول من عملية التطوير, ومن ثم يتم البدء في الدوران حول الحلزون للبدء في الطور الثاني من عملية التطوير والذي يهدف في النهاية إلى إخراج "المتطلبات" كمنتج أساسي له, وفي الطور الثالث تتم عملية "التصميم" ، أما "التنفيذ والاختبار" فتتم في الطور الرابع من عملية التطوير, وفي كل طور من هذه الأطوار يتم المرور على المناطق الأربع وتنفيذ مهام كل منطقة على حده, أي إنه في كل طور يتم تحليل المخاطر والعمل على تقليلها وتقويم البدائل المختلفة لهذا الطور في ضوء متطلبات وقيود النظام ، واستخدام النمذجة الأولية للتحقق من ملائمة أي بديل قبل اعتماده ، وكذلك ضبط خطة المشروع وضبط التكلفة والجدول الزمني بناء على تقويم العميل للمنتج في كل طور وذلك خلال المرور عبر منطقة التخطيط.

نلاحظ في الشكل رقم 3.6 أن الطور الأخير من النموذج الحلزوني (منطقة التطوير والاختبار) مناظر للمراحل الثلاثة الأخيرة من نموذج الشلال (مرحلة التصميم ، ومرحلة التطوير ، ومرحلة تسليم المنتج) مع اختلاف بعض المسميات. حيث إنه عند بداية هذه المنطقة تكون

متطلبات العميل مفهومه وواضحة ومكتملة تماماً لفريق التطوير وذلك من خلال تطوير العديد من النماذج الأولية ، بالإضافة إلى تضمين جميع الاقتراحات والحلول في تطوير النموذج الأخير (العامل) لحل جميع المخاطر الرئيسية التي واجهت المشروع في الأطوار السابقة .

تدريب (2)

ارسم شكلاً توضيحياً وبين فيه التناظر بين النموذج الحلزوني ونموذج الشلال.

والسؤال الآن ماذا عن مرحلة الصيانة في النموذج الحلزوني : للإجابة على هذا السؤال يجب أن نعرف أن النموذج الحلزوني يمكن تكييفه لتطبيقيه على امتداد كامل حياة البرمجيات (أي حتى نهاية عمر البرنامج) ، حيث إن عملية الصيانة يمكن اعتبارها طور من أطوار حياة البرمجيات ويتعامل مثل الأطوار الأخرى من حيث خضوعه لتقويم المخاطر الناتجة من إجراء عمليات الصيانة المطلوبة لمعرفة إمكانية تنفيذ هذه العمليات تحت القيود المفروضة من عدمه.

أسئلة تقويم ذاتي

ما هي أهم مميزات وعيوب النموذج الحلزوني؟.

"يقسم النموذج الحلزوني عملية تطوير البرمجيات إلى عدة نشاطات هيكلية" اشرح هذه العبارة.

6. نموذج الطرق المنهجية (Formal Methods Model)

ينطلق هذا النموذج من مبدأ : أن تطبيق علم الرياضيات بطرقه المنهجية (الصورية) في مجال تطوير البرمجيات يُعد ضرورة في جميع أطوار حياتها ، لما تحتويه هذه الطرق من آليات موثوق بها يمكن استخدامها لإزالة العديد من المشاكل التي تواجه عملية تطوير البرمجيات والتي يصعب حلها باستخدام الطرق (النماذج) التقليدية ، حيث إنه يمكن اكتشاف الأخطاء وحالات الغموض الدقيقة التي تحدث أثناء عملية التطوير

وتصحيحها ، وكذلك ضمان التوافق بين جميع مكونات البرنامج ، عن طريق التحليل الرياضي الدقيق الذي يُمكّن فريق التطوير من اكتشاف وتصحيح هذه الأخطاء التي يُمكن أن تبقى كامنة طوال فترة التطوير في حالة استخدام الطرق التقليدية التي تعتمد على المراجعة اليدوية أو البصرية أو التطبيقية.

يشمل نموذج الطرق المنهجية (الصورية) مجموعة من الأنشطة التي تُمكّن فريق التطوير من وصف متطلبات النظام والتحقق من صحة عمله عن طريق تدوين رياضي دقيق يعتمد على استخدام التحليل الرياضي المنهجي (الصوري) **(Formal Mathematical Transformation)** في وصف النماذج بمجموعة من النماذج (المعادلات) الرياضية ، ومن ثم تحويله إلى برنامج تنفيذي.

ويتميز نموذج الطرق المنهجية بما يلي :

- دقة المواصفات المستخلصة عن طريق التحليل الرياضي.
- المحافظة على صحة المعلومات أثناء العمليات التحويلية التي تقود بدورها إلى ثبات المواصفات كمحصلة نهائية .
- يتم تنفيذ جميع عملياتها بطريق آلية.
- إمكانية إعداد الرسوم البيانية المسبقة للمواصفات كدليل على تحقيقها.

ورغم أن تطبيق نموذج الطرق المنهجية يعطي برمجيات دقيقة تتوافق مع جميع متطلبات العميل ، ويُعتمد عليه في حالة تطوير أنظمة تتطلب قدراً كبيراً من الدقة والأمان **(Strict Safety)** ، والاعتمادية **(Reliability)** ، إلا أنه له سلبيات من أهمها :

- أنه مكلف ويستغرق وقتاً طويلاً في عملية التطوير.
- أنه أدوات عملية التحويل المستخدمة فيه نادرة.
- أنه قلة فقط من مطوري البرمجيات قادرون على استخدامه في تطوير البرمجيات.
- أنه يمثل مصدر صعوبة للتواصل مع العملاء حيث إنه يتطلب عميلاً متطوراً عملياً وتقنياً.

أسئلة تقويم ذاتي

1.اسرد أهم المميزات والسلبيات لنموذج الطرق المنهجية.

2. ضع علامة (√) أمام الإجابة الصحيحة وعلامة(X) أمام الإجابة الخطأثم صحح الجمل الخطأ :

أ) نماذج عمليات تطوير البرمجيات هي مجرد إستراتيجية تطوير لتمثيل أنشطة عمليات البرمجيات حيث تؤدي في النهاية إلى إنتاج منتج برمجي قابل للتنفيذ.

ب) لا يتطلب نموذج الشلال تحديد متطلبات العميل بكل دقة ودرجة وضوح كافية قبل البدء في عملية التصميم.

ج) يتم اللجوء إلى نموذج النمذجة الأولية عندما لا تتوافر معطيات كافية محدده من قبل العميل لبناء البرنامج ولكن يعرف مجموعة من الأهداف والخطوط العامة للنظام.

د) استخدام نموذج النمذجة الأولية في تطوير البرمجيات يعطي العميل فكره عامه وجيده عن النظام الفعلي.

هـ) يعتمد نموذج التطوير السريع للتطبيقات على تقسيم المشروع البرمجي إلى عدة وظائف برمجية رئيسية يتم تناول كل منها بواسطة فريق عمل مستقل.

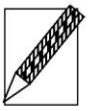
و) تقنيات إعادة الاستخدام (CASE) ، وتقسيم العمل هما حجر الزاوية في نموذج التطوير السريع للتطبيقات.

ز) لا يتطلب نموذج التطوير السريع للتطبيقات وضوح المتطلبات بدرجة عالية.

ح) النموذج التزايدي ملائم في حالة عدم توفر فريق كامل في الشركة الواحدة لإنجاز العمل المناط بها في الموعد المحدد للمشروع.

ط) النموذج الحلزوني في معظم الأحوال يدعم نموذج النمذجة الأولية بطبيعته التكرارية في الإطار النظامي لنموذج الشلال.

تدريب (3)



غالباً ما تكون نماذج تطوير عمليات البرمجيات المختلفة متحدة : ناقش هذه العبارة.

نشاط

إن أفضل نموذج لإنشاء نظام معلومات صغير في المكتب يجب أن يكون مختلفاً عن النموذج المصمم لتطوير نظام يتحكم في الأقمار الصناعية ، ناقش هذه العبارة.

تدريب (4)

أختر النماذج التي يفضل استخدامها في تطوير عمليات البرمجيات في الحالات التالية (يمكن أن تختار أكثر من نموذج مع فرض توافر باقي مقومات الاستخدام لجميع الأنظمة المذكورة) :

(1) متطلبات العميل واضحة تماماً.

○ نموذج الشلال ○ النموذج الأولي ○ النموذج الحلزوني ○ نموذج **RAD**

(2) متطلبات العميل تتغير أحياناً أثناء دورة التطوير.

○ نموذج الشلال النموذج الأولي ○ النموذج الحلزوني ○ النموذج التزايدي

(3) من الضروري البرهنة على تحقيق المتطلبات المطلوبة أثناء عملية التطوير.

○ نموذج الشلال ○ النموذج الأولي ○ النموذج الحلزوني ○ نموذج **RAD**

(4) عدم كفاية الموارد البشرية ذات الكفاءات المطلوبة لإنجاز المشروع في الوقت المحدد .

○ نموذج الشلال ○ النمـــــوذج الأولي ○ النموذج التزايدي ○ نمـــــوذج **RAD**

(5) عدم توافر موارد بشرية كافية لتكوين العدد المطلوب من فرق العمل .

○ **نموذج الشلال** ○ **النمـــــوذج الأولي** ○ **النموذج التزايدي** ○ **نموذج** RAD

(6) بعض أعضاء فريق التطوير معرضين للاستقالة أثناء عملية التطوير .

○ **نموذج الشلال** ○ **النمـــــوذج الأولي** ○ **النموذج الحلزوني** ○ **نموذج** RAD

(7) ممثلوا العملاء لن يستطيعوا متابعة عملية التطوير بصورة منتظمة وفعالة .

○ **نموذج الشلال** ○ **النمـــــوذج الأولي** ○ **النموذج الحلزوني** ○ **نموذج** RAD

(8) ممثلوا العملاء جديدون بالنسبة لتعريف النظام .

○ نموذج الشلال ○ النموذج الأولي ○ النموذج الحلزوني ○ نموذج RAD

(9) ممثلوا العملاء لديهم الرغبة في المشاركة الفعالة في جميع مراحل عملية التطوير.

○ نموذج الشلال ○ النموذج الأولي ○ النموذج الحلزوني ○ نموذج RAD

○ نموذج الشلال ○ النموذج التزايدي ○ النموذج الحلزوني ○ نموذج RAD

(10) النظام يتطلب قدراً كبيراً من الثقة والأمان والإعتمادية.

○ نموذج الشلال ○ النموذج الأولي ○ النموذج الحلزوني ○ النموذج المنهجي

(11) النظام يمثل توجه جديداً للشركة.

○ نموذج الشلال ○ النموذج الأولي ○ النموذج الحلزوني ○ نموذج RAD

(12) النظام يمثل تطويراً لنظام موجود بالفعل.

○ نموذج الشلال ○ النموذج التزايدي ○ النموذج الحلزوني ○ نموذج RAD

(13) الدعم المالي للمشروع غير مستقر أثناء عملية التطوير.

○ نموذج الشلال ○ النموذج التزايدي ○ النموذج الحلزوني ○ نموذج RAD

(14) سوف يتم استخدام النظام لفترة طويلة .

نشاط

بعد انتهاء دراستك للوحدة ارسم جدولاً قارن فيه بين نماذج عمليات تطوير البرمجيات من ناحية:

- **وضوح المتطلبات من عدمه. • نوعيات التطبيقات.**
- **الارتباط بالنماذج الأخرى.**
- **الدور الذي يقوم به النموذج للعميل.**
- **المميزات • السلبيات • وقت التطوير**
- **اعتماده على CASE.**
- **توفر الموارد البشرية (العمالة) اللازمة.**

الخلاصة

تناولت هذه الوحدة الأمور التالية:

– نموذج الشلال: الذي يبدأ بمرحلة تجميع المتطلبات بغرض توثيق متطلبات النظام ، ثم مرحلة التصميم وفيها يتم تحويل وثيقة توصيف المتطلبات إلى وثيقة توصيف التصميم ، ثم مرحلة التنفيذ وفيها يتم تحويل وثيقة توصيف التصميم إلى منتج برمجي ، ثم مرحلة الاختبار وتشخيص الأخطاء وذلك للتأكد من خلو البرمجية من الأخطاء ، ثم مرحلة الاختبار والارتقاء وفيها يتم صيانة وتغير بعض من أجزاء النظام البرمجي ، وقد عرفنا أن هذا النموذج يتميز بالبساطة في الفهم والإدارة.

– نموذج النمذجة الأولية: تعتمد فكرة هذا النموذج على التطوير السريع ويتم اللجوء إليه عندما لا تتوافر متطلبات كافية ومحددة عند العميل ، ومن عيوب هذا النموذج أنه بعد الانتهاء من تحديد متطلبات العميل النهائية لا يتم الاستفادة من الإصدارات الأولية التي تم تطويرها لهذا يطلق على هذا النموذج "أسلوب النمذجة الأولية الملقاة جانباً" ، وهناك أسلوب أخر من هذا النموذج في تطوير البرمجيات يسمى "أسلوب النمذجة الأولية الارتقائية".

– نموذج التطوير السريع للتطبيقات وهو يعتمد على تقسيم المشروع البرمجي إلى عدة وظائف رئيسة يتم تناول كلاً منها بوساطة فريق عمل مستقل ، وأهم نشاطات هذا النموذج : عملية النمذجة: (نمذجة الإعمال ،و نمذجة البيانات ،و نمذجة العمليات) وعملية بناء البرمجية (توليد التطبيق ، توليد الشفيرة ، البرمجية آلياً) ، عملية الاختبار (اختبار كامل البرمجية ، تجربة كل الواجهات – الشاشات–) .

– النموذج التزايدي: من النماذج الارتقائية التطويرية ، حيث إنه يجمع عناصر نموذج الشلال التتابعية (التحليل – التطوير – الاختبار) بصورة تكرارية ، وهذا النموذج مفيد في حالة تطوير المشاريع العملاقة.

– النموذج الحلزوني: من النماذج الارتقائية ، في كثير من الأحيان يقوم هذا النموذج بدمج نموذج النمذجة الأولية في الإطار النظامي لنموذج الشلال مع العلم بأن كارثة أو مصيبة يمكن أن تحدث خلال تطوير المشروع البرمجي ، يقسم النموذج الحلزوني عملية التطوير إلى عدد من مناطق نشاطات الهيكل وهي : منطقة تحديد الأهداف والبدائل والعوائق ، منطقة تقويم البدائل وإدارة وتحليل المخاطر ، ومنطقة التطوير والاختبار ،و منطقة التخطيط.

نموذج الطرق المنهجية : يشمل مجموعة من الأنشطة التي تمكن فريق التطوير من وصف متطلبات النظام والتحقق من صحة عملة عن طريق تدوين رياضي دقيق يعتمد على استخدام التحليل الرياضي المنهجي.

لمحة مسبقة عن الوحدة التالية

تعد هندسة متطلبات البرمجيات من أهم الأنشطة والخطوات في بناء وتطوير البرمجيات ، هذا ما سنعرفه في الوحدة التالية وهى بعنوان "هندسة متطلبات البرمجيات" وفيها سنوضح كل من مفهوم هندسة المتطلبات ، والأنشطة (المراحل) التي تشملها عملية هندسة المتطلبات ومدخلاتها ومخرجاتها.