الباب السادس

هندسة البرمجيات

٦-١ مقدمة عامة

٦-٢ منهجيات تنفيذ نظم البرمجيات

٦-٣ التصميم المنظومى والمترافق

٦-٤ تصميم وتنفيذ البرمجيات باستخدام المكونات

٦-٥ لغات البرمجة وقواعد البيانات وتكنولوچيا الإنترنت

٦-٦ نظم برمجيات المصدر المفتوح

٦-٧ نماذج قياس نضج الأداء فى هندسة البرمجيات

الباب السادس

هندســـة البرمجيـات

٦-١ مقدمة عامة

إن مرونة نظم الحاسبات تكمن فى أنها قابلة للبرمجة بحيث يمكن أن تؤدى عددا كبيرا من المهام المتنوعة حسب طبيعة هذه البرامج . وتشكل البرامج جزءا عضويا من نظام الحاسب وبعضها مثل نظم التشغيل المختلفة تتكامل بشكل كبير مع بنية الحاسب نفسه وفى بعض الأحيان يتم تطور تصميم الحاسب باضافة بعض التعليمات [Instructions] الاضافية لتسهيل تصميم برامج نظم التشغيل نفسها . وبالطبع يمكن لنظم التشغيل أن تتكامل مع عدد كبير من المكونات الجامدة Hardware للحاسبات . ولغات الحاسب المختلفة تستخدم فى تسهيل برمجة الحاسبات وتتطلب بالتالى برامج أخرى لترجمتها إلى لغة الآلة (Machine Language) . كما أن هناك برمجيات (Software) أخرى لإدارة قواعد البيانات والمساعدة على كتابة البرامج التى تتعامل معها أو برامج تطبيقات الرسومات ثنائية وثلاثية الأبعاد أو برامج معالجة النصوص وأتمته المكاتب (Office Automation) أو غيرها من البرامج المتعددة التى تساعد فى تسهيل استخدام الحاسبات فى التطبيقات المختلفة . وتعتبر كل هذه البرامج أدوات تساعد مصمم ومنفذ منظومة الحاسب المتكاملة على الوصول إلى تحقيق المهام المطلوبة من النظام .

ولذلك فإن تصميم منظومة الحاسبات يبدأ أولا بالمكون المنظومى (System Component) والتى تشتمل على المكونات الجامدة (Hardware) أو المكونات الشبكية الخاصة بربط الحاسب أو الحاسبات التى سيتم تنفيذ النظام عليها ببعضها البعض ، أو بإحدى الشبكات العالمية أو المكون اللين (Software) وهى البرمجيات المختلفة التى ستتكامل مع المكونات الأخرى . وهناك أيضا مكونات أخرى لا تقل أهمية عن تلك مثل المكون المعلوماتى والمكون الخاص بربط المستخدمين بنظام الحاسبات بالإضافة إلى ضرورة مراعاة الأبعاد الاقتصادية [Kemerer, 1998] والإيكولوجية المختلفة ومستوى الجودة المطلوب وانعكاسه على تكلفة النظام [Slaughter, 1998] . كما أن هناك بعض النظم التى تتطلب قدرا كبيرا من الاعتمادية (Reliability) والأمان ، وعلى الأخص فى التطبيقات العسكرية [Littlewood, 1992] أو التطبيقات الطبية ، التى تعتمد على شبكات المعلومات [Birman, 1996]. وعلى هذا الأساس فإن تنفيذ برمجيات فعالة يُعتمد عليها يتطلب بناء منظومة هندسية متكاملة تتوقف على اعتبارات مختلفة، مثل:

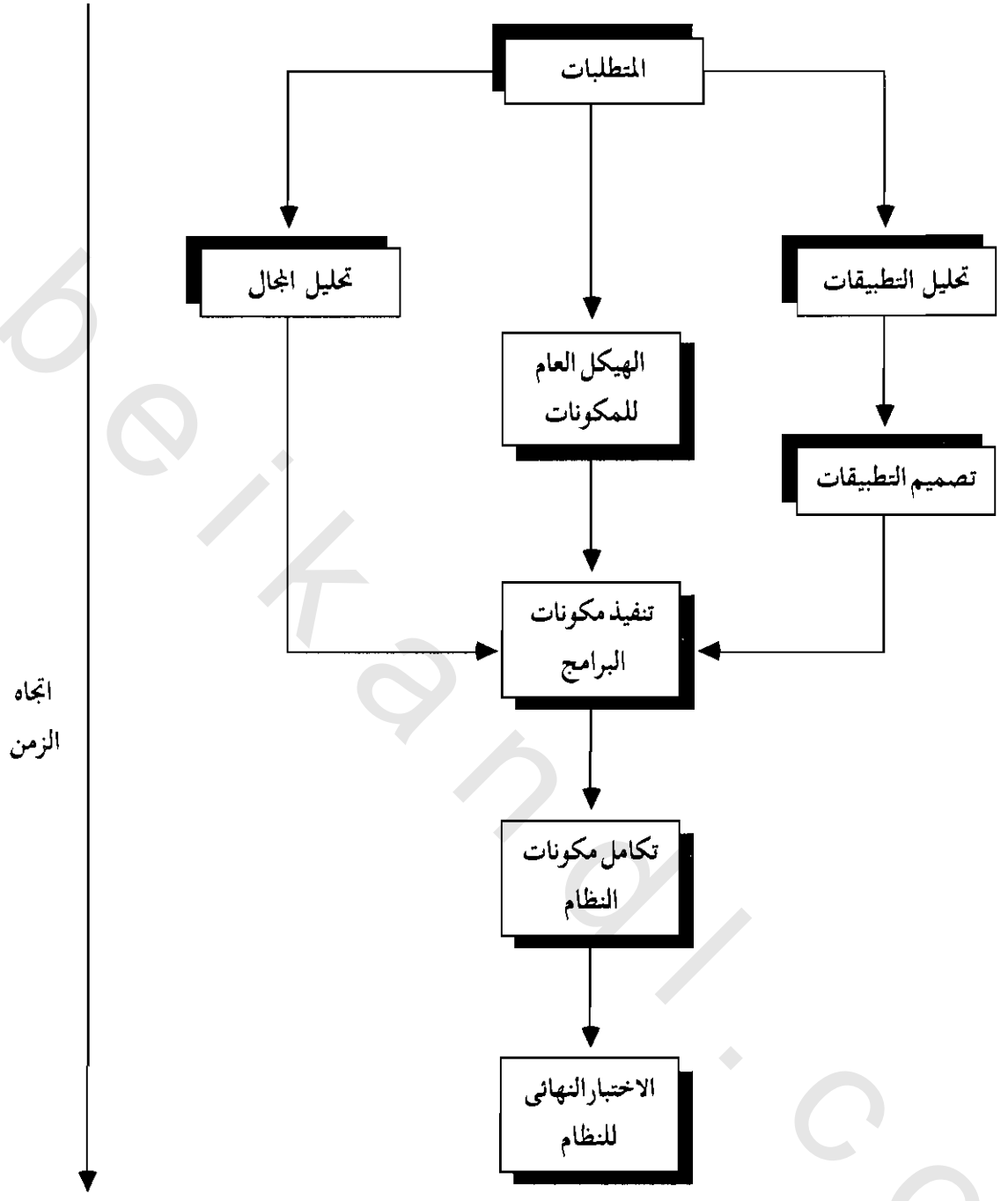
الإطار العام لتكلفة المنظومة – المجموعات التى سيخدمها النظام وهل هى

للأفراد أو لجماعات صغيرة أو مؤسسات خاصة أو عامة أو تطبيقات قومية أو تطبيقات عالمية – نوعية التطبيق وهل هى فى المجال العسكرى أو الطبى أو الترفيهى أو التعليمى أو غيرها – طريقة التعامل مع النظام من جانب المستخدمين سواء بالنسبة لفترات الاستخدام أو الخبرات المطلوبة من جانبهم [Brereton, 1999] . كل هذا ينعكس على الخبرات المطلوبة من جانب المصممين والمنفذين للنظام ، ففى بعض الأحيان يتطلب الأمر أخصائيين بمستويات مختلفة فى هندسة البرمجيات أو يتطلب أن يقوم المستخدم النهائى بعد الحصول على قدر من الدراسة والتدريب المكثف فى تصميم وتنفيذ البرمجيات الخاصة ببعض التطبيقات البسيطة [Jones, 1995] . وفى كل الحالات يتطلب الأمر وجود منهجيات محددة لتنفيذ نظم البرمجيات ، يتم اختيار المناسب منها حسب طبيعة التطبيقات والأنظمة المتكاملة .

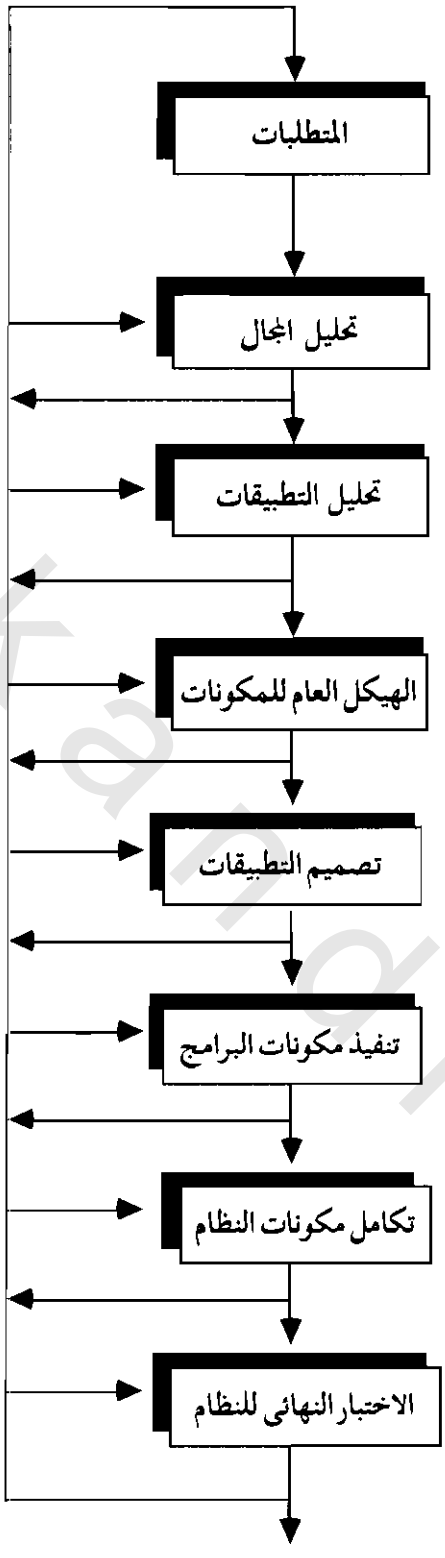
٦-٢ منهجيات تنفيذ نظم البرمجيات

هناك منهجيات متعددة لتنفيذ نظم البرمجيات ولكن سنقدم هنا فقط الإطار العام لإحداها والتى يمكن أن تصلح لبعض المشروعات الصغيرة [Russ, 2000] . ويبين الشكل (٦-١) العلاقات الزمنية بين المراحل المختلفة والتى تبدأ بمرحلة تحديد المتطلبات ، بعد ذلك تبدأ عملية التحليل التفصيلى لها سواء فيما يتعلق بالتطبيقات المطلوبة أو تحليل دقيق للمجال الذى ستعمل فيه هذه التطبيقات بالإضافة إلى تحديد الهيكل العام لمكونات النظام وطريقة ربطها والتعامل معها من جانب المستخدمين وهكذا . بعد ذلك يتم تصميم التطبيقات . وعند الانتهاء من هذه المسارات الثلاثة كما هو مبين فى الشكل (٦-١) يتم تنفيذ مكونات البرامج كل على حدة ، ثم تكاملها بعد ذلك فى منظومة متكاملة ، وفى هذه الحالة يمكن إجراء الاختبارات النهائية للنظام . ويجب ملاحظة أن هذه المراحل المختلفة يمكن أن تتضمن تعديلات متكررة ،كما هو مبين فى الشكل (٦-٢) .

وبالنسبة للمشروعات الكبيرة يمكن أن نستخدم تكنولوچيا هندسة البرمجيات بمساعدة الحاسبات Computer - Aided Software Engineering (CASE) والتى قد تعمل على زيادة إنتاجية التنفيذ وضمان جودة المنتج النهائى [Sharma, 2000] .



شكل (٦-١) : العلاقات الزمنية بين المراحل المختلفة .

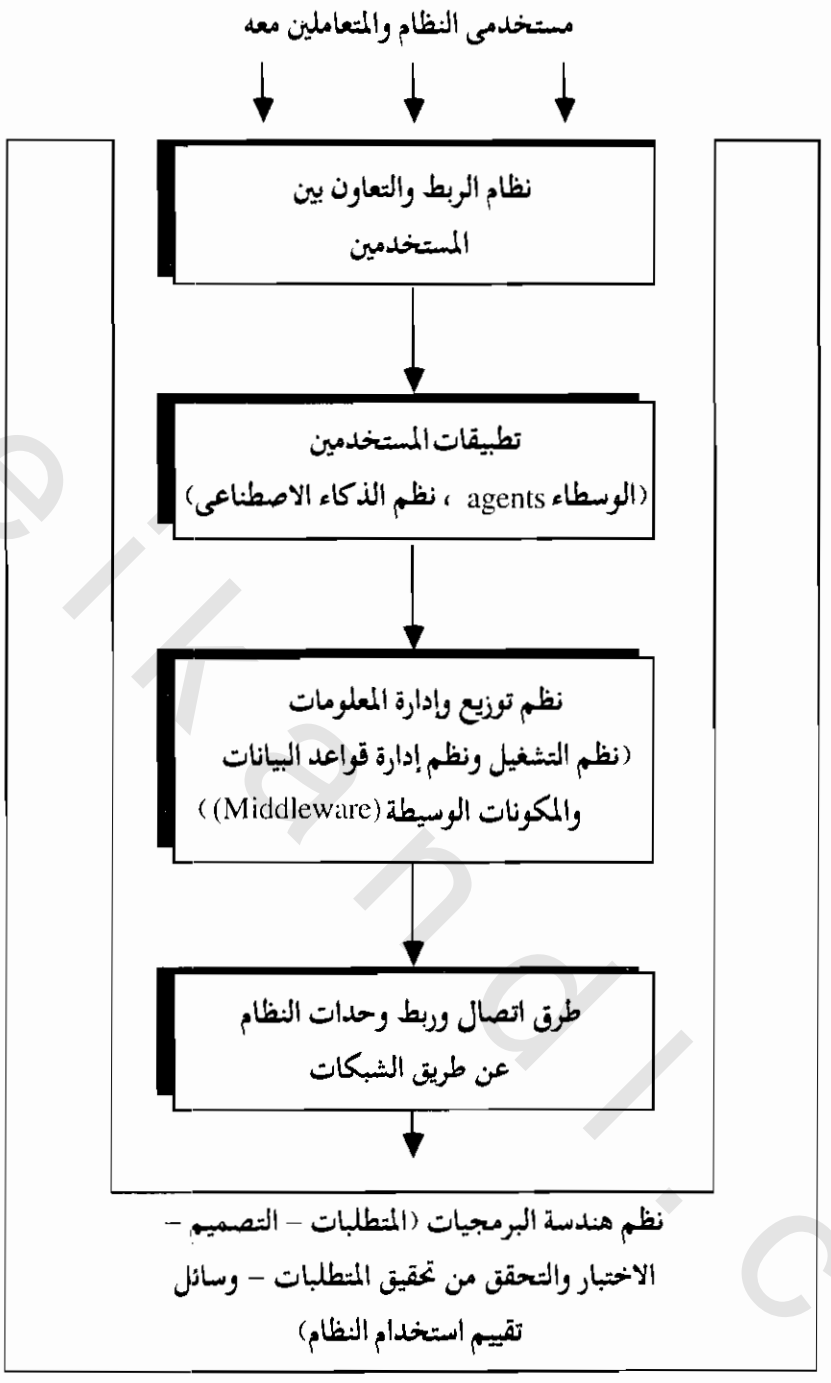


شكل (٦-٢) : نموذج التعديلات المتكررة التى يمكن أن تحدث قبل الانتهاء من التنفيذ النهائى للنظام .

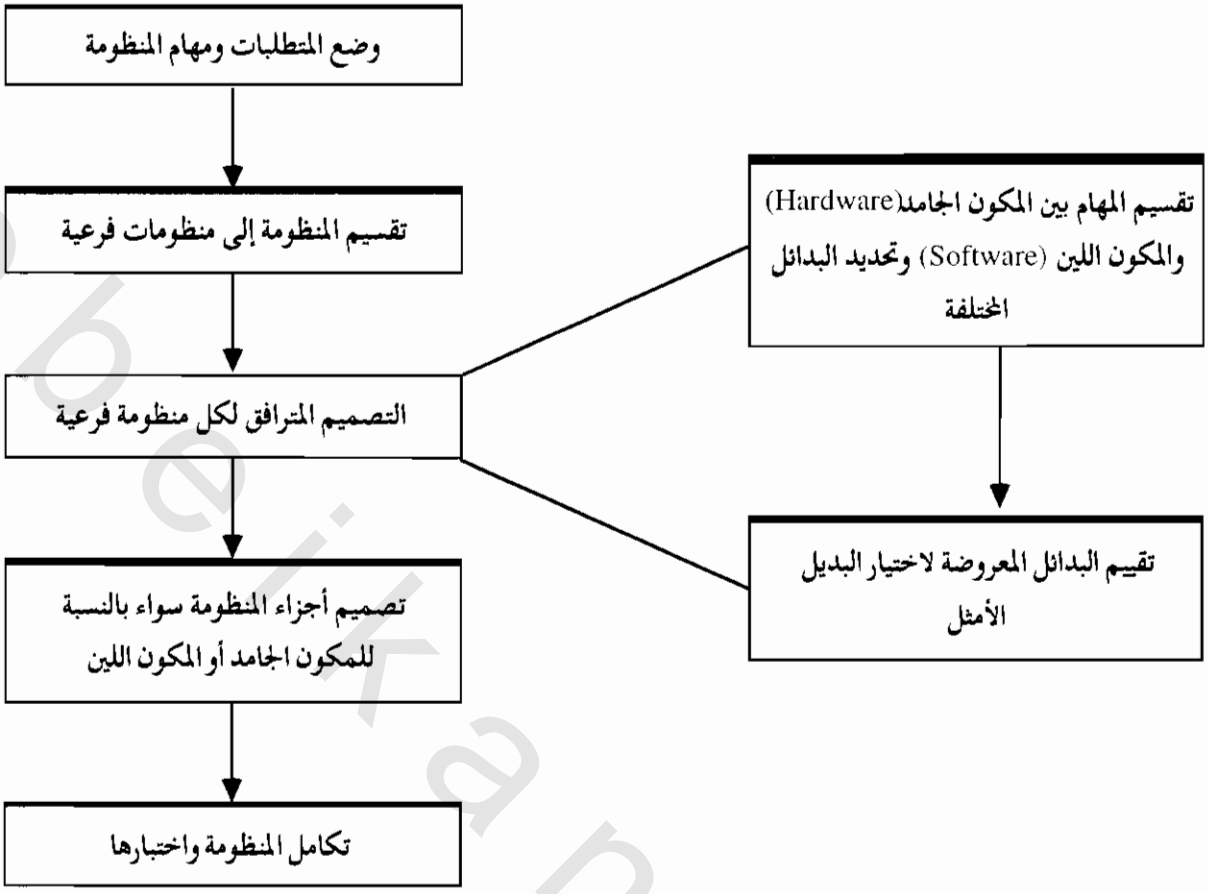
٦-٣ التصميم المنظومى والمترافق

نظرا لانتشار نظم الحاسبات فى جميع أنشطة المجالات المختلفة والخاصة بتطبيقات تكنولوچيا المعلومات فقد أصبح من الضرورى أن تتكامل هندسة البرمجيات مع المكونات المختلفة لتكنولوچيا المعلومات حتى يمكن الوصول فى النهاية إلى منظومة مترابطة تقوم فيها كل المكونات بأداء عملها فى تنسيق وتكافل [Boehm, 2000] . ويوضح الشكل (٦-٣) كيفية إجراء هذا التكامل . ويبين هذا الشكل أن نظم البرمجيات هى البوتقة التى تتكامل وتنصهر فيها بقية المنظومات الفرعية شاملة نظام الربط والتعاون بين المستخدمين والتطبيقات المختلفة ونظم توزيع وإدارة المعلومات وطرق اتصال وربط وحدات النظام عن طريق الشبكات وغيرها .

ويتطلب ذلك إعادة النظر فى تصميم نظم البرمجيات ، والتى تستخدم ما يسمى التصميم المترافق (Co - Design) والذى يبدأ أولا بتحديد مهام المنظومة وتقسيمها إذا كان ذلك مطلوبا لمنظومات فرعية . بعد ذلك يبدأ التصميم المترافق ، والذى يتم فيه تقسيم المهام بين المكون الجامد والمكون اللين ، والتى يمكن أن تشتمل على أكثر من بديل . بعد ذلك يتم تقييم هذه البدائل لاختيار أفضلها . وفى المرحلة التالية يتم تقييم أجزاء المنظومة وتكاملها واختبارها ،كما يوضح الشكل (٦-٤) . [Kumar, 1993]



شكل (٦-٣) : هندسة البرمجيات وتكاملها مع مكونات تكنولوجيا المعلومات المختلفة .



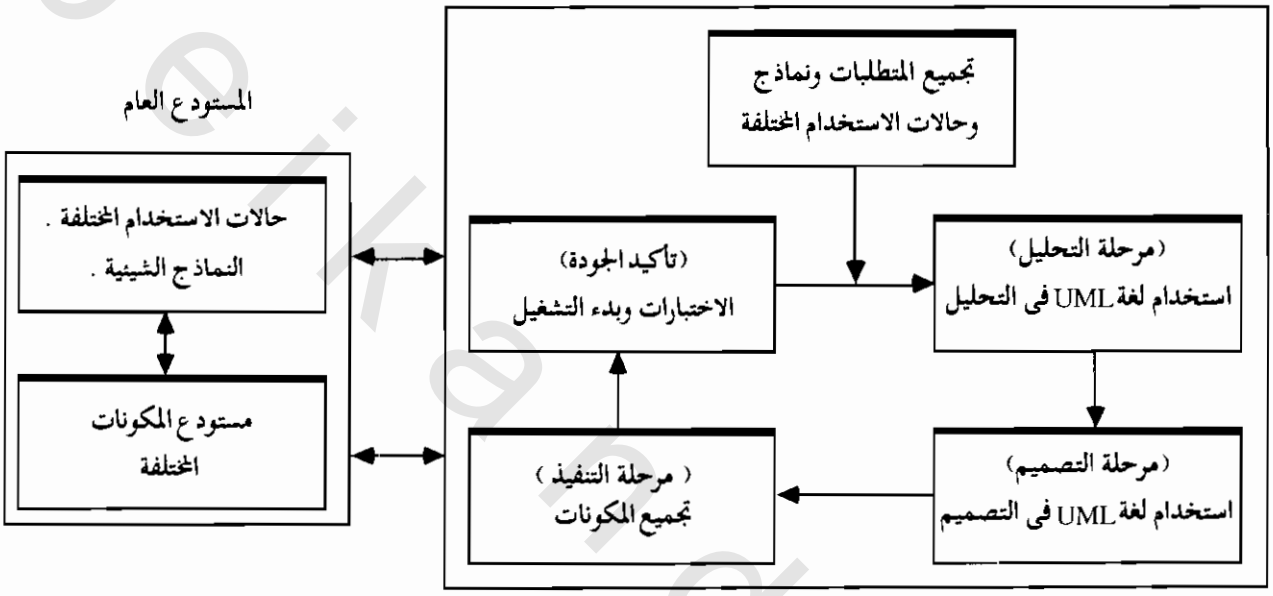
شكل (٦-٤) : الإطار العام لمراحل التصميم المترافق .

٦-٤ تصميم وتنفيذ البرمجيات باستخدام المكونات

تحظى إحدى طرق تصميم وتنفيذ البرمجيات التى تعتمد على المكونات (Component - based) فى ظل ما يسمى «الأُطُر» (Frameworks) باهتمام كبير فى الوقت الحالى . ويمكن تعريف «المكون» (Component) على أنه جزء من المنظومة من الممكن استبداله ، يتيح الاتصال والارتباط بالمكونات الأخرى عن طريق مجموعة من «الروابط البينية» (Intefaces) . أما بالنسبة للأطر فهى تتيح العناصر الأساسية والعلاقات المختلفة ، وتضمن التكامل البنائى والديناميكى للمنظومة كلها بالإضافة إلى إتاحة نقاط الامتداد التى تضمن تغيير الإطار ليصلح لتطبيق معين. وتتيح هذه الفكرة الاستفادة من التطبيقات السابقة ، سواء فيما يتعلق بنوعية الإطار المستخدم ، أو المكونات التى تساهم فى بناء منظومة معينة خاصة بتطبيق محدد [Hopkins, 2000] [Larson, 2000] . وهى تتيح أيضا الاستفادة من تكنولوجيات «الاتجاه الشيئى» (Object - Oriented) ويمكنها بذلك استخدام «لغة النمذجة الموحدة» (Unified Modeling Language) (UML) على سبيل المثال . وبذلك هناك إمكانية لزيادة الإنتاجية (Productivity) ومستوى

الجودة (Quality) والقابلية للامتداد (Extensibility) وغيرها من الخصائص المطلوبة لنظم البرمجيات .

ويمكن تبسيط عرض عملية استخدام المكونات فى تطوير برامج التطبيقات المختلفة من خلال دورة التطور الموضحة فى الشكل (٦-٥) . وهناك اهتمام كبير فى الوقت الحالى فى استخدام هذه المنهجية بالنسبة لتطبيقات التجارة الإلكترونية ، التى تتطلب قدرا كبيرا من المرونة والسرعة فى تطوير التطبيقات . [Krieger, 1998] [Fingar, 2000]



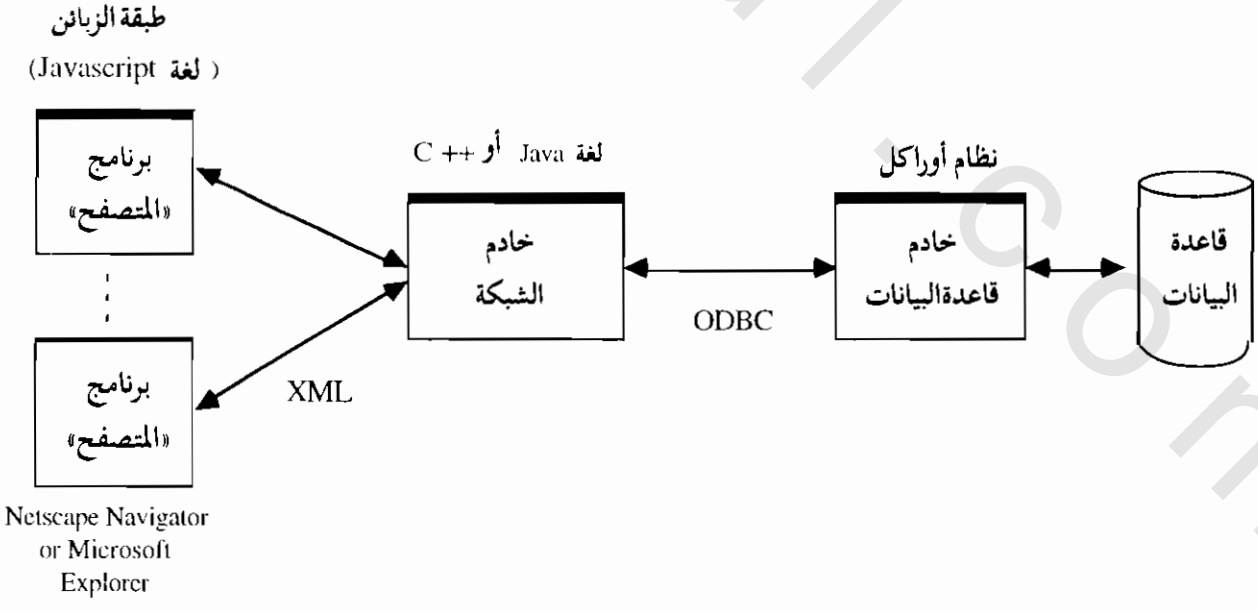
شكل (٦-٥) : دورة تطوير البرامج باستخدام النظام المبنى على المكونات .

٦-٥ لغات البرمجة وقواعد البيانات وتكنولوجيا الانترنت

منذ أن تم إستخدام لغة «فورتران» Fortran فى سنة ١٩٥٧ للتطبيقات العلمية وهناك تطوير لعديد من لغات البرمجة . كما أن نظم قواعد البيانات بدأت نواتها فى الستينيات كأدوات فعالة لنمذجة البيانات وتكاملها والتعامل معها . ومع انتشار تكنولوجيا الإنترنت وظهور الشبكة العالمية العنكبوتية World Wide Web (WWW) فى التسعينيات أثر ذلك على كل من لغات البرمجة ونظم قواعد البيانات، بالإضافة إلى التطوير الخاص بنظم شبكات المعلومات نفسها وهياكلها المختلفة [Kroenke, 2000] . ويوضح الشكل (٦-٦) ما يسمى البنية ذات الثلاث طبقات (Three-tier Structure) والتى تشتمل على طبقة «الزبائن» (Clients) والتى تتصل بالشبكة عن طريق الحاسبات الشخصية على سبيل المثال ، والطبقة الثانية هى طبقة «خادم الشبكة» (Web Server) ، والطبقة الثالثة هى طبقة «خادم قاعدة البيانات» (Database Server) وتشتمل على الحاسب الذى يتصل بإحدى قواعد

البيانات التى تحتوى على المعلومات المختلفة . وتشتمل طبقة الزبائن على برنامج يسمى «المتصفح» (Browser) والذى يتيح عرض «الصفحات» (Pages) المختلفة التى يتم الحصول عليها من خادم الشبكة أو خادم قاعدة البيانات ، وتوجد أنواع مختلفة من هذه البرامج ، أشهرها برنامج «الملاح» Navigator من شركة «نيتسكيب» (Netscape) والمستكشف (Explorer) من شركة «مايكروسوفت» (Microsoft) ، وهناك لغات مختلفة بالنسبة لأنظمة كل طبقة ، بالإضافة إلى البروتوكولات أو اللغات التى تسمح لاتصال الطبقات ببعضها . فبالنسبة لطبقة الزبائن يمكن استخدام لغة «جافا للنصوص» (Java script) وبالنسبة لخادم الشبكة يمكن إستخدام لغة «جافا» أو لغة C++ وبالنسبة لخادم قواعد البيانات يمكن استخدام نظام «أوراكل» Oracle . وبالطبع لكل طبقة يوجد نظام التشغيل الخاص بها . أما بالنسبة للربط بين طبقة الزبائن وطبقة خادم الشبكة ، فيمكن استخدام لغة تسمى «لغة العلامات الممتدة (Extended Markup Language) [Bosak, 1999] (XML) . وللربط بين خادم الشبكة وخادم قواعد البيانات يمكن على سبيل المثال إستخدام «الاتصال الشيئى بقواعد البيانات» [Kroenke, 2000] . (Object Database Connectivity) (ODBC)

كما تجدر الإشارة إلى أن شبكة الإنترنت يمكن استخدامها فى تطوير وإنتاج البرمجيات [Gao, 1999] أو استخدام لغة برمجة مثل لغة «جافا» فى إنتاج أنظمة البرمجيات التى تعتمد على ما يسمى «الحسابات المتمركزة على الشبكات» . (Hamilton, 1996) (Net-Centric Computing)



شكل (٦-٦) : البنية ذات الثلاث طبقات للاتصال بالشبكات .

٦-٦ نظم برمجيات المصدر المفتوح

هناك اتجاه فى تطوير البرمجيات يعتمد على ما يسمى «تطوير البرمجيات باستخدام المصدر المفتوح (Open Souree Software Development) . وفى هذا الاتجاه تكون تفاصيل البرمجيات التى تستخدم سواء كانت نظم تشغيل حاسبات أو غيرها من البرمجيات معروفة تماما وتفاصيل تصميمها وجميع وثائقها متاحة للشخص أو الجهة التى تقتنى هذه البرمجيات [O'Reilly, 1999] . ويمكن فى هذه الحالة للجهة التى تقتنى هذه البرمجيات بإجراء بعض التعديلات عليها أو تطويرها ويمكنها فى هذه الحالة إتاحتها للآخرين من خلال إتفاقيات خاصة . وعلى الرغم من أن هذا الإتجاه مازال فى بدايته إلا أنه يجب الاهتمام بالتطورات التى تحدث فيه على المستوى الدولى خصوصا أنه قد يتيح إضافة تعديلات على البرمجيات ، التى تصدرها بعض الشركات بلغات معينة بحيث تسمح باستخدامها بلغات أخرى ، مثل اللغة العربية على سبيل المثال .

٦-٧ نماذج قياس نضج الأداء في هندسة البرمجيات

نظرا لأهمية نظم البرمجيات فى منظومات نظم المعلومات بوجه عام فقد وضعت بعض النماذج لقياس نضج الأداء (Capability Maturity Model) (CMM) عن طريق «معهد هندسة البرمجيات» فى جامعة كارنيجى ميلون» بالولايات المتحدة الأمريكية والذى أنشأته وزارة الدفاع الأمريكية لدعم البحوث فى هندسة البرمجيات [Herbsleb, 1997] ، وعلى غرار هذا النموذج تم إقتراح نماذج أخرى لقياس كفاءة أداة الأفراد وتطويرها [Humphrey, 1996] . ويشتمل «نموذج نضج الأداء» على خمسة مستويات أقلها المستوى رقم (١) وأعلاها المستوى رقم (٥) . وخصائص المستوى رقم (١) مرتبطة بالجهات التى لا تستخدم منهجية معينة فى تطوير البرمجيات ، ويعتمد نجاح هذه الجهات على المجهودات الفردية غير المنظمة . أما بالنسبة للمستوى رقم (٢) فيكون هناك إطار لعمليات إدارة مشروع التطوير لمتابعة وجدولة العمليات وتحديد وصياغة الوظائف المختلفة . وللوصول إلى هذا المستوى يتم الاستفادة من النجاحات السابقة الخاصة بالتطبيقات المشابهة . وفى المستوى رقم (٣) تكون المؤسسة قد وصلت إلى تكامل بين أنشطة إدارة المشروع والأنشطة الفنية الأخرى، مع ضرورة توثيقها وتنميطها وتكاملها فى منهجية شاملة للمؤسسة ، سواء بالنسبة لتطوير أو صيانة البرمجيات . وفى المستوى رقم (٤) تكون المؤسسة قد وصلت إلى تحديد مقاييس تفصيلية لقياس مدى جودة عمليات إنتاج البرمجيات، مع وجود المعايير الكمية المفهومة والواضحة والتى يمكن التحكم فيها . وفى المستوى رقم (٥) تصل المؤسسة إلى أساليب واضحة فى منع الأخطاء فى نظم البرمجيات ووجود أنظمة «إدارة التغيير» (Change management) لمتابعة التطورات ، سواء على مستوى التكنولوجيا أو على مستوى العمليات التى يتم تنفيذها ، ويمكن تطبيق هذا النموذج بالنسبة لمجالات متعددة (Johnson, 2000) .