هندسة برمجيات ١

المحاضرة الأولى

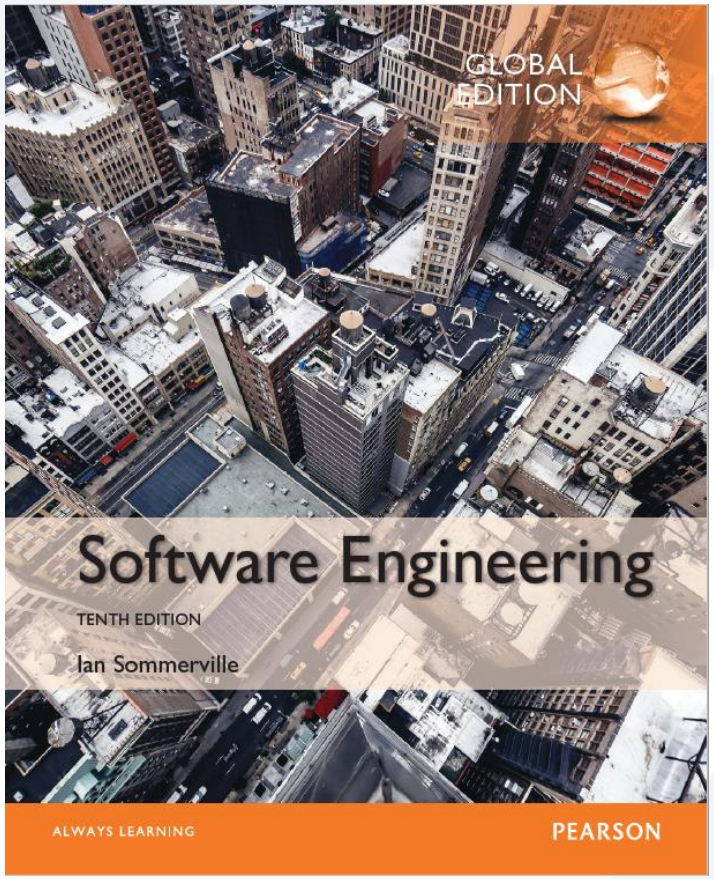
د. اناس ليلى

الفصل الدراسي الاول ٢٠٢٥ / ٢٠٢٦

جَامعة المَنارة
MANARA UNIVERSITY

مفردات المقرر

- مقدمة في هندسة البرمجيات
- الإجرائيات البرمجية Software Processes
- التطوير البرمجي الرشيق Agile Software Development
- هندسة المتطلبات Requirements Engineering
- نمذجة النظام System Modeling
- التصميم والتنجيز Design and Implementation



المرجع المعتمد:

Sommerville, I. (2016). Software Engineering (10th ed.).

ISBN:978-1-292-09613-1. Pearson Education.

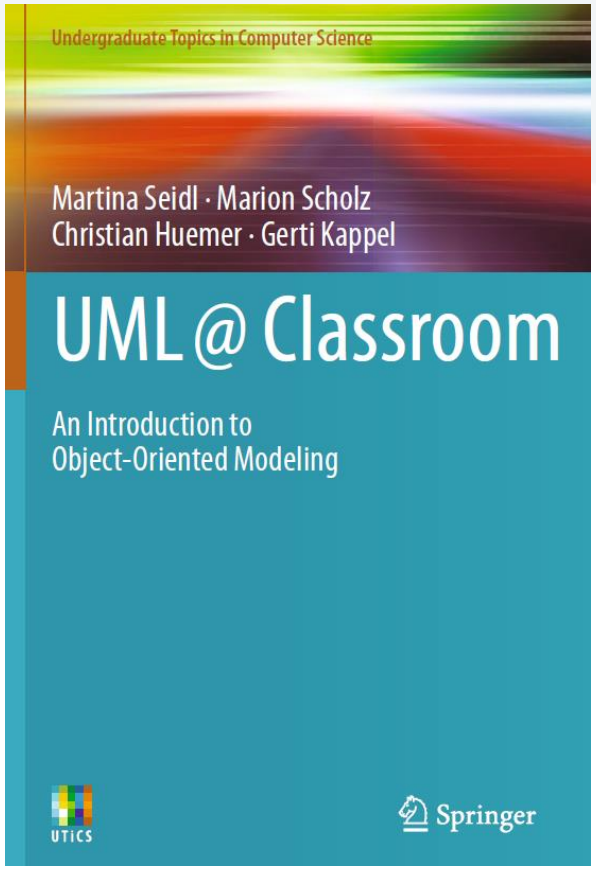
معلومات إضافية ذات صلة:

https://iansommerville.com/software-engineering-book/static/about/

مفردات المقرر - UML Part

- (Behavior Diagrams)
  - مخطط حالة الاستخدام Use Case Diagram
  - مخطط الحالة State Diagram
  - مخطط النشاط Activity Diagram
  - مخطط التتابع Sequence Diagram
- (Structure Diagrams)
  - مخطط الصفوف Class Diagram
  - مخطط الأغراض Object Diagram

المرجع المعتمد



Seidl M.; Scholz, M.; Huemer, C.; Kappel, G. (2015). UML @ Classroom: An Introduction to Object-Oriented Modeling. ISBN:978-3-319-12741-5. Springer.

معلومات إضافية:

https://www.uml-diagrams.org/

FAQs about software engineering

- What is software?
- What are the attributes of good software?
- What is software engineering?
- What is the difference between software engineering and system engineering?
- What is a software process?
- What is a software process model?
- What are the costs of software engineering?
- What are the key challenges facing software engineering?

What is software?

Many people think that software is simply another word for computer programs. But software is several separate programs and also all associated documentation, libraries, support websites, program guides, Data.

System Documentation: which describes the structure of the system.

User Documentation: which explains how to use the system.

support websites: for users to download recent product information.

Configuration files: files that contain settings and parameters that are used to configure the software to run in a specific environment.

Data: the information that the software uses or manipulates.

في هندسة البرمجيات، تعتبر التوثيق جزءًا أساسيًا من المنتج البرمجي **Software Product**

المنتجات البرمجية Software Products

يوجد نوعان من المنتجات البرمجية حسب الغرض

- **منتجات عامة (Generic Products)**

تُطوّر للاستخدام العام وليس لعميل محدد. تُباع في الأسواق وتستخدمها شريحة واسعة من الناس. أمثلة: Microsoft Office Windows المتصفحات، تطبيقات الموبايل مثل WhatsApp مميزاتها: منخفضة التكلفة على المستخدم، مدعومة لفترة طويلة.

- **منتجات مخصصة (Customized Products / Bespoke Software)**
  - المنتج البرمجي الذي تم طلبه من قبل زبون معين لتلبية احتياجاته الخاصة (Own Needs) وهذا الزبون هو من يحدد متطلبات هذه المنتجات البرمجية وهو من يتخذ القرارات بإجراء تعديلات او تغييرات على المنتج
  - تُصمّم لعميل محدد حسب احتياجاته.

مثال: نظام إدارة جامعة، نظام محاسبة لشركة معينة، نظام حجز مستشفى مميزاتها: تلبي احتياجات العميل بدقة، لكنها أعلى تكلفة.

ما الفرق الدقيق بين هندسة البرمجيات (Software Engineering) و هندسة النظم (Systems Engineering)؟

هندسة البرمجيات Software Engineering: هي العلم الذي يهتم بتحليل وتصميم وبناء واختبار وصيانة أنظمة برمجية عالية الجودة، بطريقة منهجية (Systematic) وموثوقة وقابلة للتطوير. ويكون تركيزها الأساسي على البرمجيات فقط.

هندسة النظم Systems Engineering: هي العلم الذي يهتم بتحليل وتطوير الأنظمة الكاملة (Systems) سواء كانت برمجية أو عتادية. تركيزها الأساسي على النظام بالكامل بكل مكوناته، وليس البرمجيات فقط.

تعريف النظام System: النظام هو مجموعة من المكونات (Components) المترابطة والمتفاعلة فيما بينها، والتي تعمل معًا لتحقيق هدف أو وظيفة محددة.

يتميز أي نظام بوجود

- ✓ مكوّنات (Components / Subsystems): الأجزاء التي يتكون منها النظام
- ✓ علاقات وروابط (Interactions / Interfaces): الطريقة التي تتفاعل بها المكونات معًا.
- ✓ هدف أو وظيفة ((Purpose / Function): الشيء الذي بُني النظام لأجله.
- ✓ حدود للنظام (System Boundary) يميز ما ينتمي للنظام وما هو خارج عنه.
- ✓ بيئة محيطة (Environment) كل ما يتفاعل مع النظام من الخارج.

بماذا تختلف البرمجيات عن بعضها؟ الاختلافات تأتي من عدة جوانب منها:

الغرض الأساسي للبرمجية : لعبة ≠ برنامج طبي، تطبيق محاسبة ≠ نظام تحكم في طائرة

البيئة التي تعمل فيها هناك تطبيقات سطح مكتب ،تطبيقات ويب، تطبيقات موبايل، تطبيقات مضمّنة (Embedded) مثل برمجيات الغسالات، السيارات الذكية ، تطبيقات سحابية (Cloud) وغيرها.

المتطلبات الزمنية: بعض الأنظمة تعمل بالزمن الحقيقي Real-Time مثل أنظمة الطائرات وبعضها لا يحتاج استجابة فورية.

درجة التعقيد: برنامج صغير كآلة حاسبة مقابل نظام اعقد مثل برنامج إدارة مشفى او برنامج المفاضلة الالكتروني

عدد المستخدمين: هناك برامج شخصية بمستخدم واحد ، وبرامج شبكية يستخدمها العديد من الأشخاص معاً.

حسب التوزيع: البرمجيات مفتوحة المصدر(Open Source) يكون فيها الكود متاح للجميع مثال: Linux، والبرمجيات مغلقة المصدر(Closed Source) لا يُتاح الاطلاع على الكود مثال: نظام Windows وأغلب تطبيقات الشركات.

هل تتشابه البرمجيات ؟؟

بعض البرمجيات قد تؤدي الى مشاكل كارثية ان لم تعمل بشكل صحيح هذه البرمجيات تتطلب تكلفة اضافية في مرحلة الاختبار

يمكن لبعض البرامج التي لا تعمل بشكل صحيح أن تؤدي إلى العديد من المشاكل، بما في ذلك خسارة الأموال، وقد تؤدي حتى إلى الإصابة أو الوفاة.

يمكن تقسيم تكلفة أي مشروع برمجي إلى عدة مراحل منها تكلفة مرحلة التطوير وتكلفة مرحلة الاختبار وضمان الجودة (Testing & Quality Assurance) والتي تصل في المشاريع الكبيرة الى ما يقارب ٤٠ بالمئة من التكلفة الاجمالية:

"31% of projects get cancelled before they are completed, 53% over-run their cost estimates by an average of 189% and for every 100 projects, there are 94 restarts" IBM Report

There are serious problems in the cost, timeliness, maintenance and quality of many software products

So what should we do?
How could we react?

السمات الأساسية للبرمجيات الجيدة

تُعرَّف جودة المنتج البرمجي بأنها مدى قدرة البرمجيات على تلبية **المتطلبات الوظيفية وغير الوظيفية**. وتشمل الجودة معايير رئيسية منها:

قابلية الصيانة (Maintainability)	يجب ان تصمم البرامج بحيث تكون قابلة للإصلاح والتطوير والتعديل تلبية لاحتياجات الزبائن المتغيرة. هذه سمة مهمة لأن تطوير البرنامج هو متطلب لا مفر منه في بيئة أعمال متغيرة. أي ان هذا المعيار يصف مدى سهولة تعديل النظام وإصلاحه وتطويره.
الاعتمادية والامن (Dependabilityand security)	يجب أن تكون البرمجيات جديرة بالثقة (trustworthy) وتتصف بالتوافر وتحمل الأخطاء والاستقرار. تتضمن اعتمادية البرمجية مجموعة من الخصائص بما في ذلك الموثوقية (Reliability) فلا ينبغي أن تتسبب البرامج التي يمكن الاعتماد عليها في أضرار مادية أو اقتصادية في حالة تعطل النظام. وفي (Security) يجب ألا يتمكن المستخدمون الضارون (Malicious Users) من الوصول إلى النظام أو إتلافه. ويعرف بانه قدرة البرنامج على حماية البيانات ومنع الوصول غير المصرح و يشمل: السرية، سلامة البيانات، التحقق من الهوية وعدم الإنكار.
الكفاءة (Efficiency)	تعرف بأنها مدى استخدام البرامج لموارد النظام بشكل فعال. تتضمن الكفاءة زمن المعالجة (Processing Time)، واستخدامية الذاكرة (Memory Utilization)،...
قابلية الاستخدام (Acceptability)	تعرف بأنها مدى سهولة استخدام البرمجية وفهمها من قبل المستخدمين و تشمل: سهولة التعلم والواجهة ودعم المستخدم. يجب أن يكون المنتج البرمجي مقبولاً من قبل المستخدمين المُصمّم لأجلهم.. هذا يعني أنه يجب أن يكون مفهوماً وقابلاً للاستخدام ومتوافق مع الأنظمة الأخرى التي يستخدمونها.
قابلية النقل (Portability)	إمكانية تشغيل البرنامج على منصات أو بيئات مختلفة.

Software Engineering: Definition

هندسة البرمجيات هي فرع من فروع الهندسة يهتم بتصميم وبناء واختبار ونشر وصيانة نظم البرمجيات ذات الجودة العالية.

كما تعرف بانها تطبيق مبادئ الهندسة والطرق العلمية الممنهجة وأفضل الممارسات بهدف إنتاج برمجيات: موثوقة، فعّالة، قابلة للصيانة، قابلة للتطوير تُسلّم ضمن الوقت والتكلفة المحددين.

- a high quality maintainable software
- with a given budget
- before a given deadline (on time)

The systematic approach that is used in software engineering is sometimes called a software process. A software process is a sequence of activities that leads to the production of a software product.

بعبارة أخرى، هو الإطار أو المنهجية التي تتبعها فرق تطوير البرمجيات لإنتاج برامج موثوقة وفعالة.

الاجرائية Process في هندسة البرمجيات هي الطريقة أو منهجية التطوير وهي تركز على "كيف" يُبنى البرنامج وليس على البرنامج نفسه. أما Product فهو البرمجيات النهائية الناتجة عن الاجرائية . وهناك أربع أنشطة أساسية مشتركة في جميع اجرائيات هندسة البرمجيات (Software Processes) مهما اختلف نموذج العملية (Waterfall، Agile، Spiral سنتعرف عليهم لاحقاً) وهذه الأنشطة هي:

١) **تحديد المتطلبات (Requirements Specification / Elicitation)**
   ١) جمع وتحليل احتياجات العميل وتحديد ما الذي يجب أن يقوم به النظام وما هي القيود.
٢) **تصميم النظام (Software Design)**
   ١) وضع خطة وهيكل البرنامج قبل البدء بالبرمجة. وتصميم البنية العامة للنظام (Architecture) وتصميم المكونات (Modules).
   ٢) اختيار الأساليب والخوارزميات المناسبة.
٣) **البرمجة أو التنفيذ (Implementation / Coding)**
   ١) تحويل التصميم إلى كود برمجي فعلي وكتابة التعليمات البرمجية بطريقة قابلة للصيانة والتطوير.
٤) **الاختبار (Testing / Verification & Validation)**
   ١) التأكد من أن البرمجيات تعمل بشكل صحيح وتلبي المتطلبات، كشف الأخطاء وتصحيحها قبل تسليم المنتج، ويشمل الاختبار الوظيفي (Functional) والاختبار غير الوظيفي (Non-Functional)

جَامعة
المَنارة
MANARA UNIVERSITY