Knowledge Area Description for Design (version 0.5)

Guy Tremblay
Dépt. d'Informatique, UQAM
Montréal, Qué.

Overview and rationale

As demanded in the specifications document, I have tried to be as inclusive as possible and, also, to include topics related with quality, measurements and standards. Thus, certain topics have been included in the list of topics even though they may not yet be fully considered as generally accepted (e.g., design metrics, which are formally discussed in a single software engineering book). As for the topic of standards, it was not clear how to include it, especially considering the relatively few number of such standards. Thus, a special "Existing standards" topic section has been created.

Some topics have also been included because it was asked to include topics which might be considered generally accepted within a 3-5 years time frame. In this spirit, I have included the following topics: i) elements related with software architecture ("Software Architecture", Shaw and Garlan, 1996; "Software Architecture in Practice", Bass, Clements and Kazman, 1998), including notions related with architectural styles; ii) the approach to OOD recently presented by the UML group ("The Unified Software Development Process", Jacobson, Booch and Rumbaugh, 1999). Note that although UML is not explicitly mentioned in the design notations, many of its elements are indeed present, for example: class diagrams, collaboration diagrams, deployment diagrams, object diagrams, sequence diagrams, statecharts. Note that no special section explicitly related with tools has been included as it was not clear what should be included in relation with design.

After receiving the comments from the reviewers of version 0.1, a number of modifications have been done and a single breakdown of topics has been selected. This breakdown takes the approach taken in a number of software engineering books, where the notations are presented apart from the strategies and methods that use those notations. Also, in the current breakdown of topics, the concepts related with the topics of architecture and standards have been given explicit sections, whereas the topics related with quality and metrics are grouped in a single section (because metrics can bee seen as tools to evaluate the quality of a design). Finally, the section on strategies and methods has been divided, as done in many references, in a first section that presents general strategies followed by subsequent sections that present the various classes of approaches (data- vs. function- vs. object-oriented). Note that although numerous approaches to OOD have been presented in the literature, we mention only a few, as the one recently proposed by the UML group can be considered a fusion of many existing approaches (Booch, Objectory, OMT).

Before presenting the proposed detailed breakdown of topics, let us present an overview of the major categories accompanied by a brief description.

Basic concepts and principles

The basic concepts and principles which are generally considered fundamental to software design. These are key notions that often reappear, in various forms, in the subsequent topics. They are like a foundation for understanding design.

Design quality and metrics

Quality attributes

The attributes that are generally considered important for obtaining a design of good quality. These include both attributes related with the external behavior of the software (e.g., availability, usability) or related with the quality of the internal structure of the system (e.g., maintainability, simplicity).

Quality assurance

General QA techniques that can be used to ensure the good quality of a design.

Metrics

Formal metrics that can be used to estimate various aspects of the quality of a design. Although some metrics are general, some depend on the approach used for producing the design, for example, structured vs. object-oriented design.

Software architecture

Structures and viewpoints

Various authors have different views of the differents high-level facets of a design that should be described and documented. Some of these views are presented.

Architectural styles

The notion of architectural style -- a paradigmatic architectural pattern that can be used to obtain a high-level organization of software -- has become an important notion of the field of software architecture. This section presents some of the major styles which have been identified by various authors.

Architectural descriptions

The techniques and/or notations that can be used to describe a high-level architecture, either formally (e.g., formal architecture description language) or informally (e.g., patterns). Note that these techniques generally differ from the ones used in a specific architectural design document.

Object-Oriented Frameworks: Description of the notion of framework, which has become important in the context of OO programming.

Design notations

Architectural design

Various notations that can be used to document a specific architectural design.

Detailed design

Various notations that can be used to produce the detailed design of a system.

Design strategies and methods

General strategies

General classes of strategies that can be used to create a design for a system.

Data-structure centered design

Although less popular in North America than in Europe, the work on data-structured design (mostly M. Jackson's work) seems worthy of mention.

Function-oriented design

The classical approach of function/structured design (à la Yourdon/Constantine).

Object-oriented design

Some of the strategies and methods that have been proposed for OO design, including the recently proposed Unified Software Development Process approach.

Detailed breakdown of topics

Basic concepts and principles

Abstraction

Cohesion and coupling

Decomposition

Encapsulation

Information hiding

Interface vs. Implementation

Modularity

Partitioning

Refinement

Separation of concerns

Structure

Design quality and metrics

Quality attributes
- Availability
- Correctness
- Efficiency
- Integrability
- Interoperability
- Maintainability
- Modifiability
- Portability
- Reliability
- Reusability
- Security
- Simplicity
- Testability
- Usability

Quality assurance
- Preliminary design reviews and/or design walkthroughs
- Critical design reviews
- Program design reviews

Metrics
- Graph/cyclomatic complexity metrics
- Structured design metrics
- Object-oriented design metrics

Software architecture

Structures and viewpoints
- Behavioural vs. functional vs. structural vs. data modeling
- Conceptual vs. module vs. code vs. architecture
- Logical vs. process vs. physical vs. development

Architectural styles
- Batch systems
- Blackboards
- Data abstraction and OO organization
- Distributed systems (e.g., CORBA, DCOM)
- Event-based systems
- Interactive systems
- Interpreters
- Layered systems
- Pipes and filters
- Process control
- Repositories
- Rule-based systems

Architectural descriptions
- Module Interconnection Languages
- Architecture description languages
- Patterns

Object-Oriented Frameworks

Design notations

Architectural design
- Class diagrams
- CRC (Class-Responsibilities-Collaborators) Cards
- Deployment and processes diagrams
- Module diagrams
- Structure charts
- Subsystems diagrams

Detailed design
- Collaboration diagrams
- Decision tables and diagrams
- Flowcharts and structured (box) flowcharts
- HIPO diagrams (Hierarchical Input-Process-Output)
- Jackson structure diagrams
- Pseudo-code and PDL (Program Design Language)
- Sequence diagrams
- State transition diagrams and statecharts

Design strategies and methods

General strategies
- Data-oriented decomposition
- Functional decomposition
- Information-hiding
- Object-oriented design
- Stepwise refinement

Data-structure centered design
- Jackson Structured Programming (JSP)
- Jackson System Development (JSD)

Function-oriented design
- SSADM
- Structured design

Object-oriented design
- Coad and Yourdon method
- Fusion
- Meyer's Design By Contract
- Shlaer/Mellor Method
- The Unified Software Development Process' approach to design
- Wirfs-Brock et al.'s Responsibility-driven design

Existing standards

IEEE Standard Glossary of Software Engineering Terminology

IEEE Recommended Practice for Ada as a Program Design Language

IEEE Recommended Practice for Software Design Descriptions

IEEE Guide to Software Design Descriptions

Proposed reference material

In what follows, we suggest reference material for the various topics presented in the proposed breakdown of topics. Note that, for some topics, a number of global references are given for a *non-leaf* topic, rather a specific reference for each particular leaf topic. This seemed preferable because some of these topics were discussed in a number of interesting references.

Also note that the number of references is slightly above the suggested 15 references indicated in the specification document for the knowledge area descriptions. In our case, this number is both only slightly above (17) and largerly above. Exactly 17 references have been indicated, of which four (4) are references to existing IEEE Standards. These latter were included for completeness but not all of them are directly relevant (e.g., IEE87a, IEE91). However, if complete and detailed references had been given, the number of references would have been *largely* above the suggested 15: in 3 cases, the indicated references are to publications from the IEEE Computer Society Press which are in fact collections of papers previously published elsewhere. These references are used mainly for the topics related with the various design notations and methods.

Basic concepts and principles

[BMR+96, chap. 6]
[Jal97, chap. 5]
[Pfl98, chap. 5]
[Pre92, chaps. 13 and 23]
[YC79]

Design quality and metrics

Quality attributes
- [BCK98, chap. 4]
- [BMR+96, chap. 6]
- [Jal97, chap. 5]
- [IEE91]

Quality assurance
- [BCK98, chap. 10]
- [Jal97, chap. 7]
- [Pfl98, chap. 5]

Metrics
- [Jal97, chaps. 5, 6 and 7]
- [Pre98, chaps. 18 and 23]

Software architecture

Structures and viewpoints
- [BCK98, chap. 2]
- [BMR+96, chap. 6]
- [Pfl98, chap. 5]

Architectural styles
- [BCK98, chap. 2]
- [BMR_96, chap. 2]
- [Pfl98, chap. 5]
- [SG96, chaps. 2 and 4]

Architectural descriptions
- Module Interconnection Languages [FW83, part V][SG96, chap. 7]
- Architecture description languages [BCK98, chap. 12][SG96, chaps. 6 and 7]
- Patterns [BCK98, chap. 12][BMR+96][Pre92, chap. 21]

Object-Oriented Frameworks [BMR+96, chap. 6]

Design notations

Architectural design
- [DT97, chap. 4]
- [Jal97, chaps. 5 and 6]
- [JBR99, chap. 9]
- [Pre92, chap. 14]
- [TM93, chap. 5]
- [WBWW90]
- [YC79]

Detailed design
- [DT97, chap. 4]
- [FW, parts III and VII]
- [JBR99, chap. 9]
- [Pre92, chap. 14]

Design strategies and methods

General strategies
- Data-oriented decomposition [FW83, part V]
- Functional decomposition [FW83, part V][Pre92, chaps. 13 and 14]
- Information-hiding [FW83, part V][Pre92, chap. 13]
- Object-oriented design [FW83, part VI][Pre92, chaps. 19 and 21]
- Stepwise refinement [FW83, part VII]

Data-structure centered design
- Jackson Structured Programming (JSP) [DT97, chap. 4][FW83, part VII][TM93, chap. 5]
- Jackson System Development (JSD) [DT97, chap. 4][FW83, part III][TM93, chap. 5]

Function-oriented design
- SSADM [TM93, chap. 5]
- Structured design [Jal97, chap. 5][Pre92, chap. 14][YC79]

Object-oriented design [Hut94][Jal97, chap. 6][Pre92, chap. 21]
- Coad and Yourdon method [Pre92, chap. 21][Hut94]
- Fusion [Hut94]
- Meyer's Design By Contract [Hut94][Pfl98, chap. 5]
- Shlaer/Mellor Method [Hut94]
- The Unified Software Development Process' approach to design [JBR99]
- Wirfs-Brock et al.'s Responsibility-driven design [Hut94][Pre92, chap. 21][WBWW90]

Existing standards

IEEE Standard Glossary of Software Engineering Terminology [IEE91]

IEEE Recommended Practice for Ada as a Program Design Language [IEE87a]

IEEE Recommended Practice for Software Design Descriptions [IEE87b]

IEEE Guide to Software Design Descriptions [IEE93a]

Matrix of topics vs. reference material

Note: A number in the matrix entry indicates the appropriate chapter number. A "*" indicates a general reference (no specific chapter).

	BCK 98	BMR +96	DT 97	FW 83	Hut 94	Jal 97	JBR99	Pfl 98	Pre 92	SG 96	TM 93	WBW W90	YC 79
Basic conc. and princ.		**6**				**5,6**		**5**	**13,23**				
Abstraction		6				5		5	13,23				
Cohes. and coupl.		6				5,6		5	13				
Decomposition								5					
Encapsulation		6				6			23				
Info. hiding		6				6		5	13,23				
Interf. vs. Impl.		6											
Modularity		6				5		5	13				
Partitioning						5			13				
Refinement						5			13				
Sep. of concerns		6											
Structure									13				
Design qual. and m.													
Quality attr.	4	6				5				7			
Availability	4												
Correctness						5							
Efficiency	4	6				5							
Integrability	4												
Interoperability		6											
Maintainability													
Modifiability	4	6											
Portability	4												
Reliability		6											
Reusability	4	6								7			
Security	4												
Simplicity						5							
Testability	4	6											
Usability	4												
Quality assurance	10					5		5					
PDR and/or DW	10					7		5					
CDR						7		5					
Program DR								5					
Metrics						5-7			18, 23				
Graph metrics						7			18				
Struct. des. m.						5			18				
OO des. m.						6			23				

Software architecture													
Structures and viewpoints	2	6	4					5					
Beh./func./str./data	2		4										
Conc./mod./code./ arch.	2	6						5					
Log./proc./phys./dev.	2	6											
Architectural styles	6, 8	2						5		2,4			
Batch systems										4			
Blackboards		2											
Data abs. and OO										2			
Distr. syst.	8	2						5					
Event syst.								5		2			
Interactive syst.	6	2											
Interpreters								5		2			
Layered syst.		2						5		2,4			
Pipes and filters		2						5		2			
Proc. control								5		2			
Reposit.								5		2,4			
Rule syst.										2			
Archi. descr.	12	1		V					21	6,7			
MIL				V						7			
Arch. descr. lang.	12									6,7			
Patterns	12	1							21				
OO Frameworks		6											
Des. notations													
Arch. design			4		12	5,6	9		14		5	*	*
Class diagr.						6	9						
CRC Cards					21							*	
Deploy. diagr.							9						
Module diagr.							9						
Struct. charts			4			5			14		5		*
Subsyst. diagr.							9						

Detailed design			4	III, VII		7	9		14				
Collab. diagr.							9						
Decision diagr									14				
Flowcharts				VII					14				
HIPO diagr.				III									
Jackson struct. diagr.			4										
PDL				VII		7			14				
Sequence diagr.							9						
STD			4			7	9						
Design strategies and methods													
General strategies			4	V-VII									
Data-oriented dec.			4	V									
Funct. dec.				V									
Info.-hiding				V									
OO design				VI									
Stepw. ref.				VII									
Data-str. design			4	III, VII							5		
JSP			4	VII							5		
JSD			4	III							5		
Funct.-or. design						5			14		5		*
SSADM											5		
Struct. design						5			14				*
OO design					4,7,19, 21		9	5				*	
Coad and Yourd.					4								
Fusion					7								
D BC								5					
Shlaer/Mellor					19								
UML Proc.							9						
Resp.-driven					21							*	

References

[BCK98]
L. Bass, P. Clements, and R. Kazman.
Software Architecture in Practice.
SEI Series in Software Engineering. Addison-Wesley, 1998.

[BMR+96]
F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal.
A System of Patterns.
Wiley, West Sussex, England, 1996.

[DT97]
M. Dorfman and R.H. Thayer.
Software Engineering.
IEEE Computer Society Press, Los Alamitos, CA, 1997.

[FW83]
P. Freeman and A.I. Wasserman.
Tutorial on Software Design Techniques, fourth edition.
IEEE Computer Society Press, Silver Spring, Md, 1983.

[Hut94]
A.T.F. Hutt.
Object Analysis and Design --- Description of Methods.
John Wiley & Sons, New York, 1994.

[IEE87a]
An american national standard --- IEEE recommended practice for Ada as a program design language.
IEEE Std 990-1987, IEEE, New York, NY, 1987.

[IEE87b]
An american national standard --- IEEE recommended practice for software design descriptions.
IEEE Std 1016-1987, IEEE, New York, NY, 1987.

[IEE91]
IEEE standard glossary of Software Engineering Terminology.
IEEE Std 610.12-1990, IEEE, New York, NY, 1993.

[IEE93a]
IEEE guide to software design descriptions.
IEEE Std 1016.1-1993, IEEE, New York, NY, 1993.

[Jal97]
P. Jalote.
An integrated approach to software engineering, 2nd ed.
Springer, New York, NY, 1997.

[JBR99]
I. Jacobson, G. Booch, and J. Rumbaugh.
The Unified Software Development Process.
Addison-Wesley, Reading, Ma, 1999.

[Pfl98]
S.L. Pfleeger.
Software Engineering --- Theory and Practice.
Prentice-Hall, Inc., 1998.

[Pre92]
R.S. Pressman.
Software Engineering --- A Practitioner's Approach (Third Edition).
McGraw-Hill, Inc., 1992.

[SG96]
M. Shaw and D. Garlan.
Software architecture: Pespectives on an emerging discipline.
Prentice Hall, Englewood Cliff, NJ, 1996.

[TM93]
R.H. Thayer and A.D. McGettrick.
Software Engineering: A European Perspective.
IEEE Computer Society Press, Los Alamitos, CA, 1993.

[WBWW90]
R. Wirfs-Brock, B. Wilkerson, and L. Wiener.
Designing Object-Oriented Software.
Prentice-Hall, Prentice-Hall, NJ, 1990.

[YC79]
E. Yourdon and L. Constantine.
Structured Design.
Yourdon Press, Englewood Cliffs, NJ, 1979.