APEGBC Guide for the Presentation and Evaluation of Software Engineering Knowledge and Experience

For the purpose of registering Professional Engineers in the discipline of software engineering, this document defines objective minimal requirements of knowledge and experience. The document is intended to be used by both:

- the applicants to analyze, organize and present their knowledge and experience
- the association to evaluate an application, on paper and during interviews.

We define a *capability* demonstrated by an individual as the combination of his or her *knowledge* and *experience*. A capability is acquired by formal initial education, by learning on the job and continuous professional development, and by applying knowledge and theory on real projects, first under supervision of other engineers.

The capabilities to be demonstrated by applicant engineers fall into 2 main categories:

- *General capabilities*, such as management skills, communication skills, ethics, professional practices, etc., which are covered by the CCPE rules.
- Specific *software engineering capabilities*, which are further described in this document.

There are 6 main are of software engineering capabilities:

1. Software Requirements
2. Software Design and Construction
3. Software Process Engineering
4. Software Quality
5. Software Assets Management
6. Management of Software Projects

Each main area defines software engineering capabilities. Some capabilities are further decomposed into sub-capabilities.

To each capability area (and capability where applicable), can be associated:

- the fundamental knowledge or theory, indirectly by pointing to the model academic curriculum for Software Engineering,
- the applicable industry standards
- the recognized industry practices and tools,
- and a level of mandatory or optional experience on real projects..

An applicant is invited to describe his or her experience by showing, for a number of selected software engineering projects, which capabilities and subcapabilities he or she has personally exercised on the project.

Optional capabilities are indicated by square brackets, for which not applicants may have had the opportunity of demonstrating personal experience.

1. Software Requirements

This capability is concerned with the acquisition, specification, analysis and management of software requirements.

Capabilities:

Requirements elicitation:
  Process, technique and tools for identification of the context, the users needs, the constraints.
Requirements specification
  Description software functional and non functional requirements
  [Formal specification:
  use of formal methods for the description of software requirements]
Requirements validation
  Prototyping, review, modeling
Requirements management
  Prioritization, tracing to design and test, etc.

Applicable standards:
IEEE 830-1998

2. Software Design and Construction

This capability is concerned with the techniques used to define and describe how the software will be built, and the construction and integration of a software product.

Capabilities:
Software design
  Methods, techniques, tools to make and capture decisions as to how the software will be built
  Software Architecture
  Object-oriented methods, functional methods, formal methods
  Notations (OMT, SADT, UML, etc.) for software design
  User Interface design
  [Security and safety design]
  Data engineering
  [Real-time software design]
  [Performance engineering]
Programming
Integration
  Gradual assembly of software components up to a complete product
User documentation
[Portability and reuse]

Applicable standards:
IEEE 1016, OMG's UML, IEEE 1028, IEEE 1063

3. Software Process Engineering

This capability area is concerned with the definition, measure and improvement of the software engineering process.

Capabilities:
Process definition and description
Techniques for description of the software process: e.g. IDEF0
Process measurement
For the assessment of the effectiveness of the process
Process improvement
Eviolution of the process; framework such as CMM, or SPICE

Applicable standards:
ISO/IEC 12207, ISO/IEC 15504, IEEE 1074

4. Software Quality and Testing

This capability area is concerned with the techniques and approaches to assess and ensure the software product adheres to standards, is conform to its requirements, and meets its end-user needs. Testing of software products is one of the ways to assess quality.

Capabilities:
Types and levels of testing
Taxonomy of tests, by approach and by purpose
Software Measurement
Verification and Validation
Software reviews and audits

Applicable standards:
ISO 9001-3, ISO 9126, IEEE 730, IEEE 829, IEEE 1008, IEEE 1044

5. Software Assets Management

This capability area is concerned with the safeguarding and control of the various artifacts or workproducts of software development, their version, variants, and the control of the changes done to these workproducts.

Capabilities:
Configuration management
Identification of software elements, their version and history, their relationships, archival

Change control

Identification and tacking of changes to software elements, including problem reports, or evolution

Release management and delivery

Preparation of software for release, distribution, installation and deployment into operation

Applicable standards:
IEEE 828

6. Management of Software Projects

This capability area is concerned with the aspects of engineering management that are specific to software engineering: approaches and methods for planning and measuring and controlling the progress of software projects.

Capabilities:
Project planning and scheduling

Types of process lifecycles, work breakdown structure, estimations of workload

Process enactment
Project tracking

Project Metrics of progress, quality, expenditure, etc.

Software risk management
Software maintenance

Applicable standards:
ISO/IEC 15504, MiL-STD-498

7. Related disciplines

Although not strictly software engineering capabilities, the following capabilities are part of the basic skills and techniques a software engineer must master.

Distributed systems and networking
Databases
Computer graphics

Sources:

- *National Guidelines for Admission to the practice of Engineering in Canada* CCPE doc. G01-9,
- *IEEE/ACM Guide to the Software Engineering Body of Knowledge*, version 0.7 (4/2000)
- Ian Sommerville, *Software Engineering*,5th ed. Addison-Wesley, (1995)
- *IEEE Software Engineering Standards,* Volume 1 to 4 (1999)

Suggested presentation of experience

Applicant can present their experience by first giving a list of software engineering projects to which they have contributed, giving briefly date, nature of the project. Then a table may be used to show in which project they have demonstrated the various capabilities, complementing as necessary by an indication of the specific tools, methods, techniques or standard that was applied.

Capability to demonstrate	Project name or ID (from experience)	Tool, technique, standard used
1. Software Requirements		
Requirements elicitation:		
Requirements specification		
[Formal specification		
Requirements validation		
Requirements management		
2. Software Design and Construction		
Software design		
Software Architecture		
Object-oriented methods, functional methods, formal methods		
Notations (OMT, SADT, UML, etc.) for software design		
User Interface design		
[Security and safety design]		
Data engineering		
[Real-time software design]		
[Performance engineering]		
Programming		
Integration		
User documentation		
[Portability and reuse]		
3. Software Process Engineering		
Process definition and description		
Process measurement		
Process improvement		
4. Software Quality and Testing		
Types and levels of testing		
Software Measurement		
Verification and Validation		
Software reviews and audits		
5. Software Assets Management		
Configuration management		
Change control		
Release management and delivery		
6. Management of Software Projects		
Project planning and scheduling		
Process enactment		

Project tracking		
Software risk management		
Software maintenance		
7. Related disciplines		
Distributed systems and networking		
Databases		
Computer graphics		

Example:

May 1996-Oct. 1997. FineImage Corp.

Project **SeaBottom**: analysis of underwater images to detect ferro-nickel nodules. Approx. 18 person month.

Oct 1997- June 1999. FineImage Corp.

Project **SkyHigh**: controller for scanner imaging of pollution clouds. Redesign of an older system. 50 person-month.

Etc.

Capability to demonstrate	**Project name or ID (from experience)**	**Tool, technique, standard used**
1. Software Requirements		
Requirements elicitation	--	
Requirements specification	Seebottom	IEEE 830 + Use cases
Formal specification	Skyhigh	VDM (partial)
Requirements validation	SkyHigh & Seabottom	Reviews and prototyping
Requirements management	Skyhigh	Database, DOORS
2. Software Design and Construction		
Software design		
Software Architecture		Wright + UML
Object-oriented methods, functional methods, formal methods	Seabottom Sky Plus	OMT SADT
Notations (OMT, SADT, UML, etc.)	SkyHigh & Seabottom	OMT
User Interface design	MiniCAD	User Centered design
[Security and safety design]	Skyhigh	DO-178B
Etc.		