Recommended Skills and Knowledge for Software Engineers

-Steve Tockey

Software Engineering: The Development Process, Vol I, Chapter 1

Presented by Gargi Chipalkatti

(Software Engineering II - EEL 6883)

Goal

- To propose that computer science and software engineering are distinct but related subjects
- To clearly define the relationship between computer science and software engineering
- To recommend a set of skills and knowledge that could serve to distinguish proficient software engineers

Definitions

Science –

a department of systematized knowledge as an object of study; a system of knowledge covering general truths or the operation of general laws esp. as obtained and tested through scientific method

Engineering –

… the profession in which a knowledge of the mathematical and natural sciences gained by study, experience, and practice is applied with judgment to develop ways to utilize, economically, the materials and forces of nature for the benefit of mankind

Definitions Continued…

Computer Science –

a department of systematized knowledge ***about computing*** as an object of study; a system of knowledge covering general truths or the operation of general laws of ***computing*** esp. as obtained and tested through scientific method

Software Engineering –

… the profession in which a knowledge of the mathematical and ***computing sciences*** gained by study, experience, and practice is applied with judgment to develop ways to utilize, economically, ***computing systems*** for the benefit of mankind

Inferences from the Definitions

Science –

Concerned with the continued expansion of the body of the theoretical knowledge about a certain discipline

Engineering –

Concerned with practical and economical application of that same theoretical knowledge

Equations

Engineering	=	Scientific Theory	+	Practice	+	(Engineering) Economy
Software Engineering	=	Computing Theory	+	Practice	+	(Engineering) Economy

Definitions Continued…

Skills –

a learned power of doing something competently: a developed aptitude or ability

Knowledge –

facts or ideas acquired by study, investigation, observation, or experience

Economy –

thrifty and efficient use of resources

Inferences Continued…

- Computer Science and Software Engineering, both deal with computers, computing, and software
- Science of computing, as a Body Of Knowledge is at the core of Computer Science and Software Engineering
- Computing Science is concerned with computers, computing, and software as a system of knowledge, together with the expansion of that knowledge
- Software Engineering should be concerned with the application of computers, computing, and software to practical purposes, specifically the design, construction, and operation of efficient and economical computing systems

Aim of the Recommended Software Engineering Skills and Knowledge

- Enable the proficient design, construction, and maintenance of cost-effective computing systems
- Characterize "proper professional practice" for software engineers (Non-awareness is believed to be correlated with either a decrease in an individual's proficiency or a decrease in cost-effectiveness of the resulting software)
- Person who possesses such skills and knowledge should be considered more valuable to a software organization than a person who does not

Notes

- Recommendation describes a vision of an “ideal” software engineer
- Individuals expected to have at least broad, but possibly shallow, proficiency across many skill and knowledge kernels, and much more detailed proficiency in one or more specific areas of interest to them
- At least one software engineer on each software project ought to be proficient in each relevant skill or knowledge kernel
- Software team as a whole should leverage off the proficiencies of the individual team members
- Knowledge of computing theory allows engineers to –
  - Propose a larger number of diverse designs than would otherwise be possible
  - Identify and discard proposed designs that could not work (because they violate some known theory) earlier than otherwise possible

Computing Theory

- Knowledge of computing theory allows engineers to –
  - Propose a larger number of diverse designs than would otherwise be possible
  - Identify and discard proposed designs that could not work (because they violate some known theory) earlier than otherwise possible
- Computer Science and Discrete Mathematics provide the relevant theory of computing

Recommended Computing Theory Skills and Knowledge

- Programming language concepts
- Data structure concepts
- Database system concepts
- Relational algebra
- Operating system concepts
- Software architectures
- Computer architectures
- Automata theory / Petri nets
- Computability theory / Turing machine theory
- Complexity theory
- Linguistics and parsing theory
- Computer graphics
- Set theory
- Predicate logic
- Formula proofs
- Induction

Software Practice

- Software Practice addresses the day-to-day issues encountered in industrial software settings
- This subject area is broken down into several sub-areas
  - Software Product Engineering
  - Software Quality Assurance (SQA)
  - Software Product Deployment
  - Software Engineering Management
- Skills and knowledge areas listed, apply not only to software maintenance, also to software development

Recommended Software Product Engineering Skills and Knowledge

- Task kick-offs / Previews / Readiness reviews
- Peer Reviews / Inspections / Walkthroughs
- Software project audits
- Requirements tracing/Quality Function Deployment (QFD)
- Software testing techniques
- Proofs of correctness
- Process definition and process improvement techniques
- Statistical process control
- Technology innovation

Recommended Software Quality Assurance (SQA) Skills and Knowledge

- Requirements / Analysis / Requirements engineering
- Software design
- Code optimization / Semantics preserving transformations
- Human-computer interaction / Usability engineering
- Specific programming languages
- Debugging techniques
- Software-software and Software-hardware integration
- Product family engineering techniques / Reuse techniques
- CASE/CASE tools

Recommended Software Product Deployment Skills and Knowledge

- User documentation techniques
- Product packaging techniques
- System conversion techniques
- Customer support techniques
- General technology transfer issues

Recommended Software Engineering Management Skills and Knowledge

- Risk assessment and risk management
- Project planning
- Alternative software lifecycles
- Organizational structures
- Organizational behavior
- Project tracking and oversight
- Cost management / Schedule management / Resource management
- Metrics / Goal-Question-Metric paradigm / Measurement theory
- Configuration management / Change management
- Supplier/Subcontract management
- Effective meeting skills
- Effective communication skills
- Negotiation Skills

Engineering Economy

- Ultimate aim of engineering is to create the most income from the least expense, thus maximizing profit
- Importance of Estimating the cost of programming projects
- Relevance of engineering economy to software engineering

Recommended Engineering Economy Skills and Knowledge

- Time-value of money (interest)
- Economic equivalence
- Inflation
- Depreciation
- Income taxes
- Decision making among alternatives
- Decision making under risk and uncertainty
- Evaluating replacement alternatives
- Evaluating public activities
- Break-even
- Optimization

Customer and Business Environment

- Necessary Knowledge for Developing Cost-Effective Products and Services
    - Who is the customer and what is their business?
    - What do they use our products and services for?
    - When, where, and why are our products and services used?
    - Are our products and services being used in a way different than originally intended? If so, why?
    - How do our products and services affect the customers' business?
    - What external restrictions or regulations impact the ability to deliver products and services to the customer(s)?

Recommended Customer and Business Environment Skills and Knowledge

- Customer satisfaction assessment techniques
- Competitive benchmarking techniques
- Technical communication
- Intellectual property law
- Ethics and professionalism

Practical Implications

- Software Industry has a distinct need for –
- A practitioner who will be able to rapidly assume a position of substantial responsibility in an organization
- Skills and knowledge recommended can form the basis of a standardized curriculum for software engineering degrees

Conclusions

- Difference as well as relationship between Computer Science and Software Engineering put forth
- Set of skills and knowledge recommended, that would serve to improve the standard of software engineers and new graduates

References

1. Pierre Bourque, Robert Dupuis, Alain Abran, James W Moore, Leonard Tripp, Karen Shyne, Bryan Pflug, Marcela Maya, Guy Tremblay, “Guide to the Software Engineering Body of Knowledge: A Straw Man Version”, University du Quebec a Montreal, Canada, September, 1998, (http://www.lrgl.uqam.ca/).
2. Thomas Hilburn, Donald Bagert, Susan Mengel, Dale Oexmann, “Software Engineering Across Computing Curricula”, (http://erau.db.erau.edu/~hilburn/sei-educ/guide-pub.htm).
3. Timothy C Lethbridge, “A Survey of the Relevance of Computer Science and Software Engineering Education”, Proceedings of the 11th Conference on Software Engineering Education and Training (CSEE&T ’98), IEEE Computer Society Press, February, 1998.
4. A. J. Cowling, “A Multi-Dimensional Model of the Software Engineering Curriculum”, Proceedings of the 11th Conference on Software Engineering Education and Training (CSEE&T ’98), IEEE Computer Society Press, February, 1998.
5. Steve Tockey, “A Missing Link in Software Engineering”, IEEE Software, Vol 14, No 6, November/December, 1997.