

Competencies, Patterns, and Courses: Oh My!¹

Problem Statement

The immense volume of relevant, even critical, information necessary to be truly educated in the area of computers and information technology is precipitating a crisis of identification in the computer world.

Before 1990 a clean division existed between electrical engineering combined with computer engineering; computer science; and information systems. The first focused on the hardware, the second on computing, and the third on user applications.

Post-1990 EE/CE remains; CS has fractured into theory, systems, and software engineering; and IS has split into infrastructure and applications. The entire educational system is in turmoil trying to decide which area to focus on – along with what it should be named² – or how to mix-and-match from among the areas to create a niche – still with the problem of naming.

As the amount of “in-discipline” knowledge and information has exploded, it has been necessary to exclude vast areas of critical knowledge – liberal arts, social sciences, even math and laboratory sciences, simply because there were not enough credit hours available.

Problem: how to define a discipline and a curriculum that provides an education with sufficient breadth and depth while fostering innovation and creativity?

Agonizing Re-appraisal³

Fiat. To provide focus for this discussion the realm of computing will arbitrarily be divided into three parts:

- Machination⁴ - understanding, improving, and representing the machine. This incorporates EE/CE, theory, and systems up to the boundary established with virtual machines, compilers/interpreters, device drivers, and most embedded software. If software engineering has any relevance, it would be in this area.
- Acolytism – primarily administration to the infrastructure with the addition of issues around security. Acolytes take the machines produced by the Machinators, connect them together, and keep them running. A lot of “certification” curricula would fall into this area.
- Reality Construction – exploiting the plasticity of software to provide myriad disguises for hardware and systems; enabling them to support and participate

¹ [Yes, I know it doesn't have the same rhythm as “Lions and tigers and bears, oh my!” from the Wizard of Oz, but it stuck in my head anyway.]

² Everyone wants to be identified as CS, because that is phrase with the cachet.

³ Thank you Mao Tse Tung for the eloquent phrase.

⁴ This is a neologism, so don't interpret it according to Webster.

in human nurturing and human augmenting worlds of work and play; worlds limited only by the genius⁵ of innovation and creativity.

The dictate implied by this paragraphs label: simply stated, our focus will be on Reality Construction.

L'uomo Universale. Even with a focus on reality construction, the amount of knowledge required to be a master of the discipline is staggering. Thought of in terms of a course list (taking care to include philosophy, music, art, anthropology, mathematics, scientific and metaphoric reasoning, biology, physics, history, etc. etc.), the number of distinct courses could number in the hundreds – far exceeding the 45 + 18 + 15 (~78) allowed in a typical undergraduate, masters, and Ph.D. education. A master of reality construction must, it appears, be a polymath, or at minimum a Renaissance Man⁶.

The Humanist ideal of l'uomo universale - the infinitely versatile 'universal man,' educated in all branches of human knowledge and capable of producing innovations in many of them," would be a reasonable equivalent of a master reality constructor.

Leonardo, an exemplar of the Humanist ideal, said, "*Facile cosa e farsi universale*," it is easy to become universal. In context, what he meant was, it is easy to become universal if one can perceive and understand the inter-connecting patterns that underlay the efflorescence of apparently distinct forms and processes.

"...you have nothing to lose but your chains!" In addition to re-assessing the subject matter of this new discipline, it is necessary to focus critical attention on delivery mechanisms. Disciplines, subjects, courses, credit hours, contact hours, grades, tenure, semesters/quarters, and arbitrary deadlines (e.g. four years to the BS/BA) are odious shackles indeed!

The ideal learning-teaching environment would be analogous to the bottega where Leonardo worked and studied:

- A "storefront" where goods and services are produced and delivered to paying customers.
- A workshop simultaneously engaged in the craft, in building the tools and discovering the techniques that advance and support the craft, and teaching that craft to apprentices.
- A place noisy with multiple projects and activities; walls and benches covered with works in progress and exemplars of the craft.
- A place filled with the tools of the craft (add computers and digital displays to the easels, brushes, hammers, chisels, carving, forges, kilns, model making, etc. tools found in a typical bottega). With room for lounging and eating facilities as well.
- An intellectual center that was a "must visit" for masters, scientists, and thinkers visiting the area, overseen (deliberately avoiding the term managed) by local masters and journeymen.
- A fountain of innovation and creativity!

⁵ Genius used in this context according to its roots in "family" and "spirit" or "essence" – genius as an ineffable quality, expressed in skillful application of an art, arising from membership in a human "family."

⁶ Forgive the gender bias – it is an historical anachronism.

- An environment and atmosphere that is very self-consciously multi- and inter-disciplinary; that mixes theory and practice almost without differentiation.
- A place full of music, especially “after hours.”
- A place to share food and drink (and perhaps sleep).

Would it not be a good idea to revisit the domain of expected knowledge; to identify discrete “atoms” of knowledge or practice; to identify structure and process patterns that inter-connect those atoms as “molecules”; to identify problem, pedagogical, or cognitive patterns that associate molecules into experiences or solutions; “threading” patterns that suggest learning progressions; and “packaging” patterns that provide a framework mentoring and learning interactions?

The Workshop

Audience: If you are intrigued by the idea of re-inventing computing education, this workshop is for you. If you are strangely attracted by parts of the introductory (purple) prose and the ideals and biases presented therein, this workshop is for you. If you are an educator that is “mad as hell and not going to take it anymore,” this workshop is for you. If you are a hardcore defender of the status quo, this workshop is for you. (Someone has to keep the idealists anchored in reality.)

Activities: We will simultaneously explore and discuss three inter-related threads:

- 1) A catalog (really a wiki) of “atoms” – discrete bits of knowledge, technique, skill, and/or practice that would be expected of anyone bearing the title, “Master of Reality Construction.” (Master of Software Design if we must be more prosaic.) We will label these atoms, ‘competencies,’ even though that term has connotations of vocational education (intellectual snobbery really) and therefore is held in disfavor by some.
- 2) Discovery and documentation of inter-connection patterns – the cognitive “chunks”⁷ that allow one to comprehend seemingly unrelated areas of knowledge, the kind of pattern on which Leonardo relied.
- 3) Discussion of packaging options – using pedagogical and Design⁸ patterns used to aggregate atoms and chunks and guide student acquisition of the package. Ideally, these will be conceived as if they could be delivered in a bottega environment, but pragmatically as if they had to be delivered in a three-credit semester course.

Dave West will interpret the role of “Hot Topic Leader” as agent provocateur and minimalist moderator.

⁷ After “Miller’s Magic Number Seven, Plus or Minus Two,” the psychology paper discussing one way in which human’s can overcome limits of attention and memory with regard apparently discrete bits of knowledge.

⁸ Use of the upper-case D to differentiate from the design patterns of programming.