Designing across discipline borders: obstacle or option?

Kees van Overveld

Educating technological designers is difficult. An educational curriculum should provide both sufficient discipline-related skills, and cross-disciplinary skills. In order to argue about the balance between the two, we speculate on the relation between disciplines and application domains, and we give some considerations as to what disciplinary baggage gives the best preparation for prospect interdisciplinary designers. Finally, we hint at a particular role for software designers in the process of designing across discipline borders.

Introduction: Domains and Disciplines

In the past, life was easy. Professions could be easily distinguished. If John Smith was trained as a carpenter, it was clear what to expect from him. You should see John if you needed a table or a garden fence, but if you were suffering from a headache you should go to his cousin Peter who studied medicine. And if you wanted to divorce from your husband, you should consult his other cousin Charley who went to law school.

Lawyer, medical doctor and carpenter are professions with a relatively constant definition. They have quite a long-standing history, and their educational programs stayed largely constant for extended periods.

Nowadays, examples of such stable connections between education, career, and professional activities begin to be rare. Over the last, say, 10 years we have seen both a multitude of new professions, and a multitude of new educational programs. Further, the mappings between education and professional career, between initial and final job within a career, and between a job and the tasks and activities within that job, are no longer one-to-one.

In order to argue about the causes and consequences of this decreasing transparency in education programs, careers, jobs, tasks and activities, we need some vocabulary. In particular, we want to talk about *domains* and *disciplines*.

We call a coherent set of social needs or desires a *domain*. For example, transport is a domain. It comprises the desires of people to go from A to B, the need to maintain roads, the desire to have a dense and reliable network of petrol stations, et cetera. The adjective 'social' refers to the fact that any need or desire is attributed to (a group of) people. The type of coherence in a domain is often historically determined, and it may have a degree of arbitrariness. It may change over time, and as a result domain borders are fluid. Domains are not necessarily disjoint[1]. As an example, the various departments in a national government as a partition of the national concerns form a set of domains.

We call a coherent body of (professional) knowledge, skills and attitudes a *discipline*. For example, physics is a discipline. A discipline refers to a topic that is studied in a scientific community. Although to some extent historic whimsicalities influence the scope and contents of any given discipline, the coherence in a discipline also results from knowledge

[1]Overlapping domains are a frequent source of envy among professionals with different disciplinary backgrounds. Social needs and desires that do not (yet) belong to a well-recognized domain, on the other hand, often go unnoticed for a long while and once they are accepted they may cause new disciplines to occur.

hierarchies (see below). Disciplines, with some delay, give rise to academic curricula, and they can therefore be mapped approximately to the disciplinary baggage of recently graduated practitioners in any field. Notice, therefore, a self-stabilizing mechanism, such as depicted in Figure 1.

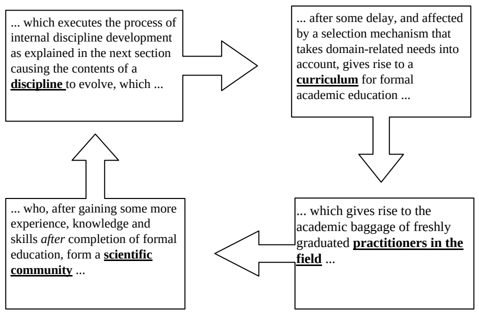
... which executes the process of internal discipline development as explained in the next section causing the contents of a **discipline** to evolve, which ...
... after some delay, and affected by a selection mechanism that takes domain-related needs into account, gives rise to a **curriculum** for formal academic education ...
... which gives rise to the academic baggage of freshly graduated **practitioners in the field** ...
... who, after gaining some more experience, knowledge and skills *after* completion of formal education, form a **scientific community** ...

Figure 1: The evolution of a discipline.

We will study the nature and the relation between domains and disciplines in the section below, and we zoom in for the case of technological disciplines. Then we discuss the (mis-)match between domains and disciplines, and we conclude with a possible approach to remedy the problem.

The nature of disciplines and their relation to domains

Disciplines, as defined above, possess a structure that is vaguely hierarchic. A body of knowledge K is rarely self-contained. It assumes fragments of knowledge that are outside that body; these fragments may be part of another body of knowledge, say K'. For example: designing wireless telecommunication systems assumes knowledge of modulation. The topic of 'modulation' assumes knowledge of high frequency oscillators. The topic 'high frequency oscillators' assumes knowledge of linear networks. The topic of 'linear networks' assumes knowledge of linear algebra. The topic of 'linear algebra' assumes basic algebra. Finally, the topic 'basic algebra' assumes 'logic'. Here, the proposition 'A assumes B' is considered to be a partial ordering. It means that in order to actively use the knowledge (and the implied skills and the implied attitudes) in A, it is necessary to believe that the knowledge in B is both available and true; further, a practitioner of the knowledge in B is, for his present purpose, not interested in, nor dependent of the knowledge in A. This partial ordering model for bodies of knowledge is overly simple and it has some fundamental problems[2] but it gives us a convenient vehicle to argue about the complexities and possible remedies of multi-disciplinary design. The above example gives rise to the fragment in figure 2 of what we will call the DAG of knowledge or K-DAG (DAG = directed a-cyclic graph). A node in the K-DAG is a small chunk of related knowledge; an arc represents the 'assumes'-relation.

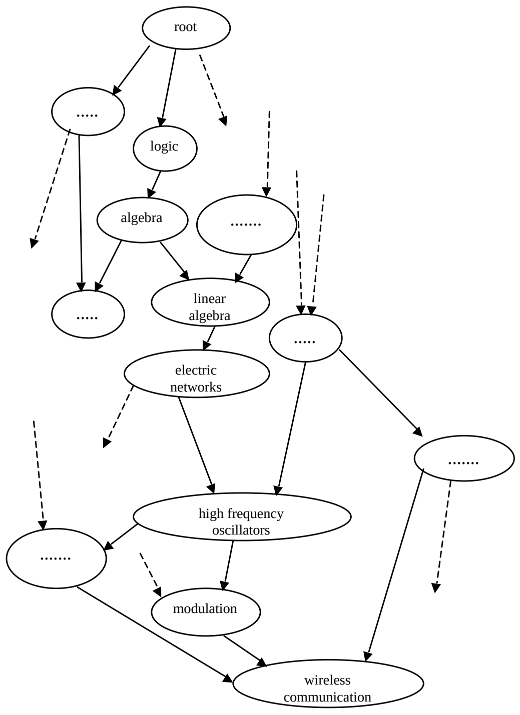

Figure 2: A fragment of the K-DAG.

[2]To mention but a few: it is not certain what constitutes the topmost chunk of knowledge (labeled 'root' in figure 2); it is not clear if there is such a thing as 'a consistent chunk of root knowledge'; it is not certain if loops in the assumptions can always be avoided, and it is not certain if the 'A assumes B' relation, apart from the knowledge components, also can be formulated for skill and attitude components of a discipline.

Nodes in the DAG close to the root represent generic notions that underlay much of scientific practice, such as ’observation’, ’hypothesis’, ’explanation’, ’model’, ’definition’, ’approximation’, ’causal relation’, as well as formal reasoning and logic. Much of what is usually called ’common sense’, as well as many abstract patterns for problem solving also reside in these nodes. Few disciplines, except from philosophy, some sub-fields of mathematics and some branches in cognitive psychology give explicit attention to these notions. In other disciplines, they are assumed implicitly, and usually they are not part of formal education. The abilities, represented by these knowledge fields are also difficult to asses by exams or tests, and therefore at present they play only a small part in our understanding of ’academic abilities’.

In the area of knowledge engineering, attempts are made to formalize (parts of) the K-DAG, including (some) root-like nodes, in terms of formal ontology - with the eventual aim to have disciplinary knowledge represented in knowledge bases that can be consulted by dedicated software applications (see for instance http://www.steplib.com). For our purpose, we will only use the terminology of graphs to argue about education and design practice.

A ’discipline’ can now be defined, more precisely, as a connected sub-graph of the K-DAG. It seems likely, however, that a discipline is typically not the result of some august body of experts, drawing endless ellipses and arcs on a huge sheet of paper. It is interesting, therefore, to speculate on the evolution of a discipline - that is, to zoom in the mechanisms that are hidden in the various quadrants in figure 1.

We can imagine that there are two basic types of mechanisms involved in the evolution of a discipline. The first is the internal evolution (upper left quadrant in figure 1). When seeking answers to ’why’ or ’how comes’ questions, a chunk of knowledge A asks for the connection to, or the development of a chunk B with which it has an ’A assumes B’-relation. Similar (although maybe less frequent) a question such as ’what can we do with this’ may lead to an instance of the opposite relation. In any case, the internal evolution maintains connectivity; it is a local process in the sense that the K-DAG grows with one arc at the time.

The second process can be called external evolution. This corresponds to the upper right quadrant in figure 1. It comes from the interplay between disciplines and domains. We remember that a domain also has an internal coherence, but this comes from social needs, governmental arbitrariness, or historical chance. The set of domains definitely does not have a DAG structure similar to disciplines - if it has any meaningful structure at all. Nevertheless, the persistence of domains causes certain needs to occur in matched combinations. For instance, in the domain of transport, the initiative of projecting a highway raises simultaneous questions in the disciplines of geography, civil engineering, economy, meteorology (with respect to the environmental consequences of the CO_2 production of the traffic on the projected road), and ecology. The merging of such (initially separate) disciplines into one new, ’multidisciplinary’ discipline as a result of external evolution can be depicted, schematically as in figure 3.

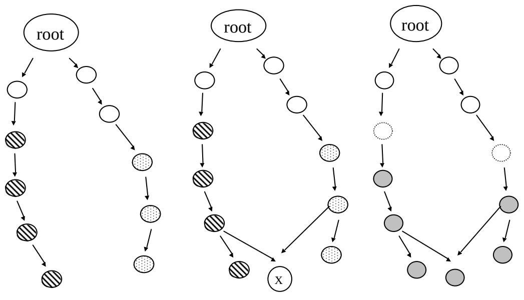

Figure 3: Three phases in the external evolution of disciplines.

In figure 3, we depict three stages of the evolution of a discipline as triggered by external factors.

In the leftmost diagram we have the initial situation. Two separate disciplines exist, both depicted by their (schematic) K-DAG. One discipline is indicated by line-textured ellipses, the other one by dotted ellipses. For simplicity, the disciplines have been drawn as simple chains of knowledge domains; in general, they will consist of many branching chains. Notice that typical disciplines don’t contain the ’root’ knowledge domain, nor knowledge domains close to the ’root’. The non-textured ellipses represent the ’implicit’ knowledge that is assumed to underlay the disciplines, but that is not part of the formal curriculum, and therefore it is not part of the *explicit* working knowledge of workers[3].

In the second stage (middle diagram in figure 3) a chunk of knowledge, labeled 'X' appears from an application domain. It assumes knowledge chunks from both disciplines. If the 'A assumes B' relations can be successfully established, and this circumstance appears sufficiently often, the third phase may occur.

In the third phase (rightmost diagram in figure 3), the two formerly separated disciplines have been merged into one new discipline (indicated by gray ellipses). For instance, if the two former disciplines were 'mechanical engineering' and 'logistics', then the new discipline could be 'transport sciences'. Notice that some ellipses, that were formerly textured, now have become blanc (indicated by dotted outlines). This is because, as we saw earlier, a discipline is carried by a curriculum, and a curriculum is the result of a resource constrained design problem. The constrained resources, in this case, are time (between 4 and 9 years for various tracks of scientific education) and learning capacity. In phase 3 we see that typically those knowledge domains are sacrificed that in the original disciplines were closer to the 'root'-nodes. The curriculum for the new discipline must contain the node formerly labeled 'X' and related nodes, and therefore doesn't have sufficient room left for the most upstream nodes in the original curricula.

This may cause a significant problem. Due to their nature, the domains close to the root are quite abstract, and many are widely applicable. Omitting these nodes makes it more difficult, in particular in freshly graduated practitioners, to see the underlying relations between (application) domains, and therefore learning these knowledge domains later (=after completion of the formal education) takes more effort. In phase 3, taught knowledge, skills and attitudes have the risk to be of a rather encyclopedic nature - which falls short in complex situations where deep and abstract understanding are required.

Disciplines in technological education

The above observations hold in arbitrary fields of science. For technological sciences, there are some additional conditions. First, the range of notions in the 'root' node and nearby nodes in the K-DAG is less broad than in general science. Many patterns can be identified that have proven adequate in a large range of situations.

Many of these have been formalized in terms of mathematical notions (e.g., 'function', 'discrete vs. continuous', 'variable', 'operation', 'state', 'singularity', 'graph', et cetera). Apart from their precise, technical meaning, they have an important value as metaphors.

For instance, even though 'monotonicity' is a formal property, typically applied to mathematical functions on sets of numbers, it is insightful to use this term in economical or even psychological contexts to indicate 'that something develops in one direction only'.

As another example, we have used the terminology of directed a-cyclic graphs in this paper to facilitate arguing about the development of disciplinary knowledge. It is important to notice that we don't (necessarily) use these terms for the purpose of technical manipulation, but we use them because of their metaphorical intuition. For instance, using the term K-DAG made it natural to talk about 'root-nodes' and 'partial ordering' without having to spend much words in explaining what these things mean. Furthermore, it invites us to think of algorithms for 'merging' and 'traversing' DAG's as metaphors for understanding the process of discipline development. Since terms such as 'merging', 'traversing', and the numerous other concepts from mathematics and computing science come with a cloud of useful associations[4], they provide convenient communication shortcuts that avoid a lot of potentially confusing verbose prose.

Moreover, once they are used in the early, exploratory phase in a communicative process (such as a design process), where terms are not yet formally defined, there is a large chance that the same terms can be used later when things get more pre-

[3]Which is not to say that they don't use this knowledge; they just don't know how to talk about it.

[4]These associations are useful in a large variety of contexts, otherwise the terms would not have gained the status of broadly used abstract notions.

cise - including (some of) their technical connotations.

Second, practitioners in technical disciplines have at least some familiarity with formal argumentation. They are trained to formally manipulate with terms and symbols in math courses. This means that the habit of using precisely defined terms comes more natural to technologically educated professionals - even when, in early stages of exploring a topic, the formal manipulations with such terms is not yet in order. There is a rich potential of problem-solving patterns hidden in the usage of precise terminology - if only this potential is recognized and stimulated by teachers and adopted and practiced by students.

Problems and solutions

After having studied the notions of disciplines and domains, and having explored the mechanisms of discipline development - both under internal and external factors - and having touched upon the particular circumstance of technological disciplines, we now arrive at the main theme of this paper.

Designing across discipline borders is a difficult process. It resembles the external evolution of a discipline as outlined in figure 3. The design problem at hand corresponds to the (application domain-induced) node 'X'. The disciplines that need to be connected are the bodies of disciplinary baggage of the involved designers. These disciplines seem unrelated because both involved designers lack a sufficient shared body of underlying, more abstract notions. As with merging disciplines, this is again a result of resource constraints (limited time, knowledge, effort and expertise of the designers at hand).

The resulting complication is most often called a 'communication problem', but it would be more appropriate to call it a problem of lacking shared, sufficiently abstract thought patterns[5].

In an attempt to remedy such 'communication problems', new 'interdisciplinary' curricula are presently developed. The underlying idea is probably that, as soon as an application domain is embedded within a discipline, and hence a curriculum is developed for this new discipline, the problem of designing across discipline borders vanishes. Indeed, in this new discipline, both earlier disciplines have been integrated, the discipline border has vanished, and the problem has gone away - or so it is hoped.

From our analysis, however, it seems that this argument is flawed. Rather, we think that the problems with interdisciplinary design increase when new 'interdisciplinary' curricula arise. Indeed, due to the resource constraints that are inherent in any curriculum such very broad and interdisciplinary curricula are increasingly devoid of the root-like nodes in the K-DAG. The ultimate version of such a trend would be that there is only one (technological?) discipline left, which would include all possible application domains. Then there are no discipline borders left, and hence no problems of cross-discipline design. In the limit case of such an ultimate interdisciplinary 'discipline', however, there would be hardly any room for abstract thought patterns, and as a consequence, there would be hardly any insight in underlying relations between (application) domains.

Instead, we recommend a more paradoxical remedy to prepare designers for interdisciplinary design challenges. Rather than spending large amounts of curriculum space to application domain-related knowledge, we propose to increase the amount of fundamental ingredients. This includes formal notions and mathematical and logical techniques. Notice: this should not be mistaken as a recommendation for 'more math'. Rather, it is a recommendation to focus on explicating thought patterns and problem solving strategies. A vehicle could be to study the intuitions behind mathematical notions, to practice with designing and studying models for the sake of understanding the methodology of model making, and to exercise definition-making skills in order to perfect precision and exactness in the expression of ideas, assumptions and propositions. Because of their abstraction and wide applicability, these skills seem to be the best candidates for dealing with arbitrary cross-disciplinary design and engineering problems.

A natural question would be what effects such an approach to cross-disciplinary design could bring. Since the distance to domain-specific applications is larger than in many 'interdisciplinary curricula', a curriculum according to the above recommendation may not be a fail-safe recipe for spectacu-

[5]Perhaps many communication problems are just the lack of sufficient shared, abstract, underlying notions.

lar innovation or for revolutionary new products. Indeed, ideas for new products often come from workers close to application domains. If such experts have less familiarity with fundamental issues, however, a thorough understanding of the underlying principles may be an underestimated ingredient - which can cause overstrained expectations and disappointing performances of hastily designed products. Rather, our recommendation for a more foundation-oriented curriculum to educate designers to work in cross-disciplinary contexts could give rise to well-structured, consistent and smooth design processes that are less hampered by communicative noise.

An option for computing science

Above we gave a recommendation that applies to curriculum design. There is, however, another route to mitigate the problems of designing across discipline borders. As follows:

Among all the sciences, computer science forms a peculiar case. In any other science, a scientific argument is judged for its convincingness with respect to (human) colleagues. In computer science, a scientific argument (e.g., an algorithm) is judged for its convincingness with respect to a formally defined machinery (namely, an (abstract) computer or some other formal framework).

This has a major consequence. In all sciences except for computer science and mathematics, there is a large amount of interpretation involved in assessing the validity of an argument. Even in empiric sciences, where so called objective observations are the cornerstones of progress in understanding, dealing with such observations often leaves room for interpretation. Interpretation, in turn, leaves room for misunderstanding, confusion or ambiguity.

A computer cannot tolerate ambiguity, and therefore a computer program cannot rely on interpretation. Hence computer scientists are trained to give precise and unambiguous definitions. At the same time, unlike some branches in mathematics, computer science is involved with modeling *reality*. A computer program has a purpose, namely to add in dealing with (aspects of) a real situation, whether this is a computer controlled machine, an administrative system or a communication network.

Therefore, by their education, computer scientists possess a rather unique combination of skills, that is essential in interdisciplinary design, namely (a) to be capable to think in terms in models (because computer programs that have something to do with real systems only do so by dealing with *models* of such systems), and (b) to be capable to formulate such models in a precise and unambiguous manner (because computers require precise and unambiguous instructions).

It is remarkable that most computer scientists only exercise this rather unique combination of skills when it comes to IT-related design. The reasons for this may be several (perhaps students choose for computer science because of the prospect of luxurious salaries, to be earned with writing software; a hobby in computer programming is the main motivation for others), but it would be tremendously helpful in all sorts of interdisciplinary design if computer scientists would offer their assistance to help clarifying interfaces between (models of) knowledge domains - irrespective whether it regards mechanical, chemical, biomedical or any other disciplines.

Maybe in the light of subsiding economic activity in the software branch, this could be an interesting option for computer scientists who are not afraid to broaden their scope.

To conclude with a paraphrase of Dijkstra's famous motto 'Beauty is our business', we might give as a characterization for this new group of professionals: 'Precision is our profession'.

About the author

Kees van Overveld (1957) obtained a MSc and PhD degree in physics at the Eindhoven University of Technology (EUT). In 1985, he joined the computing science department of the faculty of Mathematics and Computer Science of EUT as a university lecturer; since 1990 as associate professor. From 1989 to 1998 he was head of the Computer Graphics group. From November 1996 to June 1998 he was part-time employed as a Senior Researcher at Philips Research; further, he still held the position of associate professor at EUT. In June 1998 he founded 'Van Overveld Coaching', a consultancy company. In this company he works on a regular basis for Philips Research, the University of Calgary (Canada) and the University of Magdeburg (Germany). In May 2000 he left the computing science department of EUT; he changed his academic affiliation to the Stan Ackermans Institute at EUT. In 2002, he joined the Faculty of Industrial Design (ID).

Among his previous research interests are the fundamentals of raster graphics algorithms and discretisation, computer aided geometric design, interactive motion specification and (dynamic) simulation for computer animation, image processing and some aspects of 3-D computer vision. Recently, he initiated a research activity in the field of the methodology of technological design. He is currently responsible both for the teaching program and the research in this field. In ID, he is mainly involved in teaching mathematical modeling and structured creativity-related techniques.