Submitted version of paper accepted for publication in the conference proceedings of Research in Engineering Education Network (REES 2019).

The Smart Engineering Curriculum

K.E. Wolff
Stellenbosch University, South Africa
wolff.ke@gmail.com

M.J. Booysen
Stellenbosch University, South Africa
mjbooysen@sun.ac.za

***Abstract**: The increasing complexity entailed in training engineers for the Industry 4.0 workplace requires an approach beyond simply cramming more into the curriculum. The purpose of this paper is to problematise the relationship between layers of engineering that constitute the multidisciplinary systems which require contextualised and responsive engagement with data. Using a number of social realist analytical instruments, the forms of knowledge and related practices at different levels within the 'smart engineering' curriculum are interrogated and subsequently illustrated so as to guide pedagogic decisions. The intention of the research is primarily to enable students to effectively develop 1) integrated systems-level thinking, and 2) appropriate, interpretative data processing skills. There is a symbiotic and analogical relationship between the design of a curricular framework for the 'smart engineer' and the collaborative, interdisciplinary research approach: drawing on multiple perspectives and approaches from the hard and soft sciences enables a more informed educational design process.*

KEYWORDS: systems engineering; curriculum design; Legitimation Code Theory

Introduction

A widespread complaint from employers is graduate inability to apply knowledge (Griesel & Parker, 2009) acquired during their formal education, and their 'lack of technical skills' (manpowergroup.com, 2015). Attempts to introduce more practical technical elements into engineering curricula by way of project-based learning and traditional technical capstone projects are usually reported as successful in smaller-scale, well-resourced environments. For educators in large class contexts, such as in South Africa – the site of this research project – the challenges are more daunting. Of particular concern is the fact that in attempting to equip students to meet the demands of rapidly evolving technology- and information-driven sectors, the reality is that any technology introduced into the curriculum is likely to be redundant by the time the student graduates (Felder, 2012). However, the national drive to produce graduates so as to meet scarce skills demands and the Higher Education (HE) mandate to increase Science, Technology, Engineering and Mathematics (STEM) graduates has led to several curriculum redesign and renewal initiatives (as well as revised pedagogic strategies) in an attempt to produce the problem-solving professionals urgently needed to address national Sustainable Development Goals. One such initiative is the conceptual design of a theoretically-informed framework for the 21st century 'smart engineer' in the 'digital ecology' (WEF, 2016a) that characterises Industry 4.0.

> *"We are at the beginning of a global transformation that is characterized by the convergence of digital, physical, and biological technologies in ways that are changing both the world around us and our very idea of what it means to be human. The changes are historic in terms of their size, speed, and scope"* (WEF, 2016b).

With the emergence of the mobile devices that form part of the internet of things (IoT) in a massive system of interconnected devices (Things), multiple new challenges face engineers who have hitherto been exposed to a silo-structured curriculum, and siloed way of thinking. Engineering students in our context tend to be taught from the ground up, starting with core physical fundamentals (physics, chemistry, mathematics) with the objective of working towards a toolbox of disparate knowledge areas culminating in the fourth-year capstone project which is intended to integrate the preceding knowledge and is used to assess the student's ability to perform industry-mimicking tasks. The 'toolbox' approach is also evident in the divisions between the different branches of engineering. Categorisations of what have come to be called engineering 'disciplines' routinely ignore or avoid a more conceptual approach, listing fields such as software engineering, mechanical engineering, biomedical engineering and systems engineering side by side as though they were comparable. In South Africa there is little differentiation in curricula with regard to scope, scale and nature of work across engineering sectors, and the most common curricular approach for capstone projects is the Conceive-Design-Implement-Operate (CDIO) sequence (Crawley, 2001), which is appropriate to small-scale Research and Development environments. The majority of graduates, however, do not find employment in such niche areas – which operate with a fundamentally different organisational logic (Wolff, 2017); rather, employment is more common in medium to large-scale sectors where very little design (in the CDIO sense) takes place. The industry complaints with regard to 'soft skills' (Cappelli, 2014) attest to the lack of ability to work in more complex organisational structures, with the concomitant communication, teamwork and social practices.

The categorisation of engineering sectors does not take into account that software, computer and systems engineering, for example, underpin and affect *all* sectors, including those professions beyond engineering. The increasing interdisciplinarity emerging in such areas as mechatronics and industrial engineering, as well as the Industry 4.0 paradigm suggests that we have outgrown the silos and need to rethink the training of our engineers. The highlighting of graduate lack of technical and 'soft' skills, we suggest, bears testimony to our inability as educators to enable students to engage in relational, part-whole thinking. The changing technological landscape poses "major challenges requiring proactive adaptation by corporations, governments, societies and individuals" (WEF, 2016b). This is where the future SMART engineer enters the picture.

The easiest analogy to illustrate the 21st century trajectory in engineering is that of a coder versus a software systems engineer. A coder is the detailed implementation agent, with the system engineer ensuring that everything is appropriately connected and working together. These two functions require significantly different mindsets and practices. With smart systems, the engineer cannot afford to be bogged down by the detail. The engineer has to be exposed to macro ways of thinking from the outset, albeit in parallel with certain fundamentals that require micro, detailed approaches. The systems engineer essentially integrates systems that span multiple disciplines. These disciplines are not limited to engineering though, and are very reliant on the so-called "soft skills" such as ethics, environmental impacts, societal and economic impacts. Although specified by engineering qualification standards as important attributes of an engineer in any field, these attributes take on an entirely different significance in the context of the pervasive and massive scale at which smart systems will be employed in the internet of things and Industry 4.0, where machines and humans co-exist in ways never seen before, "affecting labor markets … as well as social value systems and ethical frameworks" (WEF, 2016b). The scale and dynamic pace of technology imply major environmental impact, with billions of devices 1) disposed of annually, and 2) using power and emitting interference.

The purpose of this paper is to present a conceptual framework which can illuminate the complexity facing the Smart Engineer (SE) and to suggest a curricular and pedagogic approach that may enable holistic, contextualised engagement with data generated by multidisciplinary systems in complex socio-technical contexts, with a view to the effective development of 1) integrated systems-level thinking and 2) appropriate, interpretative data processing skills at different levels of the system.

Conceptual Framework

Knowledge structures in 21st century engineering

The theoretical focus of this paper is a rigorous conceptualisation of the nature of knowledge and its concomitant practices underpinning the structure of a SE curriculum with a view to improving engineering programmes designed to meet the needs of increasingly complex 21st century socio-technical environments. It is our position that current programmes are inadequately conceptualised from the perspective of the required knowledge practices in technology-driven sectors. Engineering curricula are framed by three international engineering accords which stipulate mathematical, natural and engineering sciences as the core disciplines. The structure of these disciplines (and the implications for learning) can be described using social realist theory.

Basil Bernstein (2000) described knowledge as kinds of ‘codes’ – invisible structures based on rules established in different fields of practice. Our task, as educators is to understand how these codes arise, so as to make them explicit for the purpose of ‘learning the rules of the game’. One way in which we do this is to ‘frame’ the learning experience. ‘Framing’ is about what goes into the curriculum, in what order, at what pace, and against what criteria. Framing is also about ‘who’ controls the ‘frame’. Bernstein differentiates between three kinds of formal knowledge structures, each of which needs to be framed differently for effective learning. Hierarchical knowledge structures, represented by the natural and physical sciences, attempt “to create very general propositions and theories, which integrate knowledge at lower levels” (Bernstein, 2000). This means that new theories or concepts subsume earlier ones, creating a visible sequence over time. We see this in the traditional school science curriculum, where we start with matter then motion then energy and so on. Learning this kind of knowledge requires strong sequencing and the grasp of each preceding concept, and usually manifests as ‘teacher-centred’ pedagogy (Winberg, *et al*., 2018). In other words, hierarchical knowledge structures are acquired more effectively with the guidance of a ‘knowledgeable other’ who tightly ‘frames’ the learning experience, deciding on the content, order and criteria required to master the knowledge.

Horizontal knowledge structures, on the other hand, “consist of a series of specialised languages with specialised modes of interrogation and criteria for the construction and circulation of texts” (Bernstein, 2000). We have ‘strong’ horizontal knowledge structures and weak ones. Both types simply have different ‘languages’ of the same type of knowledge, each with its own rules. Where the rules for each language of the same type (or family) are ‘strong’, we refer to them demonstrating a ‘strong grammar’. Any mathematical theorem is a good example of a strong ‘conceptual syntax’ which is easily empirically identifiable. In the case of horizontal knowledge structures, ‘masses of particulars’ (Muller, 2009) need to be learnt independently, not necessarily sequentially, and usually in specific contexts. This means more time is required for the acquisition of this kind of knowledge. It also means the framing over what must be learned in what order can be weaker, with the student taking more control.

Horizontal knowledge structures with ‘weak grammars’ are those where the “capacity of a theory to stably identify empirical correlates” is weaker (Young & Muller, 2007). This is evident in forms of knowledge which routinely borrow concepts and terms from other fields, and which see rapid obsolescence. Human languages are a good example, as are information communication technologies! Bernstein talks about the ‘regionalisation’ of knowledge (2000), evident in such fields as engineering, where one sees the weakening of boundaries between the disciplinary bases, such as in the case of ICTs - a ‘region’ which is at the heart of 21st century multidisciplinary engineering practice. The primary disciplines underpinning ICTs are ‘logic’ and mathematics, both of which are horizontally structured.

However, the disciplinary ‘logic’ implied in engineering control systems today has become increasingly complex, with an ever-weakening ‘grammar’. Acquiring knowledge with a ‘weak’ horizontal structure, such as ‘logic programming’, means not only learning each new relevant ‘language’ as it is created or required, but staying abreast of structural and even conceptual changes to the same ‘language’, as it is the users who drive change in the field of application or social context. In this case, framing can be very weak, meaning the student controls the sequence, and needs opportunities to experience multiple types and application contexts.

Multidisciplinarity in the curriculum

Multidisciplinary engineering, such as in SE, requires the iterative navigation of these different disciplines, each of which implies a significantly different ‘code’ or way of thinking. Figure 1 captures the broad foundational systems of a multidisciplinary engineering curriculum with the associated dominant knowledge structural forms. There is an implied hierarchy as one moves from the physical systems to the ‘invisible’ control layer.

Another useful Bernsteinian concept in relation to the curriculum is that of classification. If the core disciplines are characterised by organising principles which can be clearly differentiated, then these disciplines are regarded as ‘bounded’, or insulated from other disciplines.

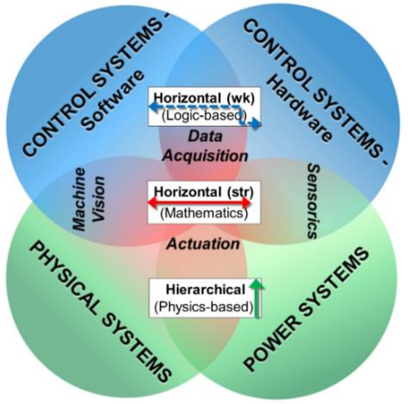

Figure 1. Multidisciplinary engineering knowledge structures

Strongly classified ‘singulars’ (such as physics and mathematics) lend themselves to a collection-type curriculum, which sees the “organisation, transmission and evaluation of knowledge as bound up with patterns of authority and control” (Bernstein, 1977). This is precisely the silo curriculum structure we still see in traditional engineering curricula (UNESCO, 2010). The emergence in 20th century education of ‘regions’ such as Medicine and Engineering weakened the classification of the pure disciplines. The boundaries between disciplines reflect social power relations. When boundaries are blurred, disciplines or their agents risk losing their specialisation and status - the social order is threatened. This may well be why there is such resistance to altering the silo curriculum structure!

A further challenge for systems engineering educators is that effective ‘systems’ are built on a productive, causal relationship between mathematics *and* logic (characterised as having strong and weak ‘grammars’ respectively). In essence, mathematics is the science of patterns. The kind of mathematical thinking implied in SE goes well beyond the confines of Calculus (the backbone of traditional engineering) or statistics. The SE needs to work in all mathematical branches, given the topological complexity of IoT systems. The reality is that the engineering ‘regions’ of the 21st century blur multiple boundaries. For a region to survive, there must be a ‘relational idea’ (Bernstein, 1975) and strong lateral relations – in other words, inter-/multi-disciplinary collaboration, and inter-/ multidisciplinary forms of thinking. What exactly does this form of thinking look like?

Analytical Tools

Legitimation Code Theory (LCT) (Maton, 2014) offers a dimension which can help to illustrate what multidisciplinary thinking looks like. The Specialisation Epistemic Plane (figure 2 - left) illustrates the relationship between a phenomenon and its approaches, the *what* and *how* of any knowledge practice. The *what* and *how* axes differentiate between how strongly ‘bounded’ a phenomenon is (‘classified’ in the Bernsteinian sense) and how fixed the approach to that phenomenon is (‘framing’ in the Bernsteinian sense). The physical sciences are strongly bounded phenomena with fixed approaches. This means that one needs *purist insight* to effectively grasp these concepts. Where the phenomenon is not important, rather the rules/strict methods are, we talk of needing *doctrinal insight*, such as mathematical functions or syntax rules. One needs *situational insight* when there are open-ended methods for a clearly defined phenomenon. When the phenomenon and approach are weaker, one talks of having ‘no insight’ or the issue is not about knowledge, but knowers in the system.

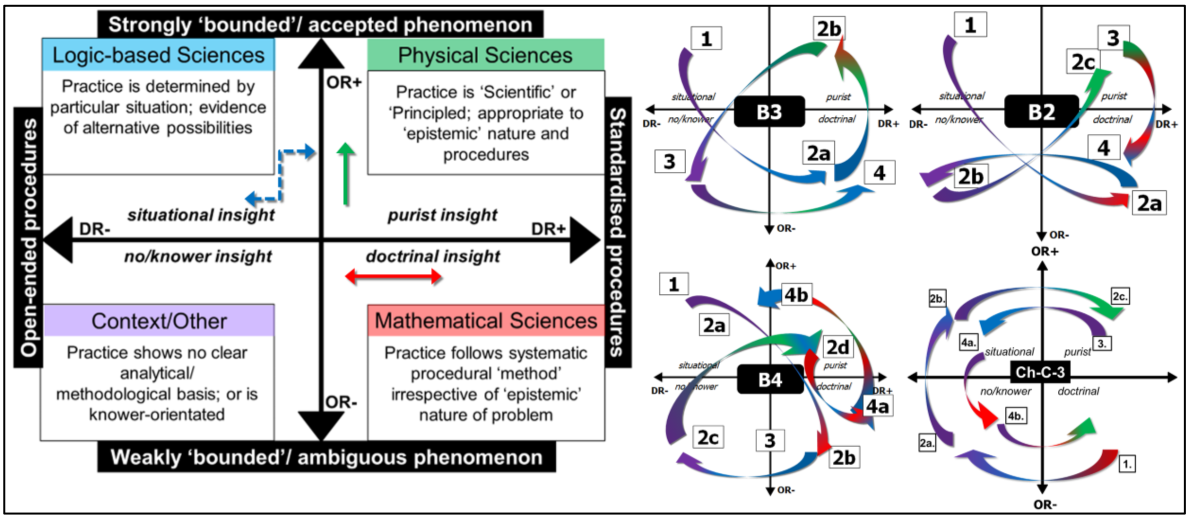

Figure 2. Knowledge structures on the epistemic plane

The Epistemic Plane has been used in engineering problem solving research to demonstrate how practitioners navigate between different disciplines as they approach (1), analyse (2), identify the cause (3) and solve (4) a controlled electro-mechanical problem in industrial contexts (Wolff, 2017). Successful practitioners (figure 2 - right) follow different patterns, depending on context and personal ‘insight’ orientation. One of the key findings from the industrial case studies is that the greatest challenge for most new graduates is shifting from the right to the left, in other words, not being able to cope with more flexible approaches in more complex contexts. The successful systems integrators in the study start with the ‘big picture’ in the ‘situational’ quadrant, and consider multiple possibilities. They also navigate more easily to the ‘contextual’ stakeholder quadrant to understand the situation better, and draw on a range of appropriate methods (most acquired in the industrial context).

What this suggests, if one returns to the knowledge structural features, is that working on the left-hand side of the Epistemic Plane (which are precisely the ‘skills’ industry requires) entails greater flexibility and a broader range of challenges. Knowledge structures situated on the left-hand side require weaker framing, in other words more control by the student. This is precisely the ‘self-regulated’ learning required to be demonstrated in Graduate Attribute 9 of the engineering qualification standards. So, what are the implications for a SE curriculum?

The SE Curriculum

Let us consider the SE domain as consisting of multiple layers. At each level there is a relationship between the fundamental sciences and context, which has implications for decisions as to curricular and pedagogic approaches (figure 3).

The physical layer (1) includes all things that can be controlled and/or measured, energy storage and harvesting, and mobility. With regards to mobility, the physical location is of special importance for this field, but included is also the dynamic physics, i.e. the way in which things that move are measured by MEMS sensors. [Despite its importance and pervasiveness in modern sensing systems, its novelty has resulted in its exclusion from traditional curricula]. The energy harvesting in this case includes power conditioning from traditional sources, although the modern trend is for distributed energy harvesting. The sensing and actuation layer represents the sensor technologies and actuation technologies. Given its tight coupling to the physical system that is controlled, it spans across the Physical domain. This layer also includes the driving circuitry, for example an H-bridge driving a stepper motor. Each of the elements in the physical layer is not only underpinned, but dictated by physics-based (hierarchical) laws, from those governing motion and thermodynamics to Ohm’s Law. Engineers working at this level require these laws at their fingertips; in other words, they require stronger *purist insight*.

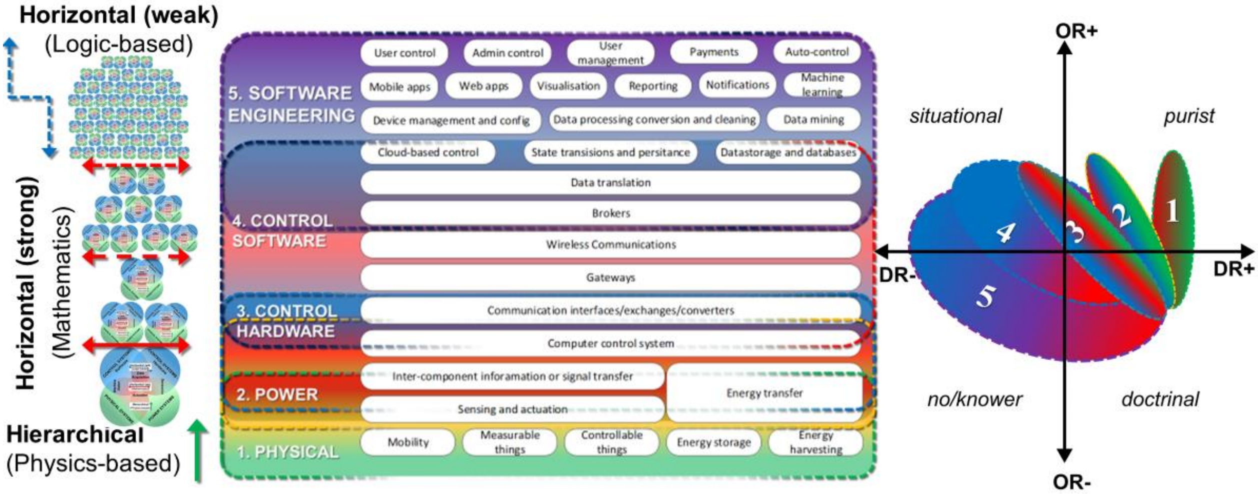

Figure 3. The SE knowledge topology

Above the sensing and actuation layers is the energy transfer and inter-component or signal transfer (2). This layer captures the method employed to convey the “sensed” quantity (e.g. sampled quantity represented as volts or communicated as serial data) and the same for actuation. This could include opto-isolated or magento-isolated sensing (e.g. signal transformers). In modern systems, this layer often also includes inter-component wireless communications, such as modbus, Zigbee, Bluetooth or WiFi. In addition, we have the transfer of energy between the energy source or storage through copper or transformer. As in the physical layer, the signal production and transfer process itself is governed predominantly by the physics of electricity/electromagnetism and classical Calculus. However, here we see the first key transition into the interpretative space, where different statistical mathematical methods (*doctrinal insight*) can be used to analyse, extract and transform signals selectively, depending on the what information the end-user (whether another machine or human) requires to continue or complete a process. This marks a step into the weak horizontal knowledge structure of logic-based sciences, where the logic is dictated by the end goal and non-physics-based determinants. In other words, here the engineer is required to move iteratively between the *doctrinal* and *situational insights*.

The computer control system (3) contains the processor and firmware, which are significantly different from historical processors and firmware architecture. The main differences relate to the locus of control, redundancy and mitigation of failure for safety and efficiency, local state persistence in the event of communications failure in the higher levels. A secondary effect of these strategies is the requirement for local storage and synchronisation thereof, for example storing schedules and target temperatures, in case the cloud control is unavailable - a highly likely scenario that must be accommodated. Although the intention is for these devices to be much less sophisticated than in historic systems, there is quite often the need for some local processing, both to reduce data transfer, and to enable local decision making. Again, as in the previous layer, the nature of and relationship between these devices is dependent on contrasting *insights*. The *situation* is that the firmware is not cast in stone, and upgrades have a knock-on effect on sizing and re-writability requirements on programme memory. However, the ability of the processor to compute and relay signals is governed by the laws of physics and Boolean arithmetic (*purist* and *doctrina*l insights). The question for engineering educators here becomes more complex: what level of understanding of the firmware and processing aspects of the control layer are required to enable effective fault-finding when managing ‘big data’?

The layer above the computer control system is required to get the remote part of the system "onto the cloud" (4). This is usually done with a modem that is either on the same PCB (sometimes embedded with the controller as a system on chip), or connected with some form of local communications, such as Zigbee or modbus. The processor therefore can either go through an exchange, condenser, convertor, aggregator, etc, or directly into the gateway, which provides the wireless communications. The available technologies in this layer, commonly termed Low-Power Wide-Area Networks (LPWAN) are evolving at an exponential rate, each with their own limitations and special features. This is characteristic of weak horizontal knowledge structures, which require the constant acquisition of "masses of particulars" (Muller, 2009), and the iterative movement between *situational* and *doctrinal* insights. Once the information is communicated from the remote device, a cloud-based plethora of systems (5) have to be in place to ensure seamless, safe, and secure system operation. The hierarchy in the software engineering block, however, is not as clearly defined, with no standardisation of components and major challenges for security (Booysen *et al.*, 2012). The reason for this is that either the practices required are all dictated by users – in other words, *knowers* in the system – or there is no clarity as to the overarching purpose – in other words '*no insight'* or no clear 'relational idea' (Bernstein, 2000). For complex regions to survive, there needs to be consensus as to the overarching purpose. One sees '*no insight'* in complex systems where there are contradictory paradigms, such as for example in education which attempts to achieve social justice through forms of managerialism designed for economic competitiveness. The moment one sees references to users ('humans') in the system, then the knowledge practices become more complex and draw on different, competing forms of knowledge.

In the software engineering layer, the user control and admin control are different access methods from clients into the system. The user typically has access only to a basic interface in which two separate types of management can be done - the devices or assets (e.g. the device being monitored and/or controlled, typically with an "overview" view and a detailed asset-bespoke interface) and account management (e.g. billing, contact details, login details, site configuration, etc.). The administrator typically has access to all devices/assets and all user profiles. Before any users can be given access, user access control must be set up, to allow users to register, log contact details, set preferences, accept terms, and setup payment options (if applicable). These core functions are paramount, but entirely dependent on *situational* and *knower* requirements currently not even considered in engineering courses.

The reporting in these types of systems presents a layer of complexity that moves beyond 'formal' knowledge: there is no use reporting content that is not understood or able to be visualised and easily interpreted. This is often where software engineers struggle to make sense of the data that is at different levels of being processed and converted, and to visualise it in a human-readable format. A key limitation here is that the approach is often a *doctrinal* (technical) one, as opposed to the *knower*-orientated approach required to produce user-friendly, visualised, and accessible information. The lack of appropriate communication skills at all levels is well reported in the literature (Cappelli, 2014). This suggests the SE student needs intimate knowledge of the physical and sensing system, all the way through to an understanding of the reporting system.

In summary, the five layers as characterised demonstrate a natural hierarchy, from traditional 'stable' disciplines to increasingly weaker 'regions'. However, the dilemma in designing both an educational framework as well as the system itself lies in the design 'direction'. Engineers follow a processor-centric development, as the processor has historically been the centre. This was mainly a top-down design when processors were central. However, with the remote access to the cloud and the shift in the loci of some fundamental control functions, the processor-centric development ends up being half top-down and half bottom-up. The requirement is now for the design to be top-down from the cloud, AND top-down from the processor, almost as two separate but equal systems. There are two major challenges in mis-directed design: If the software engineering layer design is done from a processor perspective (bottom-up into the cloud), security will be compromised, the cloud system will be very difficult to maintain, bottlenecks will occur in the cloud-remote system interface layers, systems will not be able to cater to similar solutions, as they will be developed with a “silo” mentality, rather than reusable cross-application mentality, and the wrong platforms will be chosen. The second key challenge is that software engineers used to implement what they were told the hardware or systems engineers. Now that control lies in their domain, and their domain knowledge is key to a good design, this calls into question the required scope and depth of knowledge and associated practices in order to effectively manage such complex cloud-based systems. It is therefore a daunting task to include these in any undergraduate material, and often recent engineering graduates have very little or no knowledge of these available technologies and how they compare.

Concluding Comments and Recommendations

The theoretically-informed analysis of five layers of complexity entailed in the SE system demonstrates a trajectory from traditional stable science-based disciplines in the physical systems layer towards increasingly complex and competing forms of knowledge characterised by interdisciplinary ‘regions’ in the software engineering layer. This is precisely the sequence in which traditional engineering curricula are structured. However, the analysis attempted to illuminate the fractal nature of each layer, with each consisting of sub-elements and forms of knowledge of the preceding layer. As the complexity increases, this calls into question just how much one can feasibly squeeze into an already overfull curriculum. We suggest that this ‘bottom-up’ sequence does not allow for the development of the ‘top-down’ view of the SE system. The transition in mathematical sciences through the layers as well as the dominance of horizontally structured, logic-based knowledge suggests the need for a curricular and pedagogic approach that allows for iterative and cyclical movement across different forms of knowledge and ways of thinking (*insights*) that build towards the overarching relational idea. We recommend that:

1) it is necessary for educators to reconceptualise what it means to develop systems thinking abilities, and suggest that a more explicitly relational approach to the foundation disciplines may elicit an understanding of ‘the science of patterns’ – which is how mathematics itself is described (Schoenfeld, 2009);

2) the pedagogic approach needs to shift towards a more student-centred and self-regulated learning ethic, given the shift from hierarchical knowledge structures requiring *purist insight* towards increasingly weaker horizontal knowledge structures requiring *knower insight*.

Finally, there is a symbiotic and analogical relationship between the design of a curricular framework for the ‘smart engineer’ and the collaborative, interdisciplinary research approach: drawing on multiple perspectives and approaches from the hard and soft sciences enables a more informed educational design process.

References

Bernstein, B. (1975). *Class, Codes & Control* (Vol. III). London: Routledge.

Bernstein, B. (1977). *Class, codes and control, Volume III: Towards a theory of educational transmissions.* London: Routledge & Kegan Paul.

Bernstein, B. (2000). *Pedagogy, symbolic control and identity: Theory, research, critique, rev. edn.* London: Rowman & Littlefield.

Booysen, M., Gilmore, J., Zeadally, S., & van Rooyen, G. (2012). Machine-to-Machine (M2M) Communications in Vehicular Networks. , 6, 2, (2012), . *KSII Transactions on Internet and Information Systems*, 529-546. doi:10.3837/tiis.2012.02.005

Cappelli, P. (2014, September 5). *What employers really want.* Retrieved February 12, 2019, from The Washington Post: https://chronicle-assets.s3.amazonaws.com/5/items/biz/pdf/Employers%20Survey.pdf

Crawley, E. (2001). *The CDIO syllabus: a statement of goals for undergraduate engineering education.* Massachusetts: Massachusetts Institute of Technology.

Felder, R. (2012). Engineering Education: A Tale of Two Paradigms. In McCabe, B, Pantazidou, M, & Phillips, D (Eds.), *Shaking the foundations of Geo-Engineering education* (pp. 9-14). Leiden: CRC Press. Retrieved June 17, 2012, from http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/TwoParadigms.pdf

Griesel, H., & Parker, B. (2009). *Graduate Attributes: a baseline study on South African graduates from the perspective on employers.* HESA & SAQA.

Haykin, S. (2001). Signal processing: where physics and mathematics meet,. *18*(4), pp. 6-7.

manpowergroup.com. (2015). *Talent Shortage Survey.* USA: The Manpower Group.

Maton, K. (2014). *Knowledge and Knowers: Towards a realist sociology of education.* London and New York: Routledge.

Muller, J. (2009). Forms of knowledge and curriculum coherence. *Journal of Education and Work, 22*(3), 205-226.

Schoenfeld, A. (2009). Learning to think mathematically: Problem solving, metacognition, and sense-making in mathematics. In *Handbook for Research on Mathematics Teaching and Learning* (pp. 334-370). New York: MacMillan.

UNESCO. (2010). *Engineering: issues, challenges and opportunities for development.* Paris: UNESCO Publishing.

WEF. (2016a). *The Global Information Technology Report.* Switzerland: World Economic Forum.

WEF. (2016b). *The Future of Jobs: Employment, Skills and Work Force Strategy for the 4th Industrial Revolution.* Switzerland: World Economic Forum.

Winberg, C., Wolff, K., Adendorff, H., Bozalek, V., Conana, H., Pallitt, N., . . . Roxå , T. (2018). Learning to teach the STEM disciplines in higher education: a critical review of the literature. *Teaching in Higher Education*. doi:10.1080/13562517.2018.1517735

Wolff, K. (2017). Engineering problem-solving knowledge: the impact of context. *Journal of Education and Work, 30*(8), 840-853. doi:10.1080/13639080.2017.1380299

Young, M., & Muller, J. (2007). Truth and truthfulness in the sociology of education. *Theory and Research in Education, 5*(2), 173-201.

Copyright statement