Full Paper

Introduction

This paper reports on findings from a case study of an engineering course in a New Zealand university focused on the learning and application of a 3-dimensional computer-aided design (3D CAD) software, SolidWorks, as an exploration of student understandings of software literacy. It is part of a larger two-year funded research project investigating the notion of 'software literacy' - how it is understood, developed and applied in tertiary teaching-learning contexts and how this understanding serves new learning. Software literacy incorporates understanding, applying, problem solving and critiquing software in pursuit of particular learning and professional goals (Khoo, Hight, Torrens, & Cowie, 2013; Hight, Khoo, Cowie, & Torrens, 2014), and extends current information and digital literacy frameworks that do not go far enough in examining how lecturers and students engage with specific software applications and its implications for student learning (Livingstone et al., 2013). There is emerging evidence that although the current student generation may be technologically competent, many still lack the basic academic technological literacy skills needed to successfully apply software embedded and enabled technologies effectively to enhance their formal learning (Kvavik, 2005). In relation to engineering education, there is evidence for the ways different digital technologies can significantly shape how and what millennial engineers can learn (Johri, Teo, Lo, Dufour, & Schram, 2014). This has, however, not been investigated in the New Zealand context.

Software Literacy and Engineering Education

Software studies, a research paradigm championed by Manovich and colleagues (Manovich, 2013), insists that 'software', operating at the levels of individual applications, platforms and infrastructures, is the dominant cultural technology of our time, an actor integral to many of the social, political and economic practices within contemporary society. Software users ideally need to develop a critical awareness of how software operates to contextualise and frame their agency through the logics embedded within programming code. Within this paradigm, there is a vital need for detailed empirical research into how software is understood, interpreted, and actually 'performed' by individuals and groups in specific contexts.

Our notion of software literacy is a practice-based schema which anticipates that users can scaffold from acquiring basic skills in using an application, to appreciating its affordances, and then on to develop an understanding of how software operates to frame knowledge and knowledge generation, and communication and creativity within disciplinary practices. We view software literacy as encompassing three specific levels of capabilities:

1st. a basic functional skill level, enabling the use of a particular application in order to complete a specific set of tasks;

2nd. an ability to independently problem solve issues faced when using an application for familiar tasks (which includes the ability to draw upon various resources to help solve difficulties); and, ultimately,

3rd. the ability to critique the application, including being able to apply a similar analysis to a range of software designed for similar purposes - enabling the informed selection of applications and more 'empowered' new software learning.

In these terms, the most 'critically literate' users can identify the affordances of particular software tools and are able to apply and extend their knowledge and use of these and other software tools to a range of new and different purposes and contexts. Users may acquire software literacies through a combination of means; through trial and error, learning informally, or training in a more formal or structured way. We assume most people develop proficiency with ubiquitous software packages informally through everyday engagement. Tertiary students are assumed to be able to translate these knowledge and skills into formal settings to complete learning tasks, however this is not always the case (Bennett, Maton, & Kervin, 2008).

We know very little about how students develop the skills and expertise needed to attend to the features of and use software (as application, platform and architecture) to complete everyday tasks. There is evidence that the ubiquity of software and ICT tools has led students to adopt a range of informal approaches to meet their learning needs (Peeters et al., 2014). Research also indicates that students' formal software and technology learning backgrounds are diverse (Khoo, Johnson, Torrens, & Fulton, 2011), and are highly specific to their formal and informal educational, social and cultural contexts for learning and use (Jones, Ramanau, Cross, & Healing, 2010; Valtonen, Dillon, Hacklin, & Väisänen, 2010). There is a general recognition that CAD are complicated applications to learn and that many students grapple with not only mastering the technical but also the cognitive/visual-spatial skills involved in the learning process (Akasah & Alias, 2010). The challenge is for educators to adopt flexible pedagogical strategies that address this diversity. Given the various forms of investment required in the adoption of ICTs in the tertiary sector, it is imperative to understand how to close the participatory gap for students and ensure that technology is equitably and effectively used (Jenkins, Clinton, Purushotma, Robison, & Weigel, 2006). No studies that we know of raise the role of student understanding of how software and its affordances influences knowledge generation and critique, or the influence of formal and informal learning in relation to software. This research is therefore important to investigate how students develop knowledge and skills to use software and the extent they are able to employ these to successfully learn and act in formal tertiary learning contexts.

Research Context

In this paper, we investigate the extent students are able to develop SolidWorks (CAD) software literacy (in formal learning context) and to apply and extend this understanding while on workplace experience.

The case studied engineering course is a second year course introducing students to the broad principles of engineering design process and methodology. The course offers advanced exploration into SolidWorks learning by grounding its use in real-life engineering design applications and contexts. Students attend lectures and engage in the design principles and process through examining and discussing case studies of designs. They also attend 5 two-hour weekly supervised computer labs where they are provided with tasks to help them acquire further proficiency with SolidWorks and work on individual assignments. Students also participate in a group design project as a demonstration of their SolidWorks supported design understanding and application. CAD software is considered an integral component of modern engineering and is widely used in industry. The course lecturer was keen to explore and conceptualise best practices for the teaching of software within the disciplinary framework to more effectively enhance student learning and their application of SolidWorks. The software literacy framework was adopted as it had the potential to address the lecturer's pedagogical goals for his course and better support his students' learning.

All four year engineering degrees in New Zealand require the completion of 800 hours of appropriate workplace experience. Not all work placements will include the use of CAD; however for those that do, it is useful to consider how students transition or adapt their learning (and learning strategies) from the tertiary environment to the particular demands of their workplace, including learning alternative CAD applications. Knowledge of CAD can still be useful for students not actively using the software to allow them to interpret CAD generated drawings and usefully contribute to design discussions.

Research Design

We tracked the extent the second year Engineering students were able to develop a foundational competence in SolidWorks through a combination of formal and informal learning. A smaller group of students were also recruited to study their ability to transfer and apply or adapt their SolidWorks software literacy in the more immersive and/or specialised forms of practice required within workplace settings. Having an understanding of how students approach this process, including the strategies they are encouraged to draw from, will provide valuable insights into ways to better support students learning with and through software as part of their tertiary Engineering experience.

A qualitative interpretive methodology was adopted to frame the collection and analysis of data as it allowed for careful attention to the participants' perspectives and privileges their subjective realities within their specific contexts (Maykut & Morehouse, 1994). Multiple forms of data collection were collected through an online student survey (67 students out of approximately 140 students), lab observations of students' SolidWorks learning, individual student interviews when students were on work placement (4 students), and a follow up focus group interview after students have returned from their work placement (7 students). Analysis of the data was underpinned by sociocultural theory which directed attention to the interaction between people, the tools they use to achieve particular purposes and the settings in which the interactions occur (Cole & Engestrom, 1993). Emergent themes were identified through a process of inductive reasoning (Braun & Clarke, 2006).

Emerging Findings

Four key themes emerged from the analysis of the data: 1) a general student recognition that CAD knowledge and understanding is an integral part of their disciplinary knowledge, 2) learning more complex CAD applications beyond those taught in formal coursework was necessary to address work place requirements, 3) full proficiency of SolidWorks is challenging as it is a complicated and comprehensive software, and finally, 4) informal learning initiatives, time and effort were required to use and appropriately apply SolidWorks in industries.

1) CAD knowledge and understanding is an integral part of disciplinary knowledge

Students agreed that an understanding of CAD was necessary to comprehend and contribute to the engineering design process relevant to an organisation. Student evaluation of their ability to engage with disciplinary-specific software prior to and after completing their course indicates some gains in software literacy (see Table 1). Based on the categories of 'I would need help', 'I have the basic skills' (level 1 of our model), 'I can troubleshoot problems' (level 2) and 'I can apply this software' (level 3), students at the start of their second year coursework felt they would need help to use SolidWorks (52%), or that they would only have the basic skills to use the software (39%). This decreased to 2% at the end of the course of students needing help and an increase to 45% of students who felt they now have the basic skills to use SolidWorks after learning about it in the course. Another 37% of students thought they were able to troubleshoot problems faced in using the software, an increase from 6% at the beginning of the course. Gains in these two levels (basic skills and troubleshooting ability) correspond to the first two levels of our software literacy schema. By the end of the course, only 16% however thought they could apply their skills to a wide range of tasks, an indication of a lack in achieving the third level of our software schema.

Table 1: Changes in student assessment of their ability to use SolidWorks

	How good were you in using SolidWorks	**After learning about and using SolidWorks in this paper, how**

	before enrolling in this paper?	**good would you rate yourself at using it?**
I would need some help to use this software	52%	2%
I have the basic skills to use this software	39	45
I can troubleshoot problems when using this software	6	37
I can apply this software to a wide range of tasks	3	16

These results suggest that the formal coursework focused on software learning helped to develop students' software literacy so that nearly all students reported a shift to at least tier 1 (basic ability). Very few students report achieving tier 3 of our software literacy model. However the very few who do reported on the ways SolidWorks enabled them to visualise abstract disciplinary ideas, create and manipulate 3-dimensional objects, and communicate their design ideas to others as indicated in the following student quote:

> *I guess you could say that you can make things in SolidWorks that you can't make in real life. So, […] in SolidWorks you could [drill] a hole that was in a spiral and curve round but then you can't get a drill and drill that. Yeah, just … that was a problem I came into when I was learning because I was just making models as they looked rather than how they could be made*.

Having the basic skills to use and troubleshoot problems within SolidWorks is an important part of preparation for the work place experience. Two different students in the focus group explained:

> *It is sort of expected to have some knowledge of CAD when you go into work placement. If you turn up with no background, it's a big disadvantage.*
>
> *Cause you'd always come across technical drawings so having an idea of how they're made can be a bit of a benefit especially if they're made wrong.*

Students further reported that different aspects of SolidWorks became more relevant than others for their industry design purposes which extended their understanding of the software. A student commented on an example of using the 'virtual prototyping' feature in SolidWorks in his work placement to generate simulations of different design ideas and to allow his work team to discuss and decide on an idea:

> *Yeah, so we'd use virtual prototyping if we needed to do a simulation to see how it [a design prototype] might behave under certain conditions. And then it was really good for when we had multiple ideas on the table, they were all really good ideas but we needed the final sign-off by someone else so that's when it [virtual prototypes] came in.*

2) More complex CAD applications beyond those taught in coursework was necessary

Some work placements expected students to engage with similar but different CAD applications to SolidWorks requiring them to transfer their existing skills to these contexts. An aspect that appeared to facilitate students' learning of SolidWorks was students' prior engagement with artefacts or software that had a similar conceptual basis (and similar set of affordances) which provided a pathway for them to engage with new and more advanced software learning (examples here included ProEngineer, AutoCAD, Star CCM+, Autodesk Inventor and TurboCAD). Transfer of skills and enhanced awareness of functionalities were reported by 15% of students, a finding supported by subsequent focus group discussions.

For most other students, their workplace required more specialised learning, faster and/or more complex levels of SolidWorks application to be more effective in addressing site-specific manufacturing/production processes. This was exemplified when a student learnt a new application for SolidWorks as part of his workplace experience:

> *I needed to do something and the boss pointed out another feature [in SolidWorks] that I had no idea, which was 'unpacking' or something. That opened my eyes to a whole different part, like there's an application that I had no idea existed and that I could do so much more with it.*

Another student gave the example of having to learn to also use AutoCAD and another software such as Inventor as part of his workplace requirement. He found being exposed to the contrasting features of each software useful to his software literacy development:

> *AutoCAD's got more benefits because you can export your drawing to a Paint file and you can make it to a PDF and send it in an email to your boss. You can do all that from SolidWorks as well, it's just at university you're not taught any of that stuff in SolidWorks, there's limited knowledge of what you get taught and you only scratch the surface. My boss was saying using Inventor and AutoCAD, the benefits of AutoCAD is if you have a more complex model, if you want to make a last minute change to it, its easier on AutoCAD.*

3) Full proficiency in SolidWorks is challenging

Students in general perceived SolidWorks to be a complicated, comprehensive and flexible piece of software. It was therefore not feasible to try and fully understand the breadth and depth of its hierarchies of affordances during their tertiary programme.

> *Cause there's so many tiny little individual parts about understanding SolidWorks that you get past a certain point and suddenly you don't know how to mirror a three-dimensional part (for example)*.

> *The projects that you have to at university does prepare you well but they just don't allow you to go right into what the software can do.*

As SolidWorks is a complicated application, students suggested a more in-depth grounding in conceptual frameworks in the learning of the software could facilitate their understanding and to enable them to more effectively troubleshoot their application of specific affordances they encountered in their more informal learning. For example teaching the principles of Engineering design as well as CAD conventions can enhance student understanding of the potential of the SolidWorks software. Working with real-world cases and focusing on particular applications of the software likely to be relevant were suggested by some focus group students, for example:

> *Instead of just getting a general skim of everything [in the course], have the paper [such] where we went really in-depth into the basics, for example, these are XYZ… this is how you use them, how geometry is important and then here's some features [of SolidWorks] that would be relevant to do this*.

4) Informal learning strategies needed to complement formal coursework learning

Finally, there is a recognition that the conceptual and technical complexity of SolidWorks demands a more self-directed and committed investment in time to learn the software, which required developing informal learning strategies to complement the formal training provided within their tertiary programme. Students regularly drew from informal learning strategies and networks to support their learning of SolidWorks. When asked to rank strategies most useful to their learning, the top three strategies ranked as most useful (Rank 1) were 'asking the course lecturer' (40%), followed by 'going online to refer to the Internet for instructions' (12%) and 'referring to the course or lab notes' (10%). The top three strategies student ranked as the second most useful in their SolidWorks learning (Rank 2) were 'asking a friend' (24%), 'referring to the course or lab notes' (21%) and 'reading a manual of the software' (16%). Finally the top three strategies that were ranked as third most useful in students SolidWorks learning (Rank 3) were watching someone use the software (16%), discovering through trial and error (16%), and finally going online to watch video tutorials (15%). Overall, apart from asking the course lecturer, the reported strategies tend to draw from more informal resources that occurred outside of course or lab hours.

These were confirmed by open-ended survey responses and focus group commentary. A representative focus group comment was:

> *Most of my learning on SolidWorks has been done by working on it at home or playing around at home, e.g., how to do that, learning from peers and also YouTube videos. Like, if there's no one around and you can't do it, type it into Google, type it into YouTube and hopefully you'll get something and if you don't then get some help.* (Student)

This practice of mainly drawing from informal learning strategies continued when students were in their work placement. For example, learning from peers was common informal workplace learning practice which added to students' software literacy development:

> *I know that in my work placement, I had a couple of people who knew how to do everything and I would ask them. There was some stuff that they didn't know and there were some things that I'd learnt at uni that they didn't know existed in SolidWorks...*

Another student affirmed the value of this strategy when thrown into a challenging real world context to use the software appropriately:

> *On my first day I think I was sat down and he was like, 'Right, make this' and I made it and he was like that's totally wrong and then spent like three days teaching me how to use it, just how he liked it taught so.*

One student reflected on the strategies he had developed as part of highlighting the value of learning to troubleshoot and of persistence be it in more advanced coursework or while on work placement:

> *From [first and second year] we pick up all the basic stuff and learn how to do it, but during that process we learn how to use the troubleshooting method and that's I think the most valuable thing that helped me later on ... I'm confident with even something I don't know, I know how to find it, how to learn it from online resources then I can still make that happen [on SolidWorks]. I think that's the most valuable thing, that even later when I go to my fourth year and do some more complicated thing, I know where to go.*

Discussion and Conclusion

This study adopted a software literacy framework to investigate the extent second year engineering students developed the ability to use, troubleshoot and critique SolidWorks (CAD) software literacy (in formal learning context) and to apply and extend this understanding while on workplace experience. It confirmed findings from earlier studies that SolidWorks/ CAD knowledge and use, albeit it being complicated to learn, is an important and necessary aspect of engineering disciplinary knowledge (Akasah, & Alias, 2010; Khoo, Johnson, Torrens, & Fulton, 2011). Students reported developing a range of informal learning strategies to supplement their formal coursework learning and even while on work placement to extend and develop new software literacy skills. Some students are better at transferring their knowledge and use of SolidWorks from their coursework to their workplace depending on their experience with software involving similar conceptual ideas.

These findings have three implications for engineering education in relation to CAD software learning. Firstly, pedagogical strategies that provide explicit reference to the guiding principles and conventions of engineering design principles and how these might be implemented through CAD, even before students explore specific features of the software, can help students better understand the fundamental functions of the software as well as its potential.

Secondly, the teaching and learning of CAD software could take advantage of students' informal repertoire of learning strategies and networks including their accessing of (web-based) resources and discussions with 'expert' peers. Lecturers using a range of teaching approaches (formal and informal) and being flexible to address diverse learning needs is important for supporting and facilitating students' software learning.

Finally, engineering educators need to examine how discipline-specific software teaching-and-learning is positioned in relation to local and general goals for curriculum and the kinds of software literacies expected of students. Currently at the University of Waikato and likely in many other institutions, the focus when teaching CAD is for students to develop a proficiency with the software. Often there is little emphasis on evaluating different software packages. Students would likely only begin to develop this third level of software literacy if they were exposed to multiple software packages (something difficult to achieve in a tertiary environment due to time and resource constraints). We recognise there will be competing priorities for the discipline as a whole but argue that such examination can shape curricular decisions, learning opportunities and resourcing offered for supporting students' engagement and learning with and through CAD software and its application within the wider context of the field.

References

Akasah, Z., & Alias, M. (2010). Bridging the spatial visualisation skills gap through engineering drawing using the whole-to-parts approach. *Australasian Journal of Engineering Education*, *16*(1), 81-86.

Bennett, S., Maton, K., & Kervin, L. (2008). The 'digital natives' debate: A critical review of the evidence. *British Journal of Educational Technology*, *39*(5), 775–786.

Braun, V. & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77-101

Cole, M. & Engeström, Y. (1993). A cultural-historical approach to distributed cognition. In G. Salomon (Ed.), Distributed cognitions, psychological and educational considerations (pp.1-46). Cambridge: Cambridge University Press.

Hight, C., Khoo, E., Cowie, B., & Torrens, R. (2014). Software literacies in the tertiary environment. In B. Hegarty, J. McDonald, & S.-K. Loke (Eds.), *Rhetoric and reality: Critical perspectives on educational technology. Proceedings ascilite 2014* (pp. 410–415). Dunedin, New Zealand. Retrieved from http://ascilite2014.otago.ac.nz/proceedings/

Jenkins, H., Clinton, K., Purushotma, R., Robison, A., & Weigel, M. (2006). Confronting the challenges of participatory culture: Media education for the 21st Century. Chicago, Il: MacArthur Foundation.

Johri, A., Teo, H. J., Lo, J., Dufour, M., & Schram, A. (2014). Millennial engineers: Digital media and information ecology of engineering students. *Computers in Human Behavior*, *33*, 286–301. doi:10.1016/j.chb.2013.01.048

Jones, C., Ramanau, R., Cross, S., & Healing, G. (2010). Net generation or digital natives: Is there a distinct new generation entering university? *Computers and Education*, *54*(3), 722–732.

Khoo, E, Johnson, E M, Torrens, R, Fulton, J. (2011). It only took 2 clicks and he'd lost me: Dimensions of inclusion and exclusion in ICT supported tertiary engineering education. In Y. M. Al-Abdeli & E. Lindsay (Eds.) *22nd Annual Conference for the Australasian Association for Engineering Education, 5-7 December 2011* (p.166-171). Engineers Australia: Fremantle, Western Australia.

Khoo, E., Hight, C., Torrens, R., & Cowie, B. (2013). *Copy, cut and paste: How does this shape what we know?* TLRI research in progress. Retrieved from http://www.tlri.org.nz/tlri-research/research-progress/post-school-sector/copy-cut-and-paste-how-does-shape-what-we-know or from the TLRI website http://www.tlri.org.nz/tlri-research/research-progress/post-school-sector/copy-cut-and-paste-how-does-shape-what-we-know

Kvavik, R. B. (2005). *Convenience, communications, and control: How students use technology*. Retrieved from http://universityfinancelab.com/wp-content/uploads/2011/04/Con.pdf

Livingstone, S., Wijnen, C.W., Papaioannou, T., Costa, C. & del Mar Grandío, M. (2014). Situating media literacy in the changing media environment: critical insights from European research on audiences. In N. Carpentier, K. C. Schrøder & L. Hallet (Eds.), *Audience Transformations: Shifting Audience Positions in Late Modernity*, Routledge Studies in European Communication Research and Education, Vol. 1 (pp. 210 -227). NY: Routledge.

Manovich, L. (2013). *Software Takes Command* (International Texts in Critical Media Aesthetics, Vol. 5). NY: Bloomsbury Press.

Maykut, P., & Morehouse, R. (1994). Beginning qualitative research: A philosophic and practical guide. London: Falmer Press.

Peeters, J., Backer, F. D., Buffel, T., Kindekens, A., Struyven, K., Zhu, C., & Lombaerts, K. (2014). Adult learners' informal learning experiences in formal education setting. *Journal of Adult Development*, *21*(3), 181–192. doi.10.1007/s10804-014-9190-1

Valtonen, T., Dillon, P., Hacklin, S., & Väisänen, P. (2010). Net generation at social software: Challenging assumptions, clarifying relationships and raising implications for learning. *International Journal of Educational Research*, *49*(6), 210–219. doi:10.1016/j.ijer.2011.03.001

Acknowledgements

The authors gratefully acknowledge funding support from the Teaching and Learning Research Initiative, New Zealand Council for Educational Research, Wellington, New Zealand.

Copyright