“It runs slow and crashes often”: Exploring engineering students’ software literacy of CAD software

Elaine Khoo, Craig Hight, Rob Torrens, and Mike Duke
University of Waikato, Hamilton, New Zealand
Corresponding Author Email: torrens@waikato.ac.nz

Structured Abstract

BACKGROUND
This paper reports on findings from a two-year funded research project exploring ‘software literacy’ - how it is understood, developed and applied in tertiary teaching-learning contexts and how this understanding serves new learning. We define *software literacy* as involving the expertise in understanding, applying, problem solving and critiquing software in pursuit of particular learning and professional goals. Although concepts such as computer literacy, information literacy, digital literacy, digital information literacy and related terms have been well discussed, these do not go far enough in focusing on lecturer and student critique of a particular software; critique in terms of its affordances and constraints and how these shape the way knowledge within a discipline is communicated to impact on student learning. Developing the ability to critique constitutes an essential characterisation of a 21st century learner. In this paper, we report on a case study of a tertiary engineering course in a New Zealand university where the focus is on the learning and application of a 3-dimensional computer-aided design (3D CAD) software, SolidWorks, as a focus for exploring student understandings of software literacy.

PURPOSE
The research question guiding this study is: What understandings of disciplinary-specific software do students consider they learnt as part of their Engineering tertiary coursework?

DESIGN/METHOD
A qualitative interpretive methodology was adopted in this case study of a second year Engineering course focused on engineering design and process ideas. Data collected included student focus group interviews, an online student survey and lab observations of students’ SolidWorks learning. A constant comparison inductive approach to data analysis was adopted to identify emergent themes from the data.

RESULTS
The preliminary findings suggest a majority of students considered themselves to be confident and comfortable in engaging with new technologies, expressed a preference for a range of formal and informal strategies when learning about SolidWorks and developed a rudimentary awareness of its related affordances and constraints that impacted on their learning. Significant variations however exist in students' ability to comprehend, learn and apply SolidWorks in their coursework. While most students developed some level of competency in the software, students as a whole tended to fall short in their ability to critique the ways the software shaped their understanding of disciplinary knowledge.

CONCLUSIONS
These findings illustrate the complexities and challenges of SolidWorks teaching and learning and imply a need for lecturers and course designers to give thoughtful consideration to support students’ gaining the appropriate understanding of the affordances of the software and the disciplinary principles on which the software is premised. A case can also be made for a more formal recognition of software literacy as a means to empower students to more critically engage with software and its use.

KEYWORDS
Computer-aided design (CAD), SolidWorks, software, literacy, teaching and learning, university students

Introduction

All forms of learning especially at the tertiary level are increasingly embedded within, informed by and inevitably shaped by software. This assumption is the basis of a two-year research project funded by the Teaching and Learning Research Imitative (2013-2014) exploring the significance of *software literacy* in tertiary teaching and learning. The overarching goal of the project is to ascertain how lecturers and students learn, understand and take up the use of different kinds of software, and the implications of this for their teaching and learning practice.

In this paper, we report on a case study of a tertiary engineering course in a New Zealand university where the focus is on the learning and application of a 3-dimensional computer-aided design (3D CAD) software, *SolidWorks*, as a focus for exploring student understandings of software literacy. Previous studies have found support for the ways different digital technologies can significantly shape how and what millennial engineers learn (Johri, Teo, Lo, Dufour, & Schram, 2014). This has however not been investigated in the New Zealand context.

Software literacy as an emerging field

Software studies, a research paradigm championed by Manovich and others (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 hence 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.

We define *software literacy* as involving the expertise in understanding, applying, problem solving and critiquing software in pursuit of particular learning and professional goals. 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:

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

b. 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,

c. 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 develop the ability to 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. Ideally, we envisage software literate users being able to recognise, assess and critique the nature and implications of a variety of forms of software within everyday life. Users may acquire software literacies through a combination of any number 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.

We argue that that there is a need to revisit and revise concepts such as information literacy, digital literacy, and related terms (Hegarty et al., 2010; Livingstone et al., 2013) as these do not go far enough in focusing on how lecturers and students engage with specific software applications and its implications for student learning. We argue for the need to differentiate between distinct literacies relevant to specific technologies, and to examine the nature of student critique and decision making around which tools might best serve their learning purposes.

Labels such as *digital natives* claim to describe the characteristics of a new generation of learners, capable of operating at 'twitch speed' and able to multitask, imagine, and visualize while communicating in multiple modalities (Prensky, 2001). This term tends to conflate a basic skill with new technologies with broader forms of understanding and the ability to critique aspects of technology-based cultures. We need to unpack this set of assumptions to more carefully identify the range of skills and other literacies that today's students do (and do not) bring to their tertiary learning. There is emerging evidence that although this generation may be technologically competent, many still lack basic literacies needed to successfully apply software embedded and enabled technologies effectively to enhance their learning (Kvavik, 2005). A crucial question here is whether, in an environment of universal access to digital tools, the *digital divide* is being reconfigured as inequalities in software literacy. Recent research indicates that inequalities and marginalisation persist around students' access to, and use of information and knowledge (Bennett, Maton, & Kervin, 2008). Digital inequality is not restricted to just the issue of physical access to software and hardware (Selwyn & Facer, 2007), and 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 to date 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.

Research Context

In this paper, we explore the relationship between student success in acquiring *software literacy* (as demonstrated through their learning of SolidWorks) and their broader engagement and understanding of knowledge about engineering as a discipline. The research question guiding this case study is: What understandings of disciplinary-specific software do students consider they learnt as part of their Engineering tertiary coursework?

The case studied engineering course is a second year course introducing students to the broad principles of engineering design process and methodology. It caters primarily for students majoring in the Mechanical and Electronic Engineering streams. A significant proportion of the course assessment requires students to use SolidWorks to effectively implement engineering design principles. SolidWorks is considered an integral component of modern engineering and is widely used in industry.

Building from an introductory experience to SolidWorks in their first year coursework, the course offered a more detailed discussion of and 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, process and project management through examining and discussing case studies of designs. They also attend 5 two hour weekly supervised computer lab time (workshops) where they are provided with tasks to help them acquire further proficiency with SolidWorks and work on individual assignments. Students were also required to participate in a group design project and presentation as a demonstration of their SolidWorks supported design understanding and application. They can, however, opt to learn SolidWorks at home and install it on their own computers or use the lab in their own time. In this study, we focus on students' views on their learning of SolidWorks as part of their coursework.

A qualitative interpretive methodology framed the research design and conduct of the study (Maykut & Morehouse, 1994). Data collected included student focus group interviews (6 students), an online student survey (67 students out of approximately 140 students) and lab observations of students' SolidWorks learning. Analysis of the data was underpinned by sociocultural theory which directed attention to the interaction between people, the tools they used to achieve particular purposes and the settings in which the interactions occur. A constant comparison inductive approach to data analysis was adopted to identify emergent themes from the data (Lincoln & Guba, 1985). The project received human ethical approval and all participants participated on a voluntary basis.

Findings

Four key themes emerged from investigating students' perspectives about student learning of software: 1) their general comfort level in engaging with technology; 2) students' overall preference for and/or reliance on informal learning strategies in learning to use SolidWorks; 3) their understanding of core affordances and constraints of SolidWorks; and, 4) a relative absence of critical software literacy among our participants.

1. Student comfort level with technologies

When asked about their general views towards adopting technologies, 43% of first-year students indicated they usually use new technologies when most of their friends do, 31% reported liking new technologies and using them before most people they know do, and another 10% indicated they love new technologies and are among the early adopters to use them. These results highlight therefore that a majority of students (84%) consider themselves early or quite early adopters of new technologies and are comfortable in engaging with new technologies.

2. Student preference for informal learning strategies in acquiring software skills

Students drew mostly from informal formal learning resources when learning to use 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%) (see Table 1). 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.

Table 1. Ranking of strategies useful in SolidWorks learning

Rank 1	Ask the course lecturer/ tutor	40%
	Go online/refer to the Internet for step-by-step instructions	12
	Refer to the course /lab notes	10
Rank 2	Ask a friend/peer/other student	24
	Refer to the course/ lab notes	21
	Read a paper-based manual/step-by-step instruction booklet	16

Rank 3	Watch someone using it in a face-to-face (physical) setting (not through videos)	16
	Discover through trial-and-error/ practice	16
	Go online/refer to the Internet for video tutorials (eg. Youtube)	15

Note: Rank 1 to Rank 3 denotes descending order of useful strategies in SolidWorks learning as reported by students (n=67)

Additionally, 76% of the participants reported installing SolidWorks on their own computers or laptops in order to be able to practice and use the software for their coursework. In focus groups, students reported drawing from a variety of resources to help them learn to use SolidWorks at their own pace. These include going through the SolidWorks in-built tutorials, sometimes drawing upon 'more expert' peers, practising through using their intuition and trial and error, as well as using online materials such as YouTube instruction videos (which notably involved developing an expertise in finding instructional material suited to 'their level'). An example of student drawing from more expert peers is:

> *I've been working next to a fourth year I'm friends with and he's looked at my work and gone, 'whoa, dude, hold on - let me show you how to do this' and he's stepped in and shown me a whole bunch of stuff.*

These students faced with learning more complicated discipline-specific software such as SolidWorks recognised that it demanded a greater investment in time to learn:

> *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).*

> *SolidWorks has a learning curve which can make things harder to do.*

3. Student understanding of software affordances

Students demonstrated a basic level of familiarity with SolidWorks and easily identified its key affordances and constraints. For example, when asked their views on how SolidWorks affords their addressing of engineering design issues, students indicated it allowed them to rotate and manipulate different views of their drawings (81%), and to easily modify their drawings (79%) (see Table 2).

Table 2. SolidWorks' Affordances in Addressing Engineering Problems

Allows me to rotate /manipulate different angles or views of my drawings	81%
Allows me to easily modify my drawings	79
Allows me to draw an object to see what it looks like (or to share with others my drawing so they know what I mean)	78
Helps me design and draw things before building them	78
Helps me visualise a solution that was in my mind in 3D format	75
Helps to add details to my drawings so that they could be manufactured from	61
Helps me explore effects of changes to measurements in my drawings	54

Additional responses alluded to the ways SolidWorks facilitates being able to "communicate my ideas" and to specify "properties of objects required, e.g. volume, weight" for further exploration. Another 19% of students reported using a range of software that had similar features to SolidWorks such as ProEngineer, AutoCAD, Star CCM+, Autodesk Inventor and TurboCAD. They were able to comment on the similarities between these and SolidWorks in terms of their function. Similar features between these different software were also noted such as "They have logical icons" and they have "sketch planes, extrudes, features, main interface."

Interestingly, just over half (54%) of these students were able to elaborate on the benefits of using these other similar software prior to their learning SolidWorks. These ranged from how the different software provided them with the "basic skills and familiarity" in using CAD and "how software works in general" including more shared technical understanding of "reference geometry" and so forth.

Over 28% of our participants also identified the main *constraints* of SolidWorks in relation to issues to do with:

1) accessing the software (e.g. affordability, unable to install on their personal laptops, incompatibility in opening saved files on other computers),
2) learning to use the software: "It runs slow and crashes often",
3) using particular features of the software such as, "The menus are too large, it makes it hard to find the tool you are looking for", "Can't find certain things e.g. chamfer", "Unable to mirror 3D sketches", "Moving a shape along a plane; use a plane as reference", "Change things late in design", "Move my gears after I add one assembly file to another", and "Changing 2D to 3D drawing".

The focus group participants confirmed that SolidWorks was a complicated discipline-specific software which they were not familiar with prior to tertiary study:

> *It's entry level for us, it's [SolidWorks is] super complicated - we should be learning it at that lower level, then advancing later on; or it should be a paper unto itself, because it makes the rest of the paper so difficult.*

As a result, students were less likely to identify themselves as 'highly proficient' or 'expert' in using SolidWorks at the completion of their course.

4. Relative absence of critical literacy among students

Students reported shifting in their ability to use SolidWorks after learning and using it in the course. Just over half of the students (52%) reported needing help to use the software initially before attending the course. This impressively decreased to 2% at the end of the course. Also there was an increase from 39% to 45% of students who felt they now have the basic skills to use SolidWorks after learning about it in the course (see Table 3). 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. However by the end of the course, only 16% 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 3. Student assessment of their ability to use SolidWorks

	How good were you in using SolidWorks before enrolling in this paper?	After learning about and using SolidWorks in this paper, how 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

Despite of this, 27% of students reported using SolidWorks outside of their formal coursework for a range of professional or recreational purposes. Some examples include:

> *I have many sketches which I have a hard time imagining in 3D therefore I use SolidWorks to give me a more detailed version of what I have imagined.*

Very few students however discussed how SolidWorks shaped their disciplinary knowledge (a key part of software literacy). In the focus group, three students alluded to this by noting the need to be taught the overarching principles in terms of Engineering design as well as CAD conventions can guide their SolidWorks use and enhance their understanding of the potential of the software:

> *I think what would be cool is if we had case studies or something; just some problems in class we could work through, the teacher could go through, like, 'this is something that you may encounter while you're doing CAD, this is how we've gone about it, you could do it your way but this is the procedure we've used …*

Discussion and Conclusion

Our study aimed to address the extent tertiary students are critically aware of how discipline specific software such as SolidWorks in this case, can impact their learning as a demonstration of their emerging software literacy. Our participants were generally comfortable with engaging with new technologies, identifying themselves as early to quite early adopters of technologies (84%). They reported a range of learning strategies (mostly informal apart from asking the course lecturer) and a preference for a variety of resources to draw from at their own pace when acquiring the skills to use SolidWorks. These include referring to the Internet, asking more expert peers, referring to notes and so forth. Both these findings support assumptions in the 'digital natives' label (Prensky, 2001). Further, students could successfully identify the affordances and constraints of SolidWorks use and similar functions and features of other CAD software which provided them with skills they could transfer to their SolidWorks learning. Students reported an increase in their basic and troubleshooting skills at the end of the course which signalled their growing understanding and use of the software corresponding to the first two levels of our software literacy schema.

However, student critique of how SolidWorks might *shape* their disciplinary knowledge was surprisingly superficial (In our terms, there was a clear absence of the third level of software literacy). This was largely due to the perception that SolidWorks was a complicated software to learn, one involving a steep learning curve. Although students use SolidWorks for recreational purposes outside of their formal coursework (27%), only a minority of participants (16%) thought they could successfully apply their CAD skills to a wide range of tasks.

These findings have implications for tertiary teaching and learning. Firstly, teaching and learning of courses involving a focus on software can be informed by and take advantage of students' informal repertoire of learning strategies. As CAD is a complicated software to learn, lecturers will need to provide the time, carefully plan and scaffold student learning (Akasah & Alias, 2010). Offering access to a variety of support and resources (online and paper based including expert peers/senior students) can cater to students' different learning preferences and ways of learning software (Khoo, Johnson, Torrens, Fulton, 2011).

Secondly, 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. Our previous work in CAD learning illustrated the potential of SolidWorks in scaffolding students' emergent visual spatial thinking and conceptual understanding (Johnson, Khoo, Cowie, De Lange, Torrens, 2011). There was however a need to recognise students' varying levels of background conceptual knowledge, experiences, and learning preferences in the course design and instruction (Khoo, Johnson, Torrens, Fulton, 2011). Discussing the overarching principles and making explicit links to CAD teaching and learning would be helpful in grounding student understanding of engineering design and the software tools supportive of the process.

Next, students' superficial critique of SolidWorks revealed that critical awareness does not necessarily develop naturally as a result of use of a software, rather it needs to be prompted and/ or explicitly taught. Lecturer prompting/explicit modelling the critique of software including pointing out the big ideas/ principles of CAD use can help students develop similar awareness and capacities to be more critically aware of their software use.

The next phase of the study will follow up and track the extent these students are able transfer and apply or adapt their software literacy associated with SolidWorks in the workplace setting as a more contextualised example of software use. This will provide valuable insights into ways to better support students learning with and through software as part of their tertiary Engineering experience.

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.

Hegarty, B., Penman, M., Kelly, O., Jeffrey, L., Coburn, D., & McDonald, J. (2010). *Digital Information Literacy: Supported Development of Capability in Tertiary Environments*. Wellington, New Zealand: Ministry of Education. Retrieved from http://www.educationcounts.govt.nz/publications/tertiary_education/80624

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

Johnson, E. M., Khoo, E., Cowie, B., De Lange, W., Torrens, R. (2011). ICT/elearning for developing visual spatial thinking in university science teaching. In A. Verbraeck, M. Helfert, J. Cordeiro, & B. Shishkov (Eds.), *CSEDU 2011: Proceedings of the 3rd International Conference on Computer Supported Education, 6-8 May 2011* (p.73-78). SciTePress, Noordwijkerhout: The Netherlands.

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.

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

Lincoln, Y.S.& Guba, E.G. (1985). Naturalistic Inquiry. Newbury Park, CA: Sage Publications.

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.

Prensky, M. (2001). Digital natives, digital immigrants. *On the Horizon*, *9*(5), 1–6.

Selwyn, N., & Facer, K. (2007). *Beyond the digital divide: Rethinking digital inclusion for the 21st century*. Futurelab. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.3384&rep=rep1&type=pdf

Acknowledgements

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

Copyright statement