How to decrease cognitive load to facilitate learning software applications

Ali Darejeh – University of NSW
Nadine Marcus - University of NSW
John Sweller - University of NSW

By increasing the number of features that software applications provide, they become more complex and in order to use the software efficiently, effective training methods that integrate psychologically based guidelines with modern training technologies are essential. To learn to use software efficiently, users need to learn the interface layout first, then learn the usage of each tool, finally the relations between different tools. It can be difficult for users with limited computer literacy to understand the layout and usage of different tools of a novel productivity software application that includes multiple features. The unfamiliar and complex structure of the interface can increase cognitive load and make the application incomprehensible for users (Reis et al., 2012; Fang, Luo & Xu, 2011). Therefore, finding solutions to decrease cognitive load can have a significant positive effect on learning software applications. This paper overviews the literature to provide practical solutions to facilitate learning complex software through controlling learners' cognitive load. These solutions are based on cognitive load theory principles and effects and include: providing the necessary pre-required knowledge for the software tools that are high in element interactivity; breaking down software tool instructions into smaller chunks; linking the layout and tools of the target software to software that users are familiar with; facilitating the formation of schemas in learners' long-term memory through presenting an infographic to provide a summary in the form of visual instruction at the end of the lesson to indicate each step on the software interface; applying worked examples by using a practical example for teaching each tool; preventing split attention by highlighting the target tool and its settings while they are taught; avoiding redundancy effects by not adding one screenshot twice or having descriptive text in addition to speech for explaining how to work with different tools; considering the modality effect by engaging both visual and audio channels by using the software interface as the visual element and then using simple and short speech to describe the tools (Nievelstein et al. 2013; Jin, 2012; Sweller, Ayres, & Kalyuga, 2011; Ayres & Sweller, 2014; Chen, Kalyuga, & Sweller, 2015). The proposed solutions, which reference current research related to adapting the learning environment to human cognitive structures, can increase learning performance by preventing working memory from being overwhelmed, thus facilitating schema formation resulting in more efficient and reliable learning of software with less effort (Lyra et al., 2016; Dahl, Flygare, & Zheng, 2008; Sweller, Ayres, & Kalyuga, 2011; Chen, Kalyuga, & Sweller, 2016).