MEMORY LOAD: A FACTOR THAT LINKS THE USABILITY OF INDIVIDUAL INTERACTION COMPONENTS TOGETHER

Willem-Paul Brinkman	Reinder Haakma	Don G. Bouwhuis
Brunel University	Philips Research Lab Eindhoven	Technische Universiteit Eindhoven
Uxbridge, Middlesex, UB8 3PH, United Kingdom	Prof. Holstlaan 4, 5656 AA Eindhoven, The Netherlands	PO Box 513, 5600 MB Eindhoven, The Netherlands
willem.brinkman@brunel.ac.uk	reinder.haakma@philips.com	d.g.bouwhuis@tue.nl

ABSTRACT

An underlying assumption of component-based software engineering for interactive systems is that the overall usability of a new assembled system mainly depends on the usability of its individual components. This paper challenges this assumption by presenting findings of a lab experiment. Here users were asked to use two calculators, one with a small display and one with a large display. Results show a significant change in the way users solved equations with the two calculators when faced with high memory demands. Although the effects of memory load is not new, these findings show empirically how it can also affect the interaction with components not directly responsible for it. Therefore when constructing a new system out of ready-made components, developers should still evaluate the new system as a whole since usable components tested in isolation might still have a negative effect on the way users interact with other components.

Keywords

Usability testing, component-based software engineering, memory load.

1. INTRODUCTION

Instead of building a system from scratch, Component-Based Software Engineering (CBSE) allows software engineers to assemble a new system from ready or self-made components. These components encapsulate pieces of software that can operate autonomously, free of the internal details of other components. Hiding internal operations of software parts is one of the reasons behind the success of object-oriented development. It reduces complexity and makes large applications more maintainable. Software engineers also apply CBSE for the development of interactive applications. Pop-up menus, search engines, dictionaries, or spell checkers are all examples of ready-made components that can be reused in different applications. Users can interact with these components directly or indirectly via mediation of other components creating layered interaction. HCI theories such as the Layered Protocol Theory (LPT) [3] support layered interaction, and therefore CBSE. LPT claims that protocols (read components) can be replaced by others as long as the protocols offer the same services to other layers. This suggests that using highly usable components, studied in isolation, will result in highly usable systems. However, others [5] suggest that software re-use can cause conceptual mismatches. The same concept may be used in several components, but it may not mean the exact same thing. We argue here that memory load can also cause components to affect each other's usability negatively, making an overall usability prediction of a system based on the usability of the individual components less valid. This means that although a component can be developed and tested in isolation, a usability evaluation of the entire system is still required.

The effect of mental demands on users has been the topic of extensive research, which has revealed the different strategies users apply to manage different mental demands. When faced with an interaction strategy that is mentally too demanding to maintain, users may start deploying other strategies at the expense of efficiency to reach the primary goal or to remain within acceptable operational limits [2, 7]. To make this more concrete, consider a driving example. After waiting for a traffic light to turn to green, student-drivers may turn a corner still driving in first gear and only put the car in the second gear after they have taken the corner. Although the first gear is mainly intended to get the car moving after a standstill and not to drive in, student-drivers may be unable to change up fast enough before the corner while remaining in control of the car in the bend. To avoid a loss of control over the car, students end up taking the corner at a lower speed. In this case, making it easy to shift gear is more effective than making steering easier.

Proceedings Volume 2 of the Conference

HCI 2004: Design for Life.

Leeds, UK, September 2004.

Another example is an observation made of a vision-impaired user that browsed the web with a new screenreader [8]. The user had made his own command sheet in Braille to avoid losing his train of though as he had to split his attention between understanding the new screenreader, the browser and the web site.

These examples above illustrate that mental demands caused by one part (gear or screenreader) can affect the interaction with another part (steer or web site). An explanation for this all can be found, for example, in classic limited mental capacity theories, which state that the fundamental constraint that underlies all mental operations is a limited information-processing capacity. To compensate for high demand of cognitive resources caused by the interaction with one component, users have to change their interaction strategy with other components to avoid draining the limited mental resources. Although often suggested, little research as been reported that actually presents empirical findings to support the idea of mental load linking the usability of individual components in a *single* device. Therefore, we conducted a lab study in which this idea is tested experimentally under controlled conditions by using two different versions of a calculator.

2. METHOD

2.1 Calculators

The experiment involved an application, written in Delphi 5, consisting of two calculators, and a recording mechanism to analyse the message exchange between the components [1]. The two important *interaction components* [1] identified in the development of the calculators were the so-called editor and processor component. The editor was responsible for forming the equation and passing it on to the processor. Users could enter a complete equation, including special mathematical functions (sine, ln, square root, etc), which would be passed on only after users pressed the '=' button or the 'STO' button followed by a memory button (M1 to M6). The processor processed the equation, placed the result in one of the six memory places if requested, and sent the result back to the editor. Therefore, users could only interact with the processor by mediation of the editor component. Two versions of the editor were designed: one with a large display (Figure 1, left), the other with a small display (Figure 1, right).

Figure 1: Large (left) and small display (right) calculator.

$$
\begin{aligned}
&((M1 + 61) \times 13 \times (1 - 0.0847) + \\
&\ (M4 + 74) \times 48 \times (1.0378 - 0.1237) + \\
&\ (M2 + 72) \times 12 + \\
&\ (M6 + 85) \times 11 + \\
&\ (M5 + 74) \times 15 \times 1.0378 + \\
&\ (M3 + 62) \times 21 \times (1 - 0.0847 - 0.1237) \\
&\ + 3468 + 1273 \\
&) \times (1.1442 + 0.175) + 106 =
\end{aligned}
$$

Figure 2: Example of a difficult equation, which users had to derive from a textual description. M-codes are the memory places that hold the intermediate outcomes.

The small display showed only a small part of an equation, for example: a value, an operator ('+'), a mathematical function ('sin'), a reference to a memory place ('M1'), or a list of consecutive opening and closing brackets ('((('). The large display showed the complete equation on a screen of five lines with 34 symbols each. If the equation was longer, the users could scroll up and down in the large display.

2.2 Tasks

The users were asked to calculate the cost of several building projects based on a textual description. In principle they had to solve two types of equations (easy or difficult): one that could be solved without using brackets, the other required using brackets (nesting depth of 2, Figure 2) to enforce a correct order of processing. Although the two types of equations had different levels of difficulty, the equations required the same number of keystrokes if entered in an optimal form. The cost calculation task was finished once the users had calculated the correct answer.

Expectations were that users would only change their interaction strategy when the memory demand exceeded a certain threshold. Below this threshold, users would maintain their strategy by making more mental effort. Above this threshold, users would change their strategy for a less mental demanding strategy bringing memory demand back below the threshold. To push the users towards this threshold, the memory load was increased by two factors. First, users had to memorise the memory places of intermediate outcomes, which were stored in advance in the calculators. Second, the users were interrupted and required to perform another task before they regain access to the calculator and could continue with the original task.

2.3 Hypothesis, Design, and Users

The experiment was set up to demonstrate a two-way interaction effect between the task difficulty (easy/difficult) and the version of the editor (small/large) in the way users would interact with the processor. The explanation for the change in strategy was the memory demands placed upon the users and the way the editor could help the user to cope with this. There were 24 users who completed the experiment, all students of Technische Universiteit

Eindhoven. They were expected to have extensive experience with calculators and be acquainted with mathematical priority rules. However, they were not expected to have experience with calculating the building costs for this experiment. The experiment had a 2×2 within-subject design —editor (small or large) × equation difficulty (difficult, or easy)— and was counterbalanced for learning and fatigue effects.

3. RESULTS

3.1 Overall Interaction

A doubly Multivariate Analysis Of Variance (MANOVA) was conducted on the task time and the number of keystrokes. Both measures were first logarithmically transformed to decrease the effect of extremely high values. The within-subject variables were the editor version (2) and the equation difficulty (2). The analysis only revealed a significant main effect for the equation difficulty ($F(2,22) = 33.00$; $p. < 0.001$). Inspection of the means showed that more time and keystrokes were needed in the case of a difficult equation.

3.2 Interaction with the Processor

The most efficient strategy to solve the equations did not require the users to store intermediate outcomes. Therefore to study the efficiency of the interaction strategy, the logarithmically transformed number of store requests sent to the processor was analysed with a univariate analysis of variance with repeated measures. The same within-subject variables were used as before.

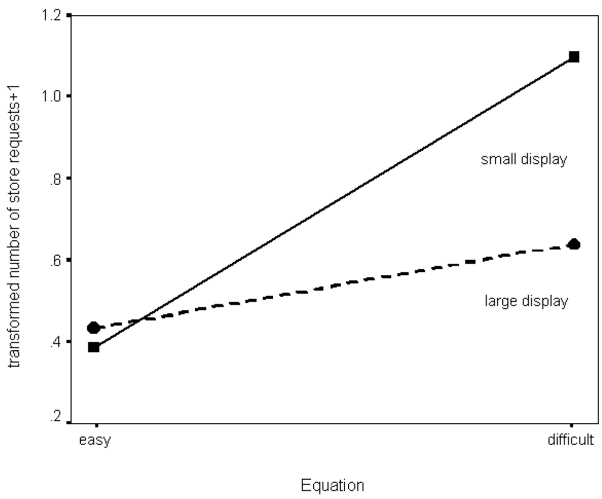

Figure 3: The mean of the logarithmically transformed number of store-request (plus 1) for the two calculators while performing an easy and a difficult equation.

This time the analysis found a significant two-way interaction effect ($F(1,23) = 6.81$; $p. = 0.016$). As can be seen in Figure 3, users more often requested to store a result when they had to solve a difficult equation with the small display calculator than in all other conditions.

3.3 Mental Effort

The mental effort was measured with a subjective and with a physiological measure. After the users solved an equation, they rated the effort on the Rating Scale Mental Effort (RSME) [10]. The other measure was based on the Heart Rate Variability (HRV) of the users in the first 5 minutes of the task. HRV has been shown to relate to a person's mental effort. A person's heart rate is more regular during the performance of effort-full mental tasks. Especially the HRV frequency band around 0.1 Hz, which relates to the regulation of short-term blood pressure, has been found to be sensitive to mental effort [6]. After pre-processing the heart rate data, adjusting for possible artefacts such as missing or extra heartbeats, a fifth order Lagrange interpolation was applied to obtain equidistant time series. The next step was the transformation of the equidistant time series to modulation index series. This is the expression of each sample as a percentage of the mean inter-beat-interval, which removes the possible effect the heart-rate may have on HRV [9]. Finally, the frequency spectrum was analysed with a Fast Fourier Transformation. Other effects on the HRV such as body temperature regulation (area between 0.02 to 0.06 Hz) and respiration related fluctuations (areas between 0.15 to 0.5 Hz) were removed by only taking the area between 0.07 and 0.14 Hz as the HRV 0.1 Hz band measure.

A doubly MANOVA was conducted on the logarithmically transformed HRV 0.1 Hz band measure and the RSME score. Again the within-subject variables were the editor version and the equation difficulty. A significant main effect for the equation difficulty was found ($F(2,21) = 14.57$; $p. < 0.001$). This effect was also found in the univariate analysis of the RSME score ($F(1,22) = 30.52$; $p. < 0.001$). The users rated the required effort lower for an easy equation than for solving a difficult equation. The doubly MANOVA also revealed a two-way interaction effect between the equation difficulty and the editor version ($F(2,21) = 4.15$; $p. = 0.030$). This effect was only found again in the univariate analysis of the HRV 0.1 Hz band ($F(1,22) = 8.01$; $p. = 0.010$). Figure 4 shows the means in the four conditions. For the small display calculator, the changeover from the easy to the difficult equation seems to be associated with an increase in the HRV (decrease in mental effort), whereas for the large display calculator a decrease is apparent (increase in mental effort).

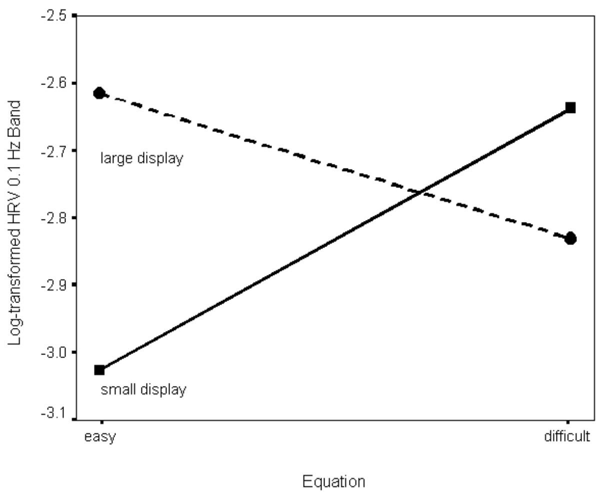

Figure 4: The mean of the logarithmically transformed HRV of the 0.1 Hz band for the two calculators while performing an easy and a difficult equation. A decrease in HRV is interpreted as an increase in mental effort.

4. DISCUSSION AND CONCLUSIONS

The experimental results support the main hypothesis that mental effort creates a link between the editor version and the efficiency of the interaction strategy with the processor. When solving a difficult equation with the small display calculator, users changed to a less efficient strategy, which however required less mental effort in solving the equations. In this condition users more often store intermediate outcomes than in the other conditions. Presumably, the users applied a so-called problem-reduction strategy [4]. This means breaking the problem into sub-problems, solving the sub-problem, storing them away and finally combining the intermediate outcomes into the overall result. In the other conditions, the users tend towards calculating the building cost within a single equation by putting in more, yet manageable, mental effort.

The results of the experiment demonstrate that a component can make users select a less efficient interaction strategy with another component in an attempt to reduce mental effort. In this experiment, the efficiency of the processor depended on the implementation of the editor. Therefore, any generalisation about the usability of the processor should take the editor into consideration. This suggests that designers of web pages, electronic documents, or other components that are only accessible by mediation of other components (e.g. browsers, screenreaders, text editors) should not only take into consideration the mental effort involved in interacting with their component but also the mental effort involved in interacting with the mediated components. These mediating components could cause a bottleneck in the interaction with their component. Although designers could test and develop a component initially in isolation, a final overall usability evaluation remains necessary to determine the usability of the component [1] in the context in which it will be deployed. For example, testing a web page with different browsers should extend the scope of only spotting the unexpected technical oddities. It should include user tests with both behavioural and mental effort measurement to understand both the interaction strategies and the mental effort involved. When observing inefficient interaction with their component, designers have to consider that this could be caused by other components. An indication of low mental effort in these cases might suggest compensational behaviour.

5. REFERENCES

[1] Brinkman, W.-P., Haakma, R., and Bouwhuis, D.G. (2004). Usability testing of interaction components: Taking the message exchange as a measure of usability. Pre-Proc. CADUI'04, Kluwer Academic Press, 159-170.

[2] Cnossen, F. (2000). Adaptive strategies and goal management in car drivers. Doctoral dissertation, Rijksuniversiteit Groningen, The Netherlands.

[3] Farrell, P.S.E., Hollands, J.G., Taylor, M.M., and Gamble, H.D. (1999). Perceptual control and layered protocols in interface design: I. Fundamental concepts. Int. Journal of Human-Computer Studies 50, 489-520.

[4] Halasz, F. G., and Moran, T. P. (1983). Mental models and problem solving in using a calculator. Proc. CHI'83. ACM Press, 212-216.

[5] Hertzum, M. (2000). Component-based design may degrade system usability: Consequences of software reuse. Proc. OZCHI 2000. Sydney: Ergonomics Society of Australia, 88-94.

[6] Mulder, G. (1980). The heart of mental effort: Studies in the cardiovascular psychophysiology of mental work. Doctoral dissertation, Rijksuniversiteit Groningen, The Netherlands.

[7] Sperandio, J. C. (1971). Variation of operator's strategies and regulating effects on workload. Ergonomics 14, 571-577.

[8] Theofanos, M.F. and Redish, J. (2003). Bridging the gap: between accessibility and usability. Interactions, 10(6), 36-51.

[9] Van Dellen, H. J., Aasman, J., Mulder, L. J. M., and Mulder, G. (1985). Time domain versus frequency domain measure of heart-rate variability. J.F. Orlebeke, G. Mulder and L.J.P. Doornen (Eds.) Psychophysiology of cardiovascular control. Plemum Press, New York, NY, USA, 353-374.

[10] Zijlstra, F. R. H. (1993). Efficiency in work behaviour: A design approach for modern tools. Doctoral dissertation, Delft University of Technology, The Netherlands.