Consistency: a Factor that Links the Usability of Individual Interaction Components Together

Willem-Paul Brinkman
Brunel University
Uxbridge, Middlesex
UB8 3PH
United Kingdom
willem.brinkman@brunel.ac.uk

Reinder Haakma
Philips Research Laboratories Eindhoven
Prof. Holstlaan 4
5656 AA Eindhoven
The Netherlands
reinder.haakma@philips.com

Don G. Bouwhuis
Technische Universiteit Eindhoven
P.O. Box 513
5600 MB Eindhoven
The Netherlands
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 device mainly depends on the usability of its individual components. This paper challenges this assumption by presenting findings from a series of lab experiments in which 48 subjects operated several consumer devices. The experiments focussed on the effect inconsistency may have on the usability of individual components. The results indicate that inconsistency could cause components, in the same or in higher layers, to activate an inappropriate mental model for other components. Furthermore, the application domain also seems to have an effect on the subjects' understanding of the functionality a component provides.

Keywords
Consistency, usability, component-based software engineering, usability testing, usability evaluation.

INTRODUCTION
Component-Based Software Engineering (CBSE) advocates the development of independent components, which can be used to create a new device. To do this, components should be autonomous units, free of the context in which they are deployed. This idea is one of the major success factors behind object-oriented development; it reduces the complexity of large software projects and improves the maintenance and reliability of a system (Cox, 1990). This approach is also used for the development of interactive systems. Interaction components such as pop-up menus, radio buttons, or more complex components such as a spell checker or an email component are developed and tested in isolation to optimise usability. HCI theories such as the Layered Protocol Theory (LPT) (Farrell, Hollands, Taylor & Gamble, 1999) support CBSE. LPT describes how interactive systems can be broken down into individual components and claims that these components can be replaced by other components without affecting the remaining part of the system as long as components provide the same services (Taylor, 1988). The underlying idea is that using highly usable components will result in highly usable systems. This claim is supported by a number of empirical studies (Brinkman, 2003). These studies did not find that the components in the systems they studied affected each other's individual usability. However, others (Hertzum, 2000) 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 inconsistency 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 device is still required.

In this paper we present the results of three experiments on the effect of inconsistency and the usability of individual components. Before the experiments are discussed, the next section gives a brief introduction on LPT and on the concept of consistency. The paper ends with a discussion about the possible implications of the findings.

LAYERED PROTOCOL THEORY
LPT is a special form of Powers' (1973, 1998) Perception Control Theory (PCT). Whereas PCT is a general theory about human interaction with the environment, LPT focuses on the human-human and human-system communication and interaction. LPT describes the user-system interaction as taking place on multiple layers. It breaks up the traditional single entity of an interactive system into individual components and explains how users control these components. Central concepts in the theory are the perception-control loop (Haakma, 1999) (Figure 1) and the accumulation of these control loops (Figure 2).

The perception-control loop emphasizes that a component ultimately interacts with users and not only with its surrounding components. By exchanging messages users are controlling their perception of the component's state. This means that users are continuously going through the four stages defined by Norman (1984): intention formation, evaluation, action

selection, and action execution. For instance, a woman is reading a book in a foreign language when she comes across a word she is unfamiliar with. She looks around and sees her electronic translator. This might trigger her in forming the intention of obtaining the English translation of the word from her electronic translator. As she picks up the translator and looks at the display, she notices that it is still displaying the result of her last search. However, since she wants the translation of another word, she presses the Clear button, which results in a blank display. She enters the word, presses the Enter button, and is satisfied when the display shows the English translation of the word she was looking for. In this short example, the button labels present *expectation* feedback (E-feedback). E-feedback guides users to establish the appropriate intentions and helps them to select the right actions. The feedback on the display is called *interpretation* feedback (I-feedback). This feedback shows the state the system is in; or more precise, how it interpreted the last message sent by the user.

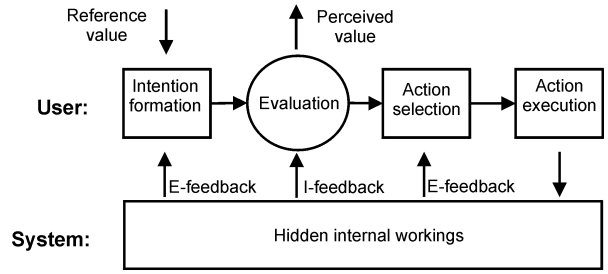

Figure 1: Perceptual-control loop.

A combination of higher-level goals (reference value) and E-feedback, on what is achievable with the system, establishes the appropriate intention. In the example, the reference value was the English translation of the word, and the E-feedback was the electronic translator as a whole. Together, these help the woman creating the intention to get the translator in a state that she could perceive the English translation of the word. This intention initiates the perceptual-control loop. The woman starts evaluating the content of the display on whether she can perceive the right translation, i.e. whether I-feedback and reference value are similar. If this is not the case, she may decide to perform an action, which is seen as sending a user message to the system. After the system receives her message, it may change its state accordingly, and it may send an I-feedback message back to inform her of the new state. This will trigger a new cycle of the loop, as she has to evaluate the new I-feedback and the reference value. The loop continues until she perceives the right I-feedback or when she changes her belief that the electronic translator can satisfy her goal.

Instead of considering one overall control loop for the whole system, the state of each component is the object of control in individual control loops according to LPT. In the case of the example this would mean a control loop for the Word Editor component, and a higher-level control loop for the Dictionary component (Figure 2). Conceptually the message exchange seems to pass from the user to the components and back within the same control loop, but in fact all messages pass down from the originating sender to components at the bottom-level layer and from that layer up at the receiving chain to the receiver that operates in the same layer as the originating sender. Only on the lowest level do users control the component directly by performing physical actions and perceiving physical signals. The interaction with the higher-level component is regarded as a virtual message exchange, which is mediated by lower-level components.

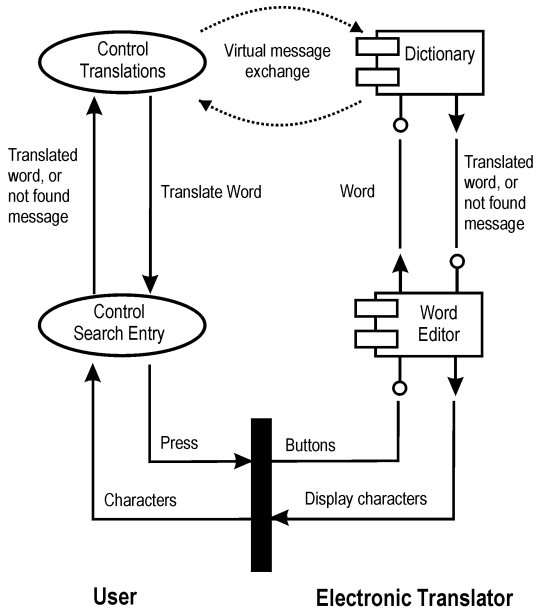

Figure 2: Layered user-system interaction between a user and the components of an electronic translator.

Interaction components are regarded as elementary units of interactive systems, on which behaviour-based evaluation is possible (Brinkman, Haakma & Bouwhuis, 2004). An interaction component is a unit within an application that directly, or indirectly via lower-level components, receives messages from the user. These messages enable the user to change the physical state of the interaction component. Furthermore, the user must be able to perceive or to infer the state of the interaction component. Therefore, an interaction component should provide I-feedback. Without the possibility of perceiving the state, users cannot control it, making their behaviour aimless. The points where input and output of different interaction components are connected demarcate the border between layers. An interaction component operates on a higher-level layer than another interaction component, when the higher-level interaction component receives its user messages from the other interaction component. In the example, the Dictionary interaction component would be considered operating on a higher-level layer as it receives its user messages from the Word Editor interaction component.

CONSISTENCY

Consistency has no meaning on its own; it is inherently a relational concept (Kellogg, 1989) and can be described as doing similar things in similar ways with agreement between agents about which things are similar (Reisner, 1993). This means that a component is regarded as consistent when both designers and users partition the interaction with the component in the same way. Furthermore, designers and users have to apply the same criteria, or dimensions, to consider the interaction with components to be similar. Likewise, inconsistency involves disagreement between designers and users about which things are similar, since what designers may find consistent may not be consistent for users at all (Grudin, 1989).

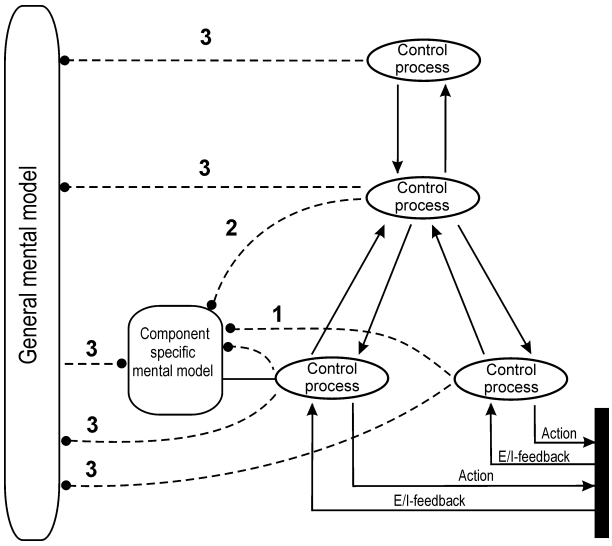

Figure 3: The proposed relations between control processes and mental models. Numbers one to three represent the three consistency relations that were studied.

Although users may establish the relation between the components, designers set the stage for possible confusion since they design feedback. Consistency is related to the I-feedback a component provides, and especially to the E-feedback that guides users in their action selection. When this E-feedback fits into the users' mental model, users can derive the consequence of an action from this mental model (Figure 3, the component-specific mental model). The E-feedback is also responsible for the users' activation of a mental model. However, if something else besides the component's E-feedback were to determine what mental model users apply, the usability of a component would be partially outside the control of its designer, which would undermine the component's autonomy. Furthermore, a mental model might be applied to control other interaction components than the one that sent the E-feedback. This would mean that users apply a mental model to control a series of interaction components in a device (Figure 3, the general mental model). This would favour the use of integral metaphors instead of a combination of multiple separate metaphors (composite metaphor) when designing an interactive system (Smilowitz, 1995). The metaphor should also properly fit the application domain since the application domain may also be partly responsible for the mental model users apply (Smilowitz, 1995).

Several studies have shown that consistency can affect the overall usability of a device (e.g. Payne & Green, 1989; Polson, 1988). However, little has been said about whether consistency can cause components to affect each other's usability. This study looks at three situations (Figure 3) where this may occur: between components in the same layer (Figure 3, relation 1); between components in different layers (Figure 3, relation 2); and between a component and an application domain (Figure 3, relation 3). All situations concern users' misinterpretation of the E-feedback because of the mental model they apply. The reason why users apply a particular mental model may depend on factors outside the component, such as feedback from other components or the application domain.

Before describing the three experiments that studied these three situations, the general experimental set-up of the experiments is presented. After the presentation of the individual experiments, a general discussion of the findings is provided.

GENERAL EXPERIMENTAL SET-UP

All three experiments were conducted simultaneously under the control of one PC application written in Delphi 5. All 48 subjects, students of Technische Universiteit Eindhoven, participated in all three experiments and received NLG 15 (roughly € 7) for their effort. In the experiment, the subjects performed tasks with three different devices in three sessions. In each session the subject performed a different task with each device. In between the sessions, the subject performed a filler task. They heard a fairy tale and were asked to count the occurrences of two words in the tale. The experimental design was counterbalanced for possible two-way interaction effects between the experimental conditions of the three experiments and the order in which they were asked to use the devices.

Throughout the task performance, the message exchange between the interaction components of the devices was recorded. This made it possible to count the number of user messages a component received directly or indirectly via lower-level layers. This number can be a powerful objective component-specific performance measure, as it represents the amount of effort a user has made to control a component (Brinkman, Haakma, & Bouwhuis, 2001). Each user message indicates a cycle of the perception-control loop. When subjects are asked to continue working as quickly as possible with the system until they fulfil their task, the number of messages received by a component relates directly to the users' ability to control a component efficiently. The advantage of using this measure is its potential statistical power compared to overall measures (Brinkman, 2003). This means a high likelihood of

detecting a difference between two versions of an interaction component by parametric statistical tests (e.g. *t*-test, or *F*-test), if of course there is a difference. The power of the measure can be higher because focusing only on a particular component reduces the variance between samples as it leaves out all the variance caused by problems users may or may not have controlling other interaction components.

INCONSISTENCY WITHIN THE SAME LAYER

The experiment to study the effect of inconsistency between interaction components within the same layer was conducted with four prototypes of a room thermostat.

Room Thermostat

The room thermostat had two very similar interaction components —daytime and nighttime temperature— which users presumably expected to be more or less similar things and therefore could be operated in a similar manner.

Figure 4 shows a part of the compositional structure of the room thermostat. To control one of the two temperature interaction components, subjects first pressed the Day or Night button (to the right of each display). This message was sent to the Router interaction component and resulted in the selection of one of the temperature components, which was made visible by turning on a light. After this, subjects could press the Left or Right button, which again sent a message to the Router interaction component. The Router component passed this message on to the selected temperature component, which adjusted its state accordingly.

Two similar versions of both components were designed, which resulted in four prototypes. In one version the temperature had a display with a moving pointer and a fixed scale (Figure 4, left image, upper display), in the other version the display had a fixed marker and a moving scale (Figure 4, left image, lower display). The Left and the Right button had an opposite effect in the two versions.

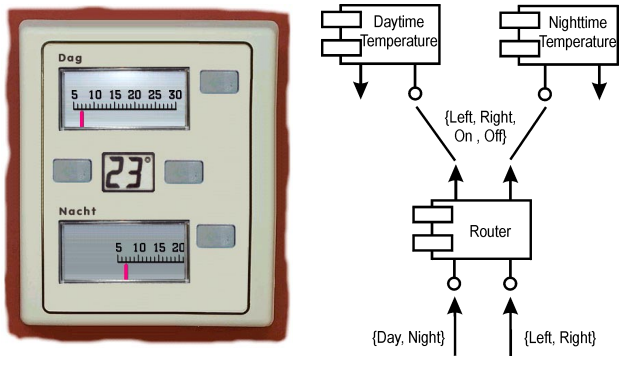

Figure 4: (left) front of an inconsistent room thermostat, and (right) part of the compositional structure of the prototypes.

The tasks subjects were asked to perform was to set both the daytime and the nighttime temperature to a specific value. All tasks required the same number of actions when performed in an optimal manner.

Results

An ANOVA was conducted on the number of messages received by the Nighttime interaction component. The analysis took the versions of the Daytime Temperature interaction component (2) and the version of the Nighttime Temperature interaction component (2) as between-subjects variables. The results reveal a significant main effect ($F(1,44) = 9.22$; $p. = 0.004$) for the version of the Nighttime component. More messages were received from the Router when the Nighttime component was implemented with the moving scale version. In addition, the analysis found a significant two-way interaction effect ($F(1,44) = 7.06$; $p. = 0.011$) between the Daytime and Nightime versions. More messages were received in the prototype that had the moving pointer version for the daytime temperature and the moving scale for the nighttime temperature (Figure 5) than in the other three prototypes. The explanation for this interaction effect is that when subjects started with setting the daytime temperature, implemented with a moving pointer, they activated a more familiar mental model than that associated with the moving scale version. The Nighttime Temperature component was interpreted in the light of this powerful mental model, which did not fit with a moving scale implementation and made subjects click on the wrong buttons.

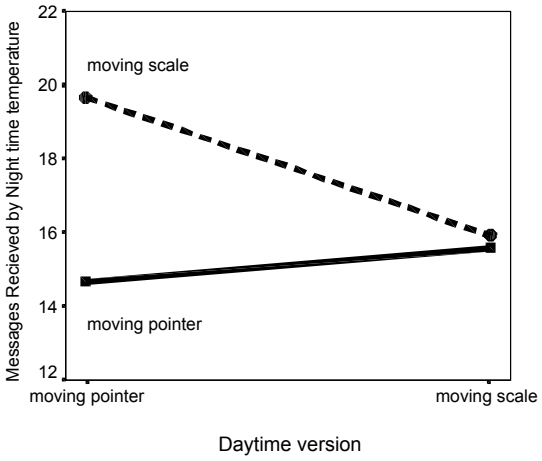

Figure 5: Number of messages received by the two versions of the Nighttime Temperature interaction component. At least 3 × (3 Left/Right messages + 1 On message + possible 1 Off message) were required to perform all tasks.

INCONSISTENCY BETWEEN LAYERS

The experiment to study the effect of inconsistency between interaction components in different layers was conducted with four prototypes of a web-enabled TV set. A mistake that novice Lynx users probably easily make, served as a model for a possible inconsistency problem between two layers. Lynx is a text-based web

browser that allows users to access the web in non-graphical environments without the use of a mouse. Users can select links with the Up and Down Arrow buttons on the keyboard. To activate the selected link, users have to press the Right Arrow button. With the Left Arrow button, users can return to the previous page. The possibility of an error may increase when links in the web page are placed on the same line. The supposed error occurs because of the activation of an inappropriate mental model —horizontal positioning with the Left and Right arrows.

Web-Enabled TV Set

The tasks the subjects had to perform, using a web-enabled TV set (Figure 6), was to find the web page that gave the departure times of a specific bus based on the bus stop, the bus number, the city and the province, which were all given in the instructions.

Figure 6: (left) linear-oriented remote control; (middle) plane-oriented remote control; (right upper corner) matrix layout; (right lower corner) list layout.

Figure 7 shows a part of the compositional structure of the web-enabled TV set. All button clicks were received by the Router, which passed it on to the Television or the Browser interaction component, depending on which one was selected at that moment. The function of the Browser was to display a web page and make it possible to select a link in the web page. When the subjects activated a link, or requested the home or previous page, the Browser sent this request to the Web Pages interaction component. The Web Pages interaction component operated as a web server as it retrieved the required web page and passed it back to the Bowser to be displayed.

The experiment had a 2 (web pages) × 2 (browser) between-subjects design. Variations in the web page's layout led to two versions of the Web Pages component. One layout, the *matrix* layout, placed the web links in a web page both on the same line and one below the other. The other layout, the *list* layout, placed all links one below the other. Variations in the remote control led to two versions of the Browser component. For one remote control, the *linear*-oriented version, the Up and Down buttons were interpreted as, "select the previous link" or "select the next link in succession". The sequence went from left to right and continued on the left of the next line. The Left and Right button were interpreted as "jumping to the previous web page" and "activate the selected link". For the other remote control, the *plane*-oriented version, the Up and Down buttons were interpreted as "select the link above" and "select the link below". Consequently, the Left and Right buttons were interpreted as "select the link left" or "select the link right". The subjects could jump to the previous page with the Back button and activate the selected link with the Middle button.

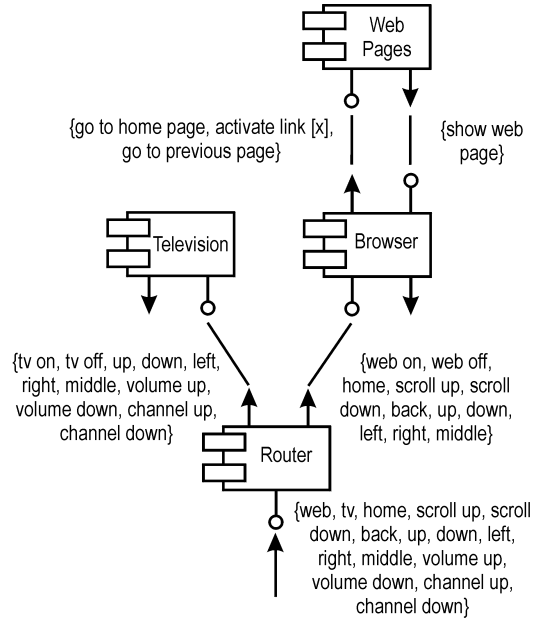

Figure 7: Part of the compositional structure of the web-enabled TV set.

Results

If subjects performed the tasks in an optimal manner, the minimal number of user messages the Browser would receive from the Router was different in the four prototypes. Therefore, instead of analysing the absolute number, the number of messages received that were needed in addition to the minimal numbers were analysed by subtracting the minimal numbers from the observed ones. The ANOVA took the versions of the Browser (linear or plane oriented) and the Web Pages (matrix or list layout) as between-subjects variables.

The results show a significant main effect ($F(1,44) = 13.78$; $p. = 0.001$) for the version of the Browser, but not a significant main effect ($F(1,44) = 3.32$; $p. = 0.075$) for the version of the Web Pages. The Browser received more additional messages when a prototype was equipped with the linear-oriented instead of the plan-oriented version of the browser (Figure 8).

The analysis also revealed a significant interaction effect ($F(1,44) = 6.79$; $p. = 0.012$) between the Browser version and the Web Pages version. The browser

received more additional messages in the prototype that combined the linear-oriented Browser version and the matrix Web Pages version than in the other prototypes (Figure 8). This demonstrates that even though the Internet architecture is developed to make web pages independent from the browsers, users might run into trouble when on a higher-level layer the web pages activate an inappropriate mental model for the interpretation of lower-level browser's E-feedback.

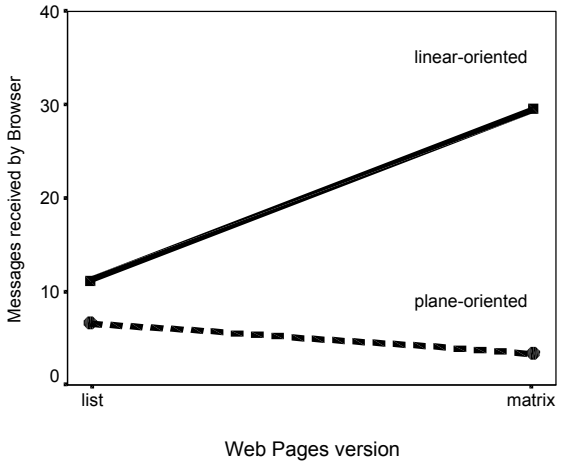

Figure 8: Number of messages received (needed in addition to the minimal number) by the two Browser interaction component versions.

INCONSISTENCY AND APPLICATION DOMAIN

Does the usability of a data browser change when it is deployed in a mobile telephone to navigate through text messages or in a digital camera to navigate through pictures? The last experiment studied the effect the application domain has on the usability of an interaction component. The application domain may activate a general mental model, which in turn may activate a component-specific mental model, which users apply to control an interaction component (Figure 3). The difference with the previous experiments is that it is not the E-feedback of other components, but the users' idea of operating a particular device that determines what component-specific mental model they apply. When a general mental model affects the usability of an interaction component, designers should apply an integrated metaphor that suits the application domain, instead of individual metaphors for each component, or metaphors that do not fit with the application domain.

Radio Alarm Clock and Microwave

The experiment took a radio alarm clock (Figure 10) and a microwave (Figure 9) as applications in which two versions of a clock were implemented. In the radio alarm clock, the clock determined when the radio should be switched on, and in the microwave, the clock determined when cooking should start. The task the subjects had to perform with the microwave was to set a timer, the cooking time and the power. For the radio alarm clock, the subjects had to set the alarm time, the radio channel and the volume. The actions required for setting the timer or the alarm time were similar when this subtask was performed in an optimal manner.

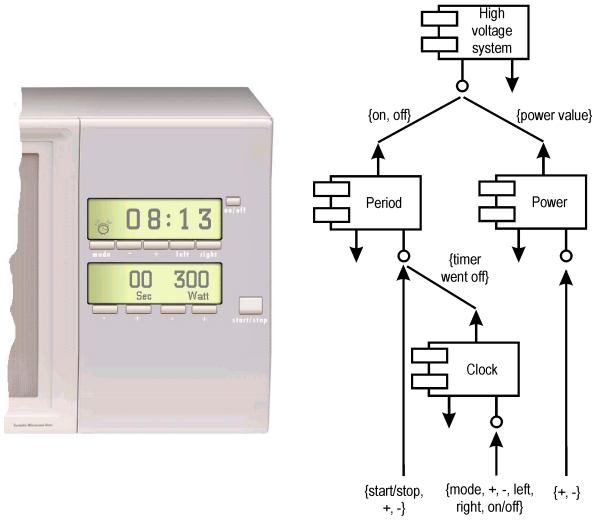

Figure 9: (left) the front of the microwave, and (right) part of the compositional structure of the device.

Both compositional structures of the radio alarm clock (Figure 11) and the microwave (Figure 9) included the interaction component Clock. This interaction component was responsible for the current time and the timer time. To see and to set these times, the subjects had to press the Mode button to put the clock in the required mode. After this, the subjects could press the + and - button to increase or decrease a digit and the Left and Right button to select another digit. The subjects activated the timer with the On/Off button. When the timer went off, the message <time went off> was sent to the Period interaction component in the case of the microwave or to the Radio Receiver interaction component in the case of the radio alarm clock.

Figure 10: Front of the radio alarm clock.

The fit or misfit between the application domain and the clock was in the clock's E-feedback that was presented along with the timer time (Figure 12). In one version, the mechanical alarm version, the symbol of a ringing mechanical alarm clock was shown, in the other version, the hot dish version, a symbol of a hot dish. The clock had four different modes: displaying the current time, displaying the timer time, setting the current time, and setting the timer time. The current time was presented along with a symbol of a clock (Figure 12, right

symbol). The timer time was presented along with the ringing mechanical alarm clock or the hot dish.

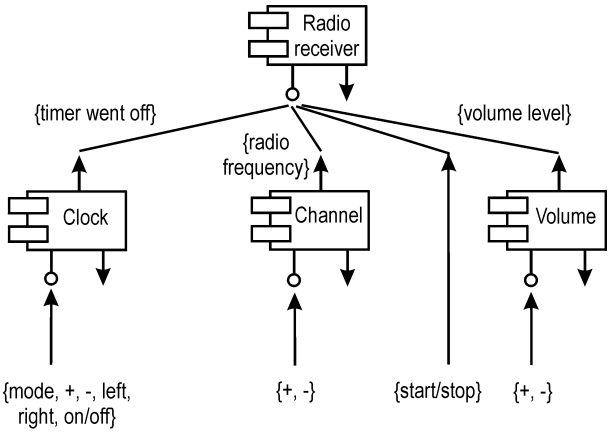

Figure 11: Part of the compositional structure of the device.

When the subjects performed a task with the radio clock, the expectation was that the task of setting the alarm of an alarm clock would activate a general mental model on alarm clocks, which subsequently activates a component-specific mental model of setting the alarm of alarm clocks. In light of this activated component-specific mental model, subjects could more easily understand the E-feedback "the time the timer will go off" represented by the mechanical alarm clock than by a hot dish. The opposite was expected for the microwave, where the E-feedback indicates "the time cooking begins" which is probably better represented by the hot dish than by a mechanical alarm clock.

Figure 12: (left) ringing mechanical alarm clock, (middle) a hot dish, and (right) normal clock symbol.

Results

An ANOVA was conducted on the number of mode change requests received by the clock. The analysis took the Clock version (2) and the Application domain (2) as between-subjects variables. The analysis did not find a significant interaction effect ($F(1,44) = 0.02$; p. = 0.887) between the two independent variables. Besides the straightforward interpretation that there is no general mental model that indirectly influences the interaction with a specific component, another interpretation is an unanticipated effect of the experimental set-up. Although the subjects may not have understood the inconsistent symbol presented with the timer, other E-feedback of the application did not suggest looking elsewhere. Consequently, the subjects were just left with the only conclusion that this inconsistent symbol had something to do with the timer time.

The ANOVA did however reveal a significant main effect ($F(1,44) = 7.57$; p. = 0.009) for the application domain. Subjects less often changed the clock mode when they operated the radio alarm than when they operated the microwave (Figure 13). The same clock function was apparently easier to use in one application domain than in the other, which suggests that the usability of a component depends on the application domain. Two explanations can be given for this finding. First, the subjects could understand the concept of a timer better in relation to the alarm clock since it is more related to the main function of the alarm clock, which is waking someone on time, than to the main function of the microwave, which is preparing a meal. Second, the other components of the microwave might have had a negative impact on the usability of the clock. When operating the microwave, the subjects may have had a problem distinguishing the Clock and the Period interaction component (Figure 9), which also deals with time —the cooking time.

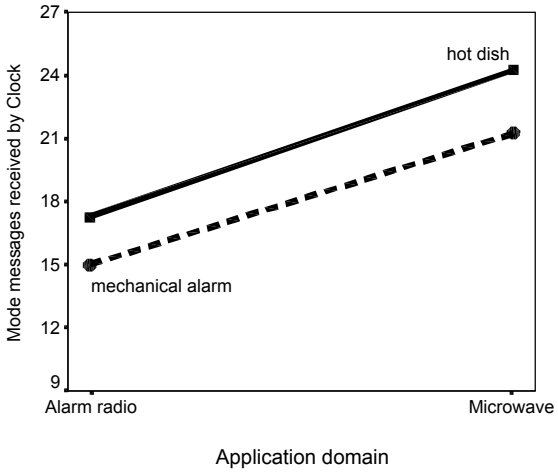

Figure 13: Number of Mode messages the two versions of the Clock interaction component received. At least 9 were required to perform all tasks.

DISCUSSION

The findings seem to confirm the main tenet of this paper —interaction components in an interactive system can affect each other's usability because of inconsistencies. Interaction components in the same layer or in other layers can activate an inappropriate component-specific mental model, which users apply to understand the interaction component's feedback to select actions. The inconsistency between the application domain and the interaction component's E-feedback was not found to affect the interaction component's usability. Whether this was only the case in this experiment or can be generalised, is a topic for further research. However, the results seem to suggest that the application domain can have an effect on the users' understanding of the functionality a component provides.

Theoretical implications

The dependency seen between components does not confirm LPT's ideas. LPT suggests that the user-system

interaction within a layer can operate rather autonomously of the other layers, to the degree that they only exchange messages. However, LPT limits itself if it suggests that control processes are solely based on the present interaction within one layer. Users rely on their knowledge gained from previous interaction. Although LPT may be right in stating that a control process aims at controlling a specific layer, it should not be concluded that it only relies on the interaction component's feedback to select actions.

These findings may be limited to the phase where users learn to control a component, as was the case in all three experiments. Once users gain experience with controlling the components it might be that the dependency between them lessens because the correct components-specific model will be activated. Users might be more guided by the E-feedback initially, and later on more by their own experience. However, this falls outside the scope of this study.

Practical implications

The findings demonstrate that designers should not assume that selecting components that may be very easy to use in other applications would automatically result in a very easy to use new application. When designers are creating a new component they should try to predict what other components will be used in relation with their component. If this is not possible, another strategy might be to design the component according to a set of specific rules. Later on, when the component is used to build an application, developers should make sure that the components they apply follow the same rules, or at least that there are no conflicting rules. These rules can be laid down in a style guide. However, this does not guarantee an application without inconsistency, because users do not have to agree with what designers consider to be consistent. Only the involvement of users can solve this problem.

Finally, when analysing the results of a usability test, evaluators should also be aware that usability problems cannot always be solely attributed to a single component, but that the combination of two or more components may have caused them.

REFERENCES

Brinkman, W.-P. (2003). *Is usability compositional?* Doctoral dissertation, Technische Universiteit Eindhoven, The Netherlands.

Brinkman, W.-P., Haakma, R., & Bouwhuis, D.G. (2001). Usability evaluation of component-based user interfaces. *INTERACT'01*, Amsterdam: IOS Press, 767-768.

Brinkman, W.-P., Haakma, R., & Bouwhuis, D.G. (2004). Usability testing of interaction components: Taking the message exchange as a measure of usability, In R.J.K. Jacob, Q. Limbourg & J. Vanderdonckt (Eds.), *Pre-Proceedings of CADUI'2004* (p. 159-170). Dordrecht, The Netherlands: Kluwer Academics.

Cox, B.J. (1990). There is a silver bullet: A software industrial revolution based on reusable and interchangeable parts will alter the software universe. *Byte, 15*, 10, 209-218.

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. *International Journal of Human-Computer Studies, 50*, 489-520.

Grudin, J. (1989). The case against user interface consistency. *Communications of the ACM, 32*, 1164-1173.

Haakma, R. (1999). Towards explaining the behaviour of novice users. *International Journal of Human-Computer Studies, 50,* 557-570.

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

Kellogg, W.A. (1989). The dimensions of consistency. In J. Nielsen (Ed.), *Coordinating user interfaces for consistency* (p. 9-20). London: Academic Press.

Norman, D.A. (1984). Stages and levels in human-machine interaction. *International journal of Man-Machine studies, 21*, 365-375.

Payne, S.J. & Green, T.R.G. (1989). The structure of command languages: An experiment on task-action grammar. *International Journal of Man-Machine Studies, 30*, 213-234.

Polson, P.G. (1988). The consequences of consistent and inconsistent user interfaces. In R. Guindon (Ed.), *Cognitive science and its applications for human-computer interaction* (p. 59-108). Hillsdale, NJ: Lawrence Erlbaum.

Powers, W.T. (1973). *Behavior: the control of perception.* New York, NY: Aldine de Gruyter.

Powers, W.T. (1998). *Making sense of behaviour*. New Canaan, CT: Benchmark Publications.

Reisner, P. (1993). APT: A description of user interface inconsistency. *International Journal of Man-Machine Studies, 39*, 215-236.

Smilowitz, E.D. (1995). *Metaphors in user interface design: An empirical investigation*. Unpublished doctoral dissertation, New Mexico State University, Las Cruces, NM.

Taylor, M.M. (1988). Layered protocol for computer-human dialogue. I: Principles. *International Journal of Man-Machine Studies, 28*, 175-218.