Interacting with Computers 22 (2010) 449–464

Contents lists available at ScienceDirect

Interacting with Computers

journal homepage: www.elsevier.com/locate/intcom

Towards analytical evaluation of human machine interfaces developed in the context of smart homes

Belkacem Chikhaoui [*], Hélène Pigot

DOMUS Laboratory, Computer Science Department, Faculty of Science, University of Sherbrooke, Sherbrooke, QC, Canada J1K 2R1

ARTICLE INFO

Article history:
Available online 6 September 2010

Keywords:
HMI evaluation
Human-computer interaction
Cognitive modeling
Smart homes
User modeling
User performance

ABSTRACT

Designing human machine interfaces that respect the ergonomic norms and following rigorous approaches constitutes a major concern for computer systems designers. The increased need on easily accessible and usable interfaces leads researchers in this domain to create methods and models that make it possible to evaluate these interfaces in terms of utility and usability. Two different approaches are currently used to evaluate human machine interfaces, empirical approaches that require user involvement in the interface development process, and analytical approaches that do not associate the user during the interface development process. This paper presents a study of user performance on two principal tasks of the contextual assistant's interface, developed in the context of smart homes, to assist persons with cognitive disabilities. We use three different methods to analyze and evaluate this interface, focusing basically on time of execution. Two of the models developed are based on cognitive models, which are ACT-R and GOMS and the third one is based on the Fitts' Law model. The results show that, all models give a good prediction of user performance, even if the cognitive models show better accuracy of the user performance. Furthermore, they provide a better insight into cognitive abilities required to interact with the interface.

1. Introduction

Smart environments constitute a technological challenge of contemporary research. According to Mark Weiser, a smart environment is "a physical world that is richly and invisibly interwoven with sensors, actuators, displays and computational elements, embedded seamlessly in the everyday objects of our lives, and connected through a continuous network" (Weiser, 1999). There is an increasing interest in developing smart environments and particularly smart homes, which can greatly enhance our life quality thanks to their services in taking care of our beloved, and aiding us when an unwelcome event or mistaken behavior occurs (Xinguo et al., 2007). These services could be among others, object localization, home cleaning, people activity monitoring and assistance.

Services included in smart homes must be easily accessible by the individual in his environment. Well-designed services, human machine interfaces and adapted appliances should help people in smart homes to increase their well being. In order to provide an intuitive and comfortable environment, attention should be brought to ensure that people should easily access and manage the information in the environment. This can be reached by providing the user an interface that is accessible, usable and efficient (Joon et al., 2007). Each human machine interface must be clear enough, to reduce the cognitive effort and allowing a good interaction with the environment. Therefore, the evaluation of human machine interfaces appears as a significant contribution in the design of applications and systems dedicated for smart homes.

Two main approaches are currently used to evaluate human machine interfaces: empirical approaches (test methods) and analytical approaches (inspection methods). Empirical approaches are essentially based on performances or opinions of users gathered in laboratories, smart homes or other experimental situations. The empirical approaches are user-focused approaches. The users are involved in the development process of the interface, especially in the specification and usability testing phase (Abras et al., 2004). The results of experiments are observed and then analyzed in order to improve and redesign the developed interface. However, evaluating interfaces in smart homes with users brings various challenges due to the variability of home settings, the experiment time which lasts for months and the software and material infrastructure to set up in the smart home (Koskela and Väänänen-Vainio-Mattila, 2004). These drawbacks should be avoided using analytical approaches. Unlike the empirical approaches, analytical approaches are not based directly on the user performance, but rather based on the automated examination of interfaces using well-defined structures and rigorous analysis techniques (Yen et al., 2005). In analytical evaluation, the users are not involved in the evaluation process. The evaluation is conducted by one or more analysts with the help of theoretical

[*] Corresponding author. Tel.: +1 819 821 8000x66124; fax: +1 819 821 8200.
E-mail address: belkacem.chikhaoui@usherbrooke.ca (B. Chikhaoui).

0953-5438/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.intcom.2010.08.003

methods such as heuristic evaluation (Nielsen and Molich, 1990; Nielsen and Phillips, 1993; Nielsen, 1993), cognitive walkthrough (Nielsen, 1995), cognitive architectures (Anderson et al., 2004; Byrne, 2001; Kieras and Meyer, 1997) and predictive methods (Bligard and Osvalder, 2007; John and Kieras, 1994; Kieras, 2001). Analytical approaches provide a theoretical background, predict and estimate the time of task realization, give quantitative and qualitative measurements, introduce behavioral errors during the task realization, compare and interpret different versions of systems and explain empirical or analytical results. According to John and Salvucci (2005), performance of users can be obtained analytically by using cognitive predictive models of user behavior.

Analytical approaches focus on different aspects of human machine interaction. Some of them emphasize the task components such as the GOMS models, in which the task is deeply analyzed, described and subdivided into subtasks to reduce its complexity (Card et al., 1983). Other approaches emphasize the physical component of the task such as the Fitts' Law, by predicting the time required to move to a target area (Fitts, 1954). Other analytical approaches focus on the cognitive component during the task realization, such as the cognitive architecture ACT-R by simulating the cognitive processes involved when the user interacts with the interface, namely the user's understanding, knowledge, memory and planning (Dix et al., 2004; Chikhaoui and Pigot, 2008a). Some others are basically focused on the perceptual and motor components of the interaction, such as the cognitive architecture EPIC (Executive-Process/Interactive-Control), which simulates how human perceives and responds to various stimuli (Kieras and Meyer, 1997).

The concept of human–computer interaction, is traditionally defined as the interaction of human with a large screen placed on a desk. However, quick development of computer systems and emergence of new technologies, leads scientists and researchers to consider the interaction with new devices. Mobile devices such as cell phones bring new issues on interacting with small devices (Salvucci and Macuga, 2002). Touch screens, which are widely used in smart homes given their accessibility and adaptation, provide new interaction issue with different kinds of applications and users (Lussier-Desrochers et al., 2007). More recently, the pervasive computing paradigm expands the human machine interaction with all the environment (Minoh and Yamazaki, 2006), which changes according to the user's behavior. This makes the evaluation process more complex, expensive and time consuming.

The DOMUS laboratory at the University of Sherbrooke, develops cognitive assistance for people with cognitive impairments. The smart home of the DOMUS laboratory aims to foster autonomy by providing cognitive assistance to complete activities of daily living (Pigot et al., 2008). The human machine interaction in this pervasive environment is not restricted to the interaction with the touch screen, but rather extended to the high speakers disseminated in each room, and lights placed all around to identify specific locations. For instance, when the user asks for an object location through an interface displayed on the touch screen, the smart home answers by blinking the locker where the object is stored, to draw the user's attention on the specific place of the object. If the user forgets a step while cooking, the smart home system recalls the user through the speaker closest to him (Pigot et al., 2008). Given these new developments, it is essential to conceive theoretical analysis for evaluating the human machine interfaces in pervasive computing environments, and predicting the user behaviors and time to perform the tasks. In the present study, we present a new orientation in the evaluation of human machine interfaces based on analytical approaches.

Three analytical methods have been selected according to the various theoretical models they are based on. An experimental evaluation is conducted to validate these analytical methods. The interface to be evaluated is part of the contextual assistant "ARCHIPEL", developed at the DOMUS laboratory to provide cognitive assistance for people with intellectual retardation while performing cooking activities (Pigot et al., 2008). The cooking activities require complex cognitive abilities, such as planning, initiation and memory.

The first model used for evaluating the contextual assistant's interface is based on a cognitive model using the cognitive architecture ACT-R. ACT-R is a production system based architecture where knowledge is represented as facts and rules (Anderson et al., 2004, 2005; Byrne, 2001). ACT-R offers an environment for implementing and simulating human cognitive tasks, and providing a way to reproduce with high fidelity human behavior. In fact, the perceptual motor component of ACT-R is inspired from the cognitive architecture EPIC, which is mainly aimed at providing a detailed study of human perceptual operations (Kieras and Meyer, 1997). However, EPIC does not focus on the cognitive components, and there is no mechanism for problem solving and learning integrated in EPIC. These drawbacks motivate our choice for the ACT-R cognitive architecture that overcomes all these issues. Moreover, ACT-R offers to model the cognitive errors that will happen during the human–computer interaction. The second model used in our evaluation is based on a predictive model, the GOMS model. GOMS (Goals, Operators, Methods and Selection rules) is a formalized representation used to predict task performance (Kieras, 2003). In the GOMS model, the user achieves goals by solving subgoals in a divide and conquer fashion (Dix et al., 2004). The GOMS model includes perceptual, cognitive, and motor components involved in task realization, and it is widely used in the evaluation of human–computer interaction (Amant et al., 2007). The third model selected is the Fitts' Law model (Fitts, 1954), which is based on a mathematical model. In Fitts' Law, the human interaction with a screen is modeled as a target reaching task (MacKenzie, 1995; MacKenzie et al., 1991; Po et al., 2004). The Fitts' Law is used in our evaluation given its capability to evaluate the physical (motor) component of the human–computer interaction. The performance of these three models will be compared to an experimental study where subjects perform the task.

This paper is organized as follows. Section 2 presents the theoretical background of the three analytical methods selected to perform our evaluation. In Section 3 we describe the task to be performed using the contextual assistant. Section 4 discusses the experimental study. In Section 5 we present the three models developed, and the results obtained are presented in Section 6, followed by a discussion in Section 7. Finally, Section 8 presents the conclusion and perspectives of our paper.

2. Analytical methods

In this section we present an overview of the cognitive architecture ACT-R, the GOMS model and the Fitts' Law.

2.1. Cognitive architecture ACT-R

The cognitive architecture ACT-R is developed to simulate and understand human cognition (Anderson et al., 2004, 2005). It consists of multiple modules (declarative, goal, perceptual-motor) integrated through a central production system in order to produce coherent cognition (Anderson et al., 2004). ACT-R is a hybrid architecture that combines two subsystems: symbolic system containing knowledge, and subsymbolic system evaluating knowledge activations. The symbolic system includes the declarative memory, which contains the semantic unities of information called chunks, and the procedural memory containing the production rules (Newell, 1990). The subsymbolic system assigns activations to chunks (semantic knowledge) and rules (procedural knowledge), which helps to choose the more predominant knowledge available at a specific time. Each module is associated with one or more buffers. The different modules are responsible to place chunks in buffers in order to be detected by the production system that responds to the information contained in buffers (Anderson et al., 2004).

In ACT-R, the perceptual and motor modules are used to simulate interfaces between the cognitive modules and the external world. The perceptual modules allow the model to attend to visual and aural stimuli, while the motor modules are responsible for preparing and executing basic motor actions such as key presses and mouse movements (Bothell, 2004; Byrne, 2001).

The visual module that is part of the perceptual modules, is decomposed on two subsystems, the positional system (where) and the identification system (what). These two subsystems work together in order to send the specified chunk to the visual buffer that the cognitive modules use. The positional system is used to find objects. When a new object is detected, the chunk representing the location of that object is placed in the visual-location buffer according to some constraints provided by the production rule. The identification system is then used to attend to locations which have been found by the positional system. The chunk representing a visual location, will request the identification system to shift visual attention to that location. The result of an attention operation is a chunk, which will be placed in the visual buffer and should be placed then in the declarative memory (Bothell, 2004; Byrne, 2001).

The motor module contains only one buffer through which it accepts requests. Two actions are available in ACT-R, to click with the mouse or press a key on a virtual keyboard.

2.2. GOMS model

GOMS (Bovair et al., 1990; Dix et al., 2004; Heim, 2007; John and Kieras, 1996b; Kieras and Polson, 1999; Kieras, 2003) stands for Goals, Operators, Methods and Selection rules. This model is based on the the knowledge that a user must have in order to carry out tasks on a device or system. It is a representation of the "how to do it" knowledge that is required by a system in order to get the intended tasks accomplished (Kieras, 1999). In the GOMS model, the task is described using these four components:

- *Goals*: the user's goals describing what the user wants to achieve. Indeed, the tasks are presented as a set of goals and subgoals. To accomplish the top level goal, the user must segment higher level goals into a set of discrete subgoals. This decomposition allows the user to reduce the complexity of tasks firstly and enables the user to evaluate his or her progress secondly.
- *Operators*: tasks can only be carried out by undertaking specific actions. The operators correspond to basic actions that the user must perform in a lowest level of analysis, in order to use the system. For instance, printing a document requires issuing a print command, which may involve clicking the printer icon (an operator). This action represents the lowest level in the model.
- *Methods*: methods are sequences of steps consisting of operators and subgoal invocations, that the user performs in order to accomplish a goal. For instance, printing a document can be achieved in different ways. The user can press the printer icon or use a key combination such as (alt+p) in (Microsoft Windows) operating system. Each of this scenarios represents a specific method of achieving the goal.
- *Selection rules*: the method that the user chooses is determined by selections rules depending on the context when choice of methods arises. For instance, if the user is involved in intensive typing, he or she may prefer to use the key combination (alt+p) method of printing. However, in the tasks that are heavily mouse oriented, he or she may choose to click the printer icon with the mouse.

Thanks to these four components, the tasks are analyzed resulting to a hierarchical description of these tasks. The time is predicted by summing the operator's time required to complete the task.

2.3. Fitts' Law

Fitts' Law is a model of human movement that predicts the time required to rapidly move to a target area. Fitts' Law has been used extensively to model user performance in various interaction tasks including pointing and dragging, and in comparisons of various input devices (Card et al., 1987; MacKenzie et al., 1991). Numerous variants and extensions to Fitts' Law have been used to better fit experimental data.

In human machine interfaces, the formulation of Fitts' Law (Fitts, 1954) states that the movement time (*MT*) is function of target amplitude (*A*) and target width (*W*), according to the following equation:

$$MT = a + b\log_2\left(\frac{A}{W} + 1\right) \qquad (1)$$

The above formulation of Fitts' Law was originally proposed by MacKenzie (1989) as an improvement to the original formulation by Fitts (1954). It is the standard formulation used to compare interaction devices and techniques in HCI, and is the one used throughout this article. The constants a and b are derived empirically. They can be interpreted respectively by the intercept and the slope of a predictive linear regression equation (MacKenzie, 1995; MacKenzie et al., 1991).

The term $\log_2(\frac{A}{W} + 1)$ of the Eq. (1) is known as the index of difficulty ID. It describes the difficulty of the motor tasks. The term $\frac{1}{b}$ is called the index of performance (IP), and measures the information capacity of the human motor system.

3. Task description

The contextual assistant is an application developed to assist persons with cognitive disabilities (Lussier-Desrochers et al., 2007; Pigot et al., 2007). The aim is to foster autonomy in the daily living tasks, and particularly during complex cooking tasks, such as preparing meals (Pigot et al., 2008). Cooking tasks are chosen given the role they play in the daily life of persons on one side, and their complexity of realization on the other side. These tasks require a multitude of cognitive skills such as planning, reasoning and problem solving. Therefore, it is essential to deeply analyze how the cognitive abilities are involved during the activities execution. In that experiment, we assist people to achieve the cooking activity thanks to a smart home controlled by a touch screen. The cooking task is decomposed of steps displayed each one at a time on a touch screen. The two first steps consist of gathering the utensils and ingredients necessary to the recipe (Fig. 1). The next steps explain how to cut the vegetables, cook the sauce, and prepare the pasta by boiling water and throwing the pasta in it. The last step consists of serving the plate.

The contextual assistance's interface displays a central image of the current step and a legend to resume the step. The user could ask for a video explaining the way to perform the current step, or ask for locating a specific object in the kitchen by selecting the button "LOOK-FOR-OBJECT" ("CHERCHER UN OBJET"). This action brings the user to the object locator displaying the objects to search. When an object is selected in this interface, the contextual assistant indi-

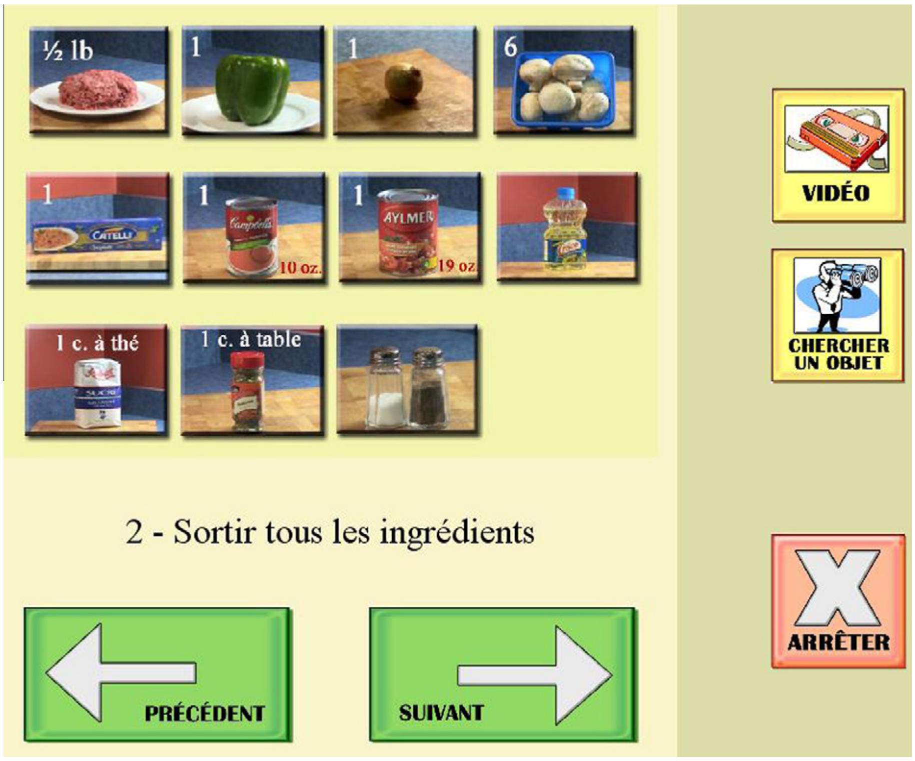

Fig. 1. Interface of the gathering ingredients step of the contextual assistant.

cates the location of that object in the environment, using pervasive computing technology. It indicates the object location by highlighting the appropriate locker containing the object as shown in Fig. 2.

The interface allows the user to progress through the steps thanks to the button "NEXT" ("SUIVANT"), and to return back to the previous steps if needed thanks to the button "PREVIOUS" ("PRÉCÉDENT") as shown in Fig. 1. The progress through the steps is conditioned by the completion of the current step, which is evaluated thanks to ubiquitous computing technics.

The contextual assistant's interface is displayed on a 1725L 17″ LCD Touch screen, with 13.3″ (338 mm) horizontal and 10.6″ (270 mm) vertical useful screen area. It is configured to 1024×768 optimal native resolution running Macintosh. The touch screen is fixed under a closet nearby the oven in order to be easily accessible and also protected against the cooking splashes.

The touch screen has been chosen because of its usability and adaptability, allowing people with cognitive deficits to easily interact with the system (Gowans et al., 2004; Lussier-Desrochers et al., 2007; Po et al., 2004; Schedlbauer et al., 2006).

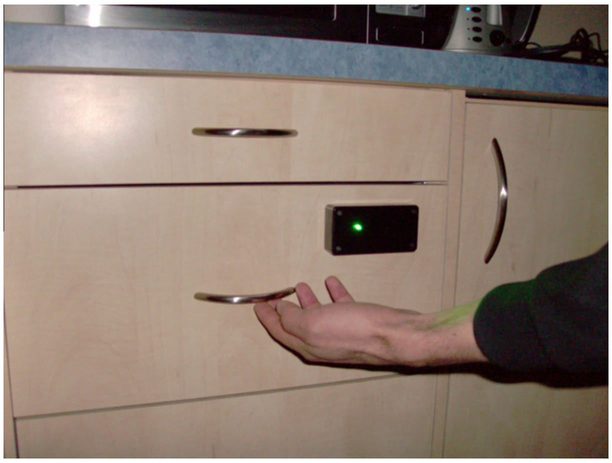

Fig. 2. Locker highlighting after an object locator request.

In this study we simulate the first two steps of preparing the spaghetti recipe. They consist of first, knowing the list of objects to gather, either utensils or ingredients, and then to use the object locator in order to find each object in the environment. During our modeling work, the interactions with the touch screen are simulated without taking into account the time taken by the subjects to pick up the objects in the environment. The two first steps require principally three subtasks (Figs. 3 and 4). The first subtask consists of activating the object locator in order to identify each object required by the recipe. This is done by pushing the button "LOOK-FOR-OBJECT" either for utensils or ingredients, which is displayed in the main interface of the contextual assistant (Fig. 1). The second subtask aims to locate each object, either utensil or ingredient needed in the current step, by pushing the button corresponding to the object in the object locator's interface. The third subtask consists of coming back to the main contextual assistant's interface in order to proceed with the next step of the recipe. The tree decomposition of the two first steps of the recipe is presented in Fig. 4. To compare the tasks tree from the interface of Fig. 1, the nodes in capital indicate the action to click on the named object, while the other nodes represent the tasks to be decomposed.

4. Experimental study

In this section we describe the procedure of the performance study conducted at the DOMUS laboratory, in terms of users, apparatus and applications used to perform this study.

4.1. Apparatus and application

The experiment consists on selecting items on the contextual assistant's interface. The items correspond to the list of utensils and ingredients needed to realize the cooking task. Each item is displayed with a large button on the contextual assistant's

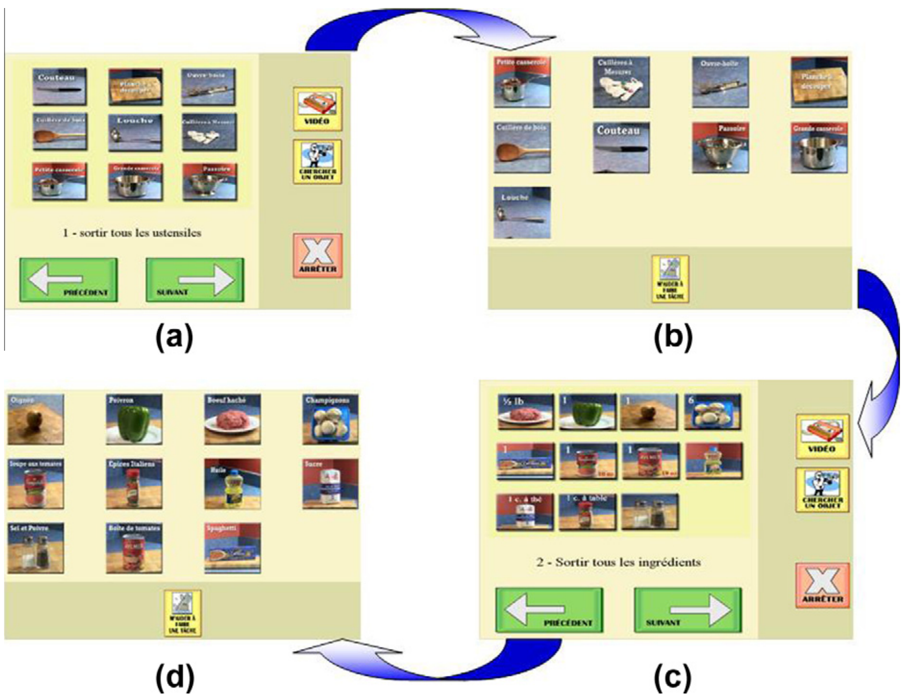

Fig. 3. Sequence of the interfaces displayed during the two tasks gathering utensils and ingredients. (a and c) Step interface. (b and d) Object locator interface.

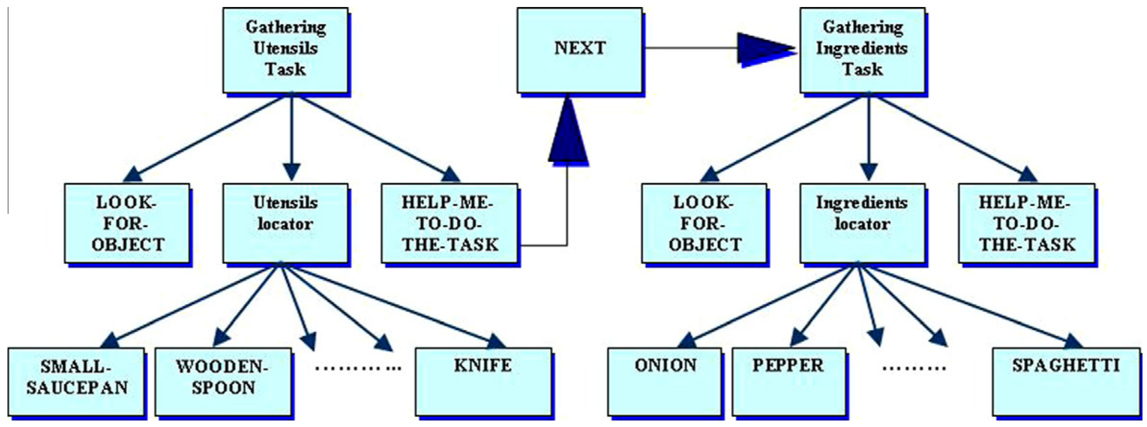

Fig. 4. Hierarchical decomposition of the gathering utensils and ingredients tasks.

interface (Fig. 1). The experiment is conducted according to three main criteria in order to be uniform for all participants.

1. Measure accurately the time of selecting items on the interface.
2. The experiment is uniform for all participants (all participants execute the same tasks).
3. The order in which the objects are displayed on the PDA (Personal Digital Assistant) does not affect the speed of selection.

To ensure the first criterion, we decide to identify the time of recognition of the object, and the time of selection (pushing) the object on the contextual assistant's interface. In order to measure the specific time of the interaction with the contextual assistant's interface, we measure the time that users take to decide which object they want to get out, and the time to select that object on the interface of the contextual assistant.

The first action corresponding to "decide which object users want to get out", can be presented experimentally using a PDA. To do so, we developed an application that displays to users, the name of the object to get out. This highlights the recognition object phase involved in the cognitive processes. It is applied for each object needed in the two tasks, gathering utensils and gathering ingredients. The specified application is set up on a PDA and executed in parallel with the contextual assistant application. The time taken to push each object on the PDA is recorded in a log file, this latter is recovered at the end of the experiment.

To ensure the second criterion, all the participants perform the same experiment with the same task and material. The objects are presented on the PDA in the same order for each subject. The third criterion necessitates to avoid an automatic selection of the objects on the contextual assistant's interface displayed on the touch screen without taking the time for recognizing them. It could happen if the objects are asked sequentially line by line. The order of objects presentation on the PDA has been chosen randomly, but remains the same for all the subjects to fulfil the second criterion.

4.2. Subjects

Ten students of the Sherbrooke University participate in the study. All subjects are male and their ages range from 27 to 32

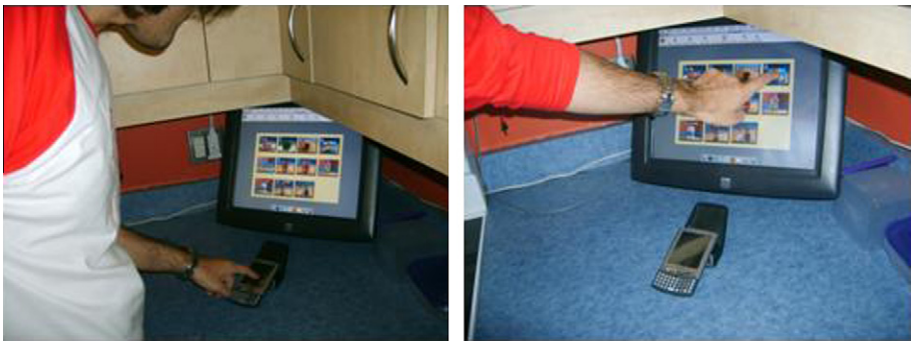

Fig. 5. Example of a participant during the experiment.

years. The subjects have corrected vision with no other physical impairments being reported. All subjects have a good knowledge of computer science, but they have no prior knowledge on the application and the cognitive assistance field.

4.3. Method

The PDA is placed at a distance of 15 cm from the touch screen, participants remain standing at a distance of approximately 30 cm from the touch screen during the entire test as shown in Fig. 5. The participants familiarize themselves with the interface during a practical stage before the experiment.

The participants look first on the PDA to know the name of the object to get out and second, select the corresponding button of that object on the contextual assistant's interface, by touching it using their index finger. Once the button is pushed, participants push the button representing the name of the object on the PDA that indicates then the next object to reach, and so on until the last object of the gathering ingredients task is reached.

During the entire test, objects displayed on the PDA were presented to participants in a randomized order. This emphasizes the recognizing objects phase in the cognitive processes, and avoid the automatic selection of objects, which can arise when the order in which objects are displayed respectively on the PDA and the touch screen is still the same.

Each participant accomplished five trials, and each trial in the study required achieving two tasks: gathering utensils and gathering ingredients. Each trial needs 25 actions "pushing button on the PDA" and 25 actions "pushing button on the contextual assistant's interface". Altogether 2500 (10 participants × 5 trials × (25 actions × 2 interfaces (PDA and contextual assistant)) = 2500) actions are observed during the experiment.

In our study, the action of getting out the objects from their locations in the environment is not modeled.

Table 1 shows the mean duration with the standard deviation for each object in the two tasks, over all participants in our study.

Table 1
User performance data across objects with mean and standard deviation.

Objects	Duration (s)	Standard deviation
LOOK-FOR-OBJECT (utensils)	5.299	1.052
CAN-OPENER	2.291	0.717
COLANDER	2.966	0.786
MEASURING-SPOON	2.167	0.605
LADLE	2.847	0.829
SMALL-SAUCEPAN	1.980	0.371
WOODEN-SPOON	2.590	0.536
KNIFE	2.328	0.430
BIG-SAUCEPAN	1.779	0.308
CUTTING-BOARD	2.000	0.309
HELP-ME-TO-DO-THE-TASK (utensils)	2.039	0.386
NEXT	2.142	0.540
LOOK-FOR-OBJECT (ingredients)	1.955	0.265
PEPPER	2.448	0.825
SPAGHETTI	1.939	0.552
TOMATOES-BOX	1.794	0.377
GROUND-BEEF	2.491	0.591
ONION	2.021	0.484
TOMATO-SOUP	1.970	0.422
SALT-AND-PEPPER	2.490	0.481
OIL	1.965	0.348
MUSHROOMS	1.809	0.369
SUGAR	1.774	0.341
ITALIAN-SPICE	1.736	0.436
HELP-ME-TO-DO-THE-TASK (ingredients)	2.432	0.614

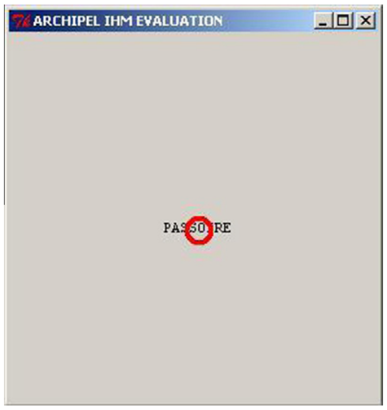

Fig. 6. Shift attention phase in the ACT-R model.

5. Modeling the interaction with the contextual assistant

In this section, we present the modeling process of the tasks involved in our study, which are gathering utensils and gathering ingredients, emphasizing firstly, the task analysis and secondly the perceptual, cognitive and motor components involved for each model and how they can be applied in our context.

5.1. Modeling the interaction with the contextual assistant using ACT-R

The model developed aims to simulate the human machine interaction with the contextual assistant. The model uses ACT-R to emphasize the cognitive processes involved, when looking for an object and choosing the button to push. The ACT-R model is

```
(P start-application
   =goal>
          ISA      begin    ; Initializing the model
 ==>
   +visual-location>
       ;Making request on the visual-location buffer
          ISA          visual-location
          :attended      nil
   +goal>
          ISA            get-object
          state          find-location )
(P attend-utensil
   =goal>
          ISA      get-object
          state    find-location
         ; Move attention to the location
         ; screen-x 122 and screen-y 250
   =visual-location>
         ISA          visual-location
         screen-x       122
         screen-y       250
   ?visual>
         state  free
 ==>
   +visual>
        ISA      move-attention
        screen-pos    =visual-location
   =goal>
        state       attend    )
```

Fig. 7. Example of some ACT-R productions responsible for the shift attention and visual encoding phases.

constructed mainly upon three phases: the visual phase, the recognition phase and the motor phase.

The visual phase itself consists of two steps: localizing the object to perceive and then identifying it. We consider that all buttons displayed on the screen are objects, either the buttons used to locate a utensil or ingredient, or the buttons to navigate in the interface. The first object corresponds to the button "LOOK-FOR-OBJECT", this object appears with the same name in both tasks as shown in Fig. 4. Thus, to avoid this confusion in this paper, a more significant name is attributed to this object according to the task to which it belongs. The first name is "LOOK-FOR-OBJECT (utensils)" for the first task, and the second one is "LOOK-FOR-OBJECT (ingredients)" for the second task. The same procedure is done with the object "HELP-ME-TO-DO-THE-TASK" as shown in Table 1. Then, all the utensils (or ingredients) needed in the recipe are presented in the visual interface of ACT-R. Finally, to complete the current step of the recipe, the button "HELP-ME-TO-DO-THE-TASK" is presented, in order to come back to the main contextual assistant's interface and pursue the next step of the recipe.

Each object of the interface is displayed at defined coordinates (x,y) on the screen. These coordinates specify the request made to the visual-location buffer of ACT-R, which creates a chunk representing the location of the specified object. After that, the identification system identifies the name of that object and creates a chunk which will be placed in the visual buffer. The location and identification phases last 185 ms (Bothell, 2004; Byrne, 2001). The objects are presented to the visual module of ACT-R by the mean of a list of all the objects (buttons of the interface) to be pushed on.

The recognition phase begins when the chunk of the object is placed in the visual buffer. This phase implies to recover that specific chunk from the declarative memory. The result of this phase is a chunk that represents the object with some characteristics as color, localization on the screen, name, and kind of object.

The motor phase consists of activating the motor actions via a request to the motor buffer, in order to click on the object. The three phases are applied for each object displayed in the interface for the two steps of the recipe. The gathering utensils and ingredients model finishes when the last object of the gathering ingredient task is reached.

In our ACT-R model, the contextual assistant's interface is simulated using a virtual display based on a vertical list in the *Lisp* environment. The virtual display maintains a representation of each object used in the interface at a given time, by displaying its name surrounded by a red circle, which reflects the shift attention to that object, as shown in Fig. 6.

Fig. 7 shows some ACT-R productions responsible for the shift attention and visual encoding phases.

The production (p start-application) is responsible to initialize the ACT-R model by making the goal state to "begin", makes the request on the "visual-location" buffer of the visual module, and finally changes the goal state to "find-location". The "visual-location" request asks the vision module to find a location of an object in its visual scene (which is the simulated interface in our model) that respects the specified requirements. Once the requirements are satisfied, a chunk that represents the location of that object is created and placed in the "visual-location" buffer. In the second production (p attend-utensil), the goal state is changed to "find-location", and the chunk representing the location of the object is recovered. A request is made on the "visual" buffer in order to shift attention to the location specified by the "visual-location" indicated by the slots (screen-x and screen-y). The result of this request is a chunk that is placed in the "visual" buffer.

The ACT-R model is developed using the ACT-R 6 environment. All memory chunks get the same value of activation and all requests to the retrieval buffer are correctly satisfied without errors. These characteristics lead to a deterministic model where no error could happen.

The visual phase is principally based on the shift attention and the visual encoding actions, followed by the recognition phase, and finally the motor phase takes place to select physically the perceived and recognized object. The process of visual encoding needs 0.185 s to be completed and stores the chunk into the visual buffer. During the recognition phase, a retrieval request is made on the retrieval buffer in order to recover the specified chunk from the declarative memory. Finally a request on the motor buffer is made to simulate the physical interaction with the interface.

In the experimental study, users interact with the PDA and the screen. Each device involves three cognitive processes including the visual, cognitive and motor phase. Therefore, the ACT-R model simulates the time twice during the interaction with each object, on the PDA firstly and on the touch screen secondly. The simulation time is computed as the summation of the time estimated for each object to click on the PDA and on the touch screen.

5.2. Modeling the interaction with the contextual assistant using GOMS

The second model used in our study is a GOMS model in which a hierarchical representation is adopted for describing methods of task analysis and operators to accomplish specific goals. The first two steps of the recipe, gathering utensils and gathering ingredients can be interpreted in the GOMS language by a principal method that defines the main goal as shown in Fig. 8. The syntax used and the durations of steps follow the guidelines of (Kieras, 1999) in his work about GOMSL and GLEAN3.

For each task in our study, a method is defined following the concepts of GOMS methods in the definition of goals and subgoals. The GOMS model is based on a hierarchical representation of goals, in fact, the user achieves goals by solving subgoals (Dix et al., 2004), until reaching the basic operations called "operators" which can not be subdivided. The methods are described as a hierarchical structure, where a method may call for subgoals to be accomplished (John and Kieras, 1996a). The main goal in the GOMS model constitutes the root of tree hierarchy, and all the other sub-goals are generated automatically using the divide-and-conquer

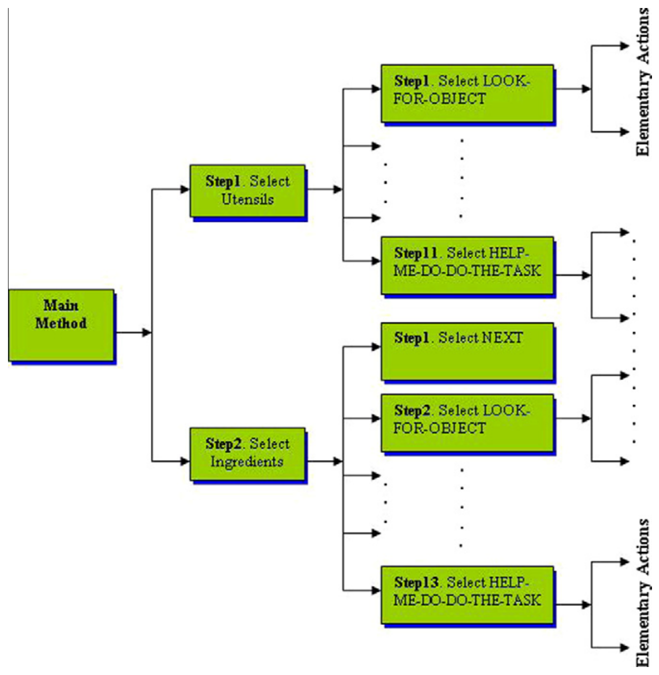
Main Method
Step1. Select Utensils
Step2. Select Ingredients
Step1. Select LOOK-FOR-OBJECT
Step11. Select HELP-ME-DO-DO-THE-TASK
Step1. Select NEXT
Step2. Select LOOK-FOR-OBJECT
Step13. Select HELP-ME-DO-DO-THE-TASK
Elementary Actions
Elementary Actions

Fig. 8. Tree decomposition of the main method in the GOMS model.

technique (Dix et al., 2004). In our GOMS model, each object is defined as a visual object and the methods have the same form for all objects. Fig. 8 shows the tree decomposition corresponding to the main goal of the GOMS model. The duration of a step in the GOMS model can be defined as the sum of the production cycle duration which equals to 50 ms and the duration of all actions included in the body of the step (Kieras et al., 1996; Amant et al., 2007) such as key presses which take 280 ms to be performed.

Our model is executed using the GLEAN modeling tool (Kieras et al., 1996). GLEAN (GOMS Language Evaluation and Analysis) is a tool for GOMS simulation, GLEAN generates quantitative predictions from a GOMS model and a set of well defined tasks (Kieras et al., 1996). GLEAN acts as interpreter for the GOMS language models by processing and executing these models in a simulated environment (Amant and Riedl, 2001).

5.3. Modeling the interaction with the contextual assistant interface using Fitts' Law

In our study, the user-interface interaction is based on the use of a touch screen, assuming that users remain standing at a distance of 30 cm from the touch screen, and point directly on the displayed objects by touching them using their index finger. After each pointing action, users returned their index finger to the start position, and the procedure continued like that for all the needed objects.

In order to calculate the time taken to push an object on the PDA, we apply the Fitts' Law as described in Section 2.3 during the interaction with the interface displayed on the PDA. We consider that the user's hand is held at a distance of 5 cm from the PDA. Thus, the predicted time to push each object using the Fitts' Law, can be expressed according to the following equation:

$$MT = MT_{touchscreen} + MT_{PDA} \tag{2}$$

By replacing the $MT_{touchscreen}$ and MT_{PDA}, the result equation takes the following form:

$$MT = \left(a_1 + b_1 \log_2\left(\frac{A_1}{W_1}+1\right)\right) + \left(a_2 + b_2 \log_2\left(\frac{A_2}{W_2}+1\right)\right) \tag{3}$$

where A_1 = 30 cm, and A_2 = 5 cm. W_1 and W_2 corresponds to the size of objects displayed on the touch screen and the PDA respectively. a_1, a_2 correspond respectively to the intercept of the linear regression equation applied to the touch screen and the PDA respectively. b_1 and b_2 represent respectively the slope of the linear regression equation applied to the touch screen and the PDA respectively.

By replacing the constants a_1, a_2, b_1 and b_2 by their appropriate values, the Eq. (3) becomes :

$$MT = \left(325.32 + 189.18 \log_2\left(\frac{A_1}{W_1}+1\right)\right) + \left(525.77 \log_2\left(\frac{A_2}{W_2}+1\right)\right) \tag{4}$$

For example, for the button "LOOK-FOR-OBJECT" in the Fig. 1, the movement time to select this button will be: $MT = 0.7137 + 0.6146 = 1.3283$ s. W_2 has the same value for all the objects (4 cm). W_1 varies according to the size of each button. The total time of the whole task applying the Fitts' Law is estimated using the following equation:

$$MT_{Total} = \sum_{i=1}^{n} MT_i \tag{5}$$

where n represents the number of objects displayed on the interface, and MT_i the corresponding movement time of each object.

6. Results and comparison

We describe the performance of our models upon two criteria: the task criterion which means the accuracy with which our models predict the overall duration of tasks, and the object criterion which represents the accuracy to predict the duration to push each object displayed in the interface.

6.1. Object performance

Table 2 shows summary of the user performance upon the object criteria and the ACT-R, GOMS and Fitts' Law model predictions. Values in parentheses represent the smallest and greatest value of user data observed for each object.

As shown in Table 2, the predicted time of objects using the ACT-R model is quite similar for all objects with a little difference of about 65 ms, except the object "SMALL-SAUCEPAN" which takes less time to be selected. The predicted time using the GOMS model is the same for all objects except the two objects: "LOOK-FOR-OBJECT (utensils)" and "LOOK-FOR-OBJECT (ingredients)". The same finding observed for the GOMS model is true for the Fitts' Law model with the addition of three other objects that have different predicted time than the others. These objects are: "HELP-ME-TO-DO-THE-TASK (utensils)", "NEXT" and "HELP-ME-TO-DO-THE-TASK (ingredients)". These variations in the predicted time for all models are explained in more details in the next sections.

According to Table 2, both ACT-R and GOMS models give a good approximation of user performance, unlike the Fitts' Law model that performs relatively poorly and gives less predicted time than the observed time.

Table 2 shows that, the predicted time to push each object is more less using the Fitts' Law than the cognitive models ACT-R and GOMS. For example, the object "CAN-OPENER" needs (2.165 s) to be pushed using ACT-R model and (2.550 s) using GOMS model. However, it needs (1.228 s) to be pushed using the Fitts' Law model. The predicted time using GOMS model and ACT-R model, is close to very close to the time obtained in the experimental study to push the same object, which equals to (2.291 s).

Table 2
Comparison of user data, ACT-R, GOMS and Fitts' Law model predictions by object.

Objects	User performance (s)	ACT-R (s)	GOMS (s)	Fitts' Law (s)
LOOK-FOR-OBJECT (utensils)	5.299 (3.838 7.052)	2.230	4.250	1.328
CAN-OPENER	2.291 (1.308 3.676)	2.165	2.550	1.228
COLANDER	2.966 (1.703 4.130)	2.250	2.550	1.228
MEASURING-SPOON	2.167 (1.206 3.262)	2.250	2.550	1.228
LADLE	2.847 (1.434 4.143)	2.165	2.550	1.228
SMALL-SAUCEPAN	1.980 (1.351 2.652)	2.080	2.550	1.228
WOODEN-SPOON	2.590 (1.561 3.622)	2.250	2.550	1.228
KNIFE	2.328 (1.676 2.947)	2.165	2.550	1.228
BIG-SAUCEPAN	1.779 (1.284 2.330)	2.165	2.550	1.228
CUTTING-BOARD	2.000 (1.600 2.451)	2.165	2.550	1.228
HELP-ME-TO-DO-THE-TASK (utensils)	2.039 (1.349 2.680)	2.165	2.550	1.328
NEXT	2.142 (1.289 3.082)	2.165	2.550	1.168
LOOK-FOR-OBJECT (ingredients)	1.955 (1.384 2.362)	2.165	2.650	1.328
PEPPER	2.448 (1.159 3.847)	2.250	2.550	1.228
SPAGHETTI	1.939 (1.311 3.180)	2.165	2.550	1.228
TOMATOES-BOX	1.794 (1.265 2.382)	2.165	2.550	1.228
GROUND-BEEF	2.491 (1.634 3.643)	2.165	2.550	1.228
ONION	2.021 (1.391 2.906)	2.165	2.550	1.228
TOMATO-SOUP	1.970 (1.373 2.714)	2.165	2.550	1.228
SALT-AND-PEPPER	2.490 (1.546 3.227)	2.165	2.550	1.228
OIL	1.965 (1.389 2.544)	2.165	2.550	1.228
MUSHROOMS	1.809 (1.232 2.477)	2.250	2.550	1.228
SUGAR	1.774 (1.213 2.300)	2.250	2.550	1.228
ITALIAN-SPICE	1.736 (1.217 2.772)	2.250	2.550	1.228
HELP-ME-TO-DO-THE-TASK (ingredients)	2.432 (1.265 3.654)	2.165	2.550	1.328

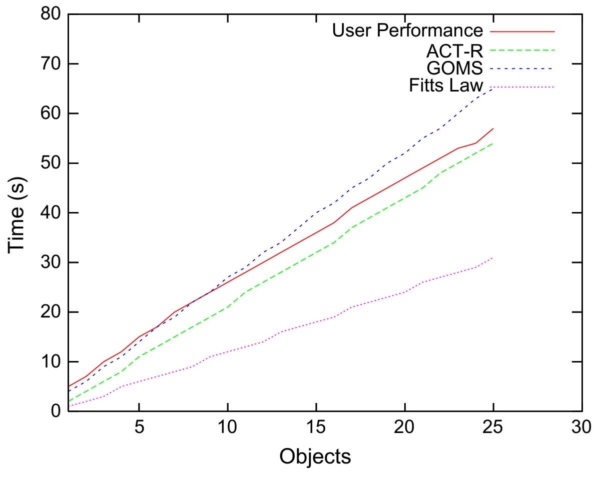

Fig. 9. Progression in pushing objects over the time for all models.

Despite the effectiveness and the accuracy of the Fitts' Law to predict time execution of tasks in such models elsewhere (Chikhaoui and Pigot, 2008b; MacKenzie, 1995), in the present study the Fitts' Law produces a less predicted time. This is supported by some related works in the literature, in which the Fitts' Law performs poorly (Amant et al., 2007).

Fig. 9 shows the progression in pushing objects during the whole task over the time for all models.

According to Fig. 9, all models follow a linear model.

6.2. Task performance

The performance of our models is presented at the object level as well as the task level. The performance of our models at the task level represents the capability of our models to predict the overall duration of tasks. Table 3 shows the user performance data, ACT-R, GOMS and Fitts' Law model predictions for both tasks: gathering utensils and gathering ingredients. The total time of each task is obtained by summing the time to select each object included in the task. The results obtained by the ACT-R and GOMS models, shown in Table 3, are very close to those obtained in the user performance study. Therefore, ACT-R and GOMS perform well at both object an task levels.

The Fitts' Law model does less well than the ACT-R and GOMS models. The results obtained by Fitts' Law in both tasks are about the half of the time observed in the user performance study. The Fitts' Law results are not really surprising, but rather they confirm the results obtained in the literature (Amant et al., 2007). Fig. 10 shows the same data in a detailed graphical form.

Figs. 11 and 12 show respectively the ACT-R, GOMS and Fitts' Law model predictions for gathering utensils and ingredients tasks.

Table 3
Comparison of user data, ACT-R, GOMS and Fitts' Law model predictions by task.

Task	User performance (s)	ACT-R (s)	GOMS (s)	Fitts' Law (s)
Gathering utensils	27.193	24.050	29.750	13.708
Gathering ingredients	27.236	30.650	35.800	17.332

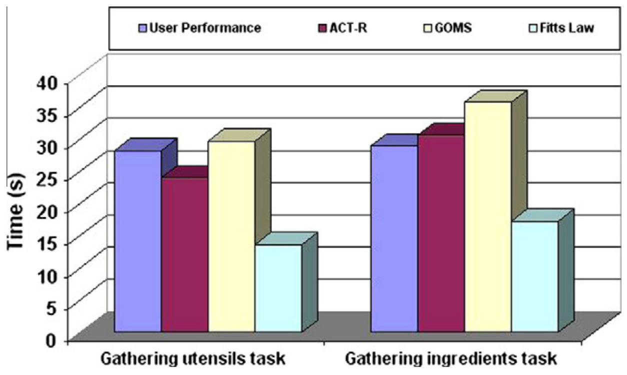

Fig. 10. User data, ACT-R, GOMS and Fitts' Law model predictions by task.

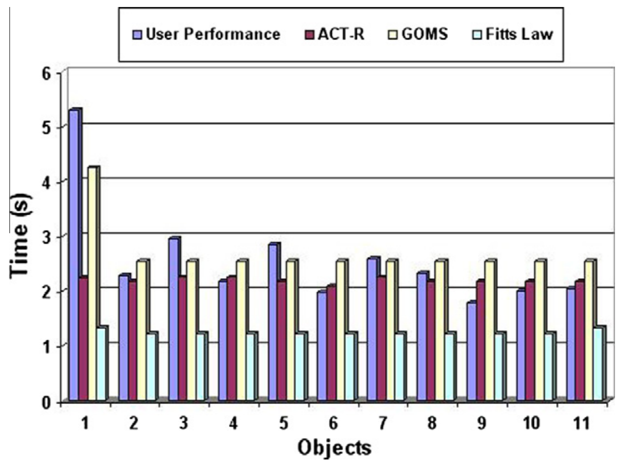

Fig. 11. ACT-R, GOMS and Fitts' Law model predictions for gathering utensils task.

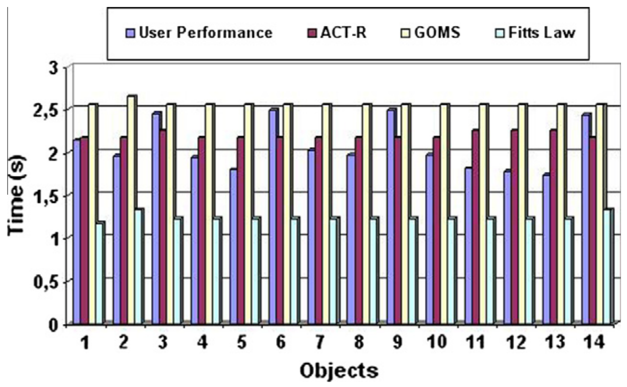

Fig. 12. ACT-R, GOMS and Fitts' Law model predictions for gathering ingredients task.

7. Discussion

In our study, we used three models to evaluate the contextual assistant's interface, developed for cognitively impaired people. The first two models are based on cognitive models, and the third one is based on the Fitts' Law model. All models have proved robust and efficient in our study. The cognitive models perform better than the Fitts' Law model, and give a better prediction of the user behavior at a more detailed level. In this section, we stress different issues in the HMI modeling. First of all, we show how a model based only on physical elements involved in the HMI gives a pretty good accuracy in the simulation results, but not precise as the cognitive models. Second, the regression models help to qualify the accuracy of the models. The last issue concerns how the three analytical methods cope with specific differences in time execution and sequences of actions. We first stress how the variability is present in the experimental results presented in that study. We then present the advantages and drawbacks of the three models to consider when choosing an analytical method. Several criteria must be weighed up depending on the goals of the simulation and the involvement to design the model. The characteristics of the three models studied here are resumed in Table 5 to help deciding which one is more suitable. More than one model should be used, especially because every model brings a specific analysis of the HMI. The discussion ends on an open issue: the learning process and errors simulation. We show how ACT-R and GOMS should take into account these human behaviors.

7.1. Physical and cognitive simulation

The results of the three simulations correspond to the execution time gathered during the experiment. As mentioned previously the two cognitive models, ACT-R and GOMS, show better accuracy than the Fitts' Law. The explanation is inherent to the principles underlying the Fitts' Law. The human machine interaction involves several human capabilities. It supposes that the user determines a goal he wants to reach, perceives the interface he faces, evaluates the correspondence between his goal and the interface proposed, solves the problems and finally plans an interaction he will then execute on the interface (John and Kieras, 1996c).

The Fitts' Law model focuses on the final step of the interaction, where the user plans his movement and executes it. Therefore, the interaction time is inversely proportional to the width of the target to reach on the interface and to the distance between the user and the target. In the Fitts' Law, the cognitive process is just restrained to the movement planning. In our experiment, the user needs to reach two targets, the first one on the object icon displayed on the touch screen, the second one on the PDA where the object name is written. The Fitts' Law simulates these two actions, forgetting the time the user needs to relate the object name (on the PDA) to the object image (on the touch screen). In contrast, GOMS takes into account the time required to perceive the interface, to set the goal and execute the action. The GLEAN tool of GOMS generates these basic operations to estimate the interaction time (Kieras et al., 1996). The ACT-R architecture provides visual and motor modules in a production system, modeling then all the steps involved during the interaction. The time taken by the user to evaluate the situation and solve the problem is for one part automatically calculated in GOMS and for the other part let at the discretion of the model designer in ACT-R. Table 4 presents some generic task actions involved during the HMI, and how they are incorporated in ACT-R, GOMS, and Fitts' Law. The human–computer interaction is defined in each model with certain precisions. As shown in Table 4, the HMI supposes four principal generic actions: (1) the initialization where the user determines the goal he wants to reach, (2) the visual phase where the user perceives the interface he looks at mainly under visual features, (3) the cognitive phase where he evaluates the adequacy between his goal and the interface and decides how to reach his goal, and finally (4) the user plans the motor action and executes it on the interface (John and Kieras, 1996b).

7.2. Linear regression model

To express how the cognitive models fit the experimental data we use a linear regression model as a formal representation for the ACT-R and GOMS models. As shown in Fig. 9 the progression of the task over time follows a linear curve. This is due for the most part to the task protocol where the user has to push buttons one after another progressively where each push button time is quite equivalent. We could then approximate the experimental data and the cognitive model results with a linear equation. This formal representation helps us to quantify the accuracy of the two cognitive simulations. The linear model obtained predicts time execution T of tasks by the number of objects to push n, plus a constant of the linearity. The aggregate linear model can be expressed by the following equation:

$$T = 2.2904n + 2.087 \tag{6}$$

To know how well the results of the regression model will be predicted, the coefficient of determination R^2 of the aggregate linear model is calculated. The greater the value of R^2 is, the more the regression model perfectly fits the data. In our case, $R^2 = 0.96$ which means that the aggregate linear model perfectly fits the data.

Table 4
Comparison of ACT-R, GOMS and Fitts' Law model for generic task actions.

Phase	ACT-R	GOMS	Fitts' Law
Task initialization	– Encode a procedure that will initiate the task	– Include mental operator	– No
Visual	– Encode objects as chunks – Make requests to visual and visual-location buffers	– Encode objects as visual objects – Call a method for the perceptual processor	– No
Cognitive	– Make requests to declarative memory to retrieve chunks	– Execute methods to accomplish goals using information in long term memory and properties of tasks	– No
Motor	– Make requests to motor module	– Call a method for the motor processor	– Physical movement from source to target

7.3. Time variability in the experimental study

Table 1 displays the experimental time needed to press each button, either an object image or a control button. This experimental time is the summation of the time to press the buttons, first on the touch screen and then on the PDA. The mean time to press each button remains quite the same allowing approximating the whole task with a linear curve, as shown in Section 7.2. Nevertheless, in the experimental results specific buttons require more time, which could be explained for various reasons.

The first button to press in the task, "LOOK-FOR-OBJECT" button, is the one requiring the most time (5.299). It also presents the most variation among results (standard deviation = 1.052). It is clearly shown on the Fig. 11, where the first blue column, indicating the time needed to press the first button in the experimental results, exceeds by around twice the others. This result indicates that a user takes time at the beginning of a task to determine the goal he wants to achieve. The time to execute the first action increases for all the users, all along the five experiments. It indicates the need for the user to evaluate the situation and determines his goal.

The time to press a button varies as shown in Table 1. Multiple causes should explain this variation. It first appears that the objects clearly defined and easily recognizable by the icon displayed on the touch screen require less time to be pushed. The sugar, mushroom or saucepans take less than 2 s to be pressed. They are frequently used in the every day life and the images represent clearly the concept. By contrast, an image badly recognizable needs more time to be selected. For instance, the time obtained to select the object "CAN-OPENER" is relatively longer than the one obtained to select the other objects. The object "CAN-OPENER" is not easily identifiable on the picture as shown in Fig. 15. This is due probably to the image chosen for the object that does not perfectly shows the specified object and the execution time indicates clearly the difficulty encountered by the subjects to select this object. Confusion between objects should provoke a delay in the execution time. "PEPPER" and "SALT-AND-PEPPER" take more time to be pressed even if these ingredients are very common and the image is clearly identified in Fig. 1. The delay to press is due to the time required by the subject to discriminate the wrong answer before selecting the good one.

Finally, the place where the image is displayed should influence the time execution. No evidence rises from the experimental result and no data have been gathered on eye tracking to ensure this hypothesis. The overall duration of the two tasks bring two remarks. First of all, time duration decreases slightly from the first to the fifth experiment for each user and for all the two tasks (Figs. 13 and 14). It underlines a tendency toward a learning process, even if sometimes distraction should provoke casual increase time as user 3 and 6 during the second task of the third experiment or user 4 during the first task of the fourth experiment. Second, the comparison between the duration of the first task and the second task shows equivalent results, respectively (28.290 s) and (28.973 s). This finding is surprising due to the number of objects to be pushed in each task, respectively 11 for the first task and 14 for the second task. The linearity of the model should lead to a significant time arising to perform the second task, around 6 s more (three objects for two execution seconds in average). It could be explained by the learning, where, during the second task occurring every time after the first one, the subject is more familiar with the experiment and interacts faster with the interface. This interpretation is invalidated by the fact that the two tasks are performed sequentially for each experiment. The Figs. 13 and 14 display a learning process inside each task as discussed previously. The second interpretation is related to the nature of the objects displayed on the contextual assistant's interface. The second task includes several ingredients to gather which have been identified as provoking increasing time. Some are difficult to identify on the image ("CAN-OPENER") while others offer confusions ("PEPPER" and "SALT-AND-PEPPER").

The experimental study presents several characteristics highlighting the features of the model. It focuses on how the motor control is used to select buttons on a PDA or on a touch screen, how the user repeats automatically these actions, how the cognitive processes are involved even if simple. Namely, it implies to perceive on the PDA the name of the next button to press, either action name or object name, to rely the object name with the appropriate image in the semantic memory, and to select it on the touch screen. This experiment allows showing several cases of time variability we discussed previously but neglects others. In order to compare the three models, we now mention other important HMI situations that could be simulated by one or another model. The movement executed by the user remains quite similar. The user first pushes on the touch screen and then on the PDA. Therefore, the distance between the two successive GUIs remains the same, equal to the distance between the touch screen and the PDA. No movement occurs between two GUIs of the same screen, either PDA or touch screen. The protocol of the experiment is rigid. The users are asked to follow a specific method without choosing any alternative.

7.4. HMI and Fitts' Law model

The Fitts' Law model assimilates the interface to a set of multiple targets to reach Table 5. The simulation consists of calculating the time needed to execute the physical movements between all the GUI, modeled as targets. It simulates the time according to the width of the target. Less time is needed to reach a larger target. It explains why in Table 2 all the object icons, displayed on buttons of the same size, required the same time (1.228). It explains also why the "NEXT" button, the larger in the screen, is reached faster (1.128) while the three remaining buttons, the smallest of the interface, are reached in a slower time (1.328). The first button to press in the task, "LOOK-FOR-OBJECT" button, is the one requiring the most time (5.299) to be pressed by the subjects. The Fitts' Law is unable to model the time needed to initiate a task neither

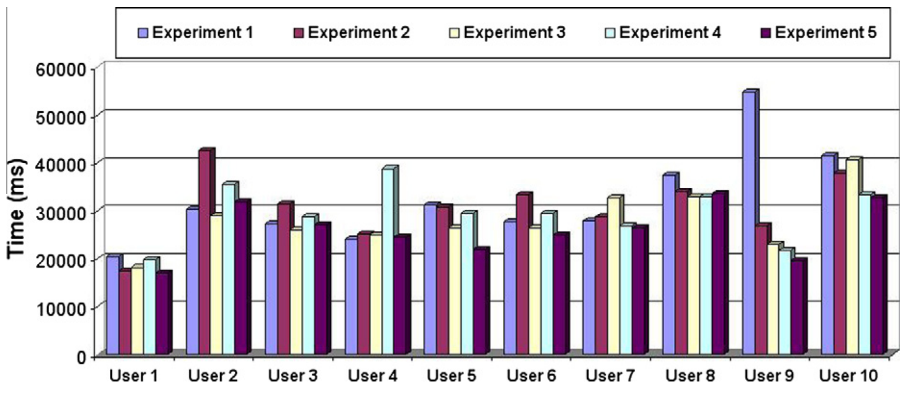

Fig. 13. Variation of time in accomplishing the first task over experiments.

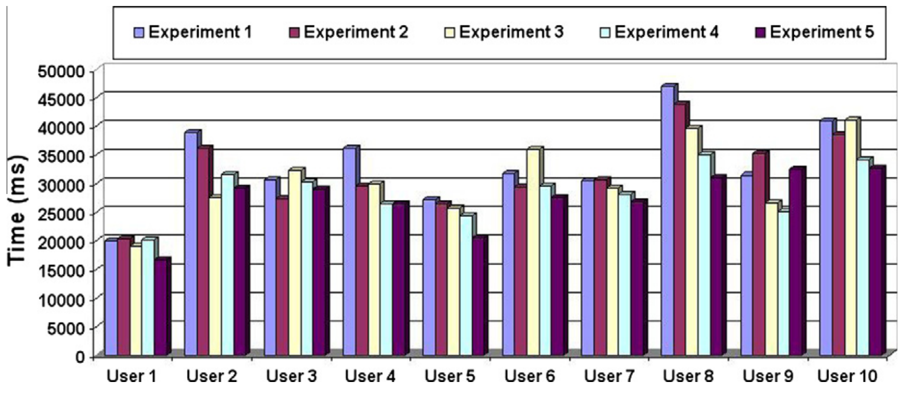

Fig. 14. Variation of time in accomplishing the second task over experiments.

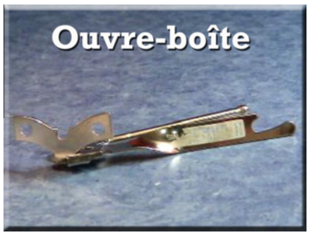

Fig. 15. Example of an object not easily identifiable.

the learning nor the individual characteristics. However, the Fitts' Law offers a good accuracy to simulate the time needed to select various GUIs on the same interface that are reached in a given order. The Fitts' Law approximates the time depending on the distance between the two sequential GUIs. In our model, the time varies only according to the object width, because the distance between a button and the next one to press is constant as mentioned earlier. Our experiment forces the sequence of the GUI to press. The next action to perform is determined with no choice between various alternatives. Therefore, our Fitts' Law model does not present alternatives. However, the Fitts' Law gives good accuracy between alternatives by summing the time of all the GUIs used for each alternative. The Fitts' Law model should then be very helpful to choose between various designs and to evaluate the speed of each alternative. But when an interface necessitates too much time to perceive the stimuli or solve problems, the Fitts' Law is unable to estimate accurately the use of that interface. In addition to the previous points, the Fitts' Law performs poorly to evaluate the time needed by normal persons to select an object displayed on the interface. Therefore, with persons with special needs such as persons with cognitive or motor disabilities, it will also perform poorly given the errors these persons will commit during the task realization. The Fitts' Law will be discarded when evaluating the HMI with this kind of population, and the focus will be only on cognitive models that incorporate perceptual, cognitive and motor functions.

7.5. HMI and GOMS model

The GOMS model focuses on the task to accomplish when using an interface. The task is decomposed hierarchically in subtasks up to the unit elements, called operators. The way the user interacts with the interface is analyzed according to perceptual and cognitive backgrounds. The results of the simulation consist of the time needed to perform the task and the task decomposition. It highlights the way the user evaluates the interface, plans his actions and evaluates the effects Table 5. The GOMS model performs very well to estimate the beginning of the task. It models a difference between the first action, press "THE LOOK-FOR-OBJECT" button, and the others (4.250 versus 2.550) as shown in Table 2. Also, the model slightly increases the time at the beginning of the second step where the time to press the "LOOK-FOR-OBJECT" button is raised to 2.650 instead of 2.550. These findings are in agreement with the experimental result where the subjects take more time to initiate the action (5.299). This increasing time to start a task is inherent to the GOMS model where beginning a task is considered as a mental operator that requires more time. Except the first button, the time required to press each button is equal to (2.550 s) as shown in Table 2. This time gives a good accuracy of the average time in the experiment, a little bit slower than the subjects as shown in Fig. 10. It still corresponds to the experimental data as demonstrated previously by the linear regression model. The

Table 5
Comparison of three analytical models: Fitts' Law, GOMS and ACT-R. GUI (Graphical User Interface).

	Fitts' Law	GOMS	ACT-R
Concepts of the model	– Physical principle – The interaction with the interface is modeled as sequences of targets to reach	– Hierarchical decomposition of the task, in subtasks – Cognitive and perceptual components in HMI are generated automatically	– Task decomposition in procedures (rules). – Description of the stimulus displayed on the screen. – Perceptual and motor interaction modeled with visual, oral and motor modules – Cognitive abilities modeled with declarative and procedural modules
Modeling process	– Decompose the tasks in sequence of actions – Measure all the GUI needed in the interface and the distance between them – Calculate the individual execution time and add them	– Decompose the tasks in sequence of subtasks and actions – Describe actions in operators – Generate automatically the time of all the actions	– Describe the elements needed in the interface in terms of declarative and visual chunks – Decompose the tasks in rules – Code the declarative and procedural memory in ACT-R – Insert the sub symbolic system – Generate automatically samples of interactions – Sum over the samples to obtain mean time
Simulation results	– Time execution	– Time execution – Decomposition of the task	– Time execution – Trace of the simulated execution that displays the procedures and memory chunks used
Modeling the physical features of the interface	– Width and height of the control GUI to press – Distance between the control GUI to reach – No visual aspects of the other GUI presented in the interface (size of the police, etc.)	– Automatic and uniform way to perceive the interface whatever is the aspect	– Localization and identification features to determine the visual and oral stimulus presented to the user
Modeling the cognitive and perceptual abilities required to interact with the computer	– The attention required to reach a target influences the time to reach a target – It depends on the target size and target distance	– Automatic and uniform way to perceive the interface – Cognitive operators automatically generated for each action	– Perceptual and motor interaction modeled with visual, oral and motor modules – Cognitive abilities modeled with declarative and procedural modules
Modeling the action sequence in the task	– The execution time results from the summation of time estimated to interact with each GUI involved in the task. It depends on the width of the control GUI to be reached and the distance from the control source GUI to the next control GUI to be reached	– The time of the execution sequence is estimated as the summation of all the atomic operators used to realize the sequence order	– The time results from the summation of all the procedures time triggered during the interaction and the time to retrieve the chunks in the declarative memory – The procedure time depends on the visual, cognitive and motor chunks and rules involved during the interaction
Modeling the time to initiate the interaction	No	– Automatically generated	– Not taken into account automatically – Should be modeled
Modeling the alternatives in the actions sequence	– Estimation of the time execution for each alternative and average according to the likelihood between all the alternatives used – Estimation of the worse and the best path	– Estimation of the time execution for each alternative and average according to the likelihood between all the alternatives used – Estimation of the worse and the best path	– Alternatives are modeled through the sub symbolic system applied to the procedural memory – Simulate the choice between two alternatives depending on the previous success of the alternatives -Situation 1: the execution time is calculated by generating all the alternatives. It calculates the best and worse path. – Situation 2: the choice of the alternatives is automatically generated by the sub symbolic system and the average time is estimated over the samples generated
Modeling the individual characteristics	No	– The time execution is averaged among experimental results – It could represent an expert user	- Individual cognitive characteristics are modeled through the sub symbolic system applied to the declarative memory. – Simulate arousal and memory characteristics
Modeling the learning	No	– Can predict the relative learning time for each design	– By success accumulated when choosing a good alternative

(continued on next page)

Table 5 *(continued)*

	Fitts' Law	GOMS	ACT-R
			– Through the learning process provided in ACT-R where procedural actions are learned over the time
Modeling the execution errors	No	No	– Models errors of omission and confusion between the objects of the interface by the sub symbolic system – Models execution errors which occur by selection of a wrong procedure
Deterministic simulation	Yes	Yes	No, if the sub symbolic system is activated

GOMS takes in account the perceptual and cognitive components in the HMI, which gives good approximations. But that components are automatically simulated making difficult to analyze and simulate specific situations or individual characteristics. The user is categorized expert that does not make errors. The performances are then averaged without modeling the user reactions following specific cognitive demand, such as attention requirements or planning solicitation during the task. The GOMS does not estimate the curve of learning neither models the execution errors. The GOMS helps to decide which alternative is more suitable. It offers a way to express alternatives when the interface proposes several methods to perform the task. It simulates then the worst time and the best one. GOMS does not generate a trace of execution where the user is simulated, with various performances. It remains deterministic. The GOMS then helps to describe a task and simulates the time required to achieve it using an interface, which could present alternatives. By taking into account the initiation process and the perceptual and cognitive demands in the HMI, the accuracy of the estimation is pretty well. Thanks to the perceptual modeling, GOMS takes in account the complexity of the interface in terms of information displayed.

7.6. HMI and ACT-R model

The ACT-R models how a user interacts with an interface by the means of modules namely the perceptual, procedural and declarative modules. It necessitates describing the task in procedures and memory chunks. It simulates the time to perform the task by adding the time needed to select the procedures and retrieve the chunks. ACT-R provides the time of execution as well as the trace of actions executed, procedures and chunks activated by a simulated user Table 5. ACT-R performs very well in the prediction of the time of execution of tasks comparing to GOMS. As shown in Table 2, the time estimated by ACT-R to press each button is the same with a little bit difference (2.16 s and 2.25 s). This is due to the fact that the perceptual, cognitive and motor components are applied for each object. The introduction of these components in the HMI evaluation allows to give more accurate time as shown in Table 2. The predicted time of the ACT-R model is very close to that obtained by participants, except some objects that have a time slightly greater or smaller than the predicted time. ACT-R gives good predictions of the time at the object level as well as the task level as shown in Fig. 10. Some differences in the predicted time of the ACT-R model are observed in Table 2 (2.16 s and 2.25 s). This is due to several rules involved during the visual processing when a new object is detected in the visual scene, or during the information retrieval or motor actions. In ACT-R the visual processing is explicitly defined using requests on the visual buffers, first to identify the object (visual-location buffer) and then to recognize it (visual buffer). The first object selected in both tasks, "LOOK-FOR-OBJECT" button, is the one that requires the most time (5.29 s) by the subjects. Our ACT-R model estimates the time to select this object in the range of the other objects. The initiation is not implicitly taken in account in ACT-R. The model designer should explicitly code the procedures that will be triggered to initiate the goal the user wants to achieve when using the interface.

ACT-R can simulate different users skills and is not limited to only expert users. The individual characteristics can be modeled using the sub symbolic system applied to both procedural and declarative memories. For example, in the case of persons with Alzheimer disease, the progression of the dementia can be modeled by changing the sub symbolic parameters that model the cognitive abilities (Serna et al., 2007). Modeling the individual characteristics in ACT-R allows more understanding of the human behavior when interacting with HMI, and predicting accurately the time needed to perform tasks. ACT-R has been used to explain a wide range of cognitive phenomena by producing models that fully characterize the human behavior, and make accurate predictions about choices, latencies, learning, cognitive errors, stress and fatigue (Gunzelmann et al., 2007; Jongman and Taatgen, 1999). As mentioned earlier, ACT-R has two memory systems, a declarative memory and a procedural memory. For each memory is associated a learning mechanism that adds and maintains the knowledge in them (Taatgen, 1996). Indeed, the activation value of a chunk in declarative memory is based on its past use according to the number of times the chunk was needed, and how long ago this was done. The more the chunk is used, the more its activation value increases. In the same way, knowledge in procedural memory is represented by production rules. To each rule, is associated a number of parameters such as the "Strength parameter" that reflects the past use of the production rule. All these aspects of learning are managed by the sub symbolic mechanism of ACT-R.

The human machine interaction is modeled in ACT-R through the simulation of the user as well as the interface the user is using. ACT-R offers a mechanism which is able to model even the cognitive errors that happen during the interaction. For example, confusion errors, that can happen when two objects have a great similarity, can be modeled using the sub symbolic mechanism by increasing the "similarity parameter" between the two objects. Other types of errors such as omission, commission, anticipation, and perseveration errors can be modeled using the sub symbolic mechanism as described in Serna et al. (2007). The execution errors should occur in a nondeterministic way thanks to the sub symbolic system of ACT-R.

7.7. Learning process and behavioral errors

User learning and errors are two important aspects for interactive systems. In fact, the introduction of these aspects might reduce/increase the modeled timings. In the Fitt's Law, there is no mechanism for incorporating learning or execution errors. But with the user experience, the more the interface is used and practiced by the user, the more the interface is learned, and the more the time is reduced. As mentioned in Table 5, the GOMS model can predict the relative learning time for a specified design as well as the execution time from number of rules of the procedural knowledge represented in methods (John and Kieras, 1994). In the GOMS model, similar methods can also reduce the learning time.

The strength of ACT-R is that it is able to model both learning and execution errors as mentioned in Table 5. In fact, when good alternatives are successfully chosen, this accumulated success leads to a learning of these alternatives. In the same way, when production rules are fired several times, they will be learned progressively over time and will have more chance to be selected during the execution process. This is made by the sub symbolic mechanism provided in ACT-R. Regarding execution errors, when the wrong chunk is retrieved instead of the good one, or the wrong production is selected instead of the good one, this might generate execution errors. ACT-R incorporates several types of execution errors that can be modeled using the sub symbolic mechanism. A list of these errors and how they are modeled in ACT-R can be found in Pigot and Dion (2009) and Serna et al. (2007). However, our deterministic model does not take into account behavioral errors neither learning. They should be included in our perspectives.

8. Conclusion and perspectives

This study empirically demonstrated that analytical models are a powerful tool for evaluating interfaces and predicting user performance. The main goal of our study is to evaluate the contextual assistant's interface by simulating the human machine interaction, focusing on the time execution of tasks. We used three efficient analytical models to evaluate the specified interface, two of them are based on cognitive models which are ACT-R and GOMS, and the third one is the Fitts' Law model. The results show that, all models predict well the time execution of tasks. The present study shows the effectiveness of the cognitive models in the prediction of user performance. Indeed, the GOMS model predicts user performance at good level, and the ACT-R model can predict user performance at more detailed level and performs almost as well, which prove that, the models developed are powerful and realistic. However, the Fitts' Law offers a simple way to estimate the speed of using an interface, taking apart the cognitive process. It helps to determine if the design in the interface is time consuming. Thanks to the Fitts' Law, one should choose between different controls in terms of types, place on the interface and width in order to diminish the execution time (Raskin, 2000). It also helps to decide the faster order to arrange the controls to diminish the time interaction during a specific task. In the case of a sequence of actions, the Fitts' Law estimates the worst and the best path to reach the specified GUI, by taking into account the distance between the different GUIs. This allows the designer to select the optimized path to reach the target GUI.

This study constitutes a new direction in the evaluation of HMI that takes into account the user as well as the interface with which the user interacts. In fact, the use of analytical methods allows on one hand, to simulate the user in terms of behavior, individual characteristics, and cognitive and motor capacities, and simulate the HMI in terms of the interface characteristics (GUI, interaction mode (mouse, keyboard, tactile, etc.), size of objects, colors, text, etc.) on the other hand. The description of these characteristics allows to explain and understand at a very detailed level the process of the evaluation when and where needed. Moreover, analytical methods generate a huge quantity of simulated data about the system being evaluated. This could be helpful when conducting HMI evaluations in diverse situations where experimental studies are very difficult and dangerous such as risk situations (airplane pilot, nuclear center, Space Shuttle, etc.), specific type of population such as persons with cognitive disabilities, or even when the installation is costly in time and equipment. The three models selected in our evaluation give a good approximation of the user performance. These models propose a detailed and complementary analysis on different aspects of the HMI, such as the physical part of the HMI (Fitts' Law), the task analysis (GOMS), and the cognitive processes involved in the HMI (ACT-R).

Our models are principally simulating people with special needs interacting in a smart home. Therefore, three components should be modeled: the cognitive abilities of the subject, the interaction with interface and the environment itself.

First, the model must help us to provide analysis on the behavior of the person and his characteristics, and to get a clear idea about the cognitive components involved during the HMI. Second, the description and modeling of the interface gives explanation about the way the person interacts in the smart home in order to accomplish daily living tasks. This description is of great importance given the need to take into account the person characteristics. The description of the interface includes its utility and usability, the cognitive load, etc. Finally, the description of the environment in which the tasks are performed allows to study how the environment changes according to the person behavior, and can help him to accomplish tasks using pervasive technologies. According to the results obtained, the three models could be used to improve the design of the interface and to optimize it in order to be more accessible and easily usable. One important remark that should be mentioned is the capacity of ACT-R to model and predict cognitive errors that can occur during tasks realization or during the HMI. This constitutes a very strong advantage for using ACT-R. In fact, in smart home environment where the assistance of persons takes a high priority, we can take advantage of this capacity of ACT-R to improve the quality of assistance and avoid risk situations by predicting the errors that can happen in smart homes.

In addition to the interesting, exhaustive and comprehensive study for evaluating HMI in the context of smart homes presented in this paper, our work makes several contributions: (1) an evaluation study of new human–computer interaction tasks performed in pervasive environments (smart homes); (2) new efficient and generic models developed for accurately predicting task performance; (3) implementation of new applications to support our modeling and make into practice the analytical evaluation; and finally (4) an empirical demonstration of the effectiveness of cognitive models compared to the Fitts' Law.

Some improvements should be brought to our models, particularly the ACT-R model. First, our ACT-R model is deterministic and does not make errors. It should be extended to allow errors in the pointing actions such as: pushing an object several times before or after to look for its location in the environment, or pushing an object instead of another one. These errors are essentially related to memory problems that may occur in the task modeling (Pigot and Dion, 2009; Serna et al., 2007) and during the interaction with the contextual assistant's interface. In contrast, the GOMS model in its current version does not take into account the possibility of modeling errors. In fact, GOMS is developed in an effort to provide theoretical predictions of expert performance on several dedicated situations.

Second, since the contextual assistant is designed to assist cognitively impaired people in smart homes, it would be interesting to do some experiments with this population, which allows us to study the behavior of our models in real situations and to evaluate their performance and effectiveness.

Third, eye tracking systems are now inexpensive, reliable, and precise enough to significantly enhance system evaluations. It would be interesting in the future to model this part in order to provide an explanation of the detailed visual strategies used to accomplish tasks, and to provide the scientific basis guidelines for screen layout design.

Finally in our models, the action of searching an object is resumed to the interaction with the touch screen. The contextual assistant offers an interaction with the smart home to help people recovering utensils and ingredients dispatched in the kitchen. It would be interesting in the future to model this part by simulating the shift of attention in the environment and the movement of users picking up the objects in the kitchen.

References

Abras, C., Maloney-Krichmar, D., Preece, J., 2004. User-centered Design. W. Encyclopedia of Human–Computer Interaction. Sage Publications, Thousand Oaks. 1–14.

Amant, R.S., Horton, T.E., Ritter, F.E., 2007. Model-based evaluation of expert cell phone menu interaction. ACM Transactions on Computer–Human Interaction 14 (1), 1–24.

Amant, R.S., Riedl, M.O., 2001. A perception/action substrate for cognitive modeling in HCI. International Journal of Human–Computer Studies 55 (1), 15–39.

Anderson, J.R., Bothell, D., Byrne, M.D., Douglass, S., Lebiere, C., Qin, Y., 2004. An integrated theory of the mind. Psychological Review 111, 1036–1060.

Anderson, J.R., Taatgen, N.A., Byrne, M.D., 2005. Learning to achieve perfect time sharing: architectural implications of Hazeltine, Teague, & Ivry (2002). Journal of Experimental Psychology: Human Perception and Performance 31 (4), 749–761.

Bligard, L.-O., Osvalder, A.-L., 2007. An analytical approach for predicting and identifying use error and usability problem. In: Holzinger, A., (Ed.), HCI and Usability for Medicine and Health Care, vol. 4799/2007, pp. 427–440 (Chapter 38).

Bothell, D., 2004. Act-r 6.0 Reference Manual. Working Draft, Carnegie Mellon University.

Bovair, S., Kieras, D.E., Polson, P.G., 1990. The acquisition and performance of text-editing skill: a cognitive complexity analysis. Human–Computer Interaction 5 (1), 1–48.

Byrne, M.D., 2001. ACT-R/PM and menu selection: applying a cognitive architecture to HCI. International Journal of Human–Computer Studies 55, 41–84.

Card, S.K., English, W.K., Burr, B.J., 1987. Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys, for text selection on a CRT. Human–Computer Interaction: A Multidisciplinary Approach, 386–392.

Card, S.K., Newell, A., Moran, T.P., 1983. The Psychology of Human–Computer Interaction. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

Chikhaoui, B., Pigot, H., 2008a. Analytical model based evaluation of human machine interfaces using cognitive modeling. International Journal of Information Technology 4 (4), 252–261.

Chikhaoui, B., Pigot, H., 2008b. Simulation of a human machine interaction: locate objects using a contextual assistant. In: Proceedings of the 1st International North American Simulation Technology Conference, pp. 75–80.

Dix, A., Finlay, J., Abowd, G.D., Beale, R., 2004. Human–Computer Interaction. Pearson. Prentice-Hall, Inc, England.

Fitts, P.M., 1954. The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology 47 (6), 381–391.

Gowans, G., Campbell, J., Alm, N., Dye, R., Astell, A., Ellis, M., 2004. Designing a multimedia conversation aid for reminiscence therapy in dementia care environments. In: Proceedings of the Human Factors in Computing Systems Conference. ACM, New York, NY, USA, pp. 825–836.

Gunzelmann, G., Gluck, K.A., Kershner, J., Van Dongen, H.P.A., Dinges, D.F., 2007. Understanding decrements in knowledge access resulting from increased fatigue. In: Proceedings of the 29th Annual Conference of the Cognitive Science Society.

Heim, S., 2007. The Resonant Interface: HCI Foundations for Interaction Design. Addison Wesley (Paperback).

John, B.E., Kieras, D.E., 1994. The GOMS family of analysis techniques: tools for design and evaluation. Technical Report CMU-CS-94-181, Carnegie Mellon University School of Computer Science.

John, B.E., Kieras, D.E., 1996a. The GOMS family of user interface analysis techniques: comparison and contrast. ACM Transactions on Computer–Human Interaction 3 (4), 320–351.

John, B.E., Kieras, D.E., 1996b. Using GOMS for user interface design and evaluation: which technique? ACM Transactions on Computer–Human Interaction 3 (4), 287–319.

John, B.E., Kieras, D.E., 1996c. Using GOMS for user interface design and evaluation: which technique? ACM Transactions on Computer–Human Interaction 3 (4), 287–319.

John, B.E., Salvucci, D.D., 2005. Multi-purpose prototypes for assessing user interfaces in pervasive computing systems. IEEE Pervasive Computing 4 (4), 27–34.

Jongman, L., Taatgen, N.A., 1999. An ACT-R model of individual differences in changes in adaptivity due to mental fatigue. In: Proceedings of the Twenty-first Annual Conference of the Cognitive Science Society, pp. 246–251.

Joon, L.S., Ilseok, K., Wook, K.M., 2007. A user interface for controlling information appliances in smart homes. In: Nguyen, N.T., et al. (Eds.), Agent and Multi-Agent Systems: Technologies and Applications, vol. 4496/2007, pp. 875–883 (Chapter 92).

Kieras, D., Polson, P.G., 1999. An approach to the formal analysis of user complexity. International Journal of Human–Computer Studies 51 (2), 405–434.

Kieras, D.E., 1999. A guide to GOMS model usability evaluation using GOMSL and GLEAN3. Technical Report, University of Michigan.

Kieras, D.E., 2001. Using the keystroke-level model to estimate execution times. Techical Report, University of Michigan.

Kieras, D.E., 2003. GOMS models for task analysis. In: Diaper, D., Stanton, N., (Eds.), Task Analysis for Human–Computer Interaction, Lawrence Erlbaum Associates, pp. 83–116.

Kieras, D.E., Meyer, D., 1997. An overview of the epic architecture for cognition and performance with application to human–computer interaction. Human–Computer Interaction 12 (4), 391–438.

Kieras, D.E., Wood, S.D., Abotel, K., Hornof, A., 1996. Glean: a computer-based tool for rapid GOMS model usability evaluation of user interface designs. In: Proceedings of the 8th ACM Symposium on User Interface Software and Technology, pp. 91–100.

Koskela, T., Väänänen-Vainio-Mattila, K., 2004. Evolution towards smart home environments: empirical evaluation of three user interfaces. Personal Ubiquitous Computing 8 (3-4), 234–240.

Lussier-Desrochers, D., Lachapelle, Y., Pigot, H., Bauchet, J., 2007. Apartments for people with intellectual disability: promoting innovative community living services. In: Proceedings of the 2nd International Conference on Intellectual Disabilities/Mental Retardation.

MacKenzie, I.S., 1989. A note on the information-theoretic basis of Fitts' law. Journal of Motor Behavior 21 (3), 323–330.

MacKenzie, I.S., 1995. Movement Time Prediction in Human–Computer Interfaces. Human–Computer Interaction: Toward the Year 2000. San Francisco, CA, USA, pp. 483–492 (Chapter 7).

MacKenzie, I.S., Sellen, A., Buxton, W., 1991. A comparison of input devices in elemental pointing and dragging tasks. In: Proceedings of the CHI '91 Conference on Human Factors in Computing Systems. ACM, New York, pp. 161–166.

Minoh, M., Yamazaki, T., 2006. Daily life support experiment at ubiquitous computing home. In: Proceedings of the Information Processing and Management of Uncertainty in Knowledge-Based Systems Conference.

Newell, A., 1990. Unified Theories of Cognition, Cambridge, Mass: Harvard University Press Edition.

Nielsen, J., 1993. Usability Engineering. Academic Press, Boston, MA.

Nielsen, J., 1995. Usability inspection methods. In: Proceedings of the Conference Companion on Human Factors in Computing Systems. ACM, New York, NY, USA, pp. 377–378.

Nielsen, J., Molich, R., 1990. Heuristic evaluation of user interfaces. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, New York, NY, USA, pp. 249–256.

Nielsen, J., Phillips, V.L., 1993. Estimating the relative usability of two interfaces: heuristic, formal, and empirical methods compared. In: Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems. ACM, New York, NY, USA, pp. 214–221.

Pigot, H., Bauchet, J., Giroux, S., 2007. Assistive Devices for People with Cognitive Impairments. The Engineering Handbook on Smart Technology for Aging, Disability and Independence. John Wiley and Sons. pp. 217–236 (Chapter 12).

Pigot, H., Bauchet, J., Giroux, S., 2008. A smart home to assist in recipe completion. In: Technology and Aging Selected Papers from the 2007 International Conference on Technology and Aging, vol. 21. Assistive Technology Research Series.

Pigot, H., Dion, A., 2009. Dysexecutive syndrome model using ACT-R. In: Poster presented at the 31st Annual Conference of the Cognitive Science Society.

Po, B.A., Fisher, B.D., Booth, K.S., 2004. Mouse and touchscreen selection in the upper and lower visual fields. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, New York, NY, USA, pp. 359–366.

Raskin, J., 2000. The Humane Interface: New Directions for Designing Interactive Systems. Addison-Wesley.

Salvucci, D.D., Macuga, K.L., 2002. Predicting the effects of cellular-phone dialing on driver performance. Cognitive Systems Research 3 (1), 95–102.

Schedlbauer, M.J., Pastel, R.L., Heines, J.M., 2006. Effect of posture on target acquisition with a trackball and touch screen. In: Proceedings of the 28th International Conference on Information Technology Interfaces. IEEE, pp. 257–262.

Serna, A., Pigot, H., Rialle, V., 2007. Modeling the progression of Alzheimers disease for cognitive assistance in smart homes. User Modeling and User-Adapted Interaction 7, 415–438.

Taatgen, N., 1996. Explicit learning in ACT-R. In: Schmid, U., Krems, J., Wysotki, F., (Eds.), Mind Modelling: A Cognitive Science Approach to Reasoning, Learning and Discovery, pp. 233–252.

Weiser, M., 1999. The computer for the 21st century. SIGMOBILE Mobile Computing and Communications Review 3 (3), 3–11.

Xinguo, Y., Bin, X., Weimin, H., Boonfong, C., Junfeng, D., 2007. A framework of context-aware object recognition for smart home. In: Proceedings of the 5th International Conference on Smart Homes and Health Telematics, pp. 9–19.

Yen, B., Hu, P., Wang, M., 2005. Towards effective web site designs: a framework for modeling, design evaluation and enhancement. In: Proceedings of the IEEE International Conference on e-Technology, e-Commerce, and e-Service (EEE05), pp. 1–6.