A Comparison of Shortcut and Step-by-Step Adaptive Menus for Smartphones

Sara Bouzit[1,2], Gaëlle Calvary[2], Denis Chêne[1], Jean Vanderdonckt[3]
[1]Orange Labs, 28 chemin du Vieux Chêne, F-38240 Meylan (France)
[2]Univ. Grenoble Alpes, LIG,
CNRS, LIG, F-38000 Grenoble (France)
[3]Université catholique de Louvain, Louvain School of Management, B-1348 Louvain-la-Neuve (Belgium)
{sara.bouzit, denis.chene}@orange.com, {sara.bouzit, gaelle calvary}@imag.fr, jean.vanderdonckt@uclouvain.be

This paper reports on the results of an experiment comparing two graphical adaptive menus for smartphones in order to improve their hierarchical navigation: "Shortcut Menu" and "Step-by-Step Menu" keep constant the actual presentation of initial menus and overlay them with a prediction window displaying the most frequently used menu items, wherever they are located in the hierarchy. In order to reach predicted items, the "Step-by-Step Menu" maintains the consistency with the initial menu through a level-by-level navigation while the "Shortcut Menu" directly moves the end user towards the predicted menu item, thus shortcutting the hierarchical navigation. Thirteen subjects performed fifty tests each on smartphones and data were collected about their item selection time and error rate. The "Step-by-Step Menu" has a positive impact on both variables, whether frequently used menu items are accurately predicted or not. The "Step-by-Step Menu" is fast, but could induce some problems when prediction is wrong.

Adaptation. Adaptive menu. Navigation. Nested lists. Prediction. Prediction window.

1. INTRODUCTION

One of the most predominant interaction technique in Graphical User Interfaces (GUIs) remains the *hierarchical unfolding menu*, such as a cascading menu pulled down from a menu bar on a desktop or a nested list found on mobile phones (Ahlström, 2006; Cockburn & Gin, 2006; Bailly *et al*., 2009). Instead of providing the end user with all menu items possible, the end user progressively moves between its sub-menus. Selecting an item therefore consists of a series of sub-selections, each corresponding to one sub-menu at a time, a process we hereby refer to as *hierarchical navigation*. Designing menus involves a multi-factorial approach (Norman, 1991) balancing *intrinsic* factors to the hierarchical navigation (e.g., the menu depth, the menu breadth at each level, the menu presentation, activation area) as well as *extrinsic* factors (e.g., dwell time, item selection time, and reaction time to pointing movements).

One determinant factor is related to the breadth vs depth trade-off (Jacko & Salvendy, 1996): the deeper the menu level is, the more time the end user needs to navigate to reach it, unless a shortcut is offered, which is rarely the case on a cascading menu (Ahlström, 2006) and never on a nested list on smartphones (Park *et al*., 2007). In the past, it was recommended to minimize the menu depth since it negatively impacts both the item selection time and the error rate (Kiger, 1984).

This trend has been recently inverted with the advent of smartphones which cannot display a large amount of menu items at once: to reduce breadth while increasing depth, since end users prefer to be guided towards the desired item. They care less about item selection time provided that they can reach the item (Pirolli, 2009).

A second factor lies in the hierarchy complexity: while interacting, the end user has to create a mental model of the menu hierarchy, learn it, and constantly update it according to her needs. Novice users start with a minimal, if not non-existing, mental model, therefore spending time to discover the hierarchy. Expert users, although they maintain an advanced mental model, still need to spend time to traverse the hierarchy until the desired item is reached, then go back level by level, which is not efficient and frustrating. A novice user is expected to become an expert as soon as interacting with the system becomes more frequent. End users are never happy with the navigation on smartphones (Huang *et al*., 2006).

Several studies attempted to address this problem by different ways: accelerating the hierarchical navigation, accelerating menu item selection (Cockbrun & Gin, 2006), optimising the overall menu performance (Bailly *et al*., 2013; Matsui & Yamada, 2008), recommending predicted menu items to users (Shin *et al*., 2012) or facilitating the learning of the menu structure (Park *et al*., 2007).

© The Authors. Published by BCS Learning and Development Ltd.
Proceedings British HCI 2016 - Fusion, Bournemouth, UK

This paper presents an alternate method to this problem: instead of adapting the menu itself by some method, which affects menu stability, a *prediction window* with *predicted items* (e.g., the most frequently selected items, the most critical, important for the task) is overlaid on the initial menu letting the user to rely on it if desired, thus accelerating the hierarchy traversal, or to abandon it if she prefers to stay with the menu structure.

For this purpose, two adaptive menus are compared: a “Step by Step” adaptation aimed at facilitating hierarchical navigation and helping the end user to learn the hierarchical structure and a “Shortcut” adaptation aimed at supporting end users to reach directly a desired item and bringing her back to the root efficiently. This adaptation does not support the user in learning the hierarchy as the first does.

The remainder of this paper is structured as follows: Section 2 reviews significant progress on menu interaction techniques, particularly with respect to hierarchical and adaptive menus, Section 3 discusses how the two adaptive menus for hierarchical navigation were motivated, designed, and developed, Section 4 reports on the experiment testing their overall performance, Section 5 concludes the paper by summarizing the benefits of these two techniques, by discussing their limitations, and by presenting some future avenues to this work.

2. BACKGROUND LITERATURE

A significant amount of menu interaction techniques has been researched in the literature, as reported in MenUA (Bailly *et al*., 2009), the most extensive review we are aware of in this domain. Therefore, we only discuss selected techniques related to our problem of supporting both novice and expert users.

Hesselmann *et al*. (2009) developed *stacked half-pie menus* displaying menu items as circles in half pie on a tabletop surface. This interaction technique tends to make this design unlimited in terms of menu depth and breadth while still maintaining the form of the menu. This menu is limited for small screen devices like smartphones where there is not enough space on the screen. In addition, the navigation in the pie menu may be a constraint for novice users.

PocketMenu (Pielot *et al.*, 2012) exploited the idea of changing the modality for menu selection: menu items are laid out along the border of the touch smartphone within the hand comfort zone, tactile features guide the hierarchical navigation, a vibrotactile feedback with speech allows identifying the items non-visually. This interaction technique is particularly useful for end users with visual disabilities.

MenuDesigner (Vanderdonckt, 1999) is aimed at automatically generating a menu bar, associated cascading menus and menus items based on an activity chaining graph representing possible hierarchical navigation based on a task model. This approach remains static (the menu structure is generated once for all), without any adaptation and could lead to inconsistent menus when items are arranged.

MenuOptimizer (Bailly *et al*., 2013) is aimed at helping designers and developers to optimize the menu structure by maximizing consistency vs performance based on ant colony algorithm. While MenuOptimizer reveals the popularity of menu items by a colour line under each menu item, thus leaving the menu structure untouched, it does not provide end users with an adaptive menu. Matsui & Yamada (2008) relied on a genetic algorithm to generate a menu structure that is optimized for its usage.

Ahlström *et al*. (2010) performed a large analysis of several research studies to propose the Search, Decision, and Pointing (SDP) model. They suggest *Square Menus* for improving pointing performance, especially for experts. It was shown as a promising solution compared to traditional linear menus and to pie menus. It reduces Fitts’ Law pointing time for experts and novice users performed better with traditional menus and even worse than with pie menus.

Flower menus (Bailly *et al*., 2008) extended *Marking menus* with opportunity to draw straight lines or curved ones. A comparative study of flower, linear and polygon menus shows that polygon and flower menus offer better performance for learning the expert mode as compared to linear menus. But as for previous techniques, flower menu do not accommodate well with screen real estate.

Cockburn & Gin (2006) made manipulation easier with *enlarged activation areas* in cascading menus. The objective was to eliminate the explicit delay for activation found in several implementations. Study shows that enlarged activation area and zero delays improve cascade-item selection by up to 29% in comparison to traditional methods. However, cascading menus are not an easy way to deal with hierarchies on tiny screens, they need motor abilities.

Bubbling menus (Tsandilas & Schraefel, 2007) is a design for cascading drop-down menus aimed at accelerating the selection of the frequently used items by directly jumping to them one by one.

Fish-eye menus (Bederson, 2000) display items with a font size that increases or decreases depending on the distance with respect to cursor position: the closer, the larger, the further, the smaller.

Hyperbolic menus (Lamping *et al*., 1995) is a Focus + Context technique for displaying and manipulating large hierarchies. It displays several hierarchy levels at once according to a hyperbolic view. Since parts of the hyperbolic view are expanding and collapsing depending on the position of the cursor, menu items are never displayed at the same place, thus inducing spatial instability, which is difficult to use for novice users and without fine visio-motor coordination.

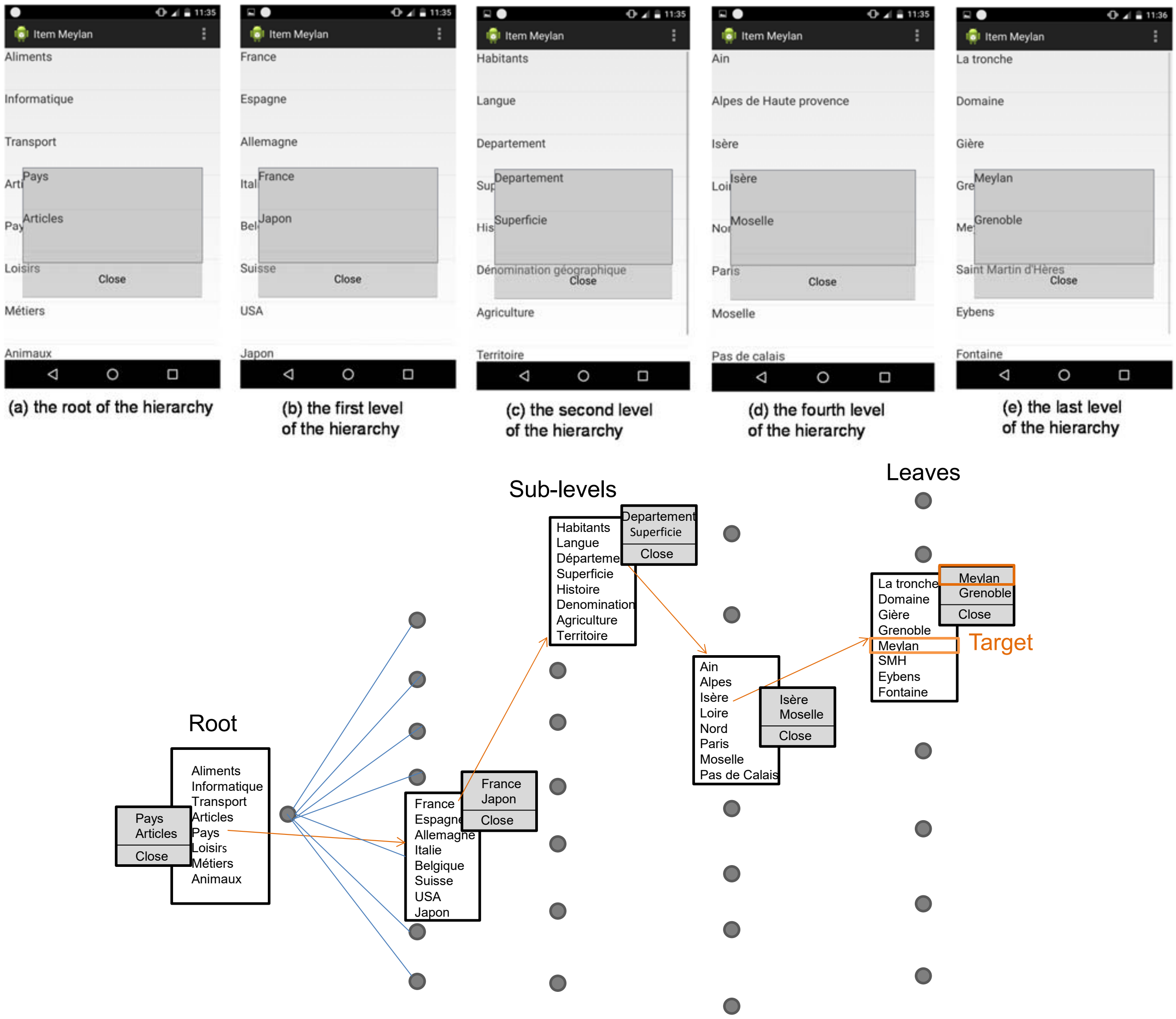


***Figure 1**: The Step by Step design.*

Space Tree (Plaisant *et al.*, 2002) consists in a tree browser that dynamically rescales tree branches to accommodate the constraints imposed by the screen resolution. It also adds icon preview to summarize the topology of the collapsed branches. This design was compared to Microsoft Explorer and to the hyperbolic browser (Lamping *et al.*, 1995): the space tree is significantly faster than hyperbolic one, Explorer is significantly faster than the two others, especially when the first task involves learning. For local topology (the example of finding 3 nodes that have more than 10 direct descendants), Hyperbolic was significantly faster than the space tree but not significantly faster than Explorer. In terms of user preferences, users significantly found Explorer less “cool” than the other ones and no significant difference was found between space tree and hyperbolic. There were no significant differences between interfaces in term of preference for future usage.

Tree Maps (Johnson & Shneiderman, 1991) display a hierarchy as a nested collection of rectangular bounding boxes representing the tree structure. Each bounding box could be displayed with a coding scheme (e.g., by size, by colour, by texture) depending on some parameters (e.g., importance, size).

Multi-layers interfaces (Kang *et al.*, 2003) are hierarchical interfaces that evolve over time, such as *Training wheels* (Catrambone & Carrol, 1987). Shneiderman (2003) defined a multi-layer UI where the initial layer contains only basic menu items. Reduced menu items may be beneficial to novice users, but the whole set of menu items is lots in the translation, which is not appreciated by expert users.

Mixed layers (Clark & Matthews, 2005) are aimed at simplifying UI usage by displaying only the basic options, such as basic menu items. When the user acquires adequate accuracy, the adaptive UI fulfils missing items. New items are only displayed when the application is launched, thus avoiding disrupting users by adding features at runtime, but not between interactive sessions. Adaptivity (the system adapts menu items) is combined with adaptability (the user is also enabled to modify menu items at run-time).

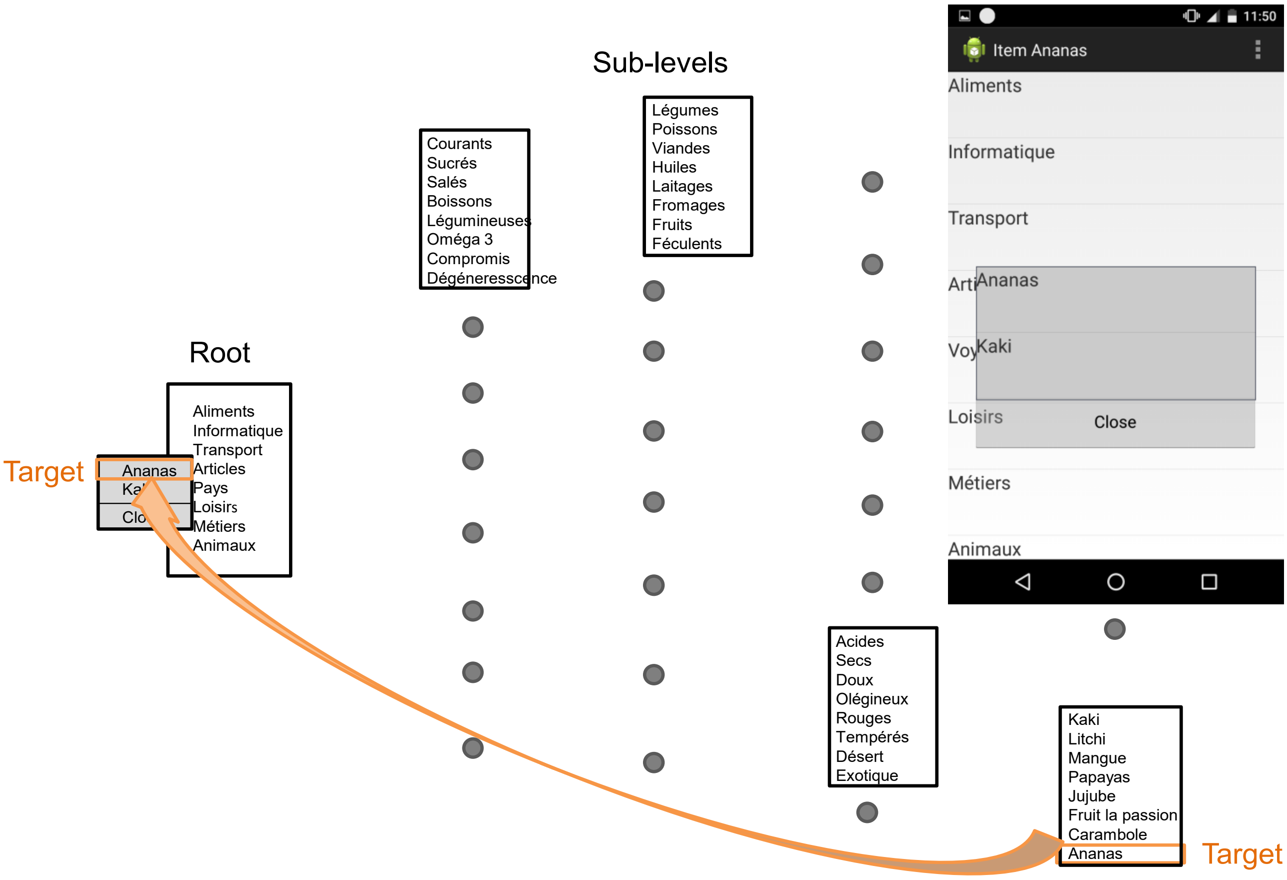


***Figure 2**: The Shortcut design.*

Adaptivity not only has shortcomings, e.g., user disruption, cognitive load, but also offers clear benefits, such as user subjective satisfaction and task performance (Lavie & Meyer, 2010). To sum up, cascading menus, pie menus, flower menus, square menus, hyperbolic menus, space tree, explorer, and Tree Maps are mainly focusing on hierarchy navigation. Bubbling menu, multi-layers and mixed layers introduce action prediction, but do not address the trade-off between usable adaptivity and false predictions: inadequate prediction may lead to user error and frustration as soon as interaction is endangered.

Adaptive techniques should accommodate simultaneously large hierarchies, tiny screens, minor visiomotor accuracies, learning capacities, good and wrong prediction, and both rapid, easy-to-use and secure navigation inside the menu hierarchy, which represents a significant set of constraints to satisfy. Previous studies show that prediction displayed through a modal window seems a good candidate to address these challenges (Bridle & McCreath, 2005), except perhaps for learning opportunities.

Huang *et al*. (2006) suggest that end users prefer a less extensive menu structure on a small screen device. Their investigation also covered factors of category classification and item labelling influencing user performance in menu selection. Research findings suggest that proper modifications in these areas could significantly enhance the usability.

Ephemeral adaptation (Findlater *et al*., 2009) is aimed at preserving the menu stability when presenting predicted menu items: instead of changing the menu, predicted items are first displayed until the full menu arrives. This has also the advantage of not producing any alternate menu since the predicted items are subject to an animated transition that preserves the context of use. After a time out of approximately 200 msec, the full menu appears as normal.

Adaptivity Animated transitions have also been successfully used to convey to the end user how a UI adaptivity has been conducted (Dessart *et al*., 2011): each adaptation operation performed on a GUI is captured, scripted and could be played or replayed at the end user's pace, thus providing some visual explanation of the adaptation. The major drawback was the lack of animation control: not all steps should be animated equally to understand.

Appearing and disappearing ephemeral adaptation (Bouzit *et al*., 2014) and subsequently *Evanescent adaptation* (Bouzit *et al*., 2015) are extensions of ephemeral adaptations where different animated transitions are ensured to convey predicted items to the end users with different visual effects. These various interaction techniques are primarily targeting hierarchical menus for stand-alone applications, and not specifically for supporting adaptivity on mobile phones, which is different from UI adaptivity for a desktop platform (Arhippainen *et al*., 2008).

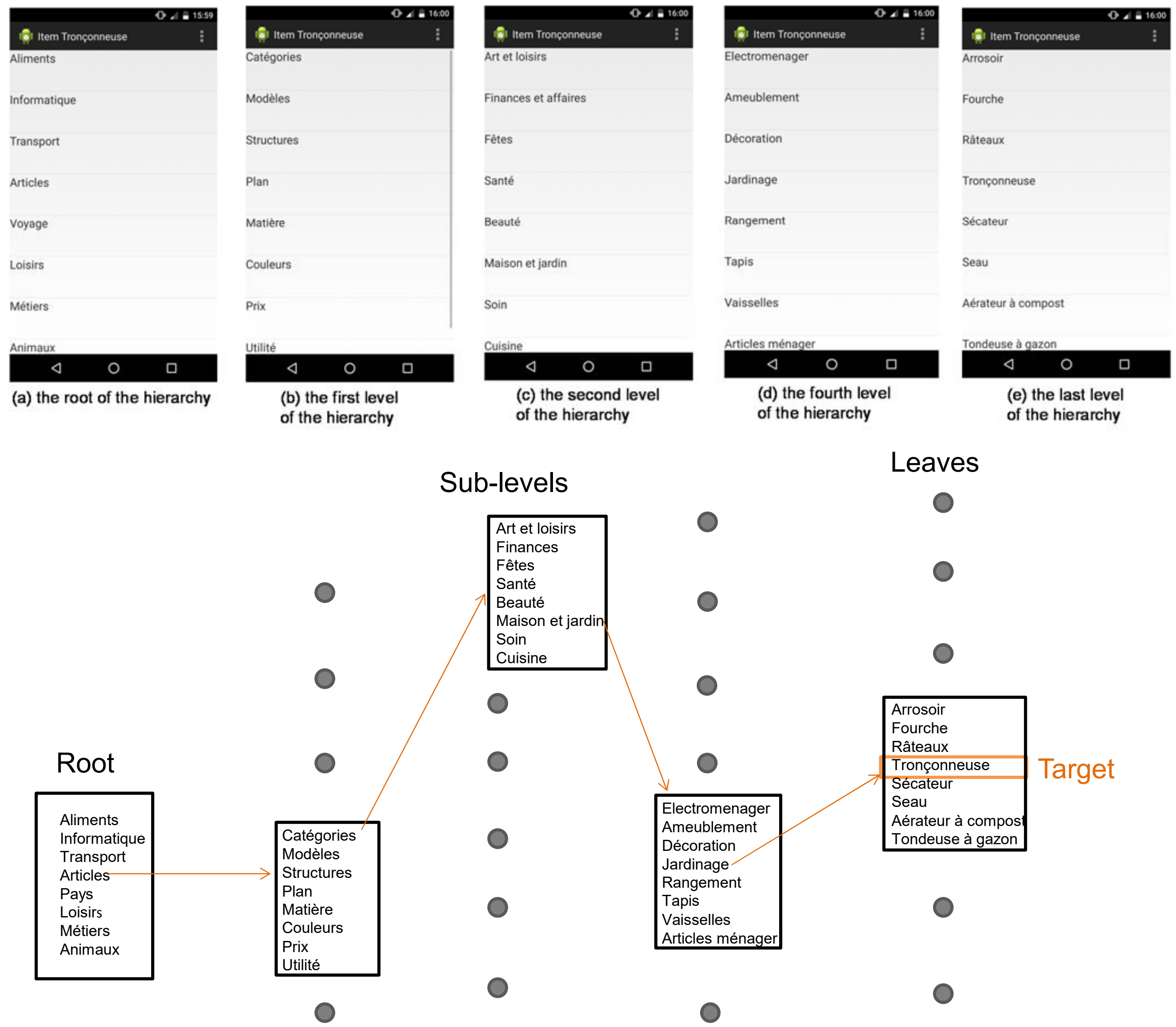


Figure 3*: The Control condition.*

For all the aforementioned reasons, we decided in this paper to investigate whether adaptivity of menu items could be effectively and efficiently supported on mobile phones and smartphones by designing, developing, and testing two adaptive menus that also preserve the menu stability, while offering the access to far away items in the hierarchy, and not just the first level.

3. DESIGN OF SHORTCUT, STEP BY STEP AND CONTROL PROTOTYPES

The first prototype, called “Step by Step” (Figure 1), displays a constant homogeneous hierarchy. At any level in the hierarchy, the most likely items to be searched by the end user is shown in a *predictive window*, which is superimposed by alpha-blending (Baudisch & Cutwin, 2004). This window enables the end user to access easily the next level of hierarchy in order to reach the final target. When the menu opens, the end user is located at the root of the hierarchy. A prediction window is available, containing the most probable items for reaching the level 1 of the hierarchy (Figure 1a). When the user makes a good selection, she goes to the next level (1) in the hierarchy in which she can also find another subsequent prediction window. This window becomes a superimposed menu containing the most probable items enabling user access to level 2 of the hierarchy (Figure 1b), and so on (Figure 1c, 1d) until a leaf in the hierarchy is reached where the last prediction window contains the user target (Figure 1e). Predictions are made only in local of each hierarchy level. Prediction is computed only on the current list of items belonging to each level. The prediction does not consider information extracted from other levels. The aim of “Step By Step” prototype is to help user in creating a mental model of the tree structure. Successive predictions are correlated to the effective paths to navigate through all the hierarchical levels. Each predictive window displays the most probable item from the current list that enable the end user to reach the next level which is on the way of the final target.

It should help user to learn the structure as he is going through each level, but it should also be efficient as, in case of accurate prediction the manipulation to operate is always the same: the focused item is always under the finger and action can be executed on repetitive basis. The prediction window can be controlled by the user who can make it disappearing (Bouzit *et al.*, 2015), especially when prediction is wrong, inaccurate, or fuzzy. The next prediction window is not displayed, otherwise the end user would have to spend extraneous time in closing each prediction window having false prediction at each level.

The second prototype is aimed at accelerating interaction with the use of a shortcut to the final target as soon as possible. This gave rise to "Shortcut" design (Figure 2), in which the predicted target is presented at the root of the hierarchy. In this case, predicted items do not come only from the current level but from all levels of the hierarchy. At the root level, the end user finds a prediction window containing the item supposed to be the desired target considered as a leaf of the hierarchy, thus making it very straightforward: in one operation (e.g., a click if the user is using a mouse or a touch event in case of a touch smartphone), the end user is reaching the target. A negative effect is that shortcutting does not help the end user in creating a complete mental model of the hierarchy since sub-levels are shortcut.

For the purpose of the experiment, the two prototypes are completed by a third one, dedicated to the Control condition. This hierarchy is traversed by normal menu structure, without any adaptation, without any prediction window or any visual effect. Only the nested lists of items are available (Figure 3).

4. EXPERIMENTAL STUDY

The objective of the experiment is to compare two different adaptations of hierarchical menus on small screen devices (smartphones). A reasonable compromise between accelerating the interaction and helping user learning the interface structure has to be found. For this purpose, three applications were developed in Java for Android on a smartphone that correspond to the Step by Step (Fig. 1), Shortcut (Fig. 2), and Control condition (3), each with 5 levels of hierarchies. The experimental plan is the following: 13 Subjects *S* x 3 interface types *I* × 2 Prediction accuracy levels P (low vs high) = 78 configurations.

4.1 Method

There are two independent factors: the interface type (I), which is a within-subject factor:

1) *Control menu*. In this Control condition, there is no help for the user who must find the target, navigating through the hierarchical structure.
2) *Step by Step Menu*. In the Step by Step condition each level of the hierarchy displays the prediction window and offers to select the menu item leading to the next level of the target path.
3) *Shortcut Menu*. In the Shortcut condition, the final target is displayed in a prediction window at the root level of the hierarchy.

The second factor is the prediction accuracy level (P). High prediction accuracy level corresponds to *correct prediction* (the selected menu item belongs to the prediction window), while low prediction accuracy level corresponds to *incorrect prediction* (the selected menu item does not belong to the prediction window). In both cases of correct and incorrect predictions, the target item is always in the complete list of items. I and P orders were selected randomly.

4.2 Hypotheses

The assumptions made for this experiment depend on the level of prediction accuracy:

H1. Speed

- For a high prediction accuracy level

Minor differences between Shortcut condition and Step By Step condition. When prediction is correct in Step By step condition, user will produce 5 consecutive selection in the prediction window to reach the target. These repeated actions should not constrain the end user in any way.

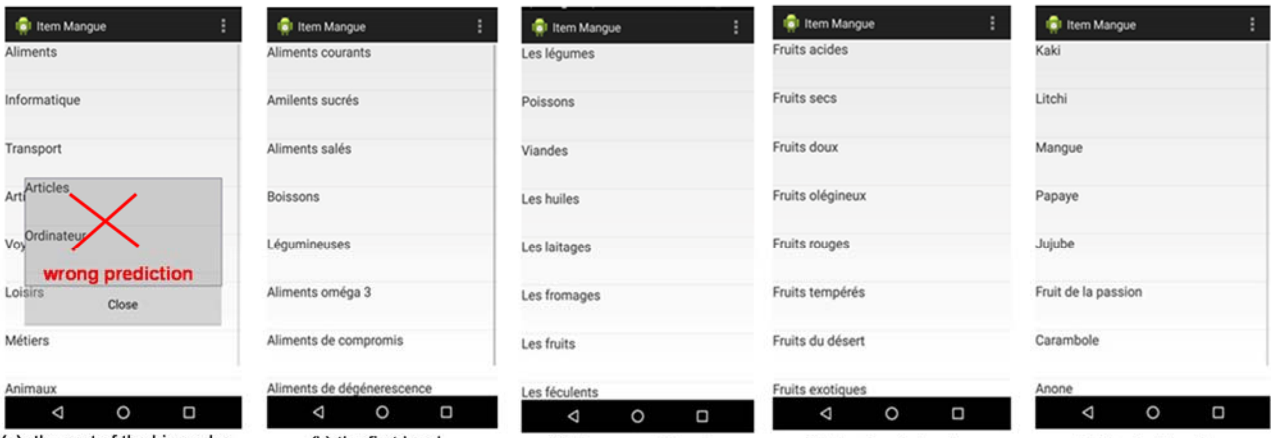
Step by step condition will be faster than control condition. In Step by Step condition, the target is highlighted in each level which may help the user to achieve the final target faster than in the Control condition.

Shortcut condition will be faster than Control condition. In this case, the target to find is presented at the root of the hierarchy which enables the end user to reach the target quickly. As opposed to Control condition, the user must traverse the hierarchy and could hesitate at each level to find the target.

- For low prediction accuracy level

Step by Step condition will be faster than the Shortcut and close to Control condition. When prediction is wrong, the user will be confused and lost in Shortcut condition because this mechanism accelerates the user interaction but does not allow her to learn the hierarchical structure. In the Step by Step condition, the end user is progressively guided by the system and learns the structure. This may help her in reaching the target in case of wrong prediction. In Control condition, the user should have learned the structure during prior interaction.

Control condition will be better than Shortcut condition. In the Control condition, the end user explores the hierarchy which may enable her to create a mental model of the hierarchical structure which is not the case in the Shortcut condition.



(a) the root of the hierarchy: the prediction is incorrect
(b) the first level of the hierarchy
(c) the second level of the hierarchy
(d) the fourth level of the hierarchy
(e) the last level of the hierarchy

Figure 4: *The Step by Step prototype principle: when prediction at the root of the hierarchy is incorrect, the system does not continue presenting incorrect predictions in each level. The prediction is presented when the user starts a novel task.*

H2. Errors

- For a high prediction accuracy level

Errors will be less frequent in Shortcut condition than in Step by Step and Control condition. When prediction is correct, the target is presented in the first interaction stage in Shortcut condition which may reduce errors compared to the Step by Step and Control conditions where the end user has to make more actions to reach the desired item.

Errors will be less frequent in Step by Step condition than in Control condition. In the Step by Step condition, the target to select is highlighted in each level of the hierarchy, which may reduce the errors compared to the control condition having no adaptation.

- For a low prediction accuracy level

Errors will be less frequent in Step by Step condition and Control condition than in the Shortcut. The Step by Step condition and the Control condition should be in principle more appropriate to find the target item in the hierarchical structure since they support some form of structure awareness while interacting.

4.3 Participants

Thirteen volunteers (men) participated to this experiment. They were all recruited from a mailing list in our organisations. They are aged between 22 and 58 (μ=31.5), all being regular users of smartphones.

4.4 Task

Users has to perform a sequence of selections. For each condition, i.e., Control, Step by Step and Shortcut, a message appears on the screen prompting the end user to select a target menu item in the hierarchy. Then the list of items appears. In Shortcut condition, the target is presented in the prediction window at the root of the hierarchy in case of correct prediction. When prediction is wrong, the user can close the prediction window and goes through the hierarchy in order to reach the target, which is a leaf node in the hierarchy. In Step by Step condition and when prediction is correct, the best candidate of the current list that leads to the final target is displayed in the prediction window. When prediction is wrong, the user can close the predictive window at the root and searches for the correct item in the complete list. In the next level of the hierarchy, the prediction window is not displayed and the user has to navigate in the same way as for Shortcut or Control conditions (Figure 4). In the Control condition, the end user has to go through the hierarchy to reach the target item which is a leaf node (5^{th} level of the hierarchy). The target item remains displayed at the top of the screen as a reminder. User selects target from the prediction window or from the full list of items. If selection is incorrect, an error message is displayed that invites the end user to find a requested target for moving to a new selection. When the user succeeds in selecting the right item, she moves to next level of the hierarchy until reaching final target. A new message then appears that specifies the name of the new target item to select. At the end of the test, a thank you message is displayed informing the user that the test is complete.

4.5 Quantitative and qualitative measures

Two dependent variables were measured. The first dependent variable is speed measured as correct menu item selection time. It is the time elapsed from opening the menu until a correct selection of the target menu item is obtained. For example, if the user has to go through 5 levels to reach the target, the time selection is calculated from opening the menu until arrival to the final level of the hierarchy. The speed is measured in seconds. The second dependent variable is the task completion. Error rates were recorded each time a deviation from the correct menu item is detected.

4.6 Apparatus

The materials used in this experiment were off-the-shelf Android smartphones. Experimental data were recorded in a database selection time (in seconds), scrolling time (in milliseconds) and error rate.

4.7 Procedure

Before starting the test, the principle of each condition was explained to each participant without mentioning the two prediction accuracy levels (high and low). Short training test was provided and each participant was able to complete ten menu item selections. The hierarchical lists used in this training test were different from those used in the test conditions. The requested targets were different in each condition. In adaptive conditions (Shortcut and Step By Step), participants were asked to find 10 targets when prediction is correct, then the same 10 targets with wrong predictions. An entire test for each participant is thus composed of 50 targets as follows:

- 20 targets for Step by Step condition divided in two parts: 10 targets when prediction is correct and 10 targets when prediction is wrong. In both cases, each participant performs 5 selections for achieving one final target because it is always a leaf node in the menu hierarchy.
- 20 targets for Shortcut condition divided also in two parts: 10 targets when prediction is correct and 10 targets when prediction is wrong. In this last case, the participant has to go down 5 levels in the hierarchy to achieve one target. In these two adaptive conditions, each target is requested twice. The first time in the case of correct prediction and the second is when the prediction is wrong. The goal is to test whether the user tends to create a mental model of the hierarchy when prediction is presented step by step, which could facilitate the task completion in the case of wrong prediction compared to the Shortcut method.
- 10 targets for control condition, the participant performs 5 selections for reaching each target that is always in the last level of the hierarchy (leaf).

The total amount of samples is therefore: 78 configurations x 50 targets = 3,900 samples.

4.8 Results

Levene's test and Brown-Forsythe's test were applied for testing the homogeneity of variance. It could not be distinguished, so non-parametric Friedman's ANOVA by Ranks and Wilcoxon Signed Ranks tests were applied for data analysis.

Speed

Obtained results show that when prediction is correct, Step by Step and Shortcut conditions are significantly faster than Control condition (Step by Step: M = 6.42, SD = 2.81, Control: M = 14.73, SD = 4.11, $W(24) = -3.18$, $p < 0.005$) – (Shortcut: M = 1.77, SD = 1.47, Control M = 14.73, SD = 4.11, $W(24) = -3.18$, $p < 0.005$). When prediction is correct, Shortcut is significantly faster ($W(24) = -3.18$, $p < 0.005$) than Step by Step condition (Shortcut M = 1.77, SD = 1.47, Step by Step M = 6.42, SD = 2.81).

However, when prediction is wrong, Step by Step condition is significantly faster ($W(24) = 2.13$, $p < 0.05$) than Shortcut condition (Step By Step: M = 14.51, SD = 3.67, Shortcut: M = 20.94, SD = 8.06). When prediction is wrong, there is no significant difference ($W(24) = -0.14$, $p > 0.5$) between Step by Step condition and Control condition (Step By Step: M = 14.51, SD = 3.67, Control: M = 14.73, SD = 4.11). Moreover, Control condition is significantly faster ($W(24) = 2.34$, $p < 0.05$) than Shortcut condition when prediction is wrong (Control: M = 14.73, SD = 4.11, Shortcut: M = 20.94, SD = 8.06).

As expected, in both Step by Step and Shortcut adaptive condition, participants were significantly faster when prediction is correct than when prediction is wrong (Step by Step with correct prediction: M = 6.42, SD = 2.81, Step by Step with wrong prediction: M = 14.51, SD = 3.67), $W(24) = 3.18$, $p < 0.005$) – (Shortcut with correct prediction: M = 1.77, SD = 1.47, Shortcut with wrong prediction: M = 20.94, SD = 8.06), $W(24) = 3.19$, $p < 0.005$).

Overall, Step by Step condition is faster than Control condition, which in turn is also revealed as faster than Shortcut condition (Step by Step: M = 10.48, SD = 5.24, Control: M = 14.73, SD = 4.11, Shortcut: M = 17.1, SD = 32.39). Speed time results are graphically depicted in Figure 5. Note that the Shortcut bar suffers from a high deviation with respect to the average.

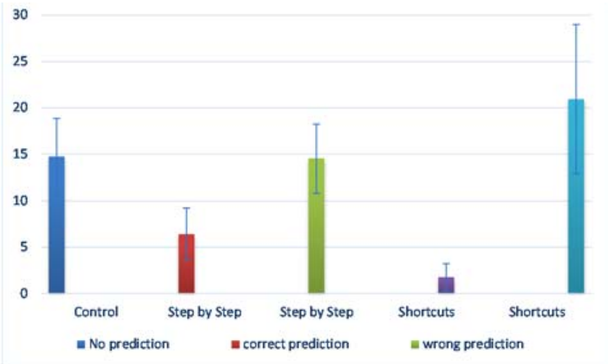


Figure 5: *Selection time for all conditions*

Errors

In the case of correct prediction in both Step by Step and Shortcut conditions, errors were recorded significantly less frequent than when prediction is wrong (Step by Step with correct prediction: M = 0.23, SD = 0.6, Step by Step with wrong prediction: M = 4.46, SD = 2.93), $W(24) = -3.20$, $p < 0.005$) – (Shortcut with correct prediction: M = 0, SD = 0, Shortcut with wrong prediction: M = 13.46, SD = 15.99), $W(24) = 3.19$, $p < 0.005$). There is no significant difference in errors ($W(24) = 1.34$, $p > 0.1$) between Step by Step and Shortcut condition when prediction is correct (Step by Step: M = 0.23, SD =

0.6, Shortcut: M = 0, SD = 0). The amount of errors was rather similar. Nevertheless, for both conditions, errors are systematically less frequent than in Control condition (Step by Step: M = 0.23, SD = 0.6, Control: M = 4.31, SD = 4.37), $W(24) = -2.83$, $p < 0.005$) – (Shortcut: M = 0, SD = 0, Control M = 4.31, SD = 4.37), $W(24) = -2.94$, $p < 0.005$).

But when prediction is wrong, errors are significantly ($W(24) = -2.34$, $p < 0.5$) less frequent in Step by Step than in Shortcut condition (Step by Step: M = 4.46, SD = 2.93, Shortcut: M = 13.46, SD = 15.99). There is no significant difference between Step by Step condition when prediction is wrong and Control condition (Step by Step: M = 4.46, Control: M = 4.31, SD = 4.37), $W(24) = 0.07$, $p > 0.5$). However, when prediction is wrong in Shortcut condition, error rate is significantly ($W(24) = 1.85$, $p > 0.5$) higher (M = 13.46, SD = 15.99) than in Control condition (M = 4.31, SD = 4.37). Error rate results are graphically depicted in Figure 6. Note that the Shortcut bar is null: there is no error since, when the predicted item is presented at the root of the prediction window, there is no need to go further in the hierarchy.

Figure 6: *Error rate for all conditions*

4.9 Discussion

H1. Speed

Table 1: *Pairwise comparisons for speed (P+ means Correct Prediction and P- means Incorrect Prediction)*

Menu	Mean	Standard Deviation	Measure	Value
Control Step By Step (P+)	14.73 6.42	4.12 2.82	-3.18	0.001
Control Shortcut (P+)	14.73 1.77	4.12 1.47	-3.18	0.001
Step By Step (P+) Shortcut (P+)	6.42 1.77	2.81 1.47	-3.18	0.001
Control Step By Step (P-)	14.73 14.51	4.12 3.67	-0.14	0.888
Control Shortcut (P-)	14.73 20.94	4.12 8.06	-2.34	0.019
Step By Step (P-) Shortcut (P-)	14.51 20.94	3.67 8.06	-3.18	0.001

- For a high accuracy prediction level (P+)

No significant difference between Shortcut condition and Step By Step condition. Not supported.
When prediction is correct, participants are faster in Shortcut condition because the target is presented in the first interaction stage. In the Step by Step condition, participants were guided until reaching the target, and results showed that it takes more time even when such actions are repeated, thus suggesting that no learning effect was observed.

Step by Step condition will be faster than Control condition. Supported.
Helping participants to reach a target item in the hierarchy by guiding them throughout each hierarchy level seems to be a real benefit for the end user and contributes to reduce searching time. It was expected and is confirmed for Shortcut condition, but it was not obvious for Step by Step condition. In case of high accuracy prediction level, Step by Step is not as efficient as Shortcut was observed, but it is really faster than no prediction at all (Control). Step by Step boosts performance in case of high accuracy level prediction. But boosting performance may be considered not enough. Let us look at what occurs in case of prediction inaccuracy, where awareness about the hierarchy could introduce a difference.

- For a low accuracy prediction level (P-)

Step by Step condition will be faster than the Shortcut and close to Control condition. Supported.
The guidance of participants in Step by Step condition enables them to create a mental model of the hierarchical structure. Hence, compared to Shortcut condition when prediction is wrong, participants are not lost and are able to reach the target. Similarly, participants are lost when prediction is wrong because they had only access to parts of the structure. They were directly jumping from the root to the target item, which is a leaf node in the hierarchy. In Shortcut condition, finding the right menu item requires a long time and induce many errors. Step By Step and Control conditions seem to be equivalent in case of wrong prediction.

Control condition will be better than Shortcut condition. Supported.
In Control condition, participants have to explore the hierarchy for reaching the target because there is no adaptation mechanism supporting this activity. As for Step by Step condition, this helps them to understand and to learn progressively the hierarchical structure. In terms of efficiency, Step by Step seems to be a good compromise as it is faster than Control in case of correct prediction and not worse than Control in case of wrong prediction, contrarily to Shortcut that dramatically cripples interaction.

Results regarding speed are summarised in Table 1.

H2. Errors

Table 2: *Pairwise comparisons for errors (P+ means Correct Prediction and P- means Incorrect Prediction)*

Menu	Menu	Standard Deviation	Measure	Value
Control Step By Step (P+)	4.31 0.23	4.37 0.56	-2.83	0.004
Control Shortcut (P+)	4.31 0	4.37 0	2.94	0.003
Step By Step (P+) Shortcut (P+)	0.23 0	0.56 0	1.34	0.179
Control Step By Step (P-)	4.31 4.46	4.37 2.93	0.07	0.944
Control Shortcut(P-)	4.31 13.46	4.37 15.99	1.85	0.064
Step By Step (P-) Shortcut (P-)	4.46 13.46	2.93 15.99	-2.34	0.018

Results regarding errors are summarized in Table 2.

- For a high prediction accuracy level (P+)

Errors will be less frequent in Shortcut condition than in Step by Step and Control condition. Partially supported.
There was no significant difference between these conditions in the case of correct prediction because in both cases, i.e., Shortcut and Step by Step, target items are easily identified by participants. Nevertheless, errors are less frequently recorded than in Control condition.

Errors will be less frequent in Step by Step condition than in control condition. Supported.
Highlighting the target item in each level of the hierarchy in Step by step condition reduces the number of errors compared to Control condition. For Shortcut and Step by Step, errors are in minor quantities while they are numerous in Control condition. This confirms the interest of a predictive interaction method. As for speed, a low amount of errors is a potential gain of such an adaptive menu.

- For a low prediction accuracy level (P-)

Errors will be less frequent in Step by Step and Control conditions than in the Shortcut condition. Supported.
When prediction is wrong in Step by Step and Control conditions, errors are less frequent because participants had time previously to explore the hierarchical structure and probably better know where to go. Conversely, in the Shortcut condition, participants experienced trouble in find the target item and produce many errors that prevent them to efficiently reach the target item.

Overall, the Step By Step condition makes a good compromise in terms of both speed and errors. It also represents a good compromise between the two cases: correct and incorrect prediction. The Step By Step adaptation technique applied in the context of hierarchies helps users to create a mental model of the interface. In this way, it enables them to reach the target quickly while avoiding selection errors. But in this experiment, there were only 5 levels in the hierarchy. Therefore, these results may not be similar when the menu hierarchy has a more profound depth than five. Generalization of these results should therefore be the subject of further experiments exploring the effect of menu depth. Shortcut adaptation technique is interesting in the case of correct prediction because it reduces time and errors. But when prediction is wrong, this technique is really penalizing the end user. Therefore, the Shortcut could be recommended as a guideline in two cases: 1) when the accuracy of prediction algorithm is very high (for example, up to 80%) and the prediction provided is supposed be acceptable most of the time. 2) when the user has achieved a reasonable level of experience and she already knows the hierarchical structure. Even when the prediction is wrong, the user is not penalized and user interaction is not slowed down.

5. CONCLUSION

This paper reported on the results of an experiment comparing two graphical adaptive menus: “Step by Step” and “Shortcut” preserve the presentation of the initial menu and overlay it with an adaptive menu rendering the most frequently used menu items, wherever they are located in the hierarchy. This prediction window is superimposed on top of the original menu by semi-transparency with alpha blending. The Step by Step Menu enables end users to create a mental model of the hierarchical structure by guiding them in each step of their interaction. The Shortcut Menu allows reducing navigation time as well as the visual search time. The experiment suggests that Step by Step Menu is a promising adaptive menu since making a good compromise between both cases, when prediction is correct and when prediction is wrong.

Future work may consider multiblending (Baudish & Cutwin, 2005) instead of alpha-blending for displaying the prediction window: the trade-off of alpha blended adaptive menus is that increasing opacity to perform a better menu selection on the foreground necessarily reduce the performance on the background task. This suggests to conduct another experiment investigating the effect of prediction window presentation on the preference and the performance of menus by participants.

5. ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their constructive feedback on an earlier version of this paper and Grenoble INP for supporting this collaboration.

6. REFERENCES

Ahlström, D., A. Cockburn, C. Gutwin, and P. Irani (2010). Why it's quick to be square: modelling new and existing hierarchical menu designs. *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems CHI'2010* (Atlanta, April 10-15, 2010). ACM Press, New York, pp. 1371–1380.

Ahlström, D. (2006). Modeling and improving selection in cascading pull-down menus using Fitt's law, the steering law and force fields. *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems CHI'2006* (Portland, April 2-7, 2006). ACM Press, New York, pp. 61–70.

Arhippainen, L., T. Rantakokko, and R. Tähti, M. (2004). Navigation with an adaptive mobile map-application: user experiences of gesture- and context-sensitiveness. *Proc. of 2nd Int. Symposium on Ubiquitous Computing Systems UCS'2005* (Tokyo, Nov. 8-9, 2004). Lecture Notes in Computer Science, vol. 3598. Springer, Berlin, pp. 62–73.

Bailly, G., E. Lecolinet, L. Nigay, and C. Lebiere (2008). Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. *Proc. of the ACM Working Conf. on Advanced Visual Interfaces AVI'2008* (Napoli, May 28-30, 2008). ACM Press, New York, pp. 15–22.

Bailly, G., E. Lecolinet, and L. Nigay (2009). MenUA: A Design Space of Menu Techniques. Accessible at www.gillesbailly.fr/menua/

Bailly, G., Oulasvirta, A., Kötzing, T., and Hoppe, S. (2013). *MenuOptimizer: interactive optimization of menu systems. Proc. of the 26th ACM Symposium on User interface software and technology UIST'2013* (St Andrews, October 8-11, 2013). ACM Press, New York, pp. 331–342.

Baudish, P. and C. Cutwin (2004). Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending. *Proc. of ACM SIGCHI Conf. on Human Factors in Computing Systems CHI'2004* (Vienna, April 24-29, 2004). ACM Press, New York, pp. 367–374.

Bederson, B.B. (2000). Fisheye menus. *Proc. of 13th ACM Symposium on User Interface Software and Technology UIST'2000* (San Diego, November 6-8, 2000). ACM Press, New York, pp. 217–225.

Bouzit, S., D. Chêne, and G. Calvary (2014). From Appearing to Disappearing Ephemeral Adaptation for Small Screens. *Proc. of the 26th Australian Computer-Human Interaction Conference OzCHI'2014* (Sydney, December 2-5, 2014). ACM Press, New York, pp. 41–48.

Bouzit, S., D. Chêne, and G. Calvary (2015). Evanescent Adaptation on Small Screens. *Proc. of the 27th Australian Computer-Human Interaction Conference OzCHI'2015* (Parkville, December 7-10, 2015). ACM Press, New York, pp. 62–68.

Bridle, R. and E. McCreath (2005). Predictive Menu Selection on a Mobile Phone. *Proc. of Workshop W7 on mining spatio-temporal data*, ECML/PKDD, Portugal.

Catrambone, R., and J.M. Carrol (1987). Learning a word processing system with training wheels and guided exploration. *Proc. of the ACM SIGCHI/GI Conf. on Human Factors in computing systems and graphics interface CHI/GI'87* (Toronto, April 5-9, 1987). ACM Press, New York, pp. 169–174.

Clark, B., and J. Matthews (2005, July). Deciding layers: adaptive composition of layers in a multi-layer user interface. *Proc. of 11th Int. Conf. on Human-Computer Interaction HCI International'2005* (Las Vegas, July 22-27, 2005). C. Stephanidis (Ed.), Volume 7. Mira Digital Publishing.

Cockburn, A. and A. Gin (2006). A faster cascading menu selection with enlarged activation areas. *Proc. of Graphics Interface GI'2006* (Quebec, June 7–9, 2006). Canadian Information Processing Society, Toronto, pp. 65–71.

Dessart, C.-E., V. Genaro Motti, and J. Vanderdonckt (2011). Showing user interface adaptivity by animated transitions. *Proc. of ACM Conf. on Engineering Interactive Computing Systems EICS'2011* (Pisa, June 13-16, 2011). ACM Press, New York, pp. 95–104.

Findlater, L., K. Moffatt, J. McGrenere, and J. Dawson (2009). Ephemeral Adaptation: The Use of Gradual Onset to Improve Menu Selection Performance. *Proc. of ACM SIGCHI Conf. on Human Aspects in Computing Systems CHI'2009* (Boston, April 4-9, 2009). ACM Press, New York, pp. 1655–1664.

Hesselmann, T., S. Flöring, and M. Schmitt (2009). Stacked half-pie menus: navigating nested menus on interactive tabletops. *Proc. of the ACM Int. Conf. on Interactive Tabletops and Surfaces. ITS'2009* (Banff, November 23-25, 2009). ACM Press, New York, pp. 173–180.

Huang, S.C., I-F. Chou, and R.G. Bias (2006, February). Empirical evaluation of a popular cellular phone's menu system: theory meets practice. *Journal of Usability Studies*, vol. 1, no. 2, pp. 91–108.

Jacko, J.A., and G. Salvendy (1996). Hierarchical menu design: breadth, depth, and task complexity. *Perceptual and Motor Skills*, vol. 82, pp. 1187–1201.

Johnson, B. and B. Shneiderman (1991). Tree-Maps: a space-filling approach to the visualization of hierarchical information structures. *Proc. of the 2nd Conf. on Visualization VIS'1991* (San Diego, October 22-25, 1991). IEEE Computer Society Press, Piscataway, pp. 284–291.

Kang, H., C. Plaisant, and B. Shneiderman (2003). New approaches to help users get started with visual interfaces: multi-layered interfaces and integrated initial guidance. *Proc. of the National Conf. on digital government research* (Boston, May 18-21, 2003), pp. 1–6.

Kiger, J.I. (1984). The depth/breadth trade-off in the design of menu-driven user interfaces. *International Journal of Man-Machine Studies*, vol. 20, no. 2, pp. 201–213.

Lamping, J., R. Roa, and P. Pirolli (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems CHI'1995* (Denver, May 7-11, 1995). ACM Press, New York, pp. 401–408.

Lavie T. and J. Meyer (2010, August). Benefits and costs of adaptive user interfaces. *Int. J. of Human-Comp. Studies*, vol. 68, no. 8, pp. 508–524.

Matsui, S. and S. Yamada (2008). Genetic algorithm can optimize hierarchical menus. *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems CHI'2008* (Florence, April 5- 10, 2008). ACM Press, New York, pp. 1385–1388.

Norman, K.L. (1991). The Psychology of Menu Selection: Designing Cognitive Control at the Human/computer interface. Intellect Books.

Park, J., S.H. Han, Y. Cho, W. Park, H. Im, and S. W. Hong (2007). Usability of a Roll-Up Style Menu on a Mobile Phone. *Proc. of the 8th Pan-Pacific Conf. on Occupational Ergonomics* (Bangkok, October 15-17, 2007).

Pielot, M., A. Kazakova, T. Hesselmann, W. Heuten, and S. Boll (2012). PocketMenu: non-visual menus for touch screen devices. *Proc. of the 14th Int. Conf. on Human-computer interaction with mobile devices and services MobileHCI'2012* (San Francisco, September 21–24, 2012). ACM Press, New York, pp. 327–330.

Pirolli, P. (2009). An elementary social information foraging model. *Proc. of the 27th Int. Conf. on Human Factors in Computing Systems CHI'2009*. ACM Press, New York, pp. 605–614.

Plaisant, C., J. Grosjean, and B.B. Bederson (2002). SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. *Proc. of the IEEE Symposium on Information Visualization InfoVis'2002* (October 28 -29). IEEE Press, Piscataway, pp. 57–64.

Shneiderman, B. (2003). Promoting universal usability with multi-layer interface design. *Proc. of the 2003 Conf. on Universal Usability CUA'2003* (Vancouver, November 10-11, 2003). ACM Press, New York, pp. 1–8.

Shin, Ch., J.-H. Hong, and A.K. Dey (2012). Understanding and prediction of mobile application usage for smart phones. *Proc. of UbiComp'2012* (Pittsburgh, September 5-8, 2012). ACM Press, New York, pp. 173–182.

Tsandilas, T. and M.C. Schraefel (2007). Bubbling menus: a selective mechanism for accessing hierarchical drop-down menus. *Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems CHI'2007* (San Jose, April 28–May 03). ACM Press, New York, pp. 1195–1204.

Vanderdonckt, J. (1999). Computer-aided design of menu bar and pull-down menus for business oriented applications. *Proc. of 6th Int. Workshop on Design, Specification, Verification of Interactive Systems DSV-IS'99* (Braga, June 2-4, 1999), D.J. Duke, A.R. Puerta (Eds.). Springer, Vienna, pp. 73–88.