Integrating Adaptation and HCI concepts to support Usability in User Interfaces: a Rule-Based Approach

Luisa Fernanda Barrera¹, Angela Carrillo-Ramos¹, Leonardo Florez-Valencia¹, Jaime Pavlich-Mariscal¹ and Nadia Alejandra Mejia-Molina¹

¹Departamento de Ingeniería de Sistemas, Pontificia Universidad Javeriana, Bogotá, Colombia
{luisa.barrera, angela.carrillo, florez-l, jpaavlich, nadia.mejia}@javeriana.edu.co

Keywords: User Interfaces, Adaptation, Usability, HCI.

Abstract: A common problem in information systems development is to provide support for adaptation, to automatically adapt their services to different users and contexts. User Interfaces (UI) are required to adapt to those contexts and to satisfy specific criteria and standards to guarantee usability. Several methods have been created to ensure a degree of usability in UI. However, these methods focus mainly in the design stage of the development process. The benefits of these methods may be lost during execution time, since they do not address the necessity to dynamically adapt the interfaces both to context and users. To address this issue it is necessary to integrate User Interface Design with Adaptation, to ensure that UI usability is preserved at the execution time, for different users and contexts. This paper proposes Tukuchi, framework to dynamically generate Adaptive User Interfaces, based in HCI precepts. This guarantees their usability during execution time, while taking into account user preferences and context. Tukuchi was built as a rule-based system, including usability criteria commonly used for web pages, which were mapped to a desktop application.

1 INTRODUCTION

When users interact through a computational system, they do it through a User Interface (UI). The constant utilization of computers and the increasing sales of information technologies, have made computers part of the daily life of many people. For this reason, an adequate user interface design has become a very important aspect in software development (Stone et al., 2005).

Human-Computer Interaction (HCI) is a discipline that utilizes ideas from Psychology, Ergonomics, and other disciplines, to improve usability of user interfaces and provide better interaction between users and systems.

Some widespread criteria for usability in HCI are those proposed by Nielsen (Nielsen, 1995): efficiency, efficacy, error prevention, satisfaction, learning, and memorization. Because of the wide access of people to the Internet, these criteria have been widely studied in their utilization for web page design.

However, the heterogeneity of users and the context in which they utilize computers requires that UI could be easily adapted to perform various tasks. User interface design in HCI is commonly performed at the design stage in software development. Depending on the user accessing the system and other factors, such as the device utilized to interact with the system (e.g. mobile devices or desktop computers), some usability characteristics defined at the design stage may be lost during execution time.

To address the above issues, it is necessary to transform user interfaces during execution time, so that different users in different contexts and access devices may utilize these systems without losing usability.

In this regard, Adaptation is a discipline that addresses the necessity to adjust information systems behavior to specific user characteristics and context. An Adaptive system is a "system that can change itself in response to changes in its environment in such a way that its performance improves through a continuous interaction with such surroundings" (Parker, 1984). Adaptive systems offer services that are enriched with user and environment characteristics, to adjust the
Adaptation, however, does not prescribe ways to improve usability of information systems. To address this issue, previous work of the authors proposed Runa-Kamahaichi (Barrera et al., 2013a), a model to integrate HCI and Adaptation concepts, to improve the interaction between user and Adaptive systems, and improve usability. To validate the Runa-Kamahaichi model, a framework, called Tukuchiy (Barrera et al., 2013c) was created. Tukuchiy realizes Runa-Kamahaichi as an infrastructure to generate dynamic user interfaces. Two prototypes were created to validate the model in two application areas: Idukay [6] for education and Midiku [6] for clinic radiology.

This paper explains in detail a rule-based system utilized by Tukuchiy to dynamically adapt user interfaces, and the way Adaptation concepts can be utilized to improve usability of user interfaces.

The remainder of this paper is organized as follows. Section 2 explains basic concepts required to understand Tukuchiy. Section 3 reviews related work. Section 4 describes Tukuchiy and the way it addresses the Nielsen criteria [4]. Section 5 details Tukuchiy's rule-based system for UI generation and its application in the Midiku prototype. Section 6 presents a prioritization process for HCI and Adaptation rules in Tukuchiy. This section also describes the validation of the prototype. Section 7 concludes and describes future work.

2 BACKGROUND

This section describes background concepts required to understand Tukuchiy. Section 2.1 describes essential laws to design user interfaces. Section 2.2 describes Nielsen's usability heuristics. Section 2.3 explains the Adaptation concepts utilized in Tukuchiy.

2.1 Five User Interface Laws

Hale et al. en (Hale, 2011) indicate that there are five laws that every user interface designer should know and apply: Fitts Law (Guiard and Beaudouin-Lafon, 2004), Steering Law (Accot and Zhai, 2001), Hicks Law (Seow, 2005), Miller Law (Miller, 1956), and Practice Law (Roessingh and Hilburn, 2000).

- Fitts Law predicts the time required to access an objective (e.g., knock, press, select, etc.) with a unique movement (Guiard and Beaudouin-Lafon, 2004).
- Miller Law indicates that seven (plus or minus two) is the number of stimuli that people can best remember in short terms and can better make decisions about them (Miller, 1956).
- Steering Law indicates that the time a person moves within a tunnel is inversely proportional to its width (Accot and Zhai, 2001).
- Hicks Law indicates that the time taken by a person to make a decision depends on the number of options provided to that person (Seow, 2005).
- Practice Law, also called Power Law of Practice, predicts the time or velocity a task will be performed, based in the number of past intents (Roessingh and Hilburn, 2000).

2.2 Nielsen's Usability Heuristics.

Nielsen proposes ten heuristics to design user interfaces (Nielsen, 1994). They are as follows:

- System state visibility: keep the user informed about what is happening, providing adequate information in a reasonable time.
- Coincide real world and system: The system must speak the user's language, using words, sentences, and concepts that are familiar to the user.
- User control and freedom: Users often select system functions by mistake and will need an "emergency exit."
- Error prevention: Careful design to avoid the user to make mistakes.
- Recognizing instead of remembering: the user should not have to remember information from one part of a dialog to another.
- Flexibility and efficiency: Accelerate interaction for expert users, so that the system can be useful to users both with and without experience.
- Static and minimalistic design: Dialogs should not contain irrelevant or seldom necessary information.
- Help the user to recognize, diagnose, and recover from errors: Error messages should be expressed in a simple language (without codes), precisely indicate the problem, and suggest a constructive solution.
Help and Documentation: documentation should be easy to search, focus in user task, list specific measures to be executed, and not be too large.

2.3 Adaptation

Tukuchiy utilizes two main Adaptation concepts: User, and Context Profiles.

User profiles represent tastes, necessities, and preferences of each user in a system, and can be used to adjust the services provided by the system, according to individual user aspects.

Context profiles represent the user environment, those characteristics that may affect the system’s usability. Particularly, Tukuchiy takes into account the time of the day to adjust UI illumination. This adjustment is based in the criteria of Berry (Berry, 2013), which indicates the way to manage brightness to ensure that user interface colors are comfortable for the user and would not reduce his/her perception capabilities.

3 RELATED WORK

Table 1 shows a comparison between related work about UI usability and UI generation, based in our work in (Barrera et al., 2013c) and (Barrera et al., 2013a). Columns are the related works. Rows are the criteria to evaluate each work.

The columns 1-6 are as follows: Column 1 (Moussa et al., 2000); Column 2 (Criado et al., 2010); Column 3 (Zimmermann et al., 2013); Column 4 (Namgoong et al., 2006); Column 5 (Akoumianakis and Stephanidis, 1997); Column 6 (England et al., 2009).

Table 1: Related Work Comparison

<table>
<thead>
<tr>
<th>Criterion</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Takes into account usability criteria during Execution (E) time or Design (D) time.</td>
<td>D</td>
<td>E-D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>UI let the user recognize, diagnose, and recover from errors</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>UI include help and documentation</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Keeps consistency between the real world and the system</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Adapts to different types of users</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Takes into account user context aspects</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Most works do not focus in improving usability. Although most take into account user profile and his/her context, they do not take into account HCI standards. These works do not take into account that interfaces change during execution time and that it is necessary to avoid losing standards given during design time.

4 TUKUCHIY

Tukuchiy ("Tukuchiy" is a Quechua word that means "To Transform") is a framework based in the Runa-Kamachiy model (Barrera et al., 2013a), to generate dynamic user interfaces, adjusted to specific user characteristics, context, and presentation preferences. Tukuchiy keeps some usability standards at execution time, so that UI usability can be kept across the entire life cycle of the system (see Figure 1).

Table 2: Usability Criteria and Heuristics

<table>
<thead>
<tr>
<th>Usability Criterion</th>
<th>Heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning</td>
<td>User control and freedom. Recognize instead of Remember.</td>
</tr>
<tr>
<td>Error prevention</td>
<td>Help the user to recognize, diagnose, and recover from errors. Help and</td>
</tr>
</tbody>
</table>
4.1 Learning

Learning is related to the capacity of the software to let users learn to use its components (Carvajal and Saab, 2010). Tukuchiy uses Practice Law (Roessingh and Hilburn, 2000) to provide different types of help, with various levels of detail, depending on the user expertise. If the user is new, help is more detailed (see Figure 2a). As the user gains more experience utilizing the system, help is reduced (Figure 2b).

![Figure 2: Learning in Tukuchiy](image)

4.2 Error Prevention

An error-tolerant interface is designed to assist the user in recovering from errors (Carvajal and Saab, 2010). It is reflected in the types of help provided to the user to perform different tasks in the system. Tukuchiy utilizes tooltips with more or less information, according to the user experience level. Users with less experience received more detailed tooltips, while users with more experience received tooltips with less detail.

![Figure 3: Button Intentionality](image)

Additionally, interface buttons are associated with intentionality and a color that represents that intentionality. Buttons should not have colors that may confuse the user (Barrera et al., 2013b). Figure 3 shows the color palette utilized when one wants to change the color to the "quit" button. Since this button is associated to a "danger" intentionality, blue and green colors, which represent "harmony", are not present in the palette (Barrera et al., 2013b).

4.3 Memorization

The memorization aspect enables users to easily remember how to interact with that system, after a period without using it (Nielsen, 1995). To address the memorization aspect, this research utilized the Miller Law (Miller, 1956), which establishes that it is easier for humans to remember information of 7 ± 2 options. Tukuchiy groups buttons according to functionality based in this 7 ± 2 rule (see Figure 4). Changes performed by the user in design (personalization) of the UI persist across sessions.

![Figure 4: Memorization in Tukuchiy](image)

4.4 Efficiency

Efficiency is associated to the amount of effort required by the user to achieve a specific goal in his/her interactions with the system (Carvajal and Saab, 2010).

![Figure 5: Efficiency in Tukuchiy](image)

Tukuchiy enlarges buttons (see Figure 5), based in the Fitts Law. In addition, it reduces in the amount of colors in palettes. These two strategies reduce the user efforts to accomplish a task or to personalize the interface.

In addition, the system changes the UI colors, based in the time of the day (ambient illumination), to improve visibility of the interface functions.

4.5 Efficacy

According to ISO 9131-11, efficacy is the degree in which planned activities are performed and the planned results are achieved. In other words, can users do what they need in a precise manner?
The use of tooltips and Fitts Law to enlarge UI elements, seeks to improve the precision of the performed tasks. In addition, color transformation assists people with color blindness to properly identify colors and avoid mistakes.

As seen in Figure 6, for people with Protanopy (red color blindness), Tukuchiy changes the color palette, discarding red colors, so that the user may distinguish a broader range of colors.

4.6 Satisfaction

Satisfaction is the perception of pleasantness and positive attitude towards the utilization of a product. That perception is reflected in the physical and emotional actions of the user when utilizing the system. In other words what is the perception of the user about the ease of use of the product? How pleasant is to use it? (Carvajal and Saab, 2010). The system does not directly address this criterion. However, we sought to indirectly satisfy the user by integrating all of the other criteria. For instance, the palette change for color blind users (Flück, 2006) may be pleasant for them.

5 RULE SYSTEM

To maintain usability characteristics during execution time, Tukuchiy is based in a rule-based system (Barrera et al., 2013c), which is detailed in this section.

To test the rules, Midiku was built, a clinic radiology application, to support the diagnostic process and medical image simulation. To build the system, two groups of rules were created: HCI rules and Adaptation rules. Both are detailed in the following sections.

5.1 HCI Rules

This group of rules realizes a subset of HCI standards. To build that subset, this research verified all of the standards that could be kept during execution time. The rules are the following:

5.1.1 Physical Conditions

The system focuses in assisting two physical difficulties: color blindness and myopia (see Figure 7). Two processes are performed to assist in these difficulties: i) color simulation and polarization; and, ii) button enlargement.

For color blindness, Tukuchiy changes the palette using the following rule:

```
void UserInterface::ButtonCustomization_Pressed_UI()
{
    focus->focus_widget();
    rules->changeColorPalette(((Button*)focus));
    ((Button*)focus)->changeColor
        (user->userPhysical->getFunctionalDiversity(),
         user->colorPreferences, this->delta);
}
```

Figure 7: Physical conditions rule

For color blindness, Tukuchiy changes the palette using the following rule:

i). Color blindness identification: when entering the system, the user is shown an image corresponding to the Ishihara Test (Flück, 2009). This test determines whether the user has color blindness problems or not and which type of color blindness he/she has (see Figure 8).

For color blindness, Tukuchiy changes the palette using the following rule:

ii). Color simulation: based in the code of (Duck, 2012), a simulation is performed in which palette colors are changed, so that they could be
perceived by the person, according to his/her color blindness type.

iii). Polarization: base colors are compared with simulated colors and the difference is calculated. This is used to change the original colors when this difference is detected and change them to colors that are visible to the user [26].

Figure 9 is a fragment of the rule for color changes. This rule is utilized when the user needs to use the palette to personalize the interface colors.

```java
// Initial UCS
float L = (17.8826f * r) + (43.5361f * g) + (1.1935f * b);
float M = (3.45505f * r) + (27.1554f * g) + (3.6716f * b);
float S = (8.099556f * r) + (6.184309f * g) + (1.46709f * b);
else if (referredcolor == "Tritanope")
{
    L = 1.0f * L + 0.0f * M + 0.0f * S;
    S = 0.0f * L + 1.0f * M + 0.0f * S;
 heav = -0.395013f * L + 0.001095f * M + 0.00f * S;
}
```

Figure 9: Color transformation rule fragment (color blindness)

The Ishihara test is performed several times, to mitigate any external factors that could affect the validity of the user answers (e.g. screen resolution, ambient illumination).

![Figure 9: Color transformation rule fragment (color blindness)](image9)

As shown in Figure 10, this rule changes colors both to the buttons and the medical image being examined. The following algorithm is used for button enlargement:

i). Identify the visual problem: This is done explicitly, asking the user if he/she has myopia. This information is stored in the user profile.

ii). Button enlargement: button properties are changed, so that, whenever the user points to the button, it changes its size. The layout of buttons within the same functional group is re-arranged.

```java
void UserInterface::ButtonMouseOver_ui()
{
    focus = false;
    if (focus) Inherits("Button")
    {
        bool fittsTrue = rules->
            evaluateFitts(user->userPhysical,
            getScaleFunctionDiversity());
        if (fittsTrue)
            ((Button*)focus)->changeSizeFitts();
    }
}
```

Figure 11: Button enlargement rule

Figure 11 describes the way button scale is changed according to the user profile. This rule is based in the Fitts Rule (Guiard and Beaudouin-Lafon, 2004). Figure 12 shows an example of button enlargement in Midiku.

![Figure 11: Button enlargement rule](image11)

![Figure 12: Button Enlargement](image12)

5.1.2 Effective Color Combinations

Among the studies about the way user interface colors should be combined, Wright et al (Wright et al., 1997) indicates that colors are not visible when they overlap. To ensure that this rule is enforced during execution, Tukuchiy performs the following algorithm:

i). Parent color: Identify the color of the element that contains the component whose color is going to change.

ii). Children colors: Identify the color of the elements contained by the component whose color is going to change.

iii). Color filter: Using the above information, the color palette is filtered to eliminate the colors that, according to Wright, do not match adequately (Wright et al., 1997). Color combinations are organized in a pessimistic manner, i.e., there is a list of colors that do not match in the previous steps.

iv). Presentation of the color palette. The filtered palette is presented to the user.
Figure 13 shows the palette change rule according to color combination.

```java
void Rule::changeCustomPalette(Button *w)
{
    //quitar texto de bg y bg de texto
    removeOriginalStyleColorGroup(w);
    //quitar grupo color padre
    removeParentColorGroup(w);
    //quitar grupo color hijos
    removeChildrenColorGroups(w);
    //quitar prohibidos segun intencion
    removeIntentionGroupColor(w);
}
```

Figure 13: Color combination rule

Figure 14 is an example of a palette change, in which the text of the red button contains only the colors that combine or contrast with the button color.

```java
void Rule::removeIntentionGroupColor(Button *w)
{
    Dialog * auDialog = new Dialog(); //getDialog();
    vector<Intentions> vecAux = intentions.getIntentions();
    vector<string> colorIntention;
    for(int i=0;i<vecAux.size();i++)
    {
        if(vecAux[i].getIntentionsName()=='Chi-
            ButtonIntention')
            colorIntention = vecAux[i].getForbiddenColors();
        for(int j=0;j<colorIntention.size();j++)
        {
            (Button *)w->isDialog->removeColorFromPalettes
                (colorIntention[j]);
        }
    }
    ((Button*)w)->getMenu().setDialog(auDialog);
}
```

Figure 14: Color combination example

5.1.3 Widget Intentionality

In a study performed by Bedolla (Bedolla, 2002), colors are associated to specific psychological states and have specific intentionality. For instance, red is associated to danger situations. This is taken into account to assign to each UI element, an intentionality associated to a color, to keep each element's essence. The algorithm for this task is the following:

i). Widget intentionality: A table is created that maps intentionality to allowed and forbidden colors. Note that each UI element may or may not be associated to intentionality. An XML file is used to store that table.

ii). Allowed and forbidden color revision: when the user is going to change a color, it is evaluated whether the color keeps the same intentionality, according to the table. Similarly to the color combination rule, this filter is performed pessimistically and is presented as a palette to the user.

Figure 15: Button intentionality

Figure 15 is a fragment of the rule that performs the filter (eliminate colors of the palette) of the colors that are not allowed, according to the button intentionality.

Figure 16 shows the allowed colors for the "close" button.

5.1.4 Luminosity

Given the importance of luminosity (see Section 2.3), a rule was created to transform the colors of the entire interface. The transformation takes context into account. For instance, illumination is changed according to the time of the day.

Figure 17 is an example of the way illumination is updated according to context. This rule is executed each time the system is started. No changes are performed while the system is being utilized, to avoid being too intrusive for the user.
Figure 17: Color brightness

Figure 18 shows the way this rule is realized in Midiku. The palette in the upper part of the screen shows the bright colors. The palette in the lower part shows the darker colors.

**Figure 18: Brightness palettes in Tukuchi**

### 5.2 Adaption Rules

This section shows the Adaption rules that complement the dynamic generation of interfaces, based in user characteristics and context.

#### 5.2.1 Help

Tukuchi filters information during the system startup to evaluate the user experience level and language preferences.

This rule process is as follows:

1. **User experience level identification:** The user profile has an attribute that indicates the amount of time the user utilizes the system, which determines the experience level of the user. This information is updated each time the user closes the system.

2. **Language identification:** When starting the system, the user chooses his/her language of preference.

3. **Help changes:** From the information in i) and ii), the system changes the names in its UI elements according to the chosen language. Tooltips are automatically changed according to the experience level. Currently, the system has two levels, novice and expert. Tooltips are more detailed for novice users than for expert users.

#### 5.2.2 Color Preferences

Each user may have different preferences about colors to display each UI element. To realize this in the preferences, a rule was created that organizes the color palette, to show the UI according to the tastes of the user. This rule is utilized as follows:

1. **User preferences:** User color preferences are read. Each time a user selects a color, a counter is updated, which is used by the system to find out which the degree of color preference, i.e., which colors are more frequently selected by the user.

2. **Prioritization:** Based in the degree of color preference, the color palette is reorganized from most preferred colors to least preferred colors.

Figure 19 illustrates the way the above rule is applied before showing the user interface. Each help has an XML file that associates UI elements with different tooltips and names.

**Figure 19: Use level about use level**

- **Figure 20** is a code fragment that denotes the way the palette is organized according to the user preferences.
important to each user. In other words, to improve the system's usability, it is necessary to satisfy usability criteria and the latter is accomplished if the user experience is taken into account during the design of the UI.

Because of these user differences, two prioritization methods were developed, each one is utilized in different stages of the UI construction process: i) Design Prioritization, performed at the Design stage and ii) Dynamic prioritization, performed during execution.

6 Design Prioritization

This section describes the prioritization process for the criteria mentioned in the previous section.

To understand this prioritization, it is important to take into account the meaning of "User Experience", which according to Hassam et al. (Hassan and Martin, 2005) involves user sensations, feelings, emotional response, and satisfaction with respect to the product, all of which result from the interaction with the product.

Based in this definition, user experience goes beyond satisfying usability criteria in the design of the UI. In addition, it is necessary to take into account, cultural aspects, user goals, emotional response of the user when interacting with the application, among others.

This knowledge about the user experience is necessary for the prioritization of the usability criteria, since it provides a vision about the user factors that must be taken into account to ensure that the UI satisfies the usability criteria that are important to each user. In other words, to improve the system's usability, it is necessary to satisfy usability criteria and the latter is accomplished if the user experience is taken into account during the design of the UI.

Because of these user differences, two prioritization methods were developed, each one is utilized in different stages of the UI construction process: i) Design Prioritization, performed at the Design stage and ii) Dynamic prioritization, performed during execution.

6.1 Design Prioritization

This stage classifies users in sub-groups with similar characteristics, to identify the types of users that interact with the system. For instance, the system may classify the users by age and yield two sub-groups: young (16-26 years old) and adults (27-50 years old).

User characterization seeks to specify the features that describe each sub-group, since each sub-group may have different preferences. For instance, for a young user it may be more important the system response time and for an adult it may be more important the precision in which a task is performed.

To properly characterize user sub-groups, we propose to utilize the requirements elicitation process described by Stone et al (Stone et al., 2005), which includes user observation techniques, interviews to the users, and questionnaires to obtain user information.
6.1.2 Identification of system goals

This stage specifies the general objective of the system, the final goal of its construction. This goal is important, since if only the user goals are taken into account, the system may not perform the adequate tasks for which it was built.

Continuing with the same example, although for young users the most important aspect would be response times, if the system goal is transaction reliability (e.g., a bank system), precision in the operations would have more importance than response times.

6.1.3 Usability Criteria Prioritization

Criteria are prioritized in three stages:

i). Objective prioritization: user requirements are mapped to each usability criteria. The importance of each usability criteria will be calculated from the number of requirements mapped to each criterion. For instance, if latency has associated 4 requirements and coherence has associated 2 requirements, then latency would have more priority than coherence.

ii). Subjective prioritization: Given the empirical nature of usability (Hassan and Martin, 2005), it is necessary to subjectively prioritize criteria, based in experience and observation of those criteria. Both users and developers should assess the importance of usability criteria according to their necessities and experience. Users, on one hand, know which criteria are the more important, according to their own necessities, while developers know the technical difficulty and feasibility of implementing each criterion.

iii). Joint prioritization: the final priority values are obtained from the average of priorities obtained from i) and ii).

6.2 Dynamic Prioritization

Dynamic prioritization is utilized to adapt user interfaces to user characteristics, preferences, and context during execution time. Dynamic UI construction takes as input the prioritization of usability criteria, according to users and the system’s vision.

Figure 23: Dynamic prioritization process.

Figure 23 describes the dynamic prioritization process, which includes three stages: i) Personalized prioritization, ii) Prioritization averaging, and iii) prioritization execution.

6.2.1 Personalized Prioritization

This stage utilizes questionnaires to determine the priorities of each user, the importance of each usability criteria to users. Continuing with the previous example, for a young user the exploratory interfaces criteria has an importance of 5 (more important), while for an adult user the importance is 2 (less important). To realize the fact that different users have different priorities, this stage generates a personalized prioritization list of usability criteria. This list also takes into account the user environment.

6.2.2 Prioritization averaging

Although user necessities and preferences are important to prioritize usability criteria, the static prioritization resulting from Section 5.1.3 must also be taken into account. For that reason the user priorities (dynamic priorities) and system priorities (static priorities) are averaged.

6.2.3 Prioritization execution

The prioritization averages obtained in the previous stage are realized in the user interface.

It is important to note the difference between static prioritization, which generalizes user characteristics (e.g. young and adults) and the adapted prioritization that prioritizes necessities and characteristics specific to each individual.

7 PILOT TEST

The authors are currently developing a functional prototype called Midiku. Since this is a work in progress, its initial assessment has only been performed over the design of Midiku’s User
This design includes functionality given by Tukuchiy (see Section 4).

### 7.1 Evaluation Process

To evaluate the usability of Midiku’s interfaces, Mock-ups (Soegaard, 2004) were utilized. Mockups are a digital demonstration of the way the UI will look like in the final system. Mockups were shown to three physicians of the San Ignacio Hospital in Bogotá, Colombia. One of them is an expert radiologist with several years of experience, who is not proficient with current computing technologies. The other two are physicians who are specializing in radiology and have high proficiency utilizing current computing technologies.

The three physicians answered a survey based in QUIS (Questionnaire for User Interface Satisfaction) (Chin et al., 1988). The questions answered focused exclusively in evaluating the UI design.

Table 3: Example questions of the survey

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Question Description</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Interaction and Adaptability</td>
<td>Flexibility of the user interface</td>
<td>Very rigid, Rigid, Flexible, Very Flexible</td>
</tr>
<tr>
<td>Complexity of the user interface</td>
<td>Very hard, Hard, Easy, Very easy</td>
<td></td>
</tr>
<tr>
<td>2. Screen and Display</td>
<td>Organization of information on screen</td>
<td>Very confused, Confused, Clear, Very clear</td>
</tr>
<tr>
<td>Use of terms throughout system</td>
<td>Very inconsistent, Inconsistent, Consistent, Very consistent</td>
<td></td>
</tr>
<tr>
<td>Is the screen density:</td>
<td>Very inadequate, Inadequate, Adequate, Very adequate</td>
<td></td>
</tr>
<tr>
<td>3. Presentation and Visualization</td>
<td>Are menus distinct from other displayed information?</td>
<td>Very confused, Confused, Clear, Very clear</td>
</tr>
<tr>
<td>Does it provide visually distinctive data fields?</td>
<td>Very high grade, High grade, Low grade, Very low grade</td>
<td></td>
</tr>
<tr>
<td>Are groups of information demarcated?</td>
<td>Very confused, Confused, Clear, Very clear</td>
<td></td>
</tr>
</tbody>
</table>

Table 3 shows some of the questions, grouped by evaluation criteria. The first criterion is the user appreciation with respect to the interface. The second criterion is the organization and meaning of graphical elements in the screen. The third criterion is about the color utilization and screen zones delimitation.

Figure 24 shows a screen with part of the survey. At least 50% of the questions have images similar to the ones in the figure.

![Survey Screen Fragment](image)

#### 7.2 Pilot Test Results

Surveyed subjects were divided in two groups: expert and novice. Three questions from Criterion 1 (Interaction and Adaptability), six from Criterion 2 (Screen and Display), and four from Criterion 3 (Presentation and Visualization).

Figure 25 indicates the results of the survey. For each criterion, results are shown for the expert group, the novices group and the expected value, which is the maximum score that can be obtained in each criterion.

![Survey Results](image)
The figure indicates that the expert radiologist valued the first criterion as 66.7%, emphasizing the interface flexibility, but he expressed that the attractiveness of interface has a low level. He valued the second criterion as 54.2%, emphasizing the adequate density of elements in the screen, but indicating the difficulty to understand the meaning of buttons. He valued the third aspect as 81.3%, emphasizing the adequate use of colors.

For novice users, the answers were averaged. The first aspect was valued as 66.7%, emphasizing the ease of initial interpretation of the interface. The second aspect was valued as 68.8%, emphasizing the organization and adequate terminology, but they expressed the density of elements in the screen are inadequate. The third aspect was valued as 68.8%, emphasizing the adequate visual distinction among screen zones, but indicating the inadequate utilization of colors.

The users commented that they would want to have more intuitive and less complex radiology interfaces. They also commented that there are “dead spaces” in the screen that could be better utilized to present information. They indicated that the survey could be enriched by using videos of the mockups, to better understand the functionality of Midiku.

8. CONCLUSIONS AND FUTURE WORK

This paper presented Tukuchiy, a framework that integrates several methods and techniques in HCI with Adaptation concepts to improve user interaction with systems in changing contexts. Tukuchiy's rule-based system ensures usability criteria are preserved at execution time in changing interfaces.

This paper also presented a functional prototype (Midiku) that supports radiologists to diagnose medical images. An initial assessment at this stage has only been performed over the UI design of Midiku. This assessment was performed through a Mockup and a survey that was answered by an expert and two novice radiologists. The results emphasize positive aspects, such as low UI complexity, adequate organization of information on the screen and the ease to visually distinguish data fields. Negative aspects found are the difficulty to understand the meaning of buttons, inadequate characters visualization and inadequate terminology.

Future work includes fully developing the functionality of Midiku and performing a more detailed analysis of the capabilities of Tukuchiy in terms of efficiency, efficacy, and user satisfaction. In addition, Tukuchiy will be assessed with fully test (20 observer’s approx.). The assessment includes an initial perception test and then interaction atoms test (key functionalities).

ACKNOWLEDGMENT

The authors want to acknowledge the Systems Engineering Department and the Systems Engineering and Computing Master’s program of the Pontificia Universidad Javeriana, for supporting the development of Runa-Kamachi, Tukuchiy and Midiku.

The authors also thank Luis Felipe Uriza and his team of resident physicians of the San Ignacio Hospital, for their collaboration in the initial evaluation of Midiku.

REFERENCES


