Université Catholique de Louvain

Louvain School of Management

Travail de fin d'études

Développement d'un logiciel de support à l'évaluation heuristique de systèmes informatiques

Étudiants :

CÉDRIC CARIAT

JULIEN DUBOIS

Promoteur :

Prof. J. VANDERDONCKT

Master 60 en Sciences de Gestion (IAG2M1/GEST20M1)

Année académique 2009-2010

Nous tenons à remercier sincèrement toutes les personnes qui nous ont apporté leur soutien ou nous ont aidé dans la réalisation de ce mémoire.

Nous tenons particulièrement à remercier Monsieur Jean Vanderdonckt pour son aide précieuse, ses conseils avisés et la liberté d'action qu'il nous a offerte tout au long de la rédaction de ce travail, ainsi que ses assistants.

Merci à nos parents ainsi qu'à tous ceux qui se sont intéressés à ce projet, ne serait-ce qu'en tentant de comprendre ce que cet énoncé aux termes obscurs pouvait signifier.

Sans vous, ce travail aurait probablement été fort différent.

Merci

Table des matières

Introduction

Alors que les interfaces homme-machine ne cessent de proposer de plus en plus de fonctionnalités, l'utilisation de celles-ci n'est pas toujours aisée. Pourtant, avec les nouvelles technologies telles que le web 2.0 ou l'usage de plus en plus fréquent de la réalité augmentée dans la vie quotidienne, il devient essentiel de se pencher sur le confort de l'utilisateur afin de rendre les interfaces plus ergonomiques. Cependant, force est de constater que la tâche n'est pas simple pour les développeurs, tant pour accéder à de l'information pertinente sur le sujet que pour corriger ce qu'ils ont conçu. Il existe pourtant plusieurs méthodes qui traitent de la problématique dont une qui semble sortir du lot, la méthode heuristique.

La première partie de ce rapport comporte deux chapitres et vise à présenter cette méthode et le cadre dans lequel elle se situe. Ainsi, le lecteur a une idée claire du contexte dans lequel s'insère notre travail. Tout au long de cette partie, nous effectuons volontairement un abus de langage en assimilant et en utilisant sans réelle distinction les termes logiciel, interface, application, système informatique et site web car une analyse heuristique peut être effectuée sur chacun d'entre eux.

Le premier chapitre présente brièvement la notion d'ergonomie, son importance pour les utilisateurs et les différentes méthodes d'analyse qui ont été mises au point. L'ensemble des techniques présentées concerne principalement les méthodes d'analyse empiriques et analytiques.

Le second chapitre se concentre sur l'analyse heuristique et présente la méthode en tant que telle, les variantes existantes ainsi que la manière de procéder en pratique. Ce chapitre se conclut en mettant en évidence les différents avantages et inconvénients de cette méthode.

Pratiquement, l'évaluation de l'ergonomie d'un système informatique par la méthode heuristique consiste à étudier si des règles ergonomiques organisées en listes de vérification sont respectées. Une fois ce travail analysé, des résultats, conclusions et recommandations peuvent être déduits.

Le nombre de règles à vérifier étant généralement élevé et l'analyse des résultats comportant un important volet analytique et statistique, une analyse ergonomique est une tâche assez lourde, répétitive et automatisable. Ces caractéristiques font que l'utilisateur a un intérêt certain à se faire aider dans sa tâche par un logiciel de support. L'objectif principal de notre travail est donc ainsi défini : développer un logiciel de support à l'évaluation de l'ergonomie par la méthode heuristique.

La seconde partie de ce rapport est divisée en trois chapitres et vise à présenter le logiciel développé. Tout d'abord, nous expliquons les spécifications du logiciel. Ce premier chapitre permet de décrire les fonctionnalités de notre application indépendamment de toute implémentation. Comme une analyse ergonomique doit être adaptée au système informatique étudié et aux besoins de l'analyste, nous expliquons de quels paramètres notre logiciel doit tenir compte pour répondre à ces attentes. De plus, le logiciel doit remplir trois fonctions essentielles : saisir les paramètres de l'analyse, générer une interface permettant d'encoder le résultat de l'évaluation des règles et analyser les données recueillies. En plus de développer ces différentes fonctionnalités, nous clôturons ce chapitre en présentant un diagramme de classe.

Le chapitre suivant traite de l'implémentation du logiciel à l'aide de l'outil Microsoft Excel et du langage de programmation VBA. A chacune des fonctionnalités essentielles décrites dans le chapitre précédent est associée une interface de communication avec l'utilisateur. Celle-ci prend la forme d'une feuille de calcul et son fonctionnement est présenté en détail.

Pour terminer, quelques perspectives de développement sont abordées dans le dernier chapitre. Ces pistes de recherche permettraient d'apporter de nouvelles fonctionnalités au logiciel.

Première partie : Cadre théorique

« Si vous ne pouvez le faire bien, rendez le beau. »

William Henry Gates, dit Bill Gates

1 Ergonomie et interface homme-machine

1.1 Origine des méthodes d'analyse

1.1.1 Concept d'ergonomie

Les logiciels et le web évoluent chaque jour. Les fonctionnalités offertes sont multiples et se compliquent de plus en plus. Afin que les utilisateurs de tout âge et de toute éducation puissent continuer à les utiliser, il est nécessaire de rendre ces technologies simples et aisées à employer. Dans le cas contraire, et vu les avancées technologiques, une partie toujours plus importante de la population risque de ne pas pouvoir en bénéficier. C'est là qu'intervient l'ergonomie.

L'ergonomie est la science du travail qui a pour but d'améliorer les conditions de travail de l'homme. Un travail plus adapté permet d'augmenter non seulement le ressentit du travailleur envers la tâche à effectuer mais également son efficacité. Pour ce faire, le travail doit être évalué dans son ensemble, tant dans son aspect physique que psychologique [1].

Dans le domaine des interfaces homme-machine, l'élément essentiel de l'ergonomie appartient au domaine cognitif. Nous ne nous attarderons pas sur les aspects consistant à réaménager l'environnement du travailleur et son poste de travail mais sur le fonctionnement des logiciels. Ceux-ci doivent être en adéquation avec le comportement des utilisateurs afin de simplifier l'interaction avec ceux-ci. Pour ce faire, il est nécessaire d'optimiser l'interface en supprimant tous les problèmes ergonomiques qui seraient susceptibles d'apparaître durant sa création.

On se retrouve en présence d'un problème d'ergonomie lorsqu'un défaut de conception vient compliquer l'exécution de la tâche de l'utilisateur ou la rend inefficace lorsqu'il essaye de parvenir à un but déterminé. De plus, ce type de problème va souvent de pair avec la diminution de la satisfaction qui ressort de l'usage de l'interface. Il s'agit principalement de se recentrer sur l'utilisateur et de faciliter son interaction avec le système [2].

1.1.2 Critères d'une application ergonomique

Pour découvrir ces défauts, les différentes méthodes d'analyse, qui seront examinées plus en détail par la suite, se basent sur deux critères essentiels : l'utilité et l'utilisabilité [1].

L'utilité représente le potentiel d'aide à la réalisation de l'objectif que l'interface peut apporter à l'utilisateur. Il s'agit donc d'examiner si les fonctions offertes sont pertinentes par rapport aux fonctions recherchées par l'utilisateur pour lui permettre de parvenir à son but.

L'utilisabilité (ou usabilité) est définie par la norme ISO 9241 comme un concept regroupant l'efficacité, l'efficience et la satisfaction de l'utilisateur. L'application doit donc permettre de parvenir à l'objectif de façon fiable et performante, le tout en un minimum de temps et d'efforts afin de rendre le système intuitif et agréable à utiliser [3].

Il est important que ces deux principes soient respectés en même temps, sans quoi le système n'est pas réellement ergonomique.

Une application peut tout à fait répondre à un besoin et donc être utile sans pour autant être utilisable, c'est-à dire que son utilisation est complexe et peu intuitive par rapport au besoin qu'elle vient combler, comme l'illustre par exemple la Fig. 1.1. Cela va également dans l'autre sens. Une application peut être très simple à manier mais rester inutile en ce sens qu'elle ne fournit aucune fonction recherchée [4].

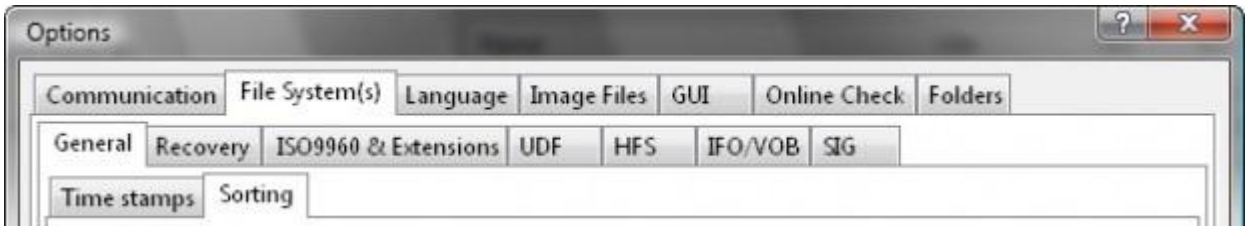


Figure 1.1: Exemple d'interface peu intuitive, Menu d'option de Isobuster: programme de récupération de données, Source : http://www.ergonomie-interface.com/mauvaises-pratiques-ergonomie/ergonomie-et-stress-informatique/

1.1.3 Importance de l'utilisabilité

Tout développeur se doit de prendre en compte l'ergonomie du système qu'il développe s'il souhaite que celui-ci devienne populaire et apprécié des gens. Il va sans dire qu'un utilisateur satisfait d'une interface, car elle est utile, intuitive et aisée à manipuler, a plus de chances de continuer à l'utiliser et en parler autour de lui. Il s'agit pour le développeur de ne pas négliger cet

aspect et de le prendre en compte au plus tôt de la phase d'élaboration [5].

De ce fait, il faut donc mettre en place un processus itératif dès la phase de conception afin de pouvoir s'adapter au plus vite à ce que souhaitent les utilisateurs. Comme en programmation, où un code n'est jamais écrit sans bugs la première fois, une interface n'est jamais satisfaisante lors de sa première version. C'est pourquoi il faut sans cesse la retravailler et la réexaminer à l'aide d'au moins une des différentes méthodes d'analyse disponibles [5].

Comme il sera examiné par la suite, certaines méthodes sont plus lourdes à mettre en place et nécessitent un capital en matériel, en ressources humaines et en temps plus important que d'autres. Cela engendre des différences de coût qui peuvent être assez élevées selon la méthode choisie.

Conscient de cette idée et des difficultés existantes, Jakob Nielsen présenta une méthode d'évaluation inspirée par ces contraintes. Il a décidé d'élaborer une série d'études qui pourraient être accessibles à moindre coût. Il a alors mis au point un processus d'évaluation qu'il a lui même qualifié de « discount », afin que chacun puisse effectuer un minimum d'analyse sur le projet qu'il développe. Selon lui, même si l'analyse n'est pas menée dans des conditions optimales et par des personnes formées, il est préférable de découvrir et corriger au moins une partie des défauts d'utilisabilité que pas du tout. Le but de ces méthodes n'est donc pas de tendre vers la perfection mais de donner l'opportunité de découvrir les problèmes principaux [6].

Dans la pratique, on constate que de nombreux développeurs, malgré leur connaissance de l'importance de l'ergonomie dans la conception, estiment que l'investissement nécessaire est inférieur au gain réalisé lorsqu'une telle étude est effectuée [7].

1.2 Différentes méthodes d'analyse

Cette section présente brièvement et de façon non exhaustive un panel de méthodes existantes permettant d'effectuer une analyse d'utilisabilité. Pour chacune d'elles, nous listons des avantages et inconvénients. L'analyse selon la méthode heuristique ne sera pas abordée ici vu qu'elle fait l'objet du chapitre suivant. Nous la positionnerons par rapport aux différentes méthodes étudiées ci-dessous. Les trois premières méthodes présentées sont celles qui sont considérées comme étant plus simples à appréhender et à mettre en œuvre [4].

1.2.1 Examen par scénario et prototypage

Méthode

Parmi les différentes méthodes permettant d'effectuer une analyse ergonomique, on retrouve l'examen par scénario et itération rapide. C'est une méthode dont le coût est relativement faible. Elle est utilisée afin de créer des prototypes ayant pour but de simplifier la complexité de certaines interfaces en se concentrant sur certains éléments essentiels et en supprimant ceux qui ne sont pas nécessaires au test.

On retrouve des prototypes horizontaux qui donnent accès aux différentes actions disponibles sans pouvoir toutes les effectuer. Cela permet d'avoir un aperçu global des fonctionnalités qui seront disponibles. Les prototypes verticaux permettent quant à eux d'avoir une version permettant de tester un nombre limité de fonctionnalités mais qui sont pleinement opérationnelles [8]. La Fig. 1.2 illustre un exemple de prototype.

Figure 1.2: Exemple de prototype papier et son équivalent fonctionnel, Source : http://www.guuui.com/issues/03_05.php

Avantages

La mise en place d'un prototype est relativement simple et, une fois combiné avec une étude de pensée à voix haute, cela permet de tester à bas prix chaque nouvelle version, tout en tenant compte des remarques précédentes.

Le prototype peut être une simple maquette en papier ou être représenté dans un environnement de programmation plus simplifié afin d'en faciliter le temps d'apprentissage. Cela permet également d'éviter d'avoir à recourir à des outils de développement avancés en début de projet et

d'accélérer ainsi sa mise en place. Ce type de méthode peut donc intervenir très tôt dans la phase de développement [4].

Inconvénients

Il s'agit d'une méthode permettant de réduire une interface à sa plus simple expression tout en permettant d'effectuer la conduite d'une analyse ergonomique. Néanmoins, il ne s'agit que d'une simulation de l'interface qui n'est possible que si l'utilisateur suit scrupuleusement un ensemble de tâches planifiées afin de rester dans le canevas mis à disposition par le prototype.

1.2.2 Pensée à haute voix

Méthode

En bref, il s'agit d'écouter un utilisateur à qui il a été demandé de penser à haute voix et qui teste le système en accomplissant une série de tâches prédéfinies. En général, ce type d'étude est mené avec l'aide d'un psychologue ou d'un autre expert qui enregistre l'ensemble de la procédure avant de revenir sur l'examen des résultats [8].

Nielsen suggère une version simplifiée de cette méthode durant laquelle, un examinateur note directement les remarques pertinentes dont l'utilisateur fait part lorsqu'il utilise le programme.

L'analyse est ensuite basée sur les notes prises durant cette phase de test. Il n'y a donc pas besoin de machine pour enregistrer et il ne faut pas passer de temps à réécouter ce qui à été dit. Cela permet de diminuer les coûts de façon importante [4].

L'utilisateur est souvent tenté de poser des questions directement à l'examinateur. Celui-ci se doit de répondre par une autre question afin de ne pas intervenir dans le test et de fausser les résultats. Par exemple, si l'utilisateur demande s'il peut effectuer une action, l'examinateur devra lui répondre en lui demandant de s'interroger sur les conséquences attachées à l'exécution de celle-ci. L'examinateur se doit d'influencer le moins possible le test, si ce n'est pour rappeler à l'utilisateur d'exprimer ses pensées [4].

Avantages

Avec cette méthode, l'examinateur peut déterminer ce que fait l'utilisateur avec le système, mais également pourquoi il le fait. Cela permet alors d'améliorer certaines incompréhensions et de redévelopper une interface de façon plus intuitive. L'examinateur peut identifier directement ce que l'utilisateur ne comprend pas ou n'interprète pas correctement. De plus, les commentaires des

utilisateurs sont souvent précis et permettent de rendre le rapport final plus aisé à retenir. Il s'agit à nouveau d'une démarche itérative où il est intéressant de corriger l'application en fonction des erreurs découvertes auparavant [4].

Inconvénients

Les utilisateurs sont rarement à l'aise en exprimant leurs pensées à voix haute tout en effectuant une tâche précise. Cela affecte souvent le processus et les utilisateurs oublient de continuer à s'exprimer durant l'usage de l'interface.

Il semblerait que ce soit encore plus difficile lorsque les utilisateurs sont des experts. Ils effectuent certaines tâches si rapidement qu'ils n'ont pas le temps de les commenter. Pour eux, certains processus sont tellement automatisés qu'ils n'ont même pas conscience de ce qu'ils font.

Cette difficulté à s'exprimer tout en effectuant le test peut avoir des conséquences sur les résultats. Par exemple, cela ralentit généralement la vitesse d'exécution de façon considérable. Il est donc impossible de mesurer la performance de l'utilisateur avec cette méthode.

De plus, il semblerait que l'utilisateur, en parlant à voix haute, s'influence lui même en effectuant la tâche et effectue ainsi moins de mauvaises manipulations. Les résultats obtenus sont donc faussés par rapport à un utilisateur qui utiliserait le programme de façon silencieuse [4].

De plus, les utilisateurs feront toujours des commentaires à propos de ce qui leur déplait dans l'interface. L'examinateur doit en tenir compte tout en restant critique. Parfois, l'utilisateur s'évertuera à critiquer un point qui n'a pas vraiment d'intérêt. Par exemple, une personne qui manie une souris pour la première fois sera plus susceptible de faire des remarques au sujet de celle-ci plutôt que sur le fond du système testé [4].

1.2.3 Adéquation aux recommandations

Méthode

Il s'agit d'examiner la conformité de l'interface avec une série de règles d'ergonomie reconnues qui devraient être respectées lors du développement d'une nouvelle interface. Pour chaque projet, on peut faire appel à plusieurs types de recommandations. On retrouve les plus générales, applicables à n'importe quel type d'interface-utilisateur ou d'autres plus spécifiques à un type de système particulier[1], jusqu'à celles qui sont spécifiques au produit en lui-même [8].

[1] On citera pour exemple les lignes de conduite Eclipse

Il est possible de trouver facilement des recommandations générales. Pour les règles plus spécifiques, il faut chercher plus en profondeur. En effet, Internet regorge d'informations sur le sujet et permet de mettre la main sur de nombreux recueils. Ceux-ci constituent souvent des agrégats de règles provenant de projets antérieurs et qui ont été généralisées afin de pouvoir s'adapter à d'autres cas semblables.

Ces différentes règles sont désignées par une multitude de termes tels que : normes, recommandations, lignes de conduites, standards, critères ergonomiques.

Les critères ergonomiques représentent des concepts plutôt abstraits tels que « la prévention des erreurs », tandis que les recommandations sont plus proches de ce que l'on peut retrouver dans une interface (par exemple : « apparition d'un message en cas d'oubli d'une zone de saisie obligatoire »). Les critères sont donc composés d'un ensemble de recommandations. Cependant, pour simplifier, certaines listes présentent parfois ces recommandations réorganisées par catégories. A titre d'exemple, on trouve la liste du BCHI [30, 38] qui regroupe un ensemble de 107 recommandations réparties en huit catégories (d'autres exemples se trouvent en annexes) :

Liste de vérification du BCHI

1. Conception globale d'un site

1) Le contenu informationnel d'un site doit être représentatif
2) Inviter le visiteur à ajouter un signet
3) La conception d'un site doit être cohérente
4) Penser à une structuration appropriée du site
5) Utiliser la hiérarchie avec soin
6) Utiliser la clôture transitive
7) Utiliser des sous-menus pour les sites volumineux
8) Prévoir nécessairement une page d'accueil
9) Prévoir éventuellement une page d'invitation
10) Prévoir similairement une page de clôture
11) Prévoir une option d'impression pour les grands sites
12) Prévoir une section « Nouveautés » évidente
13) Adopter une logique de dénomination du site
14) Adopter une logique de dénomination des fichiers du site
15) Adopter une logique d'organisation des fichiers du site
16) La conception du site doit être appropriée au contexte
17) La conception d'un site doit supporter plusieurs navigateurs
18) La conception d'un site doit viser l'ergonomie en priorité
19) Préparer les pages pour les moteurs de recherche externes
20) Ajouter un moteur de recherche interne

21) Autoriser une recherche progressivement plus détaillée
22) Assurer une certaine flexibilité linguistique

2. Navigation

23) L'accès à l'information doit être rapide
24) Considérer le temps d'accès à chaque page
25) Utiliser une carte du site pour dénoter les relations entre les informations
26) Utiliser une image réactive comme carte graphique de navigation
27) Ne pas restreindre la navigation aux graphiques
28) Inclure des repères de navigation
29) L'accès à l'information doit être rapide
30) Les repères de navigation doivent être cohérents
31) Les liens doivent pointer vers un contenu informationnel substantiel
32) Fournir des liens textuels pour chaque page
33) Les liens textuels doivent être clairs
34) Les liens textuels doivent être courts
35) Utiliser "Précédent/Suivant", "Avant/Arrière" avec précaution
36) Bannir les liens "Cliquer ici"
37) Interdire les liens "Retourner à..."
38) Tout visiteur doit distinguer un lien visité d'un lien nouveau
39) Informer le visiteur de la teneur du lien
40) Utiliser des liens intra-page dans les longues pages
41) Veiller à la maintenance des liens

3. Présentation

42) L'information la plus importante doit être placée en premier lieu
43) La présentation de toute page doit être cohérente
44) La page d'accueil doit tenir sur un seul écran
45) Ne pas contraindre physiquement la présentation de la page d'accueil
46) L'information doit être complète sur le même écran
47) Trancher entre des pages courtes ou longues
48) Utiliser une grille de présentation
49) Utiliser les techniques visuelles
50) Utiliser au maximum les tables
51) Les lignes textuelles du contenu informationnel doivent être courtes, structurées
52) Penser aux feuilles de style
53) Utiliser de l'espacement blanc
54) Être draconien avec le défilement
55) Les informations liées sémantiquement doivent être présentées conjointement

4. Formulaires, titres et en-têtes

56) Les informations à acquérir doivent être présentées dans un formulaire
57) Sélectionner les objets interactifs de manière appropriée
58) Toute page doit comporter un titre identifiant
59) Utiliser des en-têtes pour accélérer le parcours des informations
60) Pour les documents structurés, communiquer la structure au visiteur

5. Cadres et fenêtres

61) Utiliser les cadres avec parcimonie
62) Utiliser les fenêtres seulement si nécessaire
63) Le nombre de cadres, de fenêtres doit être limité
64) Utiliser les cadres, les fenêtres de manière appropriée
65) Veiller à la taille des cadres et fenêtres

6. Graphiques

66) Utiliser les graphiques de manière appropriée
67) Adjoindre du texte à chaque graphique
68) Utiliser le texte alternatif
69) Réutiliser les mêmes graphiques
70) Utiliser les graphiques pour représenter les zones du site
71) Recourir aux graphiques pour les en-têtes
72) Utiliser les graphiques pour les listes
73) Garder en tête les limites des graphiques
74) Utiliser les miniatures
75) Les dimensions des graphiques doivent être appropriées
76) Les graphiques doivent être simplifiés
77) Utiliser les GIF entrelacés, les images fractales
78) Ne pas ajouter de bords aux graphiques
79) Utiliser des graphiques transparents sur un fond d'écran tramé
80) Spécifier les dimensions d'un graphique en HTML

7. Éléments multimédia

81) Le type de fond d'écran doit être approprié
82) Préférer les fonds d'écran clairs, de basse intensité
83) Éviter les fonds d'écran avec motifs
84) Sélectionner un fond d'écran aléatoire
85) Éviter le texte en fond d'écran
86) Utiliser une couleur de fond avec le fond d'écran
87) Tester en vraie grandeur les fonds d'écran
88) Maintenir un haut contraste entre les éléments
89) Les zones de couleur peuvent être larges
90) Réduire la profondeur des couleurs
91) Utiliser les 216 couleurs principales
92) Éviter le dithering
93) Tester les couleurs
94) Utiliser les polices sans sérif pour la lecture en ligne
95) Prêter attention à la typographie
96) Prévoir les variations de taille de police
97) Recourir à des ressources multimédia de manière appropriée
98) Penser à une présentation assistée par ordinateur
99) Considérer des animations simples en GIF animé
100) Éviter les animations répétitives
101) Implémenter un système de désactivation des animations, du son

102) La page courante doit demeurer le centre d'intérêt

8. Accessibilité

103) Rendre accessibles aux visiteurs handicapés tous les éléments du site
104) Rendre accessibles les caractères et le fond d'écran
105) Recourir à des marqueurs hypertextes
106) Minimiser le nombre de liens
107) Rendre les liens accessibles

La différence entre les standards et les recommandations réside dans le fait que les premiers précisent comment une interface doit apparaître aux utilisateurs tandis que les seconds fournissent des conseils sur certains critères d'ergonomie qui devraient être respectés. En général, les standards appliquent les règles principales d'ergonomie afin que le système mis au point puisse être aussi efficace que possible.

Les normes sont quant à elles des recommandations édictées par un organe reconnu par la communauté comme ayant une autorité en la matière[2].

Avantages

Il n'est pas nécessaire d'être en présence des utilisateurs pour examiner l'adéquation avec les recommandations. De plus, celles-ci sont assez nombreuses sur Internet pour en trouver qui conviennent à l'application étudiée. N'importe quelle personne qui s'investit peut avoir accès à toute l'information nécessaire pour créer une véritable interface pratique et simple d'usage.

Inconvénients

La quantité de recommandations en tout genre et le manque d'uniformité dans leur dénomination peut décourager. Il faut déjà du temps pour se renseigner afin de savoir ce qu'on recherche et, une fois trouvés, ces recueils sont en général longs à examiner. Par exemple, une personne voulant étudier la conformité de son application avec les règles de Smith et Mosier [34] devrait prendre le temps de parcourir l'ensemble des 944 recommandations, qui ne sont pas toujours claires.

Il n'est pas rare d'avoir des documents de plus de 400 pages qui les décrivent, ce qui ne favorise pas leur consultation pendant l'élaboration du projet. Cette étape est souvent remise à plus tard

[2] On citera pour exemple les normes ISO

ou simplement ignorée car le travail nécessaire afin de trouver les critères pertinents ne semble pas en valoir la peine. Les différentes lignes de conduite évoluent selon les auteurs qui s'en servent, manquent de cohérence les unes avec les autres, manquent parfois de détails, de documentation ou d'illustrations et, enfin, sont rarement faciles d'utilisation [23].

Lorsque les développeurs en ont connaissance, leur mise en place de façon correcte est difficile car ils ne sont pas toujours aptes à reconnaitre les problèmes potentiels qui pourraient apparaitre lorsque les guides de conduites sont enfreints [7].

Souvent, les concepteurs ne parviennent pas à se mettre à la place de l'utilisateur lorsqu'ils mettent le site en place avec les caractéristiques qui lui sont nécessaires. Bien qu'ils aient connaissance des règles d'ergonomie et parviennent à identifier certains critères à respecter, les exprimer de façon correcte est rarement un succès [8].

1.2.4 Test d'utilisabilité

Il s'agit d'une méthode, aussi appelée test d'utilisateur, permettant d'obtenir les réactions des utilisateurs. Leurs comportements sont observés et enregistrés dans un environnement contrôlé. Les utilisateurs effectuent des tâches qui reflètent la façon dont ils utiliseront l'application dans une situation réelle et leur comportement est enregistré. Cela peut aller d'un simple enregistrement vidéo à l'enregistrement des entrées clavier, du mouvement des yeux, de la recherche d'aide ou des expressions faciales [2].

La récolte de données se fait en général dans un environnement contrôlé par l'enregistrement de l'ensemble des actions, par interview ou par questionnaire. Ces données sont ensuite examinées par des experts qui déterminent la manière dont les utilisateurs ont fait usage de l'interface et dans quel but. Ils essayent à la fois de découvrir la cause de certaines erreurs commises par les utilisateurs et d'expliquer la durée de certaines actions. Un rapport répertorie l'ensemble des remarques ainsi que les solutions qu'il pourrait être intéressant d'apporter aux problèmes rencontrés.

En pratique, on constate que ce type d'analyse est souvent effectué assez tard dans le développement, bien qu'elles puissent avoir lieu dès qu'il existe un premier prototype, ou une maquette de l'interface [4].

Avantages

Ce test représente la méthode la plus efficace dans l'analyse ergonomique car les données obtenues proviennent directement de l'examen des utilisateurs interagissant avec l'application.

Le test consiste à demander à quelques utilisateurs d'effectuer des tâches essentielles. En observant les utilisateurs dans ce contexte, il est possible de déterminer plus exactement à quel niveau se situe les principales lacunes ergonomiques en détectant immédiatement les difficultés rencontrées et les erreurs commises pour tenter de parvenir à l'exécution d'une tâche.

L'observation permet de recueillir de nombreuses indications qui permettront de mettre l'application en adéquation avec les besoins réels des utilisateurs [2] ainsi que des suggestions de leur part sur ce qu'ils voudraient changer [10].

Inconvénients

Ce type de test est relativement long, surtout entre la réalisation et la fin de l'examen des résultats. Il se peut que l'interface ait déjà été modifiée entre-temps. Le matériel nécessaire est également plus important, allant de la simple caméra à un système plus avancé.

Afin que les observations soient réellement pertinentes, le profil des utilisateurs potentiels doit être établi au préalable pour trouver les testeurs qui s'en rapprochent le plus, tant au niveau de l'âge que du niveau de connaissance ou encore d'éducation. Cela demande un investissement plus conséquent en temps, vu qu'il faut effectuer une sélection [10].

Un autre désavantage de ce type de méthode se situe dans l'élaboration et la réalisation du test en lui-même. Il s'agit d'une étape nécessaire mais qui reste chronophage et onéreuse dans la plupart des cas. Il faut mettre en place un ou plusieurs scénarios qui vont permettre à tous les participants d'effectuer l'examen dans les mêmes conditions. Ces scénarios décrivent le déroulement du test, les buts poursuivis ainsi que les différentes actions qui devront être effectuées.

Il faut également cerner précisément les différents objectifs et fonctions qui doivent être évalués afin de permettre l'élaboration d'un test efficace. Ces objectifs sont à la fois qualitatifs et quantitatifs. Il s'agit de découvrir ce qui manque d'efficacité dans l'interface proposée, que ce soit du point de vue de la difficulté à achever une tâche ou du temps nécessaire à l'accomplir en passant par les potentiels problèmes d'apprentissage.

1.2.5 Marche cognitive

Méthode

Cette méthode s'appuie sur les connaissances d'évaluateurs experts dans le domaine. Ils vont tenter de détecter les problèmes susceptibles de se poser aux utilisateurs sur base d'une description du système et en se concentrant sur l'objectif à atteindre.

Le concepteur de l'interface présente un ensemble de tâches qui vont être effectuées pour évaluer le système. Ensuite, il présente les différentes actions que l'utilisateur peut faire afin de parvenir à effectuer chacune des tâches. Le concepteur simule l'activité cognitive d'un utilisateur en utilisant une série de questions qui reflètent le cheminement de pensée de l'utilisateur qui tente d'atteindre son but. Il s'agit d'une sorte de jeu de rôle dans lequel l'évaluateur se prend pour un utilisateur [9].

Lors de la phase de test, les spécificités du programme sont présentées avec un scénario, et une description des futurs utilisateurs potentiels ainsi qu'une description des conditions dans lesquelles l'interface sera utilisée.

Pour illustrer, imaginons que la tâche principale est de remplir un verre de jus. Cela représente un ensemble d'actions à effectuer (prendre le récipient, prendre le verre, ouvrir le récipient, pencher le récipient,…). Si un examinateur doit connaître le profil des utilisateurs c'est pour concevoir l'interface de façon cohérente. Pour continuer dans le même exemple ; il faut connaître le type de récipient utilisé pour se poser le questionnement adéquat; une bouteille, un conditionnement en carton,…

L'examinateur se posera alors les questions de savoir si, par rapport au but recherché, un utilisateur tenterait réellement d'ouvrir le récipient. Dans le cas où d'une réponse positive, le ferait-il ? Si oui, de quelles connaissances/outils a-t-il besoin pour y parvenir. Et enfin, en disposent-ils ? L'examen des réponses apportées à ces questions permettra de mettre en avant différents problèmes à corriger dans l'interface.

Avantages

Cette méthode permet de détecter des problèmes de conception avant même le début du projet ou de la fabrication d'un prototype efficace puisqu'il s'agit essentiellement de tester le programme sur base d'une simple description des fonctionnalités de l'application. Elle permet de se

concentrer sur les éléments essentiels de la conception telle que la structuration des tâches mais elle nécessite de s'adapter correctement au profil de l'utilisateur pour être efficace.

Inconvénients

L'interprétation des résultats de ce type d'étude nécessite souvent des connaissances avancées en ergonomie étant donné que c'est souvent l'expérience de l'évaluateur qui permet de faire fonctionner correctement l'analyse. Il faut donc disposer d'évaluateurs qualifiés pour la mise en œuvre efficace d'une étude qui semble relativement simple, ce qui augmente sensiblement les coûts.

1.2.6 Analyse GOMS (Goals, operators, methods, selection rules)

Méthode

Ce type d'analyse est basé sur la psychologie cognitive qui conçoit le cerveau humain comme un processeur avec une capacité limitée. Il s'agit d'un modèle décrivant le comportement attendu d'un utilisateur expert.

Selon cette méthode, c'est la tâche à effectuer qui détermine le comportement adopté. Elle se concentre donc sur la prévision des comportements et l'identification des tâches les plus importantes. L'examen des différentes tâches disponibles permet donc d'obtenir un canevas et des objectifs pour celui qui est chargé du développement.

Ce modèle prend en compte certains facteurs dont la performance peut être mesurée comme le temps d'exécution des opérations, les entrées claviers ou les déplacements de la souris, mais également les questions que l'utilisateur se pose sur l'utilisation qu'il effectue, comme par exemple décider d'utiliser un raccourci clavier plutôt que d'accéder à un menu.

L'examinateur fixe un objectif en ayant préalablement listé les actions qui permettent d'y parvenir (*operators*) et en leur attribuant un temps de réalisation. Un temps est également attribué pour le mécanisme de décision. La méthode consiste à assembler ces différentes actions pour parvenir au but fixé en sélectionnant un moyen d'arriver au but. L'analyste peut ensuite estimer le temps nécessaire pour effectuer l'ensemble des actions en additionnant le temps de chaque étape afin de déterminer la plus efficace [11].

Avantages

Cette méthode devrait théoriquement pouvoir prédire l'utilisabilité d'une interface avant même que celle-ci ne soit opérationnelle. Cela devrait permettre d'éviter les tests avec utilisateurs mais également d'estimer les différences entre plusieurs designs potentiels sans avoir à les réaliser.

Inconvénients

Le principal inconvénient de cette méthode est qu'il ne prend pas en compte l'erreur. En effet, ce modèle se base sur le calcul du temps nécessaire pour un enchainement d'actions courtes afin de parvenir à un objectif par tous les chemins disponibles, mais sans prendre en compte le fait qu'un utilisateur pourrait se tromper et enchainer différentes actions qui ne le rapprocherait pas de son but [11].

1.3 Positionnement de la méthode heuristique

Les différentes méthodes présentées peuvent être classées de différentes façons et selon plusieurs critères. Ces classifications ont notamment pour intérêt de positionner les méthodes les unes par rapport aux autres.

Premièrement, les méthodes peuvent être décomposées en deux groupes : les méthodes *analytiques*, dont la méthode heuristique fait partie, et les méthodes *expérimentales*. Les méthodes analytiques permettent d'effectuer une évaluation sans faire appel aux utilisateurs finaux, tandis qu'ils sont essentiels à la bonne conduite de l'analyse pour les méthodes expérimentales. Cette intervention extérieure de l'utilisateur final permet de détecter des défauts ergonomiques qui lui posent de réels problèmes et qui auraient pu être ignorés lors d'une analyse par une méthode analytique. En effet, les méthodes analytiques impliquent qu'un expert est amené à supposer ce qui est le plus utile pour l'utilisateur et quel comportement il est susceptible d'adopter.

Il est également possible de classer ces méthodes selon le type de données collectées. On retrouve d'une part les données *qualitatives*, principalement obtenues par les méthodes analytiques, et d'autre part, les données *quantitatives*, qui représentent un ensemble de données chiffrées et mesurables dans un environnement contrôlé. Celles-ci sont obtenues auprès des utilisateurs finaux à l'aide de méthodes expérimentales, bien que parmi celles-ci, certaines comme le test utilisateurs donnent la possibilité d'obtenir des données tant qualitatives que

quantitatives.

Par contre, l'évaluation du système par la méthode expérimentale devient ainsi dépendante des utilisateurs choisis. Comme la capacité à évaluer correctement un système est variable d'un utilisateur à l'autre, un utilisateur moins efficace peut ainsi fausser l'ensemble d'une analyse.

L'avantage inhérent à l'observation de l'utilisateur est que cela permet d'avoir une meilleure connaissance de son comportement et de ce qu'il fait réellement avec l'interface (*behavioral*). Les méthodes sans utilisateur se fondent essentiellement sur ce que l'utilisateur prétend faire, mais pas réellement sur ce qu'il fait (*attitudinal*). Souvent, des divergences entre les deux sont constatées.

Chaque type d'analyse est susceptible d'intervenir dans un contexte différent. On peut en définir quatre ; les analyses pouvant être menées sans rien, celles effectuées sur un prototype papier, celles qui nécessitent que le développement logiciel de l'application soit suffisamment avancé et enfin, celles qui mêlent les trois contextes précédents.

La méthode heuristique étant une méthode analytique, elle a l'avantage de ne pas nécessiter la présence des utilisateurs finaux. De plus, elle peut être effectuée à n'importe quel moment du processus de développement de l'application par son concepteur, que ce soit dès l'existence d'un prototype papier jusqu'à la version finale du programme. Cependant, son usage est le plus intéressant en début de projet et, lorsqu'une version finale existe, elle n'est plus aussi avantageuse qu'un test d'utilisateur.

Bien que cette méthode permette de détecter de nombreux problèmes, elle ne permet pas de les corriger tous. De plus, les problèmes trouvés ne sont pas toujours ceux qui doivent être corrigés en priorité. Ainsi, au-delà d'un certain stade dans le processus de développement d'une interface, il devient nécessaire d'avoir l'avis d'un utilisateur final pour en améliorer l'ergonomie [43].

Selon le stade de développement du système, le type d'information que l'on souhaite obtenir d'une analyse et les ressources disponibles, certains types d'examens seront plus efficaces que d'autres. La Fig. 1.3 reprend ainsi différentes méthodes d'analyse existantes et illustre leur position les unes par rapport aux autres. On peut voir que la méthode heuristique[3] est une méthode qualitative pouvant être employée de façon très large, mais qu'elle ne suffit pas à elle seule pour comprendre le comportement réel des utilisateurs face à l'interface.

[3] La méthode heuristique est reprise dans le champ « Ethnographic field studies » dans la Fig. 1.3

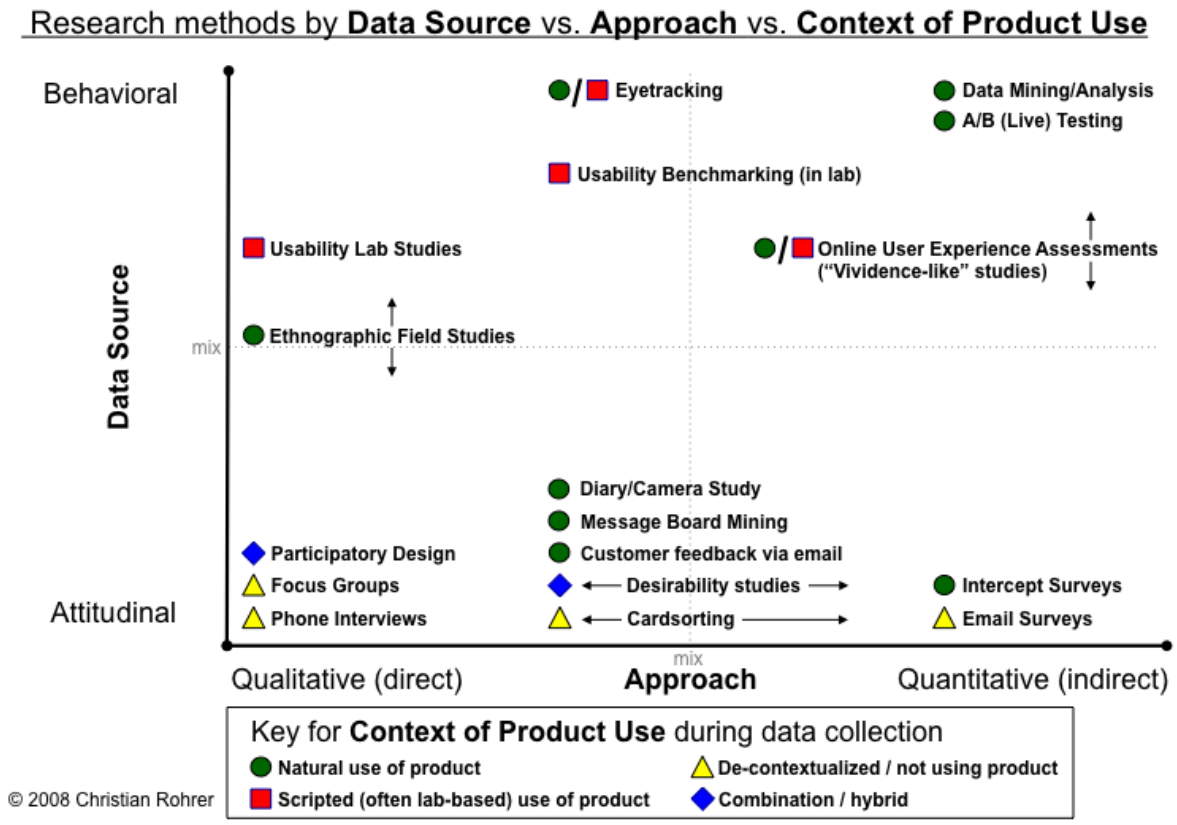


Fig. 1.3: C. Rohrer, Source: http://www.useit.com/alertbox/user-research-methods.html

En résumé, nous pouvons affirmer que, lors de la création des premières ébauches de l'interface, il faut privilégier les méthodes analytiques et se concentrer sur ce que les utilisateurs demandent, sans pour autant faire appel à eux. Ces méthodes peuvent être appliquées alors que l'interface n'est qu'au stade de projet. Par contre, plus l'interface évolue vers sa version finale, plus il faut privilégier les analyses expérimentales directement menées sur le prototype développé. Lorsque la version finale est proche, il devient logiquement plus intéressant de faire intervenir les utilisateurs finaux [44].

2 Evaluation heuristique

2.1 Description

2.1.1 Concept

L'évaluation heuristique consiste à examiner une interface pour en découvrir les bons et mauvais aspects. En principe, cette évaluation est menée en se basant sur certaines règles reconnues comme des recommandations générales d'ergonomie. Il n'y a donc pas d'intervention des utilisateurs dans cette méthode.

Lors de la conduite de l'examen, les analystes n'utilisent pas l'interface en tant que telle et l'évaluation peut donc être menée alors qu'il n'existe pas encore de version fonctionnelle. Il suffit d'un prototype comme une maquette ou une version papier, ce qui signifie qu'il est intéressant de faire intervenir très tôt ce type d'évaluation dans le développement de l'application.

En général, on constate que les développeurs prennent rarement le temps d'effectuer ce type d'analyse, pourtant mis au point afin d'être facilement utilisé et peu couteux. Cela est principalement dû au fait que certains recueils comprennent parfois jusqu'à un millier de règles à respecter, ce qui intimide les développeurs. Ceux-ci estiment que le temps nécessaire pour trouver ces règles, les comprendre, et examiner l'interface au regard de celles-ci représente un investissement trop important par rapport au bénéfice qu'ils pourraient en retirer. Néanmoins, la plupart effectuent probablement une sorte d'évaluation heuristique sur base de leur propre intuition et de ce qu'ils estiment efficace [7].

2.1.2 Listes d'évaluation

Selon Nielsen, il s'agit de trouver les problèmes d'ergonomie dans une interface par un examen systématique de celle-ci afin de corriger les problèmes avant de soumettre le système à une nouvelle analyse. Afin de découvrir les problèmes, il est nécessaire qu'un petit nombre d'évaluateurs examinent l'adéquation du système avec un ensemble de principe ergonomique reconnu, les « heuristiques ».

Nielsen a mis en avant dix heuristiques essentielles en matière d'ergonomie des interfaces homme-machine [12]. Il s'agit en réalité plus de règles logiques que de recommandations

ergonomiques spécifiques. Il s'agit [13] :

- De la **visibilité du statut du système**. Un utilisateur doit toujours savoir ce qu'il se passe dans l'application et ce dans un délai raisonnable.
- D'une **correspondance entre le système et le monde réel**. L'interface doit utiliser le langage de l'utilisateur. Elle doit faire appel à des concepts qui lui sont familiers.
- Que l'**utilisateur soit libre de ses mouvements**. A tout moment l'utilisateur doit avoir la possibilité de revenir en arrière. Il faut également qu'il puisse quitter rapidement un état dans lequel il serait arrivé par erreur.
- D'**avoir une interface constante et qui respecte les standards**. L'utilisateur ne doit pas se demander si plusieurs actions peuvent signifier la même chose, ou si une action peut avoir différents effets.
- De **prévenir les erreurs**. Une interface qui prévient qu'une erreur puisse se produire lors de la réalisation de certaines actions est plus efficace que l'apparition d'un message d'erreur.
- De donner à l'utilisateur **la possibilité de reconnaitre plutôt que de se souvenir**. Il faut minimiser l'appel à la mémoire en plaçant de façon visible les différents éléments importants avec lesquels l'utilisateur peu interagir. L'utilisateur ne doit pas se souvenir de quelque chose qu'il a vu sur une page de l'application à un autre endroit pour utiliser le système de façon correcte. L'information doit être disponible là où elle est nécessaire.
- D'avoir un **système flexible et efficace à utiliser**. Il faut la possibilité pour les utilisateurs habitués au système d'avoir des raccourcis pour les actions les plus fréquentes, afin d'être efficaces tant pour les experts que pour les novices.
- D'être **esthétique avec un design minimaliste**. L'information présentée doit être pertinente afin de ne pas diminuer la visibilité de l'information réellement recherchée.
- D'aider l'utilisateur à **reconnaitre, diagnostiquer et réparer les erreurs**. Les messages d'erreurs doivent être compréhensibles, indiquer le problème en cause et suggérer une solution afin d'y remédier.
- De fournir une **aide et de la documentation**. Même s'il est plus intéressant que le système puisse être utilisé de façon intuitive sans documentation, il faut que n'importe quelle information susceptible d'être nécessaire soit facilement accessible.

Ces heuristiques ont constitué la base de l'évaluation durant de nombreuses années et, même si de nombreuses listes de recommandations ont vu le jour depuis, ces heuristiques représentent la première étape dans cette démarche et restent toujours d'actualité.

A présent, on trouve de nombreux recueils de critères ergonomiques et des *checklists* de recommandations dont le but est de faciliter la conduite d'un examen ergonomique. Chaque critère est décomposé en une série de règles qui permettent de savoir si, après vérification, les critères sont respectés. La majeure partie de ces recueils et nouvelles listes puisent leur inspiration dans les heuristiques de Nielsen, en les adaptant aux cas plus particuliers [14].

Il existe cependant certaines heuristiques développées pour s'adapter à des cas plus concrets qui peuvent fournir des résultats plus avantageux, mais l'inconvénient est qu'il faut en général les développer soi-même en fonction des critères les plus importants de l'interface [14].

Si les experts se basent sur différentes listes de critères pour effectuer l'analyse heuristique, c'est pour que celle-ci puisse être effectuée dans un cadre objectif. Cela permet d'éviter une évaluation trop subjective, l'interprétation des recommandations faisant déjà l'objet d'une certaine subjectivité en fonction du vécu de l'expert.

2.2 Méthode

2.2.1 Evaluation

En général, une analyse menée par un seul évaluateur dure entre une et deux heures. Ce temps peut être plus long pour une interface plus complexe, mais il semble préférable dans ce cas de séparer l'examen en plusieurs sessions d'évaluation chacune concentrée sur une partie précise de l'interface.

Avant de commencer l'examen, il faut préciser le modèle d'évaluation choisi et à partir de celui-ci, décider dans quelles conditions on estime qu'une règle est enfreinte ou respectée.

La forme la plus élémentaire d'évaluation serait de choisir aléatoirement une page d'un site ou d'une interface et de demander aux évaluateurs d'examiner si les heuristiques choisies sont en adéquation avec cette page. Ils devraient facilement pouvoir dire où ils se situent dans le site et ce qu'ils peuvent faire.

Pendant l'examen, l'évaluateur parcourt l'ensemble du système à plusieurs reprises en examinant les différents éléments et leur conformité avec la liste des heuristiques d'ergonomie reconnues.

Celles-ci sont des règles censées décrire les propriétés ergonomiques générales des interfaces. De plus, l'évaluateur peut également prendre en compte les principes d'ergonomie qui lui viennent à l'esprit et qui pourraient être utile pour l'application [16].
En principe, l'évaluateur effectue l'évaluation comme il le décide. Néanmoins, il est conseillé de parcourir au moins deux fois l'interface. Une première fois afin de se familiariser et avoir un aperçu global du système et une seconde fois afin d'examiner plus en détail certains éléments plus spécifiques de l'interface, avec l'avantage de savoir comment ils interagissent avec le reste grâce à la première inspection [17].

2.2.2 Compte rendu

Une fois l'interface évaluée, l'examinateur rédige un rapport listant les différents problèmes rencontrés et les principes d'ergonomie enfreints respectifs. Ce test reste subjectif car c'est l'examinateur qui estime si une règle a ou non été enfreinte. De plus, cette méthode ne fournit pas de solutions systématiques pour résoudre les problèmes découverts. Toutefois, certains problèmes sont faciles à corriger lorsqu'ils ont été identifiés et il est donc relativement aisé de réviser au moins partiellement l'interface [4].
Une fois les rapports des différents évaluateurs reçus, ceux-ci sont compilés afin de parvenir à un rapport général contenant une liste de problèmes (et les commentaires correspondant) qui devraient être corrigés afin d'augmenter l'ergonomie de l'application.
Pour augmenter l'efficacité, il est intéressant d'effectuer un débriefing après l'analyse. Celui-ci regroupe alors les examinateurs et observateurs qui étaient présents ainsi que les développeurs. Cette séance doit avoir pour but d'examiner les solutions qu'il serait possible d'apporter aux problèmes rencontrés. Pour le développeur, il est important de bien comprendre chacun des problèmes mentionnés et de ne pas tenter d'expliquer que, pour certains problèmes, il existe une raison quelconque à leur existence. Si un examinateur estime qu'un aspect de l'application est perturbant, il y a fort à parier que de nombreux utilisateurs penseront la même chose.
Cette réunion peut aussi être l'occasion de mentionner les aspects positifs de l'interface, qui ne sont pas mentionnés dans la méthode heuristique [4, 17].

2.3 Variantes

2.3.1 Déroulement du test

1. *Multiplicité des évaluateurs*

Il existe plusieurs pratiques pour le déroulement du test. L'évaluation peut être menée de façon individuelle ou en groupe.

Dans le premier cas, chaque évaluateur examine l'interface et prend note des problèmes constatés et les résultats sont ensuite remis ensemble. L'avantage de ce système est que les évaluateurs ne doivent pas spécialement être présents pour effectuer l'examen. Ils peuvent l'effectuer de chez eux et envoyer les résultats par Internet. De plus, cette méthode permet de découvrir plus de problèmes que l'examen en équipe. L'inconvénient est qu'il faudra plus de temps pour tout remettre ensemble.

Dans l'évaluation de groupe, les experts travaillent en équipe. Ils parcourent chaque étape de l'interface tous en même temps. Ils ne doivent pas tous être d'accord sur les problèmes découverts, mais chacun doit être enregistré. L'inconvénient de cette méthode est qu'il faut que tous les examinateurs puissent être présents le même jour [16].

2. *Nombre d'évaluateurs*

En principe, un seul évaluateur peut mener à bien ce type d'analyse. Toutefois, Nielsen a découvert que dans la majorité des cas, un seul examinateur manque la majeure partie des problèmes ergonomiques. En fait, il semblerait qu'il n'en trouve en moyenne que 35%.

Lorsqu'un second évaluateur effectue l'analyse, il redécouvre certains défauts qui avaient déjà été découverts lors de la première analyse, mais il en découvre également de nouveaux. Cela augmente avec le nombre d'évaluateurs effectuant l'analyse. Il semblerait qu'au-delà de cinq évaluateurs, l'ajout d'un évaluateur supplémentaire n'apporte plus de résultat réellement significatif [18].

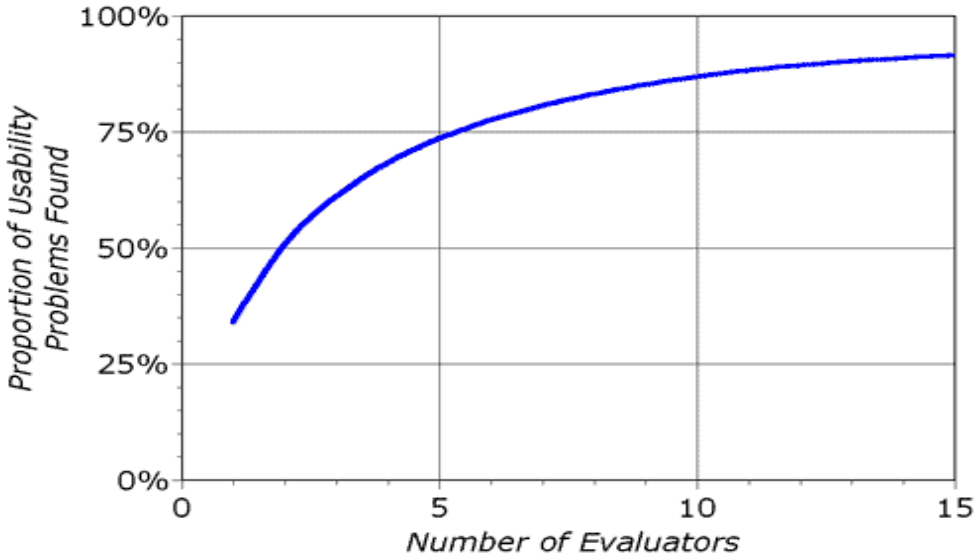


Fig. 2.1: J. Nielsen, Source : http://www.useit.com/papers/heuristic/heuristic_evaluation.html

La Fig. 2.1 montre qu'il est clairement plus avantageux d'utiliser plus d'un évaluateur. Cependant, la décision du nombre d'évaluateurs doit faire l'objet d'une analyse approfondie. Par exemple, lorsque l'ergonomie est réellement importante pour le projet et que des avantages substantiels peuvent être dégagés, il peut être intéressant de faire appel à plus de cinq évaluateurs. Dans le cas contraire, il vaut mieux se limiter à trois utilisateurs et avoir la possibilité de recommencer plusieurs fois l'analyse à différentes étapes du projet.

Si cette méthode est utilisée avec un seul évaluateur, elle ne permet pas, la plupart du temps, de découvrir beaucoup de défauts. Employée avec plusieurs évaluateurs, elle se révèle par contre être l'une des plus efficaces.

3. Compétence des évaluateurs

Il existe des différences selon les évaluateurs qui effectuent les analyses. Dans certains cas, des évaluateurs avec la même formation et du même milieu trouvent un nombre d'erreurs fort différent. Il peut donc y avoir des divergences dans les capacités à découvrir des problèmes. Les études semblent montrer qu'il existe des évaluateurs plus doués que d'autres pour découvrir des problèmes d'utilisabilité et qu'il serait avantageux de pouvoir identifier les bons évaluateurs. Cependant, pour y parvenir, il faut disposer de temps et sélectionner les plus efficaces lors de chaque analyse, ce qui représente un procédé relativement lent mais qui en plus diminue les performances des premières études effectuées [7].

Nielsen et Molich avaient initialement suggéré que les évaluateurs potentiels devaient disposer de connaissances en matière d'ergonomie sans pour autant être des experts. Néanmoins, ils ont eu l'occasion d'observer que les experts en ergonomie découvraient plus de problèmes que les non experts, et que les doubles experts (en ergonomie et dans le domaine étudié) étaient encore plus performants [19]. Cependant, ce résultat doit être nuancé. En effet, il semblerait que les experts dans le domaine étudié laissent passer un certain nombre d'erreurs et en ajoutent d'autres, qui sont plus des choix de préférence, par rapport à ceux qui n'ont pas de réelle expérience dans le domaine. Ceux-ci testent en général plus d'hypothèse tout en restant plus objectif [20].

Il faut mettre en avant le fait que, même sans formation spécifique, n'importe qui parvient à découvrir des erreurs, ce qui constitue déjà un avantage, comme l'indique la Fig. 2.2. Même si la méthodologie n'est pas optimale, il est quand même possible de découvrir quelques problèmes.

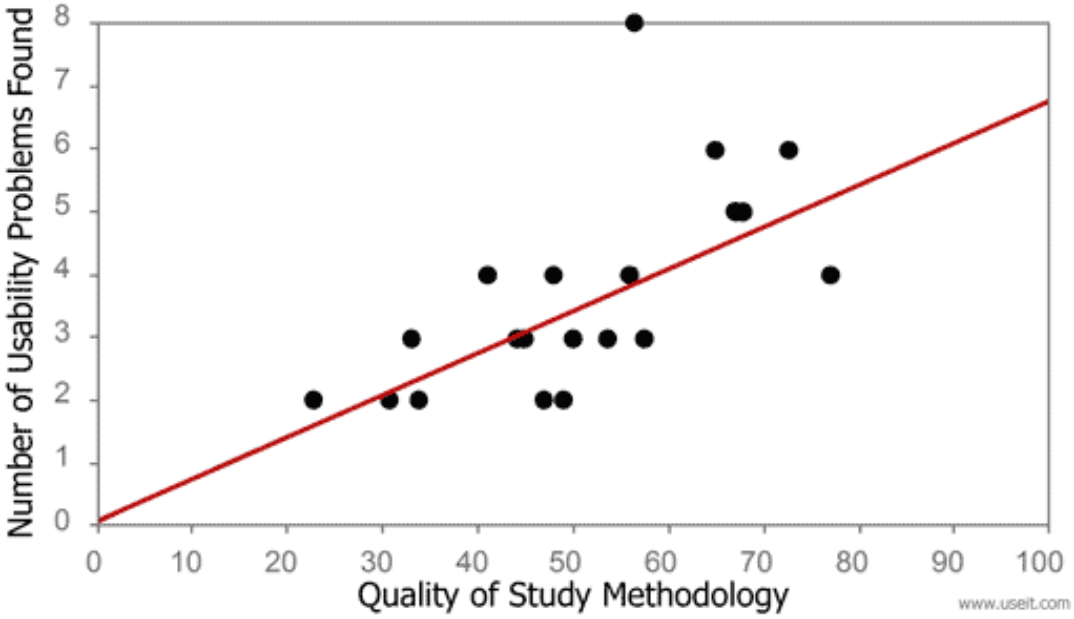


Fig. 2.2: J. Nielsen, Source : http://www.useit.com/papers/alertbox/robustness-usability-method.gif

2.3.2 Modèle avec ou sans importance

On distingue différents types d'analyse. Selon l'analyse effectuée, chaque principe ergonomique à la même valeur ou se voit attribuer un certain niveau d'importance.

Avec le modèle dit linéaire, on présume que toutes les règles ont la même importance tandis que dans le modèle avec niveau d'importance, certaines règles sont considérées comme plus importantes que d'autre, et les problèmes qui y sont liés doivent donc être traités en priorité. Ces

différents modèles seront expliqués plus en détail à la Section 3.2.

L'importance d'une règle est déterminée selon trois critères:

- La **fréquence**. Le problème apparait-il souvent?
- Les **conséquences**. Lorsque le problème survient, les utilisateurs ont-ils la possibilité de le résoudre facilement?
- La **persistance**. Une fois connu, le problème peut-il être évité, ou va-t-il survenir de façon régulière et périodique sans possibilité pour l'utilisateur de l'éviter?

Pendant l'évaluation, il est difficile pour les évaluateurs d'avoir une bonne opinion du niveau d'importance des problèmes qu'ils découvrent étant donné qu'ils sont principalement concentrés sur la recherche de nouveaux défauts [21].

Une solution est alors d'examiner la liste des problèmes découverts après l'évaluation, avec une brève description de chaque problème dans l'hypothèse où chaque évaluateur n'a pas découvert tous les problèmes.

Une autre solution peut être d'attribuer une pondération aux règles, avec un niveau d'importance plus fort pour les règles considérées comme essentielles et plus faible pour les autres.

Dans ces cas, il semblerait que trois personnes mettant en commun leur système de pondération puisse être acceptable pour la plupart des analyses [21].

2.3.3 Rapports

Les résultats peuvent être rédigés par écrit de la main de chaque évaluateur, ou alors il faut recourir à la présence d'un observateur lors de l'évaluation. Celui-ci va prendre note de ce que mentionnent les évaluateurs pendant qu'ils examinent le système.

L'avantage d'un rapport écrit est qu'il permet d'avoir un enregistrement de ce qui a été fait, malgré le fait que cela demande plus d'effort aux évaluateurs. De plus, l'ensemble des rapports doit être lu et collationné par une tierce personne.

Si rajouter un observateur augmente le nombre de personnes nécessaires pour l'évaluation, cela diminue la charge de travail des évaluateurs et donne l'opportunité d'avoir les résultats plus vite après la dernière évaluation puisque l'observateur doit simplement restructurer ses propres notes étant donné qu'il a assisté à l'ensemble des auditions. De plus, l'observateur peut assister l'évaluateur s'il a des difficultés avec l'interface ou n'est pas un expert dans le domaine [4].

La mission de l'observateur est simplement de noter les réflexions des examinateurs dans ce type

de test étant donné qu'il n'est pas chargé d'interpréter leurs actions.

Pour l'évaluation heuristique d'une application dans un domaine déterminé, l'observateur peut également intervenir et répondre aux questions des évaluateurs afin de leur permettre d'effectuer plus facilement l'analyse ergonomique tout en respectant les contraintes d'un système qu'ils ne maitrisent pas forcément [4].

De plus, l'observateur peut donner des indices si l'évaluateur a des problèmes dans l'utilisation de l'interface et qu'il en fait expressément la demande afin de gagner du temps dans l'analyse [17].

2.4 Pratiques

2.4.1 Checklists

Dans la pratique, on constate que de nombreux concepteurs évitent d'utiliser les recueils de recommandations lors du développement de l'interface. Cependant, cela ne signifie pas qu'ils se passent complètement d'une analyse, mais simplement qu'ils l'effectuent en se basant sur leur propre connaissance et sur ce qui leur semble logique.

On retrouve de nombreux sites qui permettent de faciliter la conduite d'une analyse heuristique en fournissant une *checklist* (ou liste de vérification) des critères les plus importants. Il suffit alors de cocher soit même les différentes réponses. Cependant, il reste encore à la personne qui a rempli cette grille toute la phase d'analyse des résultats. En effet, il n'y a généralement pas d'outil permettant de résumer l'ensemble de l'évaluation.

La plupart des questionnaires proposés pour aider les développeurs permettent généralement d'effectuer l'analyse à l'aide d'une échelle de Likert comme le montre la Fig. 2.3. Cela donne lieu à une analyse des données selon une approche paramétrique[4]. Or, l'analyse selon ce type d'approche ne semble pas être la plus efficace lorsque le nombre d'évaluateurs est limité. Les données obtenues avec une approche non paramétrique semblent même permettre d'augmenter la fiabilité lorsqu'il n'y a qu'un faible échantillon recueilli [22].

[4] Dans l'approche paramétrique, on considère que l'écart entre 1et 2 sur l'échelle de Likert est égal à l'écart entre 4 et 5 ou encore entre 6 et 7. Cependant, lorsque les personnes qui répondent à ce type de test sont questionnées, elles estiment qu'il y a de plus grand écarts aux extrémités de l'échelle qu'au milieu [22].

1. COMPATIBILITY		1	2	3	4	5	6	7		NA
1. Is the control of cursor compatible with movement?	Mauvais	◉	○	○	○	○	○	○	Bien	○
2. Are the results of control entry compatible with user expectations?	Mauvais	○	○	○	○	○	○	○	Bien	○
3. Is the control matched to user skill?	Mauvais	○	○	○	○	○	○	○	Bien	○
4. Are the coding compatible with familiar conventions?	Mauvais	○	○	○	○	○	○	○	Bien	○
5. Is the wording familiar?	Mauvais	○	○	○	○	○	○	○	Bien	○
2. CONSISTENCY		**1**	**2**	**3**	**4**	**5**	**6**	**7**		**NA**
6. Is the assignment of colour codes conventional?	Mauvais	○	○	○	○	○	○	○	Bien	○
7. Is the coding consistent across displays, menu options?	Mauvais	○	○	○	○	○	○	○	Bien	○
8. Is the cursor placement consistent?	Mauvais	○	○	○	○	○	○	○	Bien	○
9. Is the display format consistent?	Mauvais	○	○	○	○	○	○	○	Bien	○
10. Is the feedback consistent?	Mauvais	○	○	○	○	○	○	○	Bien	○
11. Is the format within data fields consistent?	Mauvais	○	○	○	○	○	○	○	Bien	○
12. Is the label format consistent?	Mauvais	○	○	○	○	○	○	○	Bien	○

Fig. 2.3: Exemple d'outil pour l'analyse heuristique sur Internet, Source : http://oldwww.acm.org/perlman/question.cgi

2.4.2 Logiciels d'évaluation automatique

On ne trouve que très peu de logiciels facilitant la conduite d'une analyse heuristique. Comme expliqué précédemment, la plupart des outils fournis aux concepteurs et disponibles sur Internet sont des *checklists* et de la documentation. Très peu d'outils logiciels sont présents pour venir en aide aux développeurs désireux de mener un examen de façon simple et rapide.

Il est possible de trouver quelques logiciels destinés à la gestion des bases de données dans lesquels il faudra encoder soi-même les différentes grilles de critères, mais il ne s'agit pas réellement d'un outil conçu pour aider les développeurs.

Il existe cependant un faible nombre de logiciels tels que *GUIguide* [24] qui fournissent un ensemble de *checklists* prédéfinies et la documentation correspondante, le tout provenant de différentes sources. Un des problèmes est qu'il faudra s'acquitter d'une licence avant de pouvoir l'utiliser, ce coût venant encore augmenter celui de la conduite de l'analyse.

De plus, la recherche de ce type de logiciel est compliquée par l'incroyable quantité d'information disponible. A moins d'avoir une idée très précise, d'en avoir entendu parler, ou beaucoup de patience, il est difficile de découvrir les mots-clés permettant de parvenir à un résultat réellement pertinent. La plupart des réponses obtenues n'étant essentiellement que de la documentation supplémentaire sur le sujet.

2.5 Avantages

Cette méthode est rapide à mettre en place, relativement simple à effectuer et elle peut être conduite dès les premiers stades de création d'une application. Elle permet à tout un chacun d'avoir l'opportunité de découvrir les défauts majeurs de son interface sans avoir besoin de faire appel aux utilisateurs potentiels, qui ne sont pas toujours faciles à contacter.

Son principal attrait reste les coûts relativement faibles en comparaison aux dépenses nécessaires pour le bon déroulement des autres études.

De plus, elle reste relativement intuitive, et avec un peu d'investissement, elle peut donner de très bons résultats. Elle permet de trouver un nombre très important de problèmes, même s'ils ne sont pas toujours parmi ceux à régler en priorité [27]. De plus, cette méthode peut être pratiquée par une personne sans aucune expérience, car l'information nécessaire existe en abondance et est facile à trouver [26]. Il est vrai qu'une personne sans expérience ne corrigera pas autant de problèmes que quelqu'un d'expérimenté, mais il est toujours mieux d'apporter seulement certaines corrections que pas du tout [7].

2.6 Inconvénients

Cette méthode laisse parfois passer des problèmes plus mineurs et ne permet pas toujours d'avoir une idée de l'importance de certains types de défaut, ce qui empêche parfois la résolution de problèmes prioritaires. De plus, la détection reste soumise à un certains degré de subjectivité. Elle dépend des compétences et de la façon dont l'expert aborde l'interface [27].

Le problème de ce type d'étude est qu'un expert essaie de se faire passer pour un utilisateur sans pour autant en être réellement un. Pour obtenir un vrai retour de ce que ressent un utilisateur face au programme, il est nécessaire de faire appel à d'autres types de test et d'impliquer les utilisateurs dans le développement de l'interface.

De plus, les problèmes découverts varient d'un expert à l'autre, même lorsque les deux appliquent les mêmes critères ergonomiques

Un autre problème de ce type d'étude est l'investissement nécessaire afin d'être capable d'effectuer des évaluations correctes. Les règles ne sont pas toujours compréhensibles à la première lecture, et se documenter pour connaître la signification de chacune d'elles peut prendre du temps.

Il ressort également que ce type d'examen ne permet pas d'identifier autant de problèmes ergonomiques que d'autres comme le test d'utilisabilité, qui permet de découvrir des erreurs pour lesquelles l'utilisateur est plus sensible.

Il faut également insister sur le fait que l'analyse heuristique ne semble pas être efficace pour les sites web 2.0, où le contenu est ajouté principalement par les autres utilisateurs. Les résultats d'une analyse heuristique dans ce type de cas montrent que ces sites manquent d'ergonomie et devraient poser des difficultés. Pourtant, lors des tests en présence d'utilisateurs, ceux-ci parviennent à accomplir l'ensemble des tâches qui leur sont soumises et sont plutôt satisfait de l'expérience qu'ils ont eue, leurs reproches essentiels étant adressés aux erreurs dans le contenu ajouté par les autres membres. L'analyse heuristique se concentre uniquement sur ce qui est mis en ligne par les concepteurs du site et ne permet donc pas de prendre ces reproches en considération. Elle est donc moins biens adaptée pour ce type de site [42].

Si on parle ici de sites utilisant les technologies du web 2.0, cela illustre tout de même le fait que se contenter d'un seul type d'analyse mène au risque de rater un nombre important d'erreurs. Pour mener un examen ergonomique complet, il est donc utile d'effectuer une analyse heuristique suivie d'un test d'utilisabilité. L'un comme l'autre se complètent efficacement en permettant de découvrir différents types de problèmes [27].

Seconde partie : Développement d'un logiciel de support

« Simplicity is the ultimate sophistication. »

Leonardo DaVinci

Maintenant que le contexte dans lequel la méthode heuristique s'insère et que les tenants, aboutissants, avantages et inconvénients de cette méthode ont été expliqués, nous allons présenter le logiciel que nous avons développé afin de faciliter l'évaluation de l'ergonomie d'un système informatique par cette méthode. Cette partie est divisée en trois chapitres. Dans un premier temps, nous présentons les spécifications du logiciel de support. Ensuite, nous en décrivons l'implémentation proprement dite. Enfin, nous évoquons quelques perspectives de développement qui permettraient d'encore améliorer les fonctionnalités du logiciel.

3 Spécification du logiciel de support

3.1 Choix d'analyse offerts à l'utilisateur

Une analyse ergonomique logicielle peut concerner différents types de système informatique et ces systèmes peuvent eux-mêmes être évalués en utilisant des règles ergonomiques personnalisées ou provenant de listes de vérification de référence. De plus, l'évaluation du respect des règles peut se faire selon différentes méthodes. Une analyse ergonomique n'est donc pas un procédé standard qui peut s'appliquer indifféremment à chaque analyse. Pour que celle-ci soit pertinente, l'utilisateur doit avoir la capacité de l'adapter au système étudié et à ses besoins grâce à un paramétrage. Ainsi, indépendamment de toute implémentation, il a été fait le choix d'offrir à l'utilisateur la possibilité de paramétrer le logiciel pour s'adapter à différents types d'analyse. Les différents choix proposés à l'utilisateur sont repris ci-dessous.

1. Premièrement, l'utilisateur doit pouvoir choisir la langue de l'analyse, le choix étant laissé entre le Français et l'Anglais. Il importe donc de disposer des différentes listes de vérification de référence traduites dans les deux langues.
2. L'utilisateur peut aussi choisir le type du système informatique dont l'ergonomie sera évaluée. Pour ce paramètre, trois types de système sont pris en compte dans notre travail : les

sites Internet, les applications logicielles et les applications spécifiquement destinées aux téléphones mobiles.

3. Pour chaque type de système, les listes de vérification les plus connues et les plus utilisées ont été sélectionnées et incluses au logiciel. En pratique, cela signifie que cinq listes ont été choisies à la fois pour les sites Internet et les applications logicielles et une liste pour les applications pour téléphones mobiles. L'utilisateur sélectionne parmi celles-ci les listes qu'il juge pertinentes à l'analyse à effectuer. De plus, il est aussi possible d'encoder une liste de vérification personnalisée et le logiciel doit faciliter la tâche de l'utilisateur en proposant une structure standardisée qu'il est facilement possible de remplir. Le tableau suivant reprend les listes qui ont été incluses au logiciel et présente leur origine et leurs principales caractéristiques. Les listes complètes sont toutes disponibles en annexes dans leur langue d'origine.

Sites Internet	CNRS	Cette liste est reprise du guide de recommandations ergonomiques destinées aux sites et applications web édité par la direction des systèmes d'information du Centre National de la Recherche Scientifique (CNRS) (version 2005) [28]. Elle comporte 162 règles (*guidelines*).
	Nielsen & Tahir	J. Nielsen est considéré comme un des plus grands experts en matière d'utilisabilité et d'ergonomie d'interface. Il a coécrit un livre avec M. Tahir intitulé « *Homepage usability: 50 websites deconstructed* » [29] dans lequel sont reprises 112 règles à respecter en matière de conception et de design de sites Internet à vocation organisationnelle. Ces règles sont regroupées en 25 catégories et constituent une référence dans le domaine.
	BCHI	Le Belgian Laboratory of Computer-Human Interaction (BCHI) est un laboratoire inclus dans l'unité de recherche Information System (ISYS) de l'Université Catholique de Louvain (UCL). Ses activités sont la recherche, le développement et la consultance dans le domaine de l'ingénierie des interfaces. Il a publié une liste de 107 guidelines réparties en 8 catégories traitant de la conception physique d'un site Internet [30, 38].

	W3C	Cette liste a été développée en 1999 par l'organisme de standardisation World Wide Web Consortium (W3C) et est intitulée « *Web Content Accessibility Guidelines* » [31]. Elle reprend 14 règles à respecter en matière d'accessibilité de contenu web aux personnes avec un handicap.
	MIT	Le département Information Services & Technology (IS&T) du Massachusetts Institute of Technology (MIT) a publié ses propres règles de bonne pratique sous la forme d'une liste de 60 règles réparties en 10 catégories [32].
Applications logicielles	Nielsen Norman group	Le Nielsen Norman Group est une compagnie de consultance en utilisabilité créée par trois experts dans ce domaine (J. Nielsen, D. Norman et B. Tognazzini) en 1998. Cette liste publiée en 2003 par B. Tognazzini reprend des principes fondamentaux en matière de design et d'implémentation d'interfaces [33].
	Smith and Mosier - data entry	S. Smith et J. Mosier sont deux ingénieurs travaillant pour la division des systèmes électroniques de l'armée de l'air américaine. En 1986, ils ont publié un ouvrage de référence intitulé « *Guidelines for designing user interface software* » [34] qui est un outil d'aide à la conception d'interface logicielle. Cet ouvrage est organisé en 6 parties dont seules les 2 premières ont été incluses à notre logiciel. La première partie s'intitule « *Data entry* » et traite des actions de l'utilisateur pour entrer des données dans l'ordinateur et la réponse de l'ordinateur à ces actions. Elle comporte 199 règles organisées en 12 catégories.
	Smith and Mosier - data display	La deuxième partie de l'ouvrage de Smith et Mosier [34] est intitulée « *Data display* » et traite de la sortie de données de l'ordinateur vers l'utilisateur (affichage). Elle comprend 298 règles de bonne pratique réparties en 22 catégories.
	CNRS	La direction des systèmes d'information du Centre National de la Recherche Scientifique (CNRS) a également publié en 2000 un document intitulé « *Guide de recommandations ergonomiques pour la conception et l'évaluation d'interfaces graphiques* » [35] reprenant 56

		règles à respecter. Les recommandations de ce document concernent les applications logicielles.
	Eclipse	Cette liste reprend les *guidelines* en matière d'interface avec Eclipse, le célèbre environnement de développement logiciel (*IDE* en Anglais). Elle provient d'un document intitulé « *Eclipse User Interface Guidelines* » [36] dont la dernière version date de 2004. Elle est composée de 144 règles réparties en 18 catégories.
App. pour mobiles	W3C	L'organisme de standardisation World Wide Web Consortium (W3C) a également publié une liste de 60 recommandations pour les applications web destinées aux téléphones mobiles. Le document de référence s'intitule « *Mobile Web Best Practises* » [37] et la dernière version date de 2008. Les recommandations se focalisent sur le contenu délivré à l'utilisateur.

4. Une fois les listes pertinentes sélectionnées, le travail de l'analyste consiste à vérifier si les règles sont respectées. Le logiciel doit laisser l'opportunité à l'utilisateur d'évaluer le respect des règles selon trois critères différents : le modèle linéaire, le modèle pondéré ou le modèle avec importance. Tout d'abord, il faut savoir qu'au cours d'une analyse ergonomique, il est généralement nécessaire d'étudier l'ergonomie de plusieurs pages ou écrans du système informatique (la page d'accueil ainsi que les pages des différentes sections d'un site Internet par exemple). Ces différents écrans sont évalués selon les mêmes listes de vérification (et donc les mêmes règles) et chaque règle est considérée comme respectée ou enfreinte sur chacun des écrans.
    - *Modèle linéaire* : Avec ce modèle, on attribue une valeur de respect positive à une règle (c'est-à-dire égale à 1) si celle-ci est respectée pour tous les écrans analysés. Il suffit qu'elle ne soit pas respectée sur une page ou un écran et la valeur de respect sera négative (valeur de 0). Ce modèle a l'avantage de la simplicité mais il peut être trop réducteur car il ne tient pas compte du ratio entre le nombre de fois où la règle est respectée et où elle est enfreinte.
    - *Modèle pondéré* : Un second modèle plus élaboré a dès lors été inclus au logiciel. Avec celui-ci, l'utilisateur calcule le ratio entre le nombre d'écrans pour lesquels la

règle est respectée et le nombre total d'écrans analysés. Si ce ratio est supérieur à un seuil fixé par l'utilisateur, la règle est considérée comme respectée. Considérons un exemple illustratif : l'analyse d'un site Internet porte sur 10 pages. Une règle consiste à vérifier si une barre de navigation est présente et identique sur les différentes pages du site et cette règle est vérifiée pour 8 pages sur les 10. Comme pour cet exemple le seuil de respect a été arbitrairement fixé à 0,7, la règle est considérée comme respectée. Cela n'aurait pas été le cas avec un seuil de 0,9.

- *Modèle avec importance* : Un troisième modèle encore plus élaboré a également été inclus. Pour celui-ci, l'utilisateur donne à chaque règle une valeur d'importance allant de 0 (règle n'ayant aucune importance) à 1 (règle ayant une importance maximale). Ainsi, des règles portant sur des critères strictement graphiques peuvent être considérées comme moins importantes que des règles sémantiques pour certains types de site Internet. Cette valeur d'importance est multipliée par le ratio calculé pour le modèle pondéré et le résultat est comparé à une valeur seuil. Reprenons les données de l'exemple précédent (ratio de 0,8). Si la règle à une importance de 0,9 et que le seuil est fixé à 0,7, la règle est respectée : $0{,}8 \times 0{,}9 = 0{,}72 > 0{,}7$. Par contre, si l'importance de la règle n'est que de 0,8, la règle n'est plus respectée : $0{,}8 \times 0{,}8 = 0{,}64 < 0{,}7$. A contrario, les deux modèles précédents considèrent que toutes les règles sont équiproportionnellement importantes (valeurs d'importance égales à 1).

5. Dans certains cas, une analyse séparée de certains écrans ou pages d'un système informatique peut être nécessaire. De même, une analyse ergonomique peut être divisée en plusieurs parties comprenant chacune d'elles plusieurs pages. Lors que l'utilisateur a saisi le type de système à analyser, les listes de vérification utilisées et les critères de respect des règles, il est possible de générer une feuille d'analyse avec les règles à évaluer. Pour les cas particuliers évoqués ci-dessus, une option doit permettre à l'utilisateur de sélectionner le nombre d'écrans qu'il souhaite analyser et le logiciel va ainsi dupliquer la feuille d'analyse. Les résultats récapitulatifs seront calculés pour chacun de ces écrans d'analyse.

3.2 Fonctionnalités du logiciel

Le chapitre précédent détaillait quels paramètres permettent de tenir compte des spécificités d'une analyse ergonomique. Maintenant, nous allons expliquer les fonctionnalités que le logiciel

doit apporter à l'utilisateur pour supporter cette analyse, soit l'équivalent de notre cahier des charges. Au cours d'une analyse, on compte trois fonctions chronologiques principales englobant d'autres fonctions secondaires. Ci-dessous, nous expliquons en détail le rôle de chacune d'elle.

3.2.1 Saisie des paramètres de l'analyse

Avant d'effectuer l'analyse du système, le logiciel doit permettre d'en saisir tous les paramètres. Pour rappel, cela implique les paramètres explicités dans le chapitre précédent :

- choix de la langue,
- choix du type de système informatique,
- choix des listes de vérification et prise en compte éventuelle d'une liste personnalisée,
- valeurs des niveaux de respect pour les modèles pondéré et avec importance,
- nombre d'écrans analysés,
- paramètres secondaires permettant de mettre en forme les résultats de l'analyse.

La manière de saisir les paramètres sera discutée dans le chapitre suivant relatif à l'implémentation mais, d'un point de vue fonctionnel, celle-ci doit être interactive, rapide et centralisée lors du démarrage du logiciel.

3.2.2 Création d'une feuille d'analyse

Une fois les paramètres saisis, le logiciel doit générer une feuille d'analyse qui soit conforme aux paramètres d'entrée et facilite et automatise la tâche de l'analyste. Cette étape est primordiale car une feuille d'analyse bien conçue peut mener à des gains de temps substantiels et remplir ainsi efficacement la fonction de support du logiciel. Concrètement, cette feuille doit regrouper les informations suivantes :

- Les listes de vérification qui ont été sélectionnées par l'utilisateur. Ces listes doivent être présentées sous une forme standardisée pour en faciliter la lecture et l'évaluation du respect des règles.
- Pour chacune des règles de ces listes, le logiciel doit permettre d'encoder une valeur de respect (modèle linéaire et pondéré), d'encoder la valeur d'importance de la règle le cas échéant et d'indiquer si celle-ci est pertinente pour l'analyse en question.
- La possibilité d'écrire des commentaires propres à chaque règle doit être offerte.
- Des techniques graphiques (mise en couleur par exemple) doivent faciliter la lecture des

résultats encodés. Sont particulièrement visées les valeurs de pertinence ou de respect des différentes règles. Les moyens mis en œuvre sont discutés dans la phase d'implémentation.

- Dans le cas des modèles pondérés et avec importance, le logiciel doit automatiquement calculer si les règles sont respectées à partir des valeurs de respect et des seuils encodés.

3.2.3 Génération de résultats synthétiques

Quand l'utilisateur a évalué le respect des différentes règles en complétant la feuille d'analyse, le logiciel doit générer des résultats synthétisant l'analyse (tableaux, graphiques, …). Cet outil facilite considérablement la tâche de l'utilisateur s'il doit écrire un rapport sur l'analyse par exemple.

Premièrement, il faut savoir que, malgré le choix des listes, certaines règles peuvent être jugées non pertinentes pour certaines analyses et ces règles ne doivent donc pas être prises en compte lors du calcul de statistiques permettant de synthétiser l'analyse.

La mise en forme des résultats dépend de l'implémentation retenue mais, pour chaque liste de vérification retenue, le logiciel doit tout au moins faire apparaître les informations suivantes :

- Pour la liste de vérification en entière ainsi que pour chaque catégorie de règles la composant, il devrait apparaître des tableaux récapitulatifs montrant le nombre total de règles, le nombre de règles respectées, le nombre de règles enfreintes et le ratio entre le nombre de règles respectées et le nombre de règles total. L'utilisateur a ainsi très rapidement un premier aperçu des performances globales du système informatique étudié et il peut voir si les éventuelles contre-performances sont associées à une catégorie de règles particulière. En plus d'une présentation sous forme de tableaux, ces résultats sont bien adaptés à une représentation graphique sous forme de diagrammes circulaires ou de graphiques en radar (voir implémentation).
- En plus de ces informations, le logiciel devrait calculer quelques statistiques de base : moyenne et écart-type des valeurs de respect pour les modèles linéaires et pondérés et moyenne et écart-type des valeurs d'importance le cas échéant, à la fois pour la liste entière et pour chacune des catégories. Pour ce point et le point précédent, une mise en forme des données (mise en couleur par exemple) a également pour rôle de faciliter la lecture des résultats.

- Si le modèle avec importance a été choisi, il est intéressant de savoir le nombre de règles respectées et enfreintes dans la liste pour chaque valeur d'importance. En effet, il est primordial que les règles jugées comme importantes soient respectées alors que le non respect des règles peu importantes peut être relativisé. Pour cet aspect de l'analyse, une représentation des données sous forme d'un diagramme en bâtonnet est appropriée (voir implémentation).

3.3 Diagramme de classe UML

Maintenant que les paramètres et les fonctionnalités du logiciel sont connus, il est bon de présenter un diagramme de classe qui permet de visualiser les données manipulées par le logiciel et ainsi donner une bonne synthèse des spécifications (Fig. 3.1).

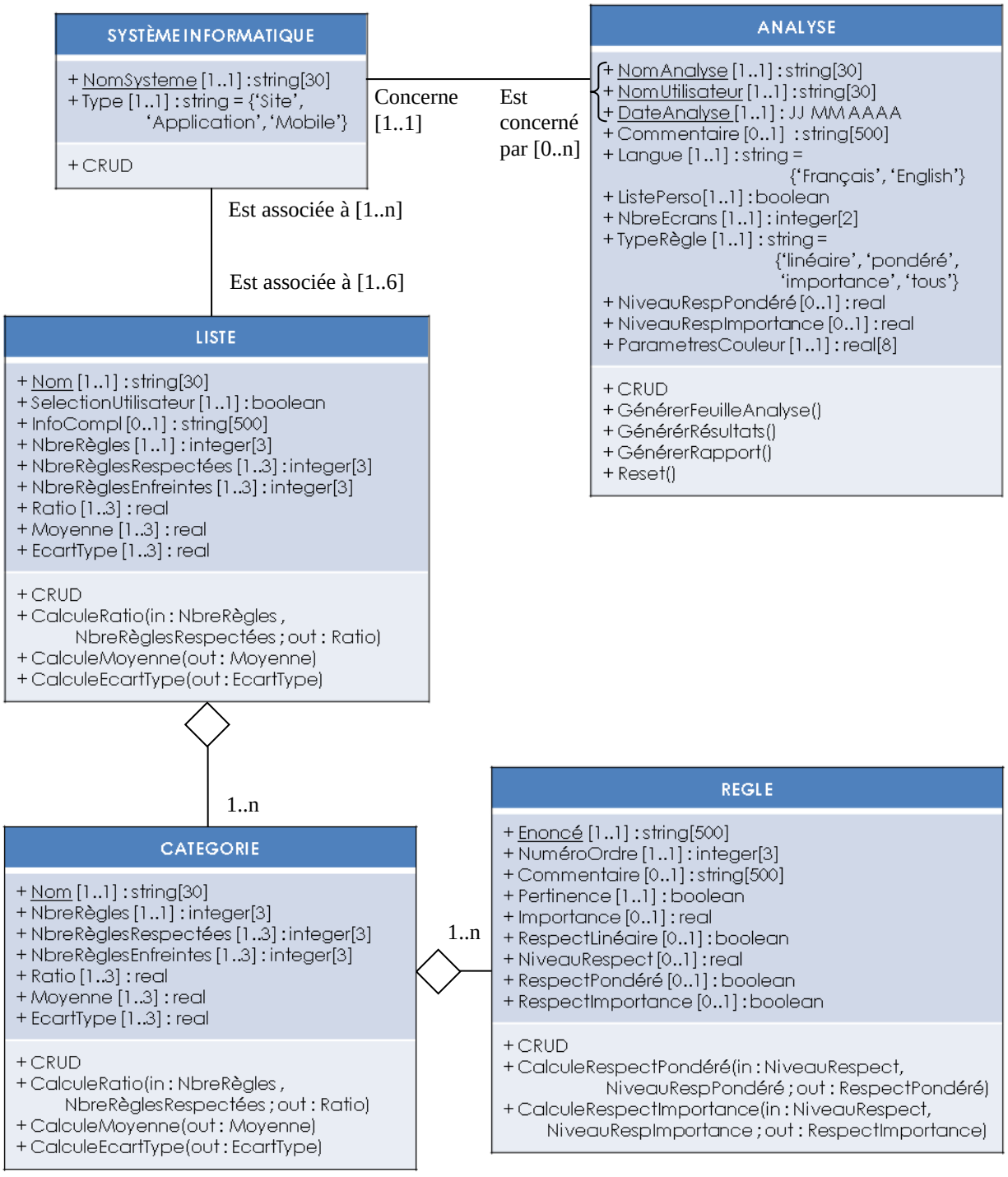


Fig. 3.1 - Diagramme de classe UML du logiciel développé

4 Implémentation du logiciel de support

4.1 Présentation générale

Le logiciel a été implémenté en utilisant l'outil Microsoft Excel (ci-après simplement dénommé Excel) de la suite bureautique Microsoft Office. En effet, cet outil est particulièrement adapté à une présentation synthétique des résultats de l'analyse sous la forme de tableaux et de graphiques. En outre, Excel permet de développer facilement des applications à l'aide du langage dédié Microsoft VBA (Visual Basic for Applications). Des facilités sont notamment proposées aux utilisateurs afin de réaliser des interfaces (menus déroulants, boutons, ..) et des formulaires ou de gérer les évènements (clic sur un bouton, sélection d'une cellule, …).

Selon l'implémentation retenue, le logiciel de support se présente sous la forme d'un classeur Excel prenant en charge les macros (fichier possédant l'extension « xlsm » avec la version de 2007 d'Excel). Un classeur Excel est un document contenant une ou plusieurs feuilles de calcul elles-mêmes composées de cellules réparties sous la forme de lignes et de colonnes. Dans le contexte étudié, on appelle *macro* un programme écrit dans le langage VBA qui permet de commander de façon automatique les fonctionnalités d'Excel.

Lors d'une analyse, le fichier permet lui-même de générer un nouveau fichier « xlsm » dans lequel seront enregistrés les paramètres, les données et les résultats de l'analyse.

4.2 Fonctionnement du logiciel

Conformément aux fonctionnalités principales décrites au chapitre 3.2, une utilisation classique du logiciel de support comporte trois étapes successives :

- Saisie des paramètres de l'analyse
- Création d'une feuille d'analyse
- Génération de résultats synthétiques

A chacune de ces phases est associée une interface de communication avec l'utilisateur. Le fonctionnement et la structure de ces interfaces sont présentés dans les paragraphes suivants. Enfin, le fonctionnement du logiciel lors de sa fermeture est évoqué dans le dernier paragraphe de cette section.

4.2.1 Interface d'accueil

Au lancement du programme (ouverture du fichier Excel) apparait directement un écran d'accueil permettant à l'utilisateur de saisir les paramètres de l'analyse (Fig. 4.1). Cette interface se présente sous la forme d'une feuille de calcul Excel sur laquelle ont été ajoutés des menus et des boutons.

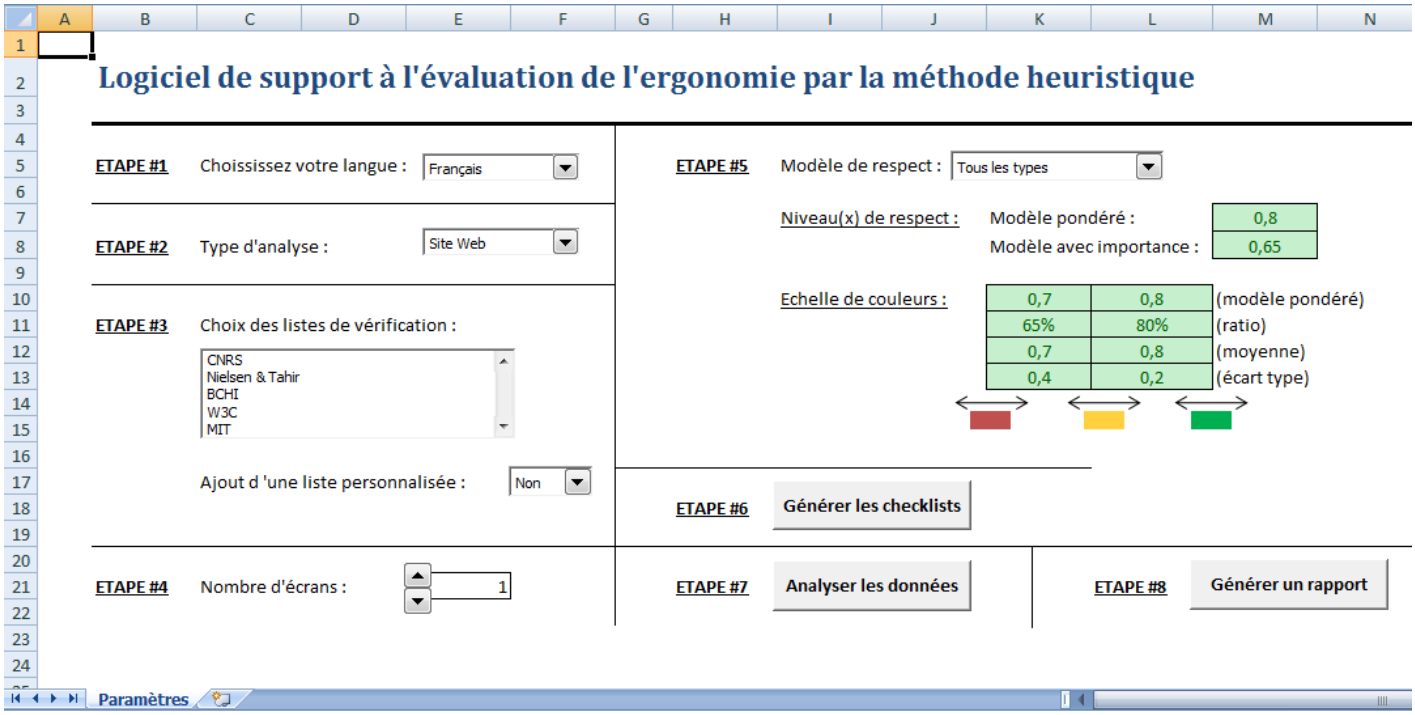


Fig. 4.1 – Interface d'accueil du logiciel

L'implémentation choisie invite l'utilisateur à procéder étape par étape bien que le logiciel permette aussi, et ce n'est pas trivial du point de vue de l'implémentation, que les paramètres soient sélectionnés dans un ordre aléatoire. Passons en revue ces différentes étapes.

1. Choix de la langue (Français, le choix par défaut, ou Anglais) via un menu déroulant. Dès que l'utilisateur change la langue en cliquant dans le menu déroulant, le logiciel exécute une fonction qui traduit instantanément le contenu de toutes les cellules et des boutons de la feuille et qui met au premier plan les menus de sélection des étapes suivantes correspondant à la langue sélectionnée. En effet, en cas de superposition d'objets, Excel permet de gérer quel objet de la feuille de calcul doit être devant les autres, et ainsi être visible pour l'utilisateur,

grâce à des commandes prédéfinies que l'on peut utiliser dans les macros.

2. Choix du type de système informatique (site Internet, le choix par défaut, application ou application pour mobile). Dès que l'utilisateur change de système dans le menu déroulant, l'évènement conduit à l'exécution d'une macro qui fait apparaitre au premier plan le menu comportant les listes de vérification relatives au type choisi afin de permettre le choix de la troisième étape.

3. Selon l'implémentation retenue, il y a trois menus permettant de choisir les listes de vérification qui se superposent mais seul l'un d'entre eux est visible. Le choix des listes se fait par un simple clic pour mettre le nom de la liste en surbrillance. Un deuxième clic permet d'annuler la sélection. De plus, un autre menu déroulant permet de choisir si on souhaite inclure une liste de vérification personnalisée dont il faudra encoder manuellement les règles par la suite. Lorsque l'utilisateur fait ce choix, un tableau apparait sur la partie droite de l'interface (Fig. 4.2). Celui-ci permet d'encoder le nombre de catégories composant la liste personnalisée (maximum 10) et le nombre de règles dans chacune de ces catégories (maximum 50). Des mises en forme conditionnelles ont été ajoutées à la feuille de calcul afin de colorer en jaune les cases qu'il faut compléter, en vert les cases correctement complétées, en rouge les cases incorrectement complétées et en gris les cases qu'il n'est pas nécessaire de compléter.

# de categories	5	(max 10)
#règles ds cat. 1	0	(max 50)
#règles ds cat. 2	6	(max 50)
#règles ds cat. 3	12	(max 50)
#règles ds cat. 4	51	(max 50)
#règles ds cat. 5	0	(max 50)
#règles ds cat. 6	0	(max 50)
#règles ds cat. 7	0	(max 50)
#règles ds cat. 8	0	(max 50)
#règles ds cat. 9	0	(max 50)
#règles ds cat. 10	0	(max 50)

Fig. 4.2 – Interface d'accueil du logiciel

4. La quatrième étape invite l'utilisateur à choisir le nombre d'écrans pour lesquels il veut une feuille d'analyse. Par défaut, le logiciel génère une seule feuille et un changement de ce paramètre s'effectue par de simples clics sur les deux boutons (haut et bas) afin d'obtenir la valeur souhaitée (maximum 50).

5. La cinquième étape traite du respect des règles ergonomiques. L'utilisateur peut choisir le modèle qu'il désire (linéaire, pondéré ou avec importance) ou une combinaison des trois (choix par défaut) grâce à un menu déroulant. Si les modèles pondéré ou avec importance ont été choisis, l'utilisateur encode le niveau de respect dans les cellules prévues à cet effet. Il peut aussi choisir la valeur des paramètres permettant de régler les échelles de couleur. Ces paramètres permettent de piloter la mise en forme des données et des résultats. Leur signification sera discutée dans les sections 4.2.2 et 4.2.3. Toutes les valeurs encodées manuellement dans les cellules sont contrôlées et mises en couleur pour guider l'utilisateur dans sa démarche de configuration (vert si la valeur est acceptable et rouge si elle doit être changée). Pour les niveaux de respect, les valeurs encodées doivent être comprises entre 0 et 1 et pour les échelles de couleurs. Une condition supplémentaire impose à la valeur de gauche du tableau d'être inférieure à la valeur de droite (sauf pour les valeurs d'écart-type ou l'inverse prévaut).

6. Une fois que les paramètres des cinq premières étapes ont été encodés, l'utilisateur n'a plus qu'à cliquer sur le bouton « *Générer les checklists / Generate checklists* » et une nouvelle interface permettant d'effectuer l'analyse ergonomique proprement dite apparait (voir section 4.2.2). Avant que n'apparaisse cette interface, certains tests sont effectués sur les paramètres saisis. Un test vérifie si au moins une liste de vérification a été sélectionnée (que ça soit une liste de référence ou une liste personnalisée). Dans le cas contraire, un message d'erreur invite l'utilisateur à sélectionner au moins une liste.

7. Une fois que le respect des règles a été évalué, l'utilisateur peut cliquer sur le bouton « *Analyser les données* / *Analyse data* » pour obtenir des résultats synthétisant l'analyse. Le fonctionnement de cette interface est détaillé dans la section 4.2.3. Si l'utilisateur essaie de cliquer sur les boutons des étapes 7 (« *Analyser les données* ») ou 8 (« *Générer un rapport* ») avant d'avoir préalablement exécuté l'étape 6, des tests sont prévus pour détecter la mauvaise manœuvre et inviter l'utilisateur à exécuter les différentes étapes dans l'ordre adéquat.

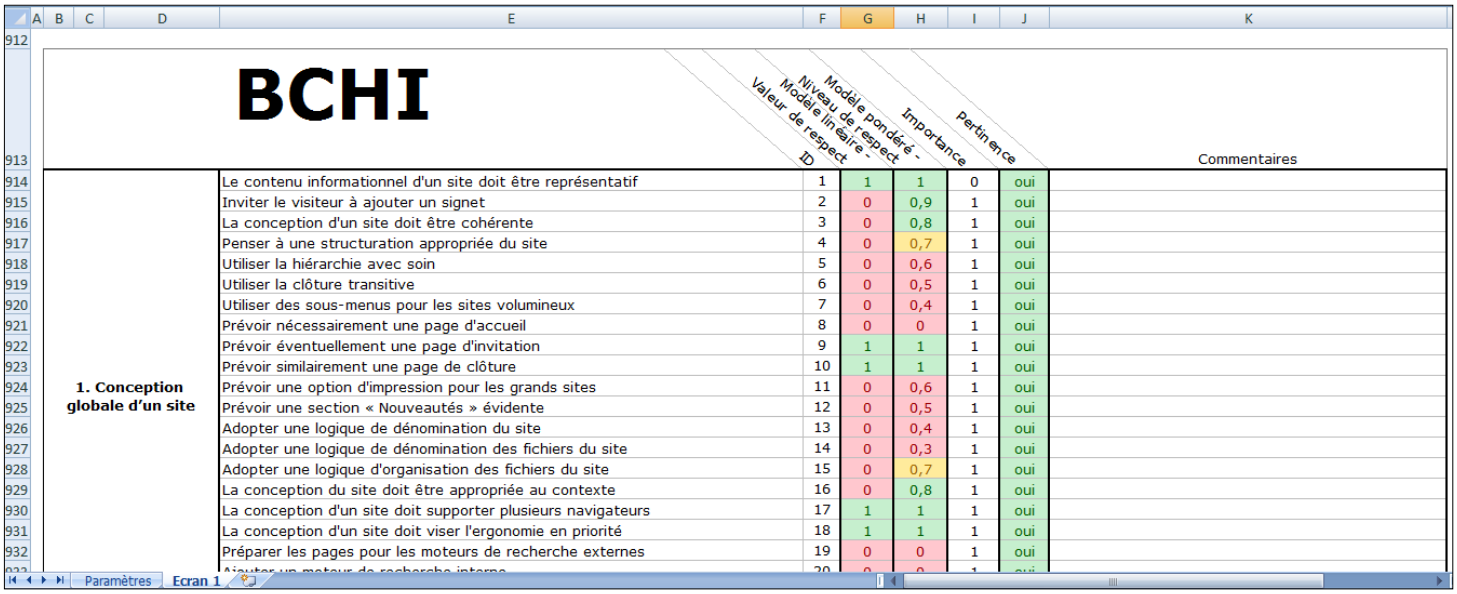
En plus de ces mécanismes de gestion des menus, une structure de donnée est créée et initialisée aux valeurs par défaut à l'ouverture du logiciel. Cette structure s'appelle *paramètres* et est composée de champs de différents types (*string*, *boolean*, *integer*) permettant de stocker les paramètres choisis par l'utilisateur. En effet, Excel offre la possibilité d'associer des évènements (comme le clic sur un bouton ou un choix dans un menu déroulant) à l'exécution d'une routine. Ainsi, à chaque utilisation des menus ou des boutons, des routines sont exécutées afin de modifier le contenu des champs de la structure en conséquence. De plus, les paramètres finaux de l'utilisateur sont affectés aux champs de la structure de donnée lorsque l'utilisateur sélectionne le bouton de l'étape 6.

Comme expliqué plus haut, ces routines permettent également de sélectionner quels menus doivent être placés au premier plan de l'interface.

Le bouton *reset* en haut à droite (non présent sur la Fig. 4.1) permet, après confirmation de l'utilisateur, de réinitialiser tous les paramètres de l'écran d'accueil ainsi que le contenu des champs de la structure *paramètres* à leur valeur par défaut. Si une analyse est en cours, toutes les feuilles de calcul qui ont été générées sont supprimées.

4.2.2 Interface d'évaluation des règles ergonomiques

Lorsque l'utilisateur clique sur le bouton de l'étape 6 de l'écran d'accueil, il souhaite commencer une analyse ergonomique et une nouvelle interface apparait donc au premier plan (Fig. 4.3). Cette nouvelle feuille de calcul Excel est composée des différentes listes de vérification sélectionnées par l'utilisateur placées les unes à la suite des autres.



BCHI

		ID	Valeur de respect - Modèle linéaire	Niveau de respect - Modèle pondéré	Importance	Pertinence	Commentaires
1. Conception globale d'un site	Le contenu informationnel d'un site doit être représentatif	1	1	1	0	oui	
	Inviter le visiteur à ajouter un signet	2	0	0,9	1	oui	
	La conception d'un site doit être cohérente	3	0	0,8	1	oui	
	Penser à une structuration appropriée du site	4	0	0,7	1	oui	
	Utiliser la hiérarchie avec soin	5	0	0,6	1	oui	
	Utiliser la clôture transitive	6	0	0,5	1	oui	
	Utiliser des sous-menus pour les sites volumineux	7	0	0,4	1	oui	
	Prévoir nécessairement une page d'accueil	8	0	0	1	oui	
	Prévoir éventuellement une page d'invitation	9	1	1	1	oui	
	Prévoir similairement une page de clôture	10	1	1	1	oui	
	Prévoir une option d'impression pour les grands sites	11	0	0,6	1	oui	
	Prévoir une section « Nouveautés » évidente	12	0	0,5	1	oui	
	Adopter une logique de dénomination du site	13	0	0,4	1	oui	
	Adopter une logique de dénomination des fichiers du site	14	0	0,3	1	oui	
	Adopter une logique d'organisation des fichiers du site	15	0	0,7	1	oui	
	La conception du site doit être appropriée au contexte	16	0	0,8	1	oui	
	La conception d'un site doit supporter plusieurs navigateurs	17	1	1	1	oui	
	La conception d'un site doit viser l'ergonomie en priorité	18	1	1	1	oui	
	Préparer les pages pour les moteurs de recherche externes	19	0	0	1	oui	

Fig. 4.3 – Interface d'évaluation des règles ergonomiques

Excel offre des fonctionnalités permettant de masquer certaines lignes et colonnes d'une feuille ou même certaines feuilles d'un classeur. Cela signifie que ces éléments font partie du fichier mais qu'ils ne sont pas apparents pour l'utilisateur.

Pour créer l'interface d'évaluation, une feuille masquée est copiée et montrée à l'utilisateur après un traitement. Il existe six feuilles, que l'on appellera feuille-type, utilisables pour cette étape car pour chaque langue, il y a trois feuilles correspondant aux trois types de système informatique. Chaque feuille-type contient toutes les listes de vérification du type de système considéré dans la langue appropriée. Selon les choix de langue et de système, la feuille-type appropriée est copiée. Ainsi, seule la copie sera modifiée et la feuille originale restera intacte pour d'autres analyses.

Comme l'utilisateur peut ne sélectionner qu'un certain nombre des listes de vérification proposées, celles qu'il ne souhaite pas sont cachées en appelant les fonctionnalités précitées d'Excel afin de masquer les lignes concernées.

Les différentes opérations expliquées ci-dessus sont mises en œuvre au moyen de tests logiques (*if-else*) sur les valeurs de certains champs de la structure de données *paramètres* afin de déterminer les conditions de l'analyse.

Chaque liste est présentée de façon standardisée pour faciliter l'adaptation de l'utilisateur. Une liste est composée de règles organisées en catégories et chaque règle occupe une ligne de la feuille. En partant de l'extrême gauche de la feuille, on trouve successivement sur chaque ligne

la catégorie à laquelle appartient la règle, la règle en elle-même, son numéro d'ordre dans la liste, sa valeur de respect selon les modèles linéaires et pondérés, sa valeur d'importance, sa pertinence et un commentaire éventuel.

Certaines colonnes peuvent être manquantes selon le modèle choisi pour évaluer le respect des règles. En effet, les colonnes superflues sont masquées juste après la copie de la feuille-type.

L'encodage de la pertinence d'une règle ou de sa valeur de respect selon le modèle linéaire se fait très facilement au moyen d'un simple clic sur la cellule correspondant. En effet, à chaque changement de sélection de cellule dans la feuille, une routine est exécutée. Cette routine teste si la nouvelle cellule sélectionnée appartient aux deux colonnes en question. Si c'est le cas et que la ligne sélectionnée correspond à une règle, le contenu de la cellule est changé pour prendre sa valeur opposée. Si la règle était pertinente, elle ne l'est plus et vice-versa et si la règle était respectée, elle ne l'est plus et vice-versa. Cette routine permet de considérablement simplifier la tâche d'encodage de l'utilisateur, et donc de gagner du temps, puisque seul un clic sur les cellules dont il souhaite inverser la valeur est suffisant.

Les valeurs encodées pour le modèle pondéré sont mises en forme à l'aide d'une échelle à trois niveaux de couleur. Cette échelle est configurée grâce aux deux paramètres de la première ligne du tableau de l'étape 5 (cf. interface d'accueil). Les valeurs comprises entre 0 et la première valeur sont colorées en rouge, celles entre la première et la seconde valeur sont colorées en orange et enfin, celles entre la seconde valeur et 1 sont colorées en vert. Le rouge correspond aux règles qui ne sont clairement pas respectées, le vert correspond aux règles qui sont clairement respectées et l'orange correspond aux situations ambigües auxquelles l'utilisateur a intérêt à porter une attention toute particulière. De même, les règles non respectées selon le modèle linéaire ou non pertinentes ont leurs cellules correspondantes colorées en rouge. Dans le cas contraire, elles sont colorées en vert.

4.2.3 Interface d'analyse des résultats

Lorsque l'utilisateur a fini d'évaluer le respect des règles, il peut générer des résultats synthétiques en cliquant sur le bouton de l'étape 7 de l'écran d'accueil (« *Analyser les données* / *Analyse data* »). Dans ce cas, une nouvelle interface apparait présentant différents graphiques et tableaux (Fig. 4.4).

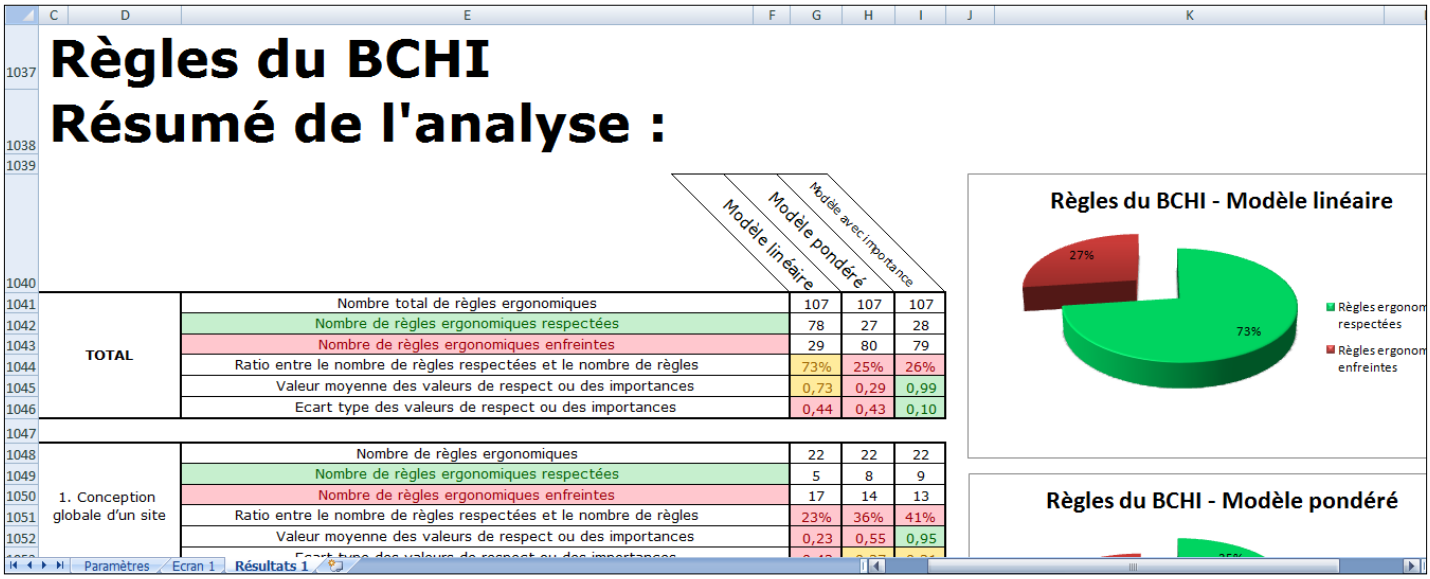


Fig. 4.4 – Interface d'analyse des résultats

Pour chaque liste de vérification sélectionnée, les mêmes résultats synthétiques sont présentés soit en l'occurrence :

- Un tableau général avec, pour chaque modèle de respect des règles choisi, le nombre total de règles, le nombre de règles respectées et enfreintes, le ratio entre le nombre de règles respectées et le nombre de règles total, la moyenne des valeurs de respect et leur écart-type (voir Fig. 4.4). Pour ces deux dernières informations, c'est la moyenne et l'écart-type des valeurs d'importance qui sont données lorsque le modèle avec importance est sélectionné. Les statistiques sont calculées à l'aide de fonctions de base fournies par Excel.
- Un tableau comportant les mêmes informations que celui décrit ci-dessus est aussi calculé pour chaque catégorie de la liste (voir Fig. 4.4).
- Pour chaque modèle de respect des règles, un diagramme circulaire (dit aussi « en quartiers de tarte ») représente le nombre de règles respectées et enfreintes dans la liste (voir Fig. 4.4).
- Un tableau synthétique par modèle de respect des règles est également calculé (Fig. 4.5). Celui-ci reprend des informations déjà calculées pour la liste et les catégories (règles respectées et enfreintes, ratios) mais présentées différemment.

Modèle pondéré :

	Nombre de règles	Nombre de règles respectées	Nombre de règles enfreintes	Ratio
TOTAL	105	71	34	68%
Conception globale d'un site	20	16	4	80%
Navigation	19	14	5	74%
Présentation	14	10	4	71%
Formulaires, titres et en-têtes	5	4	1	80%
Cadres et fenêtres	5	4	1	80%
Graphiques	15	4	11	27%
Éléments multimédia	22	14	8	64%
Accessibilité	5	5	0	100%

Fig. 4.5 – Tableau résumant l'analyse d'une liste de vérification (modèle pondéré)

- Pour chaque modèle de respect des règles, un diagramme en radar est calculé (Fig. 4.6). Celui-ci possède autant d'axes qu'il y a de catégories dans la liste. Sur chacun des axes est représenté le ratio entre le nombre de règles respectées et le nombre de règles contenu dans la catégorie. Tous ces points sont reliés de sorte que ce type de graphique se présente sous la forme d'une surface colorée proportionnelle au nombre de règles respectées. Il permet de facilement détecter si une catégorie de règles est mal respectée car la surface est irrégulière au niveau de son axe (voir la catégorie « Navigation » de la Fig.4.6 par exemple). Au contraire, une surface régulière montre que le taux de respect des différentes catégories est semblable.
- Si le modèle avec importance a été inclus dans l'analyse, un autre graphique est calculé (Fig. 4.7). Celui-ci présente sous la forme de bâtonnets le nombre de règles respectées et enfreintes en fonction de l'importance de ces règles. Selon l'implémentation retenue, on considère des pas de 0,1 entre les différentes valeurs d'importance (en allant de 0 à 1). Les données de ce graphique sont également rassemblées dans un tableau. Ce graphique est particulièrement utile pour détecter si des règles importantes ne sont pas respectées.

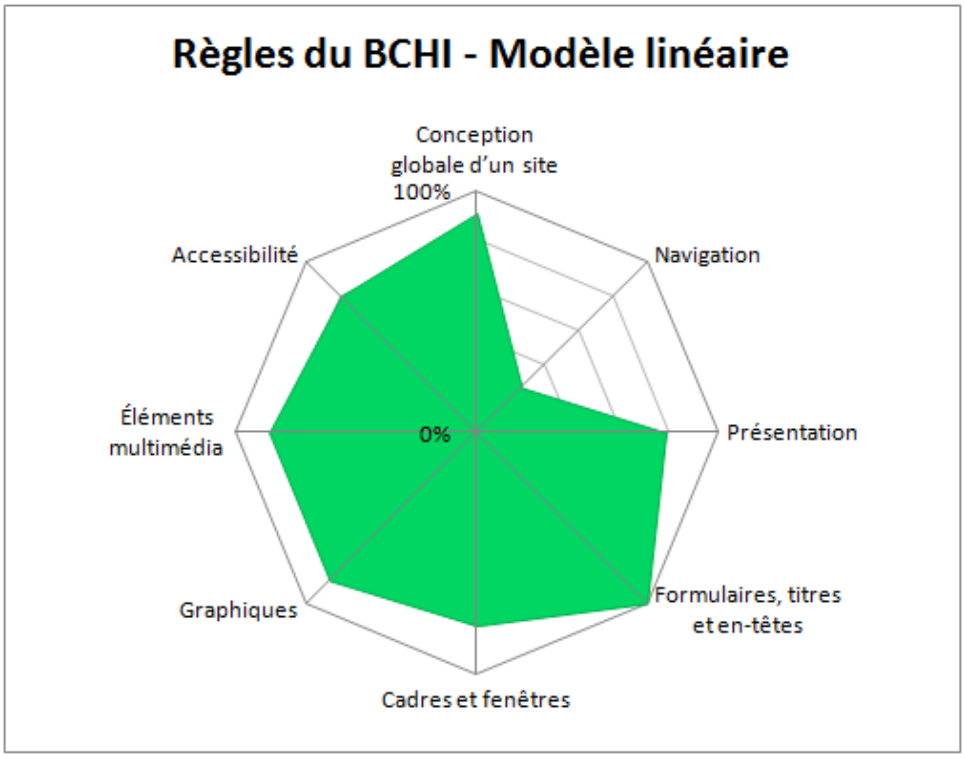


Fig. 4.6 – Diagramme en radar résumant l'analyse d'une liste de vérification (modèle linéaire)

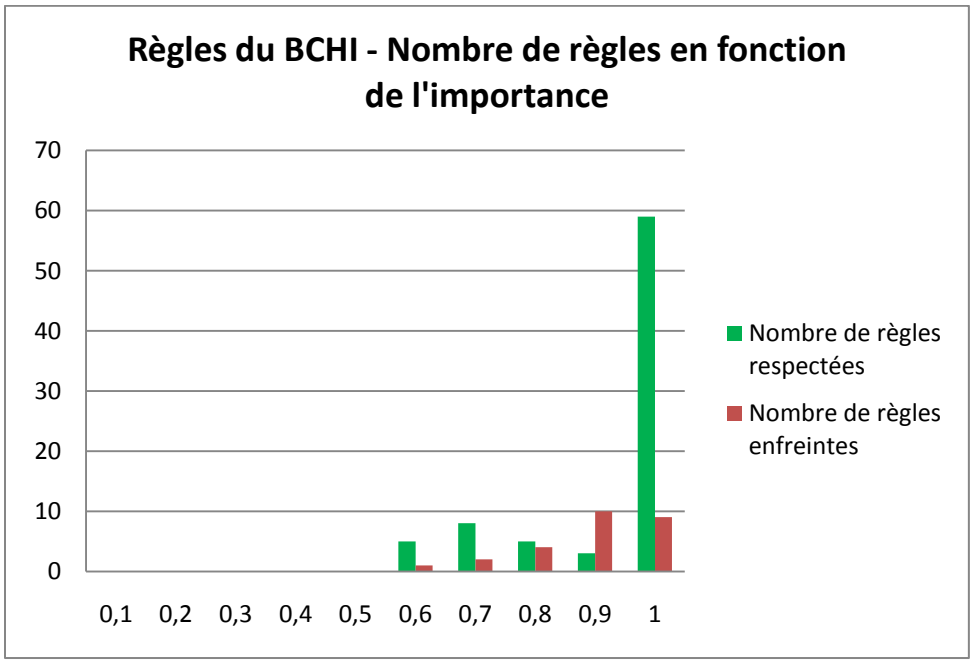


Fig. 4.7 – Diagramme en bâtonnets résumant l'analyse d'une liste de vérification

Cette feuille de calcul est directement tirée de l'interface précédente. En effet, chaque feuille-type contient déjà les résultats synthétiques placés juste après les règles de chaque liste de vérification mais ces résultats sont masqués. Lorsque les résultats sont demandés, l'interface précédente (interface d'évaluation) est copiée, les règles sont masquées et les résultats sont affichés. Si seuls certains modèles de vérification des règles ont été sélectionnés, les tableaux et graphiques relatifs aux autres modèles sont masqués de la feuille de résultats.

Comme le nombre de règles est fixe pour chaque liste, les formules permettant de calculer les graphiques et les tableaux sont pré-encodées en sélectionnant les cellules de la feuille qui nous intéressent. Excel permet de mettre à jour dynamiquement le contenu des tableaux et les coordonnées des points des graphiques.

Trois familles de données sont mises en forme à l'aide d'échelles de couleur dont les paramètres sont encodés dans le tableau de l'étape 5 de l'écran d'accueil : les ratios, les moyennes et les écarts-type. Selon le même principe que celui expliqué à la section 4.2.2 pour les valeurs de respect, trois couleurs différentes sont utilisées (rouge, orange, vert). Pour les écarts-type, la couleur verte correspond aux plus petites valeurs et la couleur rouge, synonyme de plus grande dispersion des données, aux plus grandes. Ces mises en couleur permettent de détecter directement si le respect des règles d'une catégorie est inférieur aux exigences de l'utilisateur.

4.2.4 Fermeture du logiciel

Lorsque que l'utilisateur souhaite quitter le logiciel (clic sur la croix rouge du programme Excel par exemple), une routine est exécutée afin de gérer le mécanisme de fermeture. Deux scénarios sont traités différemment. Ces deux scénarios dépendent de la valeur d'un *flag* de la structure de données *paramètres* qui vaut *true* si l'étape 6 de l'écran d'accueil a été exécutée et *false* dans le cas contraire.

- *Aucune feuille d'analyse n'a été créée* : l'utilisateur n'a jamais exécuté l'étape 6 et n'a donc pas manifesté le souhait d'effectuer une analyse ergonomique. Dans ce cas, le logiciel se ferme sans demande de confirmation et les paramètres sélectionnés ne sont pas enregistrés. Le fichier est prêt à être réutilisé ultérieurement pour une analyse et les paramètres par défaut seront chargés.

- *Une feuille d'analyse a été créée* : l'utilisateur a exécuté correctement l'étape 6 de l'écran d'accueil. Arrivé à ce stade, l'utilisateur a donc manifesté la volonté de commencer une analyse ergonomique. Dans ce cas, un menu apparait et demande à l'utilisateur s'il souhaite enregistrer l'analyse en cours (Fig. 4.8). L'exécution de l'étape 7 n'influence pas l'apparition de ce menu. Si l'utilisateur confirme son choix, un nouveau menu apparait demandant sous quel nom l'utilisateur souhaite enregistrer l'analyse (Fig. 4.9). L'analyse est alors enregistrée sous la forme d'un nouveau fichier « xslm ». Dans celui-ci, les champs de la structure de données *paramètres* sont enregistrés dans des cellules cachées et un *flag* indiquant si le fichier a déjà été enregistré passe également à *true*. Ainsi, lorsque le fichier est réouvert, les paramètres de l'analyse en cours sont connus et utilisés pour initialiser la structure de données et les menus de l'interface d'accueil. Le nouveau fichier peut donc être réutilisé pour continuer l'analyse en cours car toutes les règles ainsi que les éventuels résultats synthétiques sont enregistrés. En effet, une analyse ergonomique est un procédé qui peut être long si le système analysé est complexe ou si le nombre de listes de vérification est élevé. Comme l'analyse en cours est enregistrée dans un nouveau fichier, le fichier original n'est pas modifié et reste donc utilisable pour des analyses ultérieures. Il s'ouvrira en chargeant les paramètres par défaut.

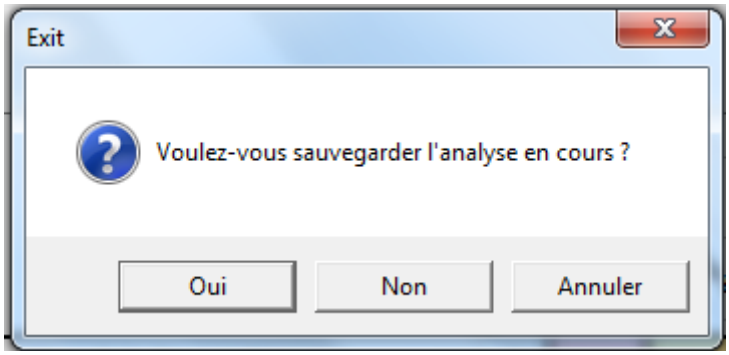


Fig. 4.8 – Interface de fermeture du logiciel

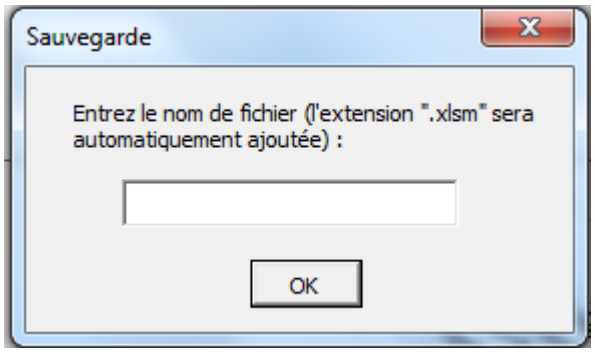


Fig. 4.9 – Interface d'enregistrement de l'analyse en cours

5 Perspectives de développement

Au-delà des fonctionnalités apportées par le logiciel développé, il est possible d'imaginer d'autres fonctionnalités qui auraient un intérêt pour l'utilisateur. Certaines idées sont évoquées dans les paragraphes suivants. Elles constituent sans aucun doute des perspectives intéressantes afin d'approfondir le travail réalisé.

5.1 Génération automatique d'un rapport

Dans certains cas, une analyse ergonomique selon la méthode heuristique est réalisée par des consultants pour le compte d'un client. Leur rôle est d'effectuer des recommandations afin d'améliorer l'ergonomie du système informatique étudié.

Généralement, les consultants rédigent un rapport écrit reprenant les résultats de l'analyse ainsi que leurs recommandations. Il leur serait dès lors très utile que le logiciel permette de créer automatiquement un document reprenant les résultats fournis par l'interface d'analyse.

L'appel à cette fonctionnalité a été inclus à l'interface d'accueil par l'intermédiaire du bouton de l'étape 8 (« *Générer un rapport / Generate repport* ») et des tests s'assurant que les différentes étapes sont exécutées dans l'ordre adéquat ont été prévus. Néanmoins, à ce stade du développement, le logiciel permet simplement de générer un document Microsoft Word vierge. Le défi consiste maintenant à transférer les tableaux et graphiques dans le document tout en respectant une structure de document type dont les titres et sous-titres indiquent les paramètres de l'analyse.

Selon nous, la piste la plus prometteuse afin de permettre cette fonctionnalité est de piloter l'outil de traitement de texte Microsoft Word par l'intermédiaire d'Excel en utilisant le langage VBA. Cette démarche s'appelle l'*automation* et elle est définie comme « *une fonctionnalité que les programmes utilisent pour exposer leurs objets aux outils de développement, aux langages macro et aux autres programmes qui prennent en charge l'automation* » [39].

En effet, les éléments utilisés dans un tableur (feuille de calcul, cellule, graphique, …) ou un logiciel de traitement de texte (document, paragraphe, phrase, …) sont appelés des *objets* et possèdent chacun un certain type. Le langage VBA permet de manipuler ces objets en appelant des méthodes auxquelles ils sont associés ou en obtenant et en définissant les propriétés de l'objet. En effet, les développeurs de la suite Microsoft Office ont pris soin de prendre en charge

l'automation et ainsi associer à chaque catégorie d'objet des méthodes qui lui sont propres. En en apprenant plus sur ces méthodes, il devrait être possible de traiter les objets pour, par exemple, transférer des objets Excel dans des objets Word et les remettre en forme.

5.2 Publication des résultats au format HTML

Une autre alternative intéressante serait d'exporter les résultats synthétiques obtenus avec Excel au format HTML. L'*HTML*, pour Hypertext Markup Language, est le langage de formatage de données utilisé pour écrire les pages web. Une fois écrites à l'aide de balises, celles-ci sont directement visualisables à l'aide d'un navigateur.

Cette approche serait complémentaire à la génération d'un rapport car elle permettrait de rendre consultables en ligne les résultats de l'analyse. La diffusion de ceux-ci peut donc être plus rapide et plus vaste. De plus, pour les personnes souhaitant simplement consulter les conclusions de l'étude ergonomique, cette solution permettrait de s'affranchir de l'achat de licences de produits Microsoft (Excel ou Word).

Il existe des solutions commerciales permettant la conversion de fichiers Excel au format HTML [40, 41], notamment le produit *SpreadsheetConverter to HTML* de la firme *Framtidsforum* qui convertit les fichiers à partir de JavaScript. Les technologies utilisées par ces produits constituent sans aucun doute de sérieuses pistes de recherche.

5.3 Aide interactive à l'évaluation des règles

Une autre fonctionnalité intéressante serait l'affichage d'une infobulle lors du survol d'une règle à l'aide de la souris. L'information affichée pourrait prendre la forme d'un exemple explicatif sur la façon dont la règle s'applique étant donné que certaines des règles composant les listes de vérification sont uniquement composées de mots-clés. En lieu et place d'une infobulle, on peut également imaginer la possibilité de cliquer sur un onglet à coté de la règle entrainant l'affichage de l'information recherchée.

Il serait intéressant d'ajouter cette fonctionnalité afin que le logiciel puisse être utilisé par une personne n'ayant pas de connaissances particulières sur la méthode heuristique. Ces messages d'aide éviteraient que l'utilisateur inexpérimenté doive constamment rechercher de l'information complémentaire sur les listes de vérification.

Conclusion

Pour rappel, l'objectif initial de ce travail était de développer un logiciel de support à l'évaluation de l'ergonomie d'un système informatique par la méthode heuristique.

Dans la première partie du travail, nous avons présenté un ensemble de méthodes permettant d'effectuer une analyse ergonomique. Parmi celles-ci, nous avons porté une attention toute particulière à la méthode heuristique.

Malgré les difficultés attachées à cette méthode, elle semble être la plus accessible, la moins compliquée à appliquer et la plus pratique. Cependant, il est intéressant de remarquer qu'une analyse ergonomique ne peut être complète en se limitant à cette seule méthode heuristique. Il faut donc faire suivre l'application de cette méthode par un test d'utilisabilité afin de connaître le vrai ressenti des utilisateurs.

Alors qu'il existe de nombreuses techniques d'analyse dites « discount », nous constatons qu'il n'y a que très peu d'outils permettant de faciliter la conduite d'une analyse ergonomique. De plus, les quelques outils existant ne sont pas facilement accessibles sur Internet. Dès lors, un développeur inexpérimenté en matière d'analyse ergonomique n'a que peu de chance de trouver les outils qu'il cherche.

En résumé, nous constatons que, malgré l'abondance d'information sur la méthode heuristique et malgré le fait que l'on tente de donner un accès plus large à l'ergonomie logicielle, les outils destinés à assister les développeurs sont incomplets, restent trop rares ou ne facilitent pas suffisamment leur tâche.

Dans la seconde partie de ce travail, nous avons présenté le logiciel que nous avons développé. Celui-ci est implémenté à l'aide de l'outil Microsoft Excel et communique avec l'utilisateur par l'intermédiaire de trois interfaces ayant chacune une fonctionnalité bien précise.

La première permet de saisir les paramètres de l'analyse ergonomique de manière centralisée et interactive. Notre logiciel supporte deux langues (Français et Anglais) et trois types de système informatique (site Internet, application et application pour téléphone mobile). Pour chacun de ces types, les listes de vérification les plus utilisées ont été incluses pour porter le nombre total de listes prises en compte à 11. Parmi les créateurs de ces listes, on compte Nielsen, Smith, Mosier et des organismes tels que le CNRS, le W3C et le BCHI. De plus, le logiciel offre la possibilité d'encoder une liste personnalisée. L'évaluation du respect des règles ergonomiques peut se faire

selon les modèles linéaires, pondérés et/ou avec importance.

La deuxième interface permet à l'utilisateur d'encoder les observations qu'il effectue au cours de l'analyse (commentaires, valeurs de respect, de pertinence et d'importance des règles ergonomiques). L'interface proposée permet de simplifier la tâche de l'utilisateur car il peut encoder rapidement certaines valeurs. Enfin, la présentation claire et standardisée des listes de vérification facilite la prise en main du logiciel.

La troisième et dernière interface propose une synthèse des données récoltées au cours de l'analyse. Des tableaux et graphiques sont calculés pour chaque liste de vérification ainsi que pour chaque catégorie de règles composant les listes. Ces résultats sont mis en forme pour en faciliter l'interprétation, et donc la déduction de recommandations.

Ce logiciel répond à son objectif initial car il accompagne l'utilisateur tout au long de sa tâche, il permet d'automatiser certains aspects de l'analyse et il aide l'utilisateur à tirer des conclusions des résultats récoltés.

Pour terminer, nous soulignons que, malgré les nombreuses fonctionnalités apportées par notre logiciel, notre travail n'a pas la prétention d'être exhaustif. En effet, des fonctionnalités additionnelles pourraient intéresser l'utilisateur. A ce sujet, nous avons émis quelques suggestions qui peuvent guider le lecteur souhaitant approfondir le travail réalisé. Parmi celles-ci, la génération automatique d'un rapport écrit à partir des résultats synthétiques obtenus avec Excel constitue un objectif important.

Bibliographie

[1] Direction des systèmes d'information du Centre National de la Recherche Scientifique, *Sensibilisation à la démarche d'analyse du travail*, 2000, Source (page consultée le 8 août 2010) : http://www.dsi.cnrs.fr/methodes/ergonomie/documentation/anatravail.pdf.

[2] L-CH. LAW et E. TH. HVNNBERG, Complementarity and Converence of Heuristic Evaluation and Usability Test: A case Study of UNIVERSAL Brokerage Platform, In *NordiCHI*, October 19-23, 2002, p. 71-80.

[3] Organisation internationale de normalisation, *Norme ISO 9241-11 : Ligne directrices relatives à l'utilisabilité*, 1998.

[4] J. NIELSEN, *Usability engineering*, Academic Press, San Francisco, 1994, ISBN 0-12-518406-9.

[5] J.M.C., BASTIEN et C. LEULIER, et D.L. SCAPIN, *L'ergonomie des sites web*, 1998, Source (page consultée le 8 août 2010) : http://educmath.inrp.fr/Educmath/dossier-manifestations/propositions-de-formation/archives/formations-educmath-2008-2009/Ressources/DocSupAtErgo-ergonomie_scapin-bastien.pdf.

[6] J. NIELSEN, *Discount usability : 20 years*, 2009, Source (page consultée le 8 août 2010) : http://www.useit.com/alertbox/discount-usability.html.

[7] J. NIELSEN et R. MOLICH, Improving a human-computer dialogue, In *Communication of the ACM*, 1990, v33-3, p. 338-348.

[8] A. CHEVALIER, Effet du niveau d'expertise des concepteurs sur la prise en compte de contraintes et sur la qualité ergonomique de maquettes de sites web, In *Tr. Hum.*, 2003, Vol. 66, p.127 - 160.

[9] J. HORN, *The usability methods toolbox handbook*, 1998, Source (page consultée le 8 août 2010) : http://www.idemployee.id.tue.nl/g.w.m.rauterberg/lecturenotes/UsabilityMethodsToolboxHandbook.pdf.

[10] M. BOUTIN, *Évaluation de l'utilisabilité d'un site Web : tests d'utilisabilité versus évaluation heuristique*, 2001, Source (page consultée le 8 août 2010) : http://www.crim.ca/Publications/2001/documents/plein_texte/ERG_BouMal_0101-08.pdf.

[11] G. ABOWD, *GOMS analysis techniques*, 1997, Source (page consultée le 8 août 2010) : http://www.cc.gatech.edu/classes/cs6751_97_fall/projects/closet2000+/FinalEssays/goms.html.

[12] J. NIELSEN et R.L. MACK, *Usability inspection methods*, John Wile & Sons, New York, 1994, p. 25-64.

[13] J. NIELSEN, *Ten usability heuristics*, 2005, Source (page consultée le 8 août 2010) : http://www.useit.com/papers/heuristic/heuristic_list.html.

[14] J. SOMERVELL et D. S. MCCRICKARD, Better discount evaluation : illustrating how critical parameters support heuristic creation, In *Interact. Comput.*,2005, vol.17, p. 592-612.

[15] A. CHEVALIER et N. FOUQUEREAU et J. VANDERDONCKT, Entre contraintes ergonomiques, créativité et esthétique : rôle d'un système à base de connaissance sur l'activité des concepteurs web, In *Tr. Hum.*, 2009, Vol. 72, p. 23-42.

[16] N. DANINO, *Heuristic evaluation – a step by step guide*, 2001, Source (page consultée le 8 août 2010) : http://articles.sitepoint.com/article/heuristic-evaluation-guide.

[17] Usabilitybok, *Methods : heuristic evaluation*, Source (page consultée le 8 août 2010) : http://www.usabilitybok.org/methods/p275?section=how-to.

[18] J. NIELSEN, *Why you only need to test with 5 users*, 2000, Source (page consultée le 8 août 2010) : http://www.useit.com/alertbox/20000319.html.

[19] J. NIELSEN, Finding usability problems through heuristic evaluation, In *CHI conference proceedings*, 1992, p. 373 – 380.

[20] C. LING et G. SALVENDY, «Effect of evaluators' cognitive style on heuristic evaluation : field dependent and field independent evaluators», *Int. J. Human-computer studies*, Vol. 67, 2009, p. 382-393.

[21] J. NIELSEN, *Severity rating for usability problems*, 2006, Source (page consultée le 8 août 2010) : http://www.useit.com/papers/heuristic/severityrating.html

[22] M. KAPTEIN et C. NASS et P. MARKOPOULOS, Powerful and consistent analysis of Likert-type rating scale, In *CHI conference proceedings*, 2010.

[23] J. VANDEDONCKT, Development milestones towards a tool for working with guidelines, In *Interact. Compu.*, 1999, vol. 12, p. 81-118.

[24] GUIguide, *The solution for managing design knowledge*, Source (page consultée le 8 août 2010): http://www.guiguide.com/.

[25] J. NIELSEN, Enhancing the explanatory power of usability heuristics, In *CHI Conference Proceedings*, 1994, p. 152-158.

[26] J. NIELSEN & R. Molich, Heuristic evaluation of user interfaces, In *CHI Conference proceedings*, 1990, p. 249-256.

[27] W.-S. TAN et D. LIU et R. BISHU, Web evaluation : heuristic evaluation vs. user testing, In *Int. J. Ind. Ergonom.*, 2009, vol. 39, p.621-627.

[28] Direction des systèmes d'information du Centre National de la Recherche Scientifique, *Guide de recommandations ergonomiques pour sites et applications web*, 2005, Source (page consultée le 30 juillet 2010) : http://www.dsi.cnrs.fr/methodes/ergonomie/documentation/Guidergoweb2005.pdf.

[29] J. NIELSEN et M. TAHIR, *Homepage Usability: 50 Websites Deconstructed*, New Riders Publishing, Indianapolis, 2001. ISBN 0-7357-1102-X.

[30] J. VANDERDONCKT, Guide ergonomique des interfaces homme-machine, Presses Universitaires de Namur, Namur, 1994. ISBN 2-87037-189-6.

[31] W3C Recommendation, *Web Content Accessibility Guidelines 1.0*, 1999, Source (page consultée le 30 juillet 2010) : http://www.w3.org/TR/WAI-WEBCONTENT/.

[32] Information Services and Technology, Massachusetts Institute of Technology, *Usability guidelines*, Source (page consultée le 30 juillet 2010) : http://ist.mit.edu/services/consulting/usability/guidelines.

[33] B. TOGNAZZINI (Nielsen Norman Group), *First Principles of Interaction Design*, 2003, Source (page consultée le 30 juillet 2010) : http://www.asktog.com/basics/firstPrinciples.html.

[34] S. SMITH et J. MOSIER, *Guidelines for designing user interface software*, The MITRE Corporation, Bedford, Massachusetts, USA, 1986, ESD-TR-86-278, Source (page consultée le 30 juillet 2010) : http://www.hcibib.org/sam/.

[35] Direction des systèmes d'information du Centre National de la Recherche Scientifique, *Guide de recommandations ergonomiques pour la conception et l'évaluation d'interfaces graphiques*, 2000, Source (page consultée le 30 juillet 2010) : http://www.dsi.cnrs.fr/methodes/ergonomie/documentation/guidergo.pdf.

[36] N. EDGAR et K. HAALAND et J. LI et K. PETER, *Eclipse User Interface Guidelines* (Version 2.1), 2004, Source (page consultée le 30 juillet 2010) : http://www.eclipse.org/articles/Article-UI-Guidelines/Contents.html.

[37] W3C Recommendation, *Mobile Web Best Practices 1.0*, 2008, Source (page consultée le 30 juillet 2010) : http://www.w3.org/TR/mobile-bp/.

[38] J. VANDERDONCKT, *Introduction à l'ergonomie des sites Internet*, Université Catholique de Louvain, Louvain-la-Neuve, 2002.

[39] Microsoft (France), *Aide et support Microsoft : Comment faire pour utiliser l'automation (OLE) avec Word*, Source (page consultée le 6 août 2010) : http://support.microsoft.com/kb/184974/fr.

[40] SpreadsheetConverter (Suède), *SpreadsheetConverter to HTML, Flash, ASP.NET or Java*, Source (page consultée le 7 août 2010) : http://www.exceleverywhere.com/products/flavors.htm.

[41] Flash Utility, *Convert Excel to HTML*, Source (page consultée le 7 août 2010) : http://www.flash-utility.com/convert-excel-to-html.html.

[42] A.J. THOMPSON et E, KEMP, Web 2.0: Extending the framework for heuristic evaluation, In *CHINZ*, 2009, p.29-36.

[43] J. NIELSEN, *Quantitative studies: how many users to test?*, 2006, Source (page consultée le 19 août 2010) : http://www.useit.com/alertbox/quantitative_testing.html.

[44] C. ROHRER, *When to use which user experience research methods*, 2008, Source (page consultée le 19 août 2010): http://www.useit.com/alertbox/user-research-methods.html.

Annexes

Ces annexes contiennent l’ensemble des listes de vérification incluses à notre logiciel de support.

Celles-ci sont reprises uniquement dans leur langue originale.

1 Site web

1.1 Liste de vérification du CNRS

1. Le système de Navigation

1) Au sein d'une application, le système de navigation doit être autonome.
2) Une barre de navigation doit être présente et identique sur toutes les pages
3) Dans une barre de navigation, l'élément actif doit être mis en évidence visuellement.
4) Mieux vaut éviter d'avoir un lien actif qui pointe sur la page courante.
5) Utiliser un chemin de navigation pour les applications dont l'arborescence compte trois niveaux ou davantage.
6) Sur chacune des pages de l'application, prévoir un lien vers la page d'accueil
7) Chaque page de l'application doit proposer au minimum un lien.
8) Un lien doit être clairement reconnaissable.
9) Il faut pouvoir repérer facilement l'unité du lien.
10) Les liens textuels doivent être différenciés du texte normal.
11) Dans une liste de liens qui ont du texte en commun, ne souligner que le texte qui est différent.
12) Dans une liste de liens positionnée verticalement, ne pas séparer des intitulés par des signes qui ressemblent à des puces.
13) Ne pas mettre de liens trop proches les uns des autres.
14) Les liens inactifs doivent être plus discrets que les liens actifs.
15) Concernant les liens graphiques: pour indiquer qu'il s'agit de liens et non d'images simples donner un léger relief aux images.
16) Eviter de placer un lien parmi un ensemble d'autres liens, surtout si ceux-ci sont de petite taille.
17) Un lien doit être explicite dans son contexte.
18) Pour les pages de longueur importante contenant essentiellement du texte, utiliser des signets permettant de revenir en haut de la page.
19) Lorsqu'un lien mène à l'extérieur de l'application, le site en question doit s'ouvrir dans une nouvelle fenêtre du navigateur.
20) Pour faire un lien textuel, sélectionner un mot ou un groupe de mots pertinents.
21) Intégrer le lien textuel au texte, c'est-à-dire positionner le lien sur un élément significatif du texte.
22) Les liens graphiques doivent être doublés d'un texte court.
23) L'intitulé du lien doit être quasiment identique au titre de la page à laquelle il renvoie.
24) Ne pas nommer de lien "page suivante" ou "page précédente".
25) Les liens doivent avoir un comportement homogène dans l'ensemble de l'application.
26) L'activation d'un lien pointant vers un document .pdf ou .doc doit entraîner l'ouverture d'une

nouvelle fenêtre du navigateur.
27) Un même lien doit toujours avoir un même effet et un même effet doit toujours être produit par un même lien.
28) Réserver les liens pour des actions de navigation et des boutons d'action pour les actions.
29) Eviter de répéter à l'intérieur d'une même page des liens ayant un même effet.

2. La présentation

30) Les pages doivent s'afficher par défaut en 800X600.
31) Présence d'une homogeneité dans la présentation des pages de l'application.
32) Les applications Web peuvent utiliser les cadres de façon judicieuse.
33) Le système de navigation doit occuper une surface minimale.
34) Les outils de navigation ainsi que les informations critiques doivent se situer en haut de page.
35) Ne pas hésiter à utiliser des lignes horizontales de séparation.
36) Les en-têtes de colonne doivent être centrés par rapport à la largeur de la colonne.
37) Le contenu des colonnes doit être cadré à gauche s'il s'agit de texte et à droite s'il s'agit de valeurs numériques.
38) Adopter un interlignage suffisant entre les lignes des tableaux afin d'optimiser la lisibilité de ceux-ci.
39) Si les lignes d'un tableau peuvent être sélectionnées, cette sélection peut s'effectuer de plusieurs manières.
40) Si les lignes d'un tableau peuvent être développées/décomposées en sous-éléments, ces lignes doivent être identifiées.
41) Si les informations détaillées relatives à chaque ligne d'un tableau sont peu nombreuses, alors l'affichage de ces informations peut se faire en-dessous du tableau.
42) Si les informations détaillées relatives à chaque ligne d'un tableau sont nombreuses, alors l'affichage de ces informations se fera dans une page spécifique.
43) Si le nombre de lignes de saisie est supérieur au nombre de lignes initialement affichées dans le tableau, une fonction « Insérer/créer une nouvelle ligne » doit être proposée.
44) Il peut dans certains cas être judicieux de donner à l'utilisateur la possibilité de modifier l'ordre de présentation des lignes du tableau.
45) Il peut dans certains cas être judicieux de donner à l'utilisateur la possibilité de faire circuler les éléments d'un tableau vers un autre tableau.
46) L'ascenseur vertical n'apparaît que si l'ensemble des lignes contenues dans le tableau n'est pas visible d'emblée.
47) Eviter les tableaux trop larges.
48) Eviter d'écrire en italique.
49) Ne pas employer le soulignement hormis pour les liens.
50) Ecrire tous les élément textuels en minuscules avec une majuscule à l'initiale.
51) Les formats de police doivent être spécifiés.
52) La taille des polices devra être proportionnelle au niveau de hiérarchie de l'information.
53) Eviter d'utiliser plus de quatre polices de caractères différentes dans une même page.
54) En règle générale, utiliser plutôt des polices sérif pour les corps de texte et des polices sans sérifs pour les titres, sous-titres, etc.
55) Plutôt que de justifier le texte, l'aligner à gauche.
56) L'espace entre les caractères doit être d'au moins un pixel.

57) Les dates seront plutôt affichées au format JJ/MM/AAAA.
58) Préférer un fond clair (voire blanc) à un fond sombre (fatigant pour les yeux).
59) Rester « sobre » dans la couleur de fond.
60) Limiter le nombre de couleurs à 7 (+ ou – 2).
61) Respecter la signification que l'utilisateur attribue à une couleur donnée.
62) Attention à la visibilité des couleurs.
63) Employer les pictogrammes pour les objets souvent utilisés.
64) Les pictogrammes utilisés sont représentatifs de l'action/du concept que l'on souhaite représenter, ils sont explicites, compréhensibles et doubler d'un libellé.
65) Homogénéité : quand un pictogramme est utilisé pour représenter quelque chose, le conserver pour l'ensemble de l'application et entre les applications.
66) Pour les applications DSI, une bibliothèque d'icônes est disponible sur demande auprès du graphiste de la DSI du BMSD.

3. Le contenu

67) N'afficher que les fonctionnalités pour lesquelles le profil utilisateur a le droit d'accès.
68) Un titre de page ne peut pas être un lien.
69) Le titre des pages précise le mode dans lequel on se trouve: consultation, mise à jour,…
70) Si seule une partie de la page concerne l'utilisateur, s'assurer que l'utilisateur comprenne qu'il y a différents cas de figure.
71) Lorsque plusieurs cas de figure d'utilisation sont possibles, l'utilisateur peut aisément reconnaître quel cas de figure lui est propre.
72) Le vocabulaire utilisé appartient au domaine d'activité de l'utilisateur
73) Les termes utilisés sont explicites et sans ambiguité, compréhensible pour l'utilisateur.
74) L'usage d'abréviation est limité au stricte minimum et fait de manière pertinente.
75) Le même objet ou la même action est contamment désigné par le même libellé.
76) Les boutons d'action sont uniquement utilisés pour les actions et non pas pour les actions de naviguation.
77) Le libellé du bouton est compréhensible, non-ambigu, pas trop long et représentatif de l'action correspondante.
78) Lorsque le bouton est inactif/inaccessible, sa couleur est estompée/grisée.
79) Les boutons de même nature fonctionnelle doivent être groupés et distinguables.
80) Lorsque la page contient des onglets, les boutons d'action associés à l'ensemble des onglets doivent se trouver à l'extérieur des onglets tandis que ceux associés à un seul onglet doivent être positionnés sur ce seul onglet.
81) La touche 'Entrée' doit être active et liée par défaut à l'action qui a la plus grande probabilité d'être choisie par l'utilisateur.
82) La page d'accueil ne contient qu'un seul écran et indique la finalité de l'application
83) Le contenu purement informatif d'une page représente plus de 50% de l'espace disponible tandis que le système de navigation n'occupe pas plus de 20% de l'espace
84) Les ascenseurs verticaux dispensables disparaîssent automatiquement
85) Les pages dépassant la longueur de l'écran disposent d'un 'sommaire' propre à la page

4. La déconnexion

86) Pour quitter l'application, il sera dans certains cas nécessaire de proposer un lien « Déconnexion ». L'activation de ce lien – généralement situé en haut à droite des pages –

entraîne la fermeture de la page courante et affiche la page de connexion.

87) Il peut être judicieux de mettre un Time Out en place. Ce système permet une déconnexion automatique à l'initiative du système et non de l'utilisateur dès qu'un certain laps de temps s'est écoulé sans que l'application ait été utilisée.

5. L'impression

88) Pour les données affichées à l'écran qui peuvent être imprimées, distinguer l'action « imprimer » de la version à imprimer.
89) La date d'impression doit systématiquement être présente sur la version imprimable.

6. La page d'identification

90) Suite à l'authentification, les fonctions auxquelles l'utilisateur a droit d'accès lui sont proposées
91) La page d'identification apparaît dans une fenêtre pop-up non redimensionnable.
92) La page d'identification contient le logo, le nom de l'application,une zone de saisie pour le code d'accès, le nom du créateur de l'application

7. La page d'accueil

93) Dans le cadre d'un site applicatif, la page d'accueil doit proposer les modules de l'application et eventuellement les actualités
94) La page d'accueil doit comporter un lien vers le formulaire de demande d'assistance
95) La page d'accueil peut mentionner qui a réalisé quel service ou application

8. Les pages de formulaires de saisie

96) La saisie effectuée doit être validée et ce via un bouton 'valider' ou 'enregistrer'
97) Une fois l'information validée, un retour système doit être fourni pour confirmer la validation
98) En cas de chronologie imposée sur les étapes de saisie, celle-ci doit être matérialisée (disposition, numérotation, fléchage, encadrement des différentes étapes).
99) Les données à saisir sont divisées en sous-ensembles avec 1 onglet par sous-ensemble et la saisie d'un s-e donné est pré-requise pour la saisie suivante
100) Le contrôle de cohérence s'effecture à chaque activation du bouton permettant de passer à l'onglet suivant. Seul le dernier onglet contiendra un bouton <Enregistrer> ou <Valider>
101) Des repères dans la chronologie de saisie doivent être indiqués afin que l'utilisateur puisse à tout moment savoir à quelle étape il se trouve au sein du processus et quel est son état d'avancement.
102) L'utilisateur doit pouvoir saisir les données dans l'ordre qu'il souhaite et doit pouvoir naviguer d'un onglet à l'autre sans restrictions
103) Les contrôles de cohérences s'effectuent uniquement via le bouton <Enregistrer> ou <Valider>
104) Les champs sont ordonnés en fonction soit de leur complémentarité, soit d'une séquence logique, soit d'un ordre d'importance
105) Le contenu d'un champ se situe sur la même ligne horizontale que le libellé du champ
106) Les libellés de champs sont verticalement alignés à droite et être suivis de ":"
107) Les champs sont verticalement alignés à gauche entre eux
108) La longueur de champ est adaptée au nombre de caractères attendu (ni trop ni trop peu)
109) Les champs d'affichage sont écrits en texte normal, hors cadre

110) Lors de la transformation d'une page de saisie en page de consultation, l'aspect des champs de saisie est modifié et devient un champ d'affichage
111) Le libellé est un nom et indique le résultat de la saisie et non l'action de saisir
112) Les libellés correspondent au vocabulaire propre aux utilisateurs
113) L'usage des sigles est banni, excepté dans le cas ou ces sigles sont connus et utilisés par les utilisateurs
114) La saisie de caractères n'est pas forcée en minuscules ou majuscules
115) Dans le cas ou des unités de mesures sont associées à un champ de saisie, celles-ci apparaîssent à la suite du champ
116) Pour les champs nécessitant un format particulier, un préformatage des données est proposé ou alors celui-ci est automatique et effectué a posteriori
117) Pour les champs de saisie libre, l'utilisateur est informé du nombre de caractères maximum acceptés
118) Pour certains champs critiques, une aide à la saisie est proposée sous forme de liste d'aide
119) Lorsque le listing est long, un positionnement direct basé sur la frappe du premier caractère est proposée
120) Les champs obligatoires ne contiennent jamais de valeur par défaut, seuls les champs de moindre importance le font
121) Les champs obligatoires sont clairement identifiés comme tels
122) Le système de codage utilisé est soit l'asterisque, soit la mise en couleur du champ obligatoire, en aucun cas les deux modes sont utilisés simultanément
123) A l'affichage de chaque page, le curseur est positionné par défaut dans le premier champ de saisie
124) Des boutons d'option sont utilisés pour des choix exclusifs
125) L'utilisateur dispose d'au moins 2 propositions. Si le choix n'est pas significatif, les options peuvent être grisées
126) Un des boutons d'option est sélectionné par défaut si une valeur par défaut est définissable. L'option "Aucun(e)"est disponible si on peut choisir de ne sélectionner aucun des choix
127) Le libellé est écrit à droite du bouton d'option
128) Pour les choix importants au niveau fonctionnel, l'usage des cases à cocher est proscrit
129) L'usage des cases à cocher est fait de manière pertinente vis-à-vis du contexte courant. Les libellés des cases non-utilisables sont grisés
130) Le libellé est écrit à droite de la case à cocher
131) La désélection d'un élément est simple et peut être réalisée par le même moyen que la sélection
132) L'élément sélectionné est immédiatement mis en évidence
133) La sélection d'un élément n'entraîne pas d'action et sa désélection n'a pas d'incidence sur la tâche en cours
134) Les listes sont utilisées pour afficher une suite d'éléments; la sélection de ces éléments est unique ou multiple
135) Les éléments sont affichés dans un ordre logique vis-à-vis de l'utilisateur ou de la fréquence d'utilisation
136) Le curseur ne peut pas "boucler" sur la liste; une fois l'extrémité atteinte, le curseur reste dessus
137) le positionnement direct basé sur la frappe du premier caractère est activé et l'initiale de

chaque élément est mise en majuscule afin de favoriser la lisibilité

138) Les boutons de validation sont positionnés au dessous des champs et centrés

9. Les pages de recherche et de résultats

139) Toutes les pages de recherche et de résultats de l'application comportent un lien vers le formulaire d'assistance

140) Le bouton qui lance le processus est appelé <Lancer la Recherche>, <Rechercher> ou <Chercher>

141) Le mode de recherche par date est facilité par un pré-formatage/guidage de l'information

142) La requête 'critères de recherche' et les résultats de celle-ci figurent sur une seule et même page ou sur deux pages distinces

143) L'affichage d'une liste de résultat sur une seule et même page est accompagné d'une indication du nombre de résultats trouvés pour la requête

144) L'affichage d'une liste de résultats répartie en plusieurs pages donne lieu à l'affichage du nombre de résultats affiché par page ainsi que du nombre total de résultats trouvés. Un moyen de naviguer entre ces différentes pages est offert.

145) Le numéro de la page courante est aussi mentionné par rapport à l'ensemble des pages de résultats ramenées

146) Un moyen de naviguer entre ces différentes pages est offert.

147) Le nombre de résultats affichés par page ou bloc est paramétrable par l'utilisateur sauf si lui laisser le choix n'est pas judicieux

148) Si le tri des listes de résultats est permis à l'utilisateur, celui-ci peut s'effectuer depuis le tableau, soit à partir d'une fenêtre pop-up réservée à cette opération ou globalement destinée au paramétrage de l'affichage du tableau

149) Un lien de retour à la liste des résultats est prévu sur les pages "détail [de l'élément n]"

10. Les methods d’impression

150) Une version imprimable, autre que la copie d'écran, est disponible via un lien <version imprimable> permettant d'imprimer une version épurée de tout ou partie de la page courante

11. L’aide systématique

151) L'ergonomie de l'application facilite son utilisation de façon considérable

152) Les messages s'affichent dans une boîte de message de type Windows,dans un pop-up Web ou dans une page html nouvelle ou courante

153) L'espace qui leur est réservé se trouve entre le titre de la page et le corps. Chaque type de message (information/confirmation/bloquant) est identifié par un pictogramme spécifique positionné en début de message à gauche.

154) Le message indique à l'utilisateur au moment opportun le résultat d'une action ou la fin d'une action. Ces actions ne présentent aucun risque pour l'utilisateur (perte de données…)

155) Ce type de message est systématiquement affiché préalablement aux 'opérations irréversibles. Elle permet à l'utilisateur de valider l'opération en connaissance de cause ou de l'annuler et de passer à une autre tâche

156) Le message indique à l'utilisateur une erreur de saisie ou une omission (champ obligatoire non saisi, problème de format…)

12. L’aide à l’initiative de l’utilisateur

157) Une aide est prévue pour les saisies comportant un risque au niveau du format qui peut

être générateur d'erreur (date). Cette aide prend la forme d'un formatage automatique ou d'un calendrier accessible sous forme de pictogramme par exemple.

158) Une aide est prévue lorsque un champ se compose d'une certain nombre de valeurs possibles. Cette aide prend la forme d'une page (pop-up) spécifique et dont la taille est adaptée au nombre de valeurs possibles (liste déroulante)

159) Lorsque qu'un champ, une rubrique ou une page donnée est susceptible de mettre l'utilisateur en difficulté, un lien sous forme de point d'interrogation est placé à côté du champ, au niveau de la rubrique ou en haut de page à droite du titre.

160) Lorsque le texte informatif est très bref, l'utilisation d'une bulle d'aide est préférée à une fenêtre pop-up.

161) Relative à l'ensemble de l'application et accessible via un lien "aide" positionnné tout en haut de l'écran, au dessus du titre de page et au niveau du nom du titre de l'application.

162) Constituée de plusieurs rubriques: objectifs, fonctionnement général, explications des modules & fonctions, les éléments d'ergonomie

1.2 Liste de vérification de Nielsen et Tahir

1. Communicating the Site's Purpose

1) Show the company name and/or logo in a reasonable size and noticeable location
2) Include a tag line that explicitly summarises what the site or company does
3) Emphasize what your site does that's valuable from users point of view
4) Emphasize the highest priority tasks so that users have a clear starting point on the homepage
5) Clearly designate one page per site as the official homepage
6) On main company website, don't use the word "website" to refer to anything but the totality of the company's web presence
7) Design the homepage to be clearly different from all other pages

2. Communicating Information about your company

8) Group corporate info, e.g. about Us, Investor Relations, Press Room, Employment etc in one distinct area
9) Include homepage link to "About Us" section to give overview of company and links to relevant details about products, services, company values, business proposition, management team etc.
10) If you want to get press coverage for your company include a "Press Room" or "New Room" link on homepage
11) Present unified face to customer in which the website is one of the touch points rather than an entity unto itself.
12) Include a "Contact Us" link on homepage that goes to a page with all contact info for your company.
13) If your provide a "feedback" mechanism specify purpose of link and whether will be read by customer service of the webmaster.
14) Don't include internal company information (which is targeted for employees and should go on the intranet) on the website.

15) If your site gathers any customer information include a "'\ "Privacy Policy" link on the homepage
16) Explain how the website makes money if it's not self-evident

3. Content Writing

17) Use customer-focused language. Label sections and categories according to the value they hold for the customer, not according to what they do for your company.
18) Avoid redundant content
19) Don't use clever phrases and marketing lingo that make people work too hard to figure out what you're saying.
20) Use consistent capitalization and other style standards.
21) Don't label a clearly defined area of the page if the content is sufficiently self-explanatory
22) Avoid single-item categories and single-item bulleted lists
23) Use non-breaking spaces between words in phrases that need to go together in order to be scannable and understood.
24) Only use imperative language such as "Enter a City or Zip Code" for mandatory tasks, or qualify the statement appropriately
25) Spell out abbreviations, initials and acronyms, and immediately follow them by the abbreviation in the first instance.
26) Avoid exclamation marks
27) User all uppercase letters sparingly or not at all as a formatting style.
28) Avoid using spaces and punctuation inappropriately for emphasis

4. Revealing Content Through Examples

29) Use examples to reveal the site's content rather than just describing it.
30) For each example, have a link that goes directly to the detailed page for that example, rather than to a general category page of which that item is a part.
31) Provide a link to the broader category next to the specific example
32) Make sure it's obvious which links lead to follow-up information about each example and which links lead to general information about the category as a whole.
33) Archives and Accessing Past Content
34) Make it easy to access anything that has been recently featured on your homepage, e.g. in last two weeks or month, by providing a list of recent features, as well as putting recent items into the permanent archives.

5. Links

35) Differentiate links and make them scannable
36) Don't use generic instructions, such as "Click Here'! as a link name
37) Don't use generic links such as "More..." at the end of a list of items.

6. Navigation

38) Locate the primary navigation area in a highly noticeable place, preferably directly adjacent to the main body of the page.
39) Group items in the navigation area so similar items are next to each other
40) Don't provide multiple navigation areas for the same type of links
41) Don't include an active link to the homepage on the homepage.
42) Don't use made-up words for category navigation choices.

43) Categories need to be immediately differentiable from each other - if users don't understand your terminology, it will be impossible for them to differentiate categories.
44) If you have a shopping Cart feature on your site, include a link to it on the homepage.
45) Use icons in navigation only if they help users to recognize a class of items immediately such as new items, sale items or video content.

7. Search

46) Give users an input box on the home page to enter search queries, instead of just giving them a link to a search page.
47) Input boxes should be wide enough for users to see and edit standard queries on the site. (25-30 characters)
48) Don't label the search area with a heading; instead us a "Search" button to the right of the box.
49) Unless advanced searches are the norm on your site, provide simple search on the homepage, with a link to advanced search or search tips if you have them.
50) Search on the homepage should search the entire site by default
51) Don't offer a feature to "Search the Web" from the sites search function.

8. Tools and Task Shortcuts

52) Offer users direct access to high-priority tasks on the homepage
53) Don't include tools unrelated to tasks users come to your site to do.
54) Don't provide tools that reproduce browser functionality, such as setting a page as the browser's default starting page or book marking the site.

9. Graphics and Animation

55) Use graphics to show real content, not just to decorate your homepage.
56) Label graphics and photos if their meaning is not clear from the context of the story they accompany
57) Edit photos and diagrams appropriately for the display size.
58) Avoid watermark graphics (background images with text on top of them.
59) Don't use animation for the sole purpose of drawing attention to an item on the homepage. Animation rarely has a place on the homepage because it distracts from other elements.
60) Never animate critical elements of the page, such as the logo, tag line or main headline.
61) Let users choose whether they want to see an animated intro to your site - don't make it the default.

10. Graphic Design

62) Limit font styles and other text formatting, such as sizes, colors and so forth on the page because over-designed text can actually detract from the meaning of the words.
63) Use high-contrast text and background colors so that type is as legible as possible
64) Avoid horizontal scrolling at 800X 600
65) The most critical page elements should be visible "above the fold" (in the first screen of content, without scrolling) at the most prevalent window size (800X600 at the time of this writing).
66) Use a liquid layout so the homepage size adjusts to different screen resolutions.
67) Use logos judiciously.

11. UI Widgets

68) Never use widgets for parts of the screen that you don't want people to click e.g. if you use graphical bullets next to text, make them clickable as well as the text.
69) Avoid using multiple text entry boxes on the homepage, especially in the upper part of the page where people tend to look for the search feature. Users sometimes confuse text entry boxes with search boxes.
70) Use drop-down menus sparingly, especially if the items in them are not self-explanatory. Users are attracted to them and they're often the least effective navigation devices.

12. Window Titles

71) Begin the window titles with the information-carrying word - usually the company name.
72) Don't include the top-level domain name, such as ".com" in the window title unless it is actually part of the company name, such as "Amazon.com".
73) Don't include "homepage" in the title. This adds verbiage without value.
74) Include a short description of the site in the window title.
75) Limit window titles to no more than seven or eight words and few than 64 total characters.

13. URLs

76) Homepages for commercial websites should have the URL httQ://www.comoanv.com (or an equivalent for your country or non-commercial top-level domain.
77) For any website that has an identity closely connected to a specific country other than the United Stages, use that country's top-level domain.
78) If available register domain names for alternative spellings, abbreviations, or common misspellings of the site name.
79) If you have alternative domain name spellings, choose one as the authorized version and redirect users to it from all the other spellings.

14. News and Press Releases

80) Headlines should be succinct yet descriptive to give maximum information in as few words as possible.
81) Write and edit specific summaries for press releases and news
82) Link headlines rather than the deck to the full news story.
83) As long as all news stories on the homepage occurred within the week there's no need to list the date and time in the deck V of each story unless it is truly a breaking news item that has frequent updates.

15. Popup Windows and Staging Pages

84) Take users to your "real" homepage when they time your main URL or click a link to your site. Splash screen must die. An exception is if the site has material inappropriate for minors or which is offensive.
85) Avoid popup windows.
86) Don't use routing pages for users to choose their geographical location unless you have versions of your site in many \ different languages with no single dominant language.

16. Advertising

87) Keep ads for outside companies on the periphery of the page.

88) Keep external ads (ads for companies other than your own) as small and discreet as possible relative to your core homepage content.
89) If you place ads outside the standard banner area at the top /' of the page, label them as advertising so users don't confuse them with your site's content.
90) Avoid using ad conventions to showcase regular features of the site.

17. Welcomes

91) Don't literally welcome users to your site. Before you give up prime homepage real estate to a salutation, consider using it for a tag line instead.

18. Communicating Technical Problems and Handling Emergencies

92) If the website is down or important parts of the website are not operational, show it clearly on the homepage.
93) Have a plan for handling critical content on your website in \ the event of an emergency.

19. Credits

94) Don't waste space crediting the search engine, design firm, favourite browser company or the technology behind the scenes.
95) Exercise restraint in displaying awards won by your website

20. Page Reload and Refresh

96) Don't automatically refresh the homepage to push updates to users
97) When doing a refresh, update only content that has actually changed, such as news update.

21. Customisation

98) If your homepage has areas that will provide customized information once you know something about the user, don't provide a generic version of the content to first-time users - craft different content for that space.
99) Don't offer users features to customize the basic look of the homepage UI such as color schemes.

22. Gathering Customer Data

100) Don't provide plain links to registration on the homepage; instead explain (or at least link to) the customer benefits of registration.
101) Explain the benefits and frequency of publication to users before asking them for their email addresses.

23. Fostering Community

102) If you support user communities or other discussion features, don’t show generic links to them.
103) Don't offer a "Guestbook" sign in for business sites.

24. Dates and Times

104) Show dates and times for time-sensitive information only such as news items, live chats, stock quotes.
105) Show users the time that content was last updated not the computer-generated current time.
106) Include the time zone you are using whenever you reference a time.

107) Use standard abbreviations, such as p.m. or P.M.
108) Spell out the month or use month abbreviations, not in numbers.

25. Stock Quotes and Displaying Numbers

109) Give the percentage of change, not just the points gained or lost in stock Quotes.
110) Spell out stock abbreviations unless the abbreviation is completely clear, such as "IBM".
111) Use a thousands separator appropriate to your locale for numbers that have five or more digits.
112) Align decimal points when showing columns of numbers.

1.3 Liste de vérification du BCHI

1. Conception globale d'un site

1) Le contenu informationnel d'un site doit être représentatif
2) Inviter le visiteur à ajouter un signet
3) La conception d'un site doit être cohérente
4) Penser à une structuration appropriée du site
5) Utiliser la hiérarchie avec soin
6) Utiliser la clôture transitive
7) Utiliser des sous-menus pour les sites volumineux
8) Prévoir nécessairement une page d'accueil
9) Prévoir éventuellement une page d'invitation
10) Prévoir similairement une page de clôture
11) Prévoir une option d'impression pour les grands sites
12) Prévoir une section « Nouveautés » évidente
13) Adopter une logique de dénomination du site
14) Adopter une logique de dénomination des fichiers du site
15) Adopter une logique d'organisation des fichiers du site
16) La conception du site doit être appropriée au contexte
17) La conception d'un site doit supporter plusieurs navigateurs
18) La conception d'un site doit viser l'ergonomie en priorité
19) Préparer les pages pour les moteurs de recherche externes
20) Ajouter un moteur de recherche interne
21) Autoriser une recherche progressivement plus détaillée
22) Assurer une certaine flexibilité linguistique

2. Navigation

23) L'accès à l'information doit être rapide
24) Considérer le temps d'accès à chaque page
25) Utiliser une carte du site pour dénoter les relations entre les informations
26) Utiliser une image réactive comme carte graphique de navigation
27) Ne pas restreindre la navigation aux graphiques
28) Inclure des repères de navigation
29) L'accès à l'information doit être rapide
30) Les repères de navigation doivent être cohérents

31) Les liens doivent pointer vers un contenu informationnel substantiel
32) Fournir des liens textuels pour chaque page
33) Les liens textuels doivent être clairs
34) Les liens textuels doivent être courts
35) Utiliser "Précédent/Suivant", "Avant/Arrière" avec précaution
36) Bannir les liens "Cliquer ici"
37) Interdire les liens "Retourner à..."
38) Tout visiteur doit distinguer un lien visité d'un lien nouveau
39) Informer le visiteur de la teneur du lien
40) Utiliser des liens intra-page dans les longues pages
41) Veiller à la maintenance des liens

3. Présentation

42) L'information la plus importante doit être placée en premier lieu
43) La présentation de toute page doit être cohérente
44) La page d'accueil doit tenir sur un seul écran
45) Ne pas contraindre physiquement la présentation de la page d'accueil
46) L'information doit être complète sur le même écran
47) Trancher entre des pages courtes ou longues
48) Utiliser une grille de présentation
49) Utiliser les techniques visuelles
50) Utiliser au maximum les tables
51) Les lignes textuelles du contenu informationnel doivent être courtes, structurées
52) Penser aux feuilles de style
53) Utiliser de l'espacement blanc
54) Être draconien avec le défilement
55) Les informations liées sémantiquement doivent être présentées conjointement

4. Formulaires, titres et en-têtes

56) Les informations à acquérir doivent être présentées dans un formulaire
57) Sélectionner les objets interactifs de manière appropriée
58) Toute page doit comporter un titre identifiant
59) Utiliser des en-têtes pour accélérer le parcours des informations
60) Pour les documents structurés, communiquer la structure au visiteur

5. Cadres et fenêtres

61) Utiliser les cadres avec parcimonie
62) Utiliser les fenêtres seulement si nécessaire
63) Le nombre de cadres, de fenêtres doit être limité
64) Utiliser les cadres, les fenêtres de manière appropriée
65) Veiller à la taille des cadres et fenêtres

6. Graphiques

66) Utiliser les graphiques de manière appropriée
67) Adjoindre du texte à chaque graphique
68) Utiliser le texte alternatif
69) Réutiliser les mêmes graphiques

70) Utiliser les graphiques pour représenter les zones du site
71) Recourir aux graphiques pour les en-têtes
72) Utiliser les graphiques pour les listes
73) Garder en tête les limites des graphiques
74) Utiliser les miniatures
75) Les dimensions des graphiques doivent être appropriées
76) Les graphiques doivent être simplifiés
77) Utiliser les GIF entrelacés, les images fractales
78) Ne pas ajouter de bords aux graphiques
79) Utiliser des graphiques transparents sur un fond d'écran tramé
80) Spécifier les dimensions d'un graphique en HTML

7. Éléments multimédia

81) Le type de fond d'écran doit être approprié
82) Préférer les fonds d'écran clairs, de basse intensité
83) Éviter les fonds d'écran avec motifs
84) Sélectionner un fond d'écran aléatoire
85) Éviter le texte en fond d'écran
86) Utiliser une couleur de fond avec le fond d'écran
87) Tester en vraie grandeur les fonds d'écran
88) Maintenir un haut contraste entre les éléments
89) Les zones de couleur peuvent être larges
90) Réduire la profondeur des couleurs
91) Utiliser les 216 couleurs principales
92) Éviter le dithering
93) Tester les couleurs
94) Utiliser les polices sans sérif pour la lecture en ligne
95) Prêter attention à la typographie
96) Prévoir les variations de taille de police
97) Recourir à des ressources multimédia de manière appropriée
98) Penser à une présentation assistée par ordinateur
99) Considérer des animations simples en GIF animé
100) Éviter les animations répétitives
101) Implémenter un système de désactivation des animations, du son
102) La page courante doit demeurer le centre d'intérêt

8. Accessibilité

103) Rendre accessibles aux visiteurs handicapés tous les éléments du site
104) Rendre accessibles les caractères et le fond d’écran
105) Recourir à des marqueurs hypertextes
106) Minimiser le nombre de liens
107) Rendre les liens accessibles

1.4 Liste de vérification du W3C

1) Fournir des alternatives équivalentes au contenu auditif et visuel.
2) Ne pas s'en remettre uniquement aux couleurs.
3) Utiliser le balisage et les feuilles de style, et cela de façon appropriée.
4) Clarifier l'utilisation du langage naturel
5) Créer des tableaux qui se transforment de façon élégante.
6) S'assurer que les pages qui contiennent de nouvelles technologies se transforment de façon élégante.
7) Assurer à l'utilisateur le contrôle des changements du contenu lorsque ce dernier varie dans le temps.
8) Assurer un accès direct aux interfaces utilisateur intégrées.
9) Conception respectant l'indépendance par rapport au périphérique.
10) Utilisation de solutions intermédiaires.
11) Utilisation des technologies et directives du W3C.
12) Fourniture d'informations de contexte et d'orientation.
13) Fourniture de mécanismes de navigation clairs.
14) S'assurer que les documents sont clairs et simples.

1.5 Liste de vérification du MIT

1. Navigation

1) Current location within the site is shown clearly.
2) Link to the site's main page is clearly identified.
3) Major/important parts of the site are directly accessible from the main page.
4) Site map is provided for a large, complex site.
5) Easy to use Search function is provided, as needed.

2. Functionality

6) Site accommodates novice to expert users.
7) Functions are clearly labeled.
8) Essential functions are available without leaving the site.
9) Plug-ins are used only if they add value.

3. User control

10) Site reflects user's workflow.
11) User can cancel any operation.
12) Clear exit point is provided on every page.
13) Per-page size is less than 50K, to accommodate slow connections.
14) All appropriate browsers are supported.

4. Langage and Content

15) Important information and tasks are given prominence.
16) Information of low relevance or rarely used information is not included.
17) "Related information or tasks are grouped:

a. on the same page or menu
b. in the same area within a page."

18) Language is simple, without jargon.
19) Paragraphs are brief.
20) Links are concise, expressive, and visible--not buried in text.
21) Terms are defined.

5. Online help and user guides

22) Site is designed to require minimal help and instructions.
23) Help and instructions, if needed, are easily accessible.

6. System and user feedback

24) It is always clear what is happening on the site -- visual hints, etc…
25) Users can receive email feedback if necessary.
26) Users can give feedback via email or a feedback form.
27) Confirmation screen is provided for form submittal.
28) All system feedback is timely.
29) Users are informed if a plug-in or browser version is required.
30) Each page includes a "last updated" date.

7. Web accessibility

31) Cascading Style Sheets are used for layout and style where possible.
32) The attribute ALT= is used for images, animations, and other objects.
33) Site uses client-side map and text for hotspots.
34) Site provides captioning and transcripts of audio and descriptions of video.
35) Web versions of PDF documents are provided.
36) Link labels make sense when read out of context; site avoids such link names as "click here".
37) Accomplished with headings, lists, and consistent structure.
38) Summaries are provided for graphs and charts, or the LONGDESC attribute is used.
39) Alternative content is provided for scripts, applets, and plug-ins in case these active features are inaccessible or unsupported.
40) For frames pages, site includes the NOFRAMES option and meaningful titles.
41) Line-by-line reading of tables is sensible, and summaries are included where possible.
42) Site has been validated using the W3C's HTML Validation Serviceand the Bobby Accessibility Checker.
43) Site has been tested on a variety of platforms (UNIX, Windows, Mac) and browsers (Netscape 3, 4, 6, 7; IE 5, 6; lynx).

8. Consistency

44) The same word or phrase is used consistently to describe an item.
45) Link reflects the title of the page to which it refers.

9. Error prevention and correction

46) Users can rely on recognition, not memory, for successful use of the site.
47) Site tolerates a reasonable variety of user actions.
48) Site provides concise instructions for user actions, including entry format.
49) Error messages are visible, not hidden.

50) Error messages are in plain language.
51) Error messages describe actions to remedy a problem.
52) Error messages provide a clear exit point.
53) Error messages provide contact details for assistance.

10. Architectural and Visual Clarity

54) Site is organized from the user's perspective.
55) Site is easily scannable for organization and meaning.
56) Site design and layout is straightforward and concise.
57) Site design and layout are redundant only when required for user productivity.
58) White space is sufficient; pages are not too dense.
59) Unnecessary animation is avoided.
60) Colors used for visited and unvisited links are easily seen and understood.
61) Bold and italic text is used sparingly.

2 Application

2.1 Liste de vérification du groupe Nielsen Norman

1. Anticipation

1) Applications should attempt to anticipate the user's wants and needs. Do not expect users to search for or gather information or evoke necessary tools. Bring to the user all the information and tools needed for each step of the process.

2. Autonomy

2) The computer, the interface, and the task environment all "belong" to the user, but user-autonomy doesn't mean we abandon rules.
3) Use status mechanisms to keep users aware and informed
4) Keep status information up to date and within easy view

3. Color blindness

5) Any time you use color to convey information in the interface, you should also use clear, secondary cues to convey the information to those who won't be experiencing any color coding today.

4. Consistency

6) Levels of consistency: The importance of maintaining strict consistency varies.
7) Inconsistency: It is just important to be visually inconsistent when things must act differently as it is to be visually consistent when things act the same.
8) The most important consistency is consistency with user expectations.

5. Defaults

9) Defaults should be easy to "blow away:" Fields containing defaults should come up selected, so users can replace the default contents with new material quickly and easily.
10) Defaults should be "intelligent" and responsive
11) Do not use the word "default" in an application or service. Replace with "Standard," "Use Customary Settings," "Restore Initial Settings," or some other more specific terms describing what will actually happen

6. Efficiency of the user

12) Increase user's productivity, not only the machine productivity.
13) Keep the user occupied
14) To maximize the efficiency of a business or other organization you must maximize everyone's efficiency, not just the efficiency of a single group.
15) The great efficiency breakthroughs in software are to be found in the fundamental architecture of the system, not in the surface design of the interface.
16) Write help messages tightly and make them responsive to the problem: good writing pays off big in comprehension and efficiency.
17) Menu and button labels should have the key word(s) first.

7. Explorable interface

18) Give users well-marked roads and landmarks, then let them shift into four-wheel drive

19) Sometimes, however, you have to provide deep ruts.
20) Offer users stable perceptual cues for a sense of "home."
21) Make Actions reversible
22) Always allow "Undo."
23) Always allow a way out.
24) However, make it easier to stay in.

8. Fitts' law

25) The time to acquire a target is a function of the distance to and size of the target.

9. Human interface objects

26) Human-interface objects can be seen, heard, touched, or otherwise perceived.
27) Human interface objects that can be seen are quite familiar in graphic user interfaces. Objects that play to another sense such as hearing or touch are less familiar. Good work has been done in developing auditory icons (Gaver).
28) Human-interface objects have a standard way of interacting.
29) Human-interface objects have standard resulting behaviors.
30) Human-interface objects should be understandable, self-consistent, and stable.

10. Latency reduction

31) Wherever possible, use multi-threading to push latency into the background.
32) Reduce the user's experience of latency.
33) Make it faster. Eliminate any element of the application that is not helping. Be ruthless.

11. Learnability

34) Limit the Trade-Offs. Usability and learnability are not mutually exclusive. First, decide which is the most important; then attack both with vigor. Ease of learning automatically coming at the expense of ease of use is a myth.

12. Use of metaphors

35) Choose metaphors well, metaphors that will enable users to instantly grasp the finest details of the conceptual model.
36) Bring metaphors alive by appealing to people's perceptions–sight, sound, touch, and kinesthesia–as well as triggering their memories.

13. Protect users' work

37) Ensure that users never lose their work as a result of error on their part, the vagaries of Internet transmission, or any other reason other than the completely unavoidable, such as sudden loss of power to the client computer.

14. Readability

38) Text that must be read should have high contrast. Favor black text on white or pale yellow backgrounds. Avoid gray backgrounds.
39) Use font sizes that are large enough to be readable on standard monitors. Favor particularly large characters for the actual data you intend to display, as opposed to labels and instructions.
40) Pay particular attention to the needs of older people. Presbyopia, the condition of hardened,

less flexible lenses, coupled with reduced light transmission into the eye, affects most people over age 45. Do not trust your young eyes to make size and contrast decisions.

15. Track State

41) Because many of our browser-based products exist in a stateless environment, we have the responsibility to track state as needed.
42) State information should be held in a cookie on the client machine during a session with a transaction service, then stored on the server when they log off. Users should be able to log off at work, go home, and take up exactly where they left off.

16. Visible navigation

42.1Avoid invisible navigation

2.2 Liste de vérification de Smith et Mosier (data entry)

1.0 General

1) Data Entered Only Once
2) Entry via Primary Display
3) Feedback During Data Entry
4) Fast Response
5) Single Method for Entering Data
6) Defined Display Areas for Data Entry
7) Consistent Method for Data Change
8) User-Paced Data Entry
9) Explicit ENTER Action
10) ENTER Key Labeling
11) Explicit CANCEL Action
12) Feedback for Completion of Data Entry
13) Feedback for Repetitive Data Entries
14) Feedback when Changing Data
15) Keeping Data Items Short
16) Partitioning Long Data Items
17) Optional Abbreviation
18) Distinctive Abbreviation
19) Simple Abbreviation Rule
20) Minimal Exceptions to Abbreviation Rule
21) Minimal Deviation from Abbreviation Rule
22) Fixed Abbreviation Length
23) Clarifying Unrecognized Abbreviations
24) Prompting Data Entry
25) Character Entry via Single Keystroke
26) Minimal Shift Keying
27) Upper and Lower Case Equivalent
28) Decimal Point Optional
29) Leading Zeros Optional

30) Single and Multiple Blanks Equivalent
31) Aids for Entering Hierarchic Data
32) Speech Input
33) Limited Vocabulary for Speech Input
34) Phonetically Distinct Vocabulary for Speech Input
35) Easy Error Correction for Speech Input
36) Alternative Entries for Speech Input
37) PAUSE and CONTINUE Options for Speech Input

1.1 Position designation

38) Distinctive Cursor
39) Nonobscuring Cursor
40) Precise Pointing
41) Explicit Activation
42) Fast Acknowledgement of Entry
43) Stable Cursor
44) Responsive Cursor Control
45) Consistent Incremental Positioning
46) Variable Step Size
47) Proportional Spacing
48) Continuous Cursor Positioning
49) Direct Pointing
50) Large Pointing Area for Option Selection
51) Cursor Control at Keyboard
52) Compatible Control of Cursor Movement
53) Minimal Use of Multiple Cursors
54) Distinctive Multiple Cursors
55) Distinctive Control of Multiple Cursors
56) Compatible Control of Multiple Cursors
57) Consistent HOME Position
58) Consistent Cursor Placement
59) Easy Cursor Movement to Data Fields
60) Display Format Protection
61) Data Entry Independent of Cursor Placement

1.2 Direction designation

62) Analog Entry of Estimated Direction
63) Keyed Entry of Quantified Direction

1.3 Text

64) Adequate Display Capacity
65) Editing Capabilities During Text Entry
66) Free Cursor Movement
67) Control Entries Distinct from Text
68) Natural Units of Text
69) Control Entry Based on Units of Text
70) Highlighting Specified Text

71) Cursor Movement by Units of Text
72) String Search
73) Upper and Lower Case Equivalent in Search
74) Specifying Case in Search
75) Global Search and Replace
76) Case in Global Search and Replace
77) Automatic Pagination Aids
78) User Control of Pagination
79) Controlling Integrity of Text Units
80) Automatic Line Break
81) Consistent Word Spacing
82) Hyphenation by Users
83) Format Control by User
84) Establishing Predefined Formats
85) Storing User-Defined Formats
86) Moving Text
87) Storing Frequently Used Text
88) Necessary Data Displayed
89) Text Distinct from Annotation
90) Text Displayed as Printed
91) Flexible Printing Options
92) Information on Printing Status
93) Auditory Signals for Alerting Users
94) Protecting Text During Page Overruns
95) Confirming Actions in DELETE Mode
96) Reversible Actions
97) User Confirmation of Editing Changes

1.4 Data Forms

98) Combined Entry of Related Data
99) Flexible Interrupt
100) Minimal Use of Delimiters
101) Standard Delimiter Character
102) Data Field Labels
103) Consistent Labeling
104) Protected Labels
105) Labels Close to Data Fields
106) Standard Symbol for Prompting Entry
107) Marking Field Boundaries
108) Prompting Field Length
109) Marking Required and Optional Data Fields
110) Field Markers Not Entered with Data
111) Automatic Justification of Variable-Length Entries
112) Explicit Tabbing to Data Fields
113) Distinctive Label Format
114) Consistent Label Format

115) Label Punctuation as Entry Cue
116) Informative Labels
117) Data Format Cueing in Labels
118) Labeling Units of Measurement
119) Familiar Units of Measurement
120) Alternative Units of Measurement
121) Form Compatible for Data Entry and Display
122) Form Compatible with Source Documents
123) Minimal Cursor Positioning
124) Data Items in Logical Order
125) Automatic Cursor Placement

1.5 Tables

126) Tables for Related Data Sets
127) Distinctive Labels
128) Informative Labels
129) Tabbing within Rows
130) Tabbing within Columns
131) Automatic Justification of Entries
132) Justification of Numeric Entries
133) Maintaining Significant Zeros
134) Aiding Entry of Duplicative Data
135) Row Scanning Cues

1.6 Graphics

136) Pointing
137) Distinctive Cursor
138) Easy Cursor Positioning
139) Confirming Cursor Position
140) Zooming for Precise Positioning
141) Selecting Graphic Elements
142) Highlighting Selected Elements
143) Changing Position (Translation)
144) Deleting Elements
145) Selecting from Displayed Attributes
146) Selecting Colors
147) Displaying Current Attributes
148) Changing Attributes
149) Consistent Method for Attribute Selection
150) Easy Storage and Retrieval
151) Naming Displays and Elements
152) Automatic Data Registration
153) Aids for Entering Hierarchic Data
154) Automatic Data Validation

1.6.1 Graphics – plotting data

155) Automated Data Plotting

156) Plotting Stored Data
157) Predefined Graphic Formats
158) Aids for Graph Construction
159) Aids for Scaling
160) Computer Derivation of Graphic Data

1.6.2 Graphics - drawing

161) Drawing Lines
162) Rubberbanding
163) Aiding Line Connection
164) Grid Reference for Alignment
165) Changing Grid Intervals
166) Constraint for Vertical and Horizontal Lines
167) Specifying Line Relations
168) Drawing Figures
169) Alternative Methods for Drawing Figures
170) Changing Size
171) Enlargement for Symbol Drawing
172) Copying Elements
173) Rotating Elements
174) Reflection of Elements
175) Grouping Elements
176) Merging Elements
177) Filling Enclosed Areas
178) Automatic Figure Completion
179) Stored Models

1.7 Data validation

180) Automatic Data Validation
181) Accepting Correct Entries
182) Non-Disruptive Error Messages
183) Deferral of Required Data Entry
184) Reminder of Deferred Entry
185) Timely Validation of Sequential Transactions
186) Optional Item-by-Item Validation

1.8 Other data processing

187) Default Values
188) Defaults for Sequential Entries
189) User Definition of Default Values
190) Display of Default Values
191) Easy Confirmation to Enter Default Values
192) Temporary Replacement of Default Values
193) Automatic Generation of Routine Data
194) Automatic Computation of Derived Data
195) User Review of Prior Entries
196) Automatic Entry of Redundant Data

197) Automatic Cross-File Updating
198) Aids for Entering Hierarchic Data

1.9 Design change

199) Flexible Design for Data Entry

2.3 Liste de vérification de Smith et Mosier (data display)

1.0 General

1) Necessary Data Displayed
2) Only Necessary Data Displayed
3) Data Displayed in Usable Form
4) Data Display Consistent with User Conventions
5) Establishing Display Standards
6) Consistent Display Format
7) Display Consistent with Entry Requirements
8) User Control of Data Display
9) User Changes to Displayed Data
10) Protection of Displayed Data
11) Context for Displayed Data
12) Familiar Wording
13) Consistent Wording
14) Consistent Wording Across Displays
15) Consistent Grammatical Structure
16) Minimal Use of Abbreviation
17) Common Abbreviations
18) Simple Abbreviation Rule
19) Distinctive Abbreviations
20) Minimal Punctuation of Abbreviations
21) Dictionary of Abbreviations

1.1 Text

22) Conventional Text Display
23) Printing Lengthy Text Displays
24) Consistent Text Format
25) Adequate Display Capacity
26) Text Displayed in Wide Columns
27) Conventional Use of Mixed Case
28) Separation of Paragraphs
29) Consistent Word Spacing
30) Minimal Hyphenation
31) Conventional Punctuation
32) Clarity of Wording
33) Sentences Begin with Main Topic
34) Simple Sentence Structure

35) Concise Wording
36) Distinct Wording
37) Affirmative Sentences
38) Active Voice
39) Temporal Sequence
40) Lists for Related Items
41) Single-Column List Format
42) Marking Multiline Items in a List
43) Arabic Numerals for Numbered Items
44) Logical List Ordering
45) Vertical Ordering in Multiple Columns
46) Hierarchic Structure for Long Lists
47) Abbreviations Defined in Text
48) Highlighting Text
49) Combining Text with Other Data
50) Placing Figures Near Their Citations

1.2 Data forms

51) Forms for Related Data
52) Visually Distinctive Data Fields
53) Data Field Labeling
54) Descriptive Wording of Labels
55) Consistent Wording of Labels
56) Distinctive Wording of Labels
57) Consistent Label Location
58) Distinctive Label Format
59) Labels Close to Data Fields
60) Labeling Units of Measurement
61) Consistent Format Across Displays
62) Form Compatible for Data Entry and Display
63) Consistent Format Within Data Fields
64) Partitioning Long Data Items
65) Distinguishing Blanks from Nulls

1.3 Tables

66) Tables for Data Comparison
67) Logical Organization
68) Table Access by Row and Column
69) Tables Referenced by First Column
70) Items Paired for Direct Comparison
71) Row and Column Labels
72) Consistent Label Format
73) Distinctive Labeling
74) Numbered Items Start with 1
75) Repeated Elements in Hierarchic Numbering
76) Labeling Units of Measurement

77) Consistent Column Spacing
78) Column Scanning Cues
79) Row Scanning Cues
80) Justification of Alphabetic Data
81) Justification of Numeric Data
82) Maintaining Significant Zeros

1.4 Graphics

83) Graphic Displays
84) Data Comparison
85) Monitoring Data Change
86) Consistency
87) Only Necessary Information Displayed
88) Highlighting Critical Data
89) Reference Index or Baseline
90) Text Annotation
91) Data Annotation
92) Consistent Annotation Format
93) Normal Orientation for Labels
94) Standard Symbols
95) Pictorial Symbols
96) Simple Texture Codes
97) Zooming for Display Expansion
98) Show Changing Scale
99) Show Overview Position of Visible Section
100) Animation for Dynamic Display
101) Highlighting by Animation
102) Printing Graphic Displays

1.4.1 Graphics-scaling

103) Scaling Conventions
104) Consistent Scaling
105) Labeling Axes
106) Linear Scaling
107) Scaling in Standard Intervals
108) Numeric Scales Start at Zero
109) Restricted Use of Broken Axes
110) Duplicate Axes
111) Single Scale Only
112) Scaling Against a Reference Index
113) Aids for Scale Interpolation
114) Unobtrusive Grids
115) Restricted Use of Three-Dimensional Scaling

1.4.2 Graphics- Scatterplots

116) Scatterplots
117) Highlighting

118) Grouping Scatterplots to Show Multiple Relations
119) Interactive Analysis of Grouped Scatterplots

1.4.3 Graphics- curves and line graphs

120) Curves and Line Graphs
121) Comparing Curves
122) Labeling Curves
123) Compatible Ordering in Legends
124) Highlighting
125) Line Coding to Distinguish Curves
126) Consistent Line Codes
127) Broken Lines for Projected Curves
128) Reference Index
129) Repeating Display of Cyclic Data
130) Direct Display of Differences
131) Surface Charts
132) Ordering Data in Surface Charts
133) Labeling Surface Charts
134) Cumulative Curves

1.4.4 Graphics-Bar graphics

135) Bar Graphs
136) Histograms (Step Charts)
137) Consistent Orientation of Bars
138) Bar Spacing
139) Reference Index
140) Highlighting
141) Paired or Overlapped Bars
142) Labeling Paired Bars
143) Stacked or Segmented Bars
144) Ordering Data in Stacked Bars
145) Restricted Use of Icons

1.4.5 Graphics- Pie charts

146) Restricted Use of Pie Charts
147) Labeling Pie Charts
148) Numeric Labels
149) Highlighting

1.4.6 Graphics-pictures and diagrams

150) Pictures
151) Diagrams
152) Linking Sectional Diagrams
153) Highlighting
154) Rotation
155) Aids for Pictorial Analysis

1.4.7 Graphics- Flowcharts

156) Flowcharts
157) Flowcharts to Aid Problem Solving
158) Logical Ordering of Steps
159) Ordering to Minimize Path Length
160) Conventional Path Orientation
161) Consistent Coding of Elements
162) Conventional Use of Arrows
163) Highlighting
164) Single Decision at Each Step
165) Logical Ordering of Options
166) Consistent Ordering of Options
167) Consistent Wording

1.4.8 Graphics-Maps and situation displays

168) Maps
169) Consistent Orientation
170) Consistent Projection
171) Labeling Maps
172) Consistent Positioning of Labels
173) Area Coding
174) Tonal Codes
175) Ordered Coding
176) Highlighting
177) Panning for Flexible Display Framing
178) Show Overview Position of Visible Section
179) Aiding Distance Judgments
180) Aids for Analyzing Maps
181) Mapping Nongeographic Data
182) Situation Displays
183) Indicating Data Change
184) Selectable Data Categories
185) Stable Reference for Changing Data

1.5 Format

186) Consistent Format
187) Distinctive Display Elements
188) Spacing to Structure Displays
189) Paging Crowded Displays
190) Related Data on Same Page
191) Page Labeling
192) Integrated Display
193) User-Defined Data Windows
194) Adequate Window Size
195) Display Title at Top
196) Command Entry, Prompts, Messages at Bottom

197) Logical Data Organization
198) Grouping for Data Comparison
199) Data Grouped by Sequence of Use
200) Data Grouped by Function
201) Data Grouped by Importance
202) Data Grouped by Frequency
203) Data Grouped Alphabetically or Chronologically

1.6 Coding

204) Highlighting Critical Data
205) Removing Highlighting
206) Coding by Data Category
207) Meaningful Codes
208) Familiar Coding Conventions
209) Definition of Display Codes
210) Consistent Coding Across Displays
211) Alphanumeric Coding
212) Consistent Case in Alphabetic Coding
213) Combining Letters and Numbers
214) Short Codes
215) Special Symbols
216) Consistent Use of Special Symbols
217) Markers Close to Words Marked
218) Shape Coding
219) Establishing Standards for Shape Coding
220) Line Coding
221) Underlining for Emphasis
222) Coding by Line Length
223) Coding by Line Direction
224) Limited Use of Size Coding
225) Adequate Differences in Size
226) Limited Use of Brightness Coding
227) Brightness Inversion
228) Color Coding for Relative Values
229) Color Coding for Data Categories
230) Easily Discriminable Colors
231) Conservative Use of Color
232) Adding Color to Formatted Displays
233) Redundant Color Coding
234) Unique Assignment of Color Codes
235) Conventional Assignment of Color Codes
236) Brightness and Saturation to Draw Attention
237) Saturated Blue for Background Color
238) Blink Coding
239) Blinking Marker Symbols
240) Optimal Blink Rate

241) Coding with Texture, Focus, Motion
242) Auditory Coding
243) Distinctive Auditory Coding
244) Voice Coding
245) Coding Synthesized Voice Alarms

1.7 Display control

246) Flexible Display Control by User

1.7.1 Display control- Selection

247) User Selection of Data for Display
248) Display Identification Labels
249) Meaningful Display Labels
250) Consistent Format for Display Labels
251) Selectable Data Categories
252) Fast Response to Display Request
253) Signaling Completion of Display Output
254) Regenerating Changed Data
255) Initial Erasure to Replace Changed Data
256) Nondestructive Overlay
257) Printing Displays Locally
258) Integrated Display
259) Easy Paging
260) Continuous Numbering in Multipage Lists
261) Labels for Multipage Tables
262) Annotating Display of Continued Data
263) Numbering Display Pages
264) Consistent Orientation - Panning vs. Scrolling
265) Panning with Free Cursor Movement
266) Functional Labeling for Display Framing
267) Labeling Panning Functions
268) Labeling Scrolling Functions
269) Panning for Flexible Display Framing
270) Zooming for Display Expansion
271) Show Changing Scale
272) Show Overview Position of Visible Section
273) Framing Integrally for All Data
274) Return to Normal Display Coverage

1.7.2 Display control- Framing

275) Automatic Display Update
276) Highlighting Changed Data
277) Readability of Changing Data
278) Visual Integration of Changing Graphics
279) Display Freeze
280) Labeling Display Freeze
281) Signaling Changes to Frozen Data

282) Resuming Update After Display Freeze
283) Prediction Display

1.7.3 Display control- Update

284) Temporary Suppression of Displayed Data
285) Labeling Display Suppression
286) Signaling Changes to Suppressed Data
287) Resuming Display of Suppressed Data

1.7.4 Display control – Window overlays

288) Temporary Window Overlays
289) Predefined Windows
290) User-Specified Windows
291) Consistent Window Control
292) Easy Suppression of Window Overlays
293) Labeling Windows
294) Indicate Active Window
295) Easy Shifting Among Windows
296) Consistent Control Within Windows
297) Nondestructive Overlay

1.8 Design Change

298) Flexible Design for Data Display

2.4 Liste de vérification du CNRS

1. L'apparence des fenêtres

1) I LA DENSITÉ DE L'AFFICHAGE
2) II LA DISPOSITION DES ÉLÉMENTS DANS LES FENÊTRES
3) III LA PRÉSENTATION DES LISTES ET DES TABLEAUX
4) IV LES ÉLÉMENTS TEXTUELS
5) V.1 LA TYPOGRAPHIE
6) V.2 LA COULEUR
7) V.3 LES ICONES
8) VI L'UTILISATION DES ONGLETS

2. Les principles de navigation

9) I.1 L'UTILISATION DE LA TOUCHE TABULATION (DÉPLACEMENT SÉQUENTIEL)
10) I.2 POSITIONNEMENT DIRECT DU CURSEUR
11) I.3 LES BARRES DE DÉFILEMENT HORIZONTAL ET VERTICAL
12) II.1 LA CONVERSATION LIBRE ET CONVERSATION GUIDÉE - Conversation libre
13) II.1 LA CONVERSATION LIBRE ET CONVERSATION GUIDÉE - Conversation guidée
14) II.1 LA CONVERSATION LIBRE ET CONVERSATION GUIDÉE - Quel type de

conversation adopter ?
15) II.2 LE NOMBRE DE FENÊTRES

3. Les principes de saisie

16) I.1 LES CHAMPS DE SAISIE - Aspect des champs de saisie
17) I.1 LES CHAMPS DE SAISIE - Initialisation des champs de saisie avec une valeur par défaut
18) I.1 LES CHAMPS DE SAISIE - Caractères à saisir
19) I.1 LES CHAMPS DE SAISIE - Saisie obligatoire et saisie optionnelle
20) I.1 LES CHAMPS DE SAISIE - Validation de la saisie
21) I.1 LES CHAMPS DE SAISIE - Retour système
22) I.2 LA SAISIE DANS UN TABLEAU
23) I.3 LA SÉLECTION D'UN NOMBRE LIMITÉ D'OPTIONS - Les options exclusives : boutons d'option
24) I.3 LA SÉLECTION D'UN NOMBRE LIMITÉ D'OPTIONS - Les options non exclusives : cases à cocher
25) I.3 LA SÉLECTION D'UN NOMBRE LIMITÉ D'OPTIONS - La liste limitée d'options liées : sélecteur rotatif
26) II LES DONNÉES SAISIES PAR SÉLECTION DANS UNE LISTE

4. Choix des actions de l’application par menu ou par bouton de commande

27) I.1 LE LIBELLÉ DES ACTIONS
28) I.2 LES RACCOURCIS CLAVIER
29) I.3 LES ACTIONS INTERDITES OU INDISPONIBLES
30) I.4 LE RETOUR-ARRIÈRE
31) I.5 LES ACTIONS PAR DÉFAUT
32) I.6 LES ACCÈS AUX ACTIONS
33) II.1 LA BARRE DE MENUS
34) II.2 LES OPTIONS DE MENU - Recommandations générales
35) II.2 LES OPTIONS DE MENU - Les menus déroulants
36) II.2 LES OPTIONS DE MENU - Les menus en cascade
37) II.2 LES OPTIONS DE MENU - Les menus contextuels (ou menus d'incrustation)
38) II.2 LES OPTIONS DE MENU - La barre d'outils
39) III.1 LA PRÉSENTATION DES BOUTONS DE COMMANDE
40) III.2 LE COMPORTEMENT DES BOUTONS DE COMMANDE
41) III.3 LE CHOIX DES BOUTONS DE COMMANDE

5. Les actions du système

42) I.1 RECOMMANDATIONS GÉNÉRALES
43) I.2 LES MESSAGES D'INFORMATION OU DE SIMPLE MISE EN GARDE
44) I.3 LES MESSAGES D'AVERTISSEMENT
45) I.4 LES MESSAGES DE CONFIRMATION
46) I.5 LES MESSAGES D'ARRÊT IMMEDIAT/BLOQUANTS
47) II LE CURSEUR/POINTEUR
48) III.1 GÉNÉRALITÉS
49) III.2 INDICATEUR DE PROGRESSION
50) IV LE SIGNAL SONORE

51) V LA CONFIDENTIALITÉ DES INFORMATIONS

6. L’aide en ligne

52) I.1 L'AIDE IMPLICITE
53) I.2 L'AIDE EXPLICITE : LES MESSAGES
54) II.1 L'AIDE À LA SAISIE
55) II.2 L'AIDE CONTEXTUELLE RAPIDE
56) II.3 L'AIDE GÉNÉRALE

2.5 Liste de vérification pour Eclipse

1. General UI guidelines

1) Follow and apply good user interface design principles: user in control, directness, consistency, forgiveness, feedback, aesthetics, and simplicity.
2) Follow the platform lead for user interface conventions.
3) Be careful not to mix UI metaphors. It may blur the original concept, and your own application.
4) If you have an interesting idea, work with the Eclipse community to make Eclipse a better platform for all.
5) Use Headline style capitalization for menus, tooltip and all titles, including those used for windows, dialogs, tabs, column headings and push buttons. Capitalize the first and last words, and all nouns, pronouns, adjectives, verbs and adverbs. Do not include ending punctuation.
6) Use Sentence style capitalization for all control labels in a dialog or window, including those for check boxes, radio buttons, group labels, and simple text fields. Capitalize the first letter of the first word, and any proper names such as the word Java.
7) Create localized version of the resources within your plug-in.
8) When an error occurs which requires either an explicit user input or immediate attention from users, communicate the occurrence with a modal dialog.
9) If a programming error occurs in the product, communicate the occurrence with a dialog, and log it.

2. Visual design

10) Re-use the core visual concepts to maintain consistent representation and meaning across Eclipse plug-ins.
11) Use the Eclipse 256 color palette for creating the active or selected state of all icon types.
12) Use the Eclipse 8 color palette for creating the enabled state of perspective, view, toolbar, toolbar wizard, and local toolbar icons.
13) Use the Eclipse 2 color palette for creating the disabled state of toolbar, toolbar wizard, and local toolbar icons.
14) Use the appropriate icon type in the location it is designed for within the user interface.
15) Follow the specific size specifications for each type of icon.
16) Cut the icons with the specific placement shown to ensure alignment in the user interface.
17) Follow the positioning guidelines for the different types of icons for optimal alignment of

these elements relative to one another.
18) Use the Eclipse special blue 183 color palette for creating wizard graphics.
19) Follow the specific size specifications for wizard graphics.
20) Cut the wizard graphics with the specific placement shown to ensure alignment in the wizard banner area.
21) Follow the predefined directory structure and naming convention.
22) Keep the original directory names provided.
23) Minimize duplication of graphics within a plugin by keeping all graphics in one, or few, first level user interface directories.
24) Use the active, enabled, and disabled states provided.
25) Abbreviate file name instead of using the full icon name, e.g. New Interface becomes "newint".
26) Use lower case characters in your file names, e.g. DTD becomes "dtd".
27) Use 10 characters or less in your file names if possible (underscores count as a character).
28) Use a file name suffix that describes its location or function in the tool, e.g. newint_wiz.
29) Use transparent *.gif format for all user interface icons and wizard graphics, unless the context requires a different file format.
30) Keep the original file names provided.

3. Component development

31) Each command must have a label, tool tip, and full color image. The label and tool tip must use Headline style capitalization.
32) The command tooltip should describe the result of the command, not the current state of the command. Use the text same as that for the command label.
33) Adopt the labeling terminology of the workbench for New, Delete and Add commands.
34) A command should only be enabled if it can be completed successfully.
35) Command enablement should be quick. If command enablement cannot be quick, enable the command optimistically and display an appropriate message if the command is invoked, but cannot be completed.

4. Dialogs

36) When a dialog opens, set the initial focus to the first input control in the container. If there are no input controls, the initial focus should be assigned to the default button.
37) Slush Bucket widget (or Twin Box) should flow from left to right with the source objects on the left hand side. It should have the >, <, >>, << control buttons in this order.

5. Wizards

38) Use a wizard for any task consisting of many steps, which must be completed in a specific order.
39) Each wizard must contain a header with a banner graphic and a text area for user feedback. It must also contain Back, Next, Finish, and Cancel buttons in the footer.
40) Start the wizard with a prompt, not an error message.
41) Seed the fields within the wizard using the current workbench state.
42) Validate the wizard data in tab order. Display a prompt when information is absent, and an error when information is invalid.
43) Only enable the Next / Finish buttons if all required information in the dialog is present

and valid.
44) Remove all programming message ID's from wizard text.
45) Use a Browse Button whenever an existing object is referenced in a wizard.
46) If a new file is created, open the file in an editor. If a group of files are created, open the most important, or central file in an editor. Open the readme.html file upon creation of an example project.
47) If a new project is created, prompt users and change the active perspective to suit the project type.
48) If a new object is created, select and reveal the new object in the appropriate view.
49) Create folder objects in a wizard if reasonable defaults can be defined.
50) Use the term "Project name" for the input field label when the item must be a Project; otherwise, use the term "Folder name". Do not qualify the term.

6. Editors

51) Use an editor to edit or browse a file, document, or other primary content.
52) Modifications made in an editor must follow an open-save-close lifecycle model.
53) Only one instance of an editor may exist, for each editor input, within a perspective.
54) It must be possible to open a separate instance of an editor for each different input.
55) The editor should be labeled with the name of the file, document, or input being edited.
56) In multipage editors, use a tab control for page activation.Tab labels should be kept to one word, and two words at most.
57) All of the commands, except for the obvious commands, available in the editor should be added to the window menu bar.
58) Use the standard format for editor contributions in the window menu bar.
59) If an editor has support for Cut, Copy, Paste, or any of the global commands, these commands must be executable from the same commands in the window menu bar and toolbar.
60) Fill the editor toolbar with the most commonly used items in the view menu.
61) Fill the context menu with selection oriented commands.
62) Use the standard format for editor context menus.
63) Fill the context menu with a fixed set of commands for each selection type, and then enable or disable each to reflect the selection state.
64) Register all context menus in the editor with the platform.
65) Implement a Command Filter for each object type in the editor.
66) If the input to an editor is deleted, and the editor contains no changes, the editor should be closed.
67) If the input to an editor is deleted, and the editor contains changes, the editor should give the user a chance to save their changes to another location, and then close.
68) If the resource is dirty, prefix the resource name presented in the editor tab with an asterisk
69) Treat read-only editor input as you would any other input. Enable the Save As if possible. Display "Read-only" in the status bar area.
70) If the data within an editor is too extensive to see on a single screen, and will yield a structured outline, the editor should provide an outline model to the Outline view.
71) Notification about location between an editor and the Outline view should be two-way. A context menu should be available in the Outline view as appropriate.

72) An error or warning image should be added to items with the error or warning respectively. A container should have a red X if it there are errors on the container itself, a gray X if any of its descendents have errors (but not the container itself), and no X if neither the container nor any of its descendents have errors.
73) If appropriate, implement the "Add Task" feature in your editor.
74) If appropriate, implement the "Add Bookmark" feature in your editor.
75) Editors with source lines of text should show the current line and optionally column numbers the status line. It's optional for the editor to show line numbers for each line in the editor itself.
76) Table cell editors should support the single-click activation model, and in edit mode, they should render complex controls upon single-click.
77) Changes made in a table cell editor should be committed when a user clicks off the cell or hits the "Enter" key. Selection should be cancelled when user hits the "Esc" key.First letter navigation should be supported as a cursoring mechanism within a cell.
78) When performing fine-grain error validation in an editor, use red squiggles to underline the invalid content. When users move the mouse over the red squiggles, display the error text in a fly-over pop up box.
79) Use the Task view to show errors found when the Save command is invoked.
80) If modifications to a resource are made outside of the workbench, users should be prompted to either override the changes made outside of the workbench, or back out of the Save operation when the Save command is invoked in the editor.

7. Views

81) Use a view to navigate a hierarchy of information, open an editor, or display the properties of an object.
82) Modifications made within a view must be saved immediately.
83) Only one instance of a view may exist in a perspective.
84) A view must be able to be opened in more than one perspective
85) A view can be opened from the Window > Show View menu
86) The view label in the title bar must be prefixed with the label of the view in the Perspective > Show View menu.
87) If a view contains more than one control, it may be advisable to split it up into two or more views.
88) When a view first opens, derive the view input from the state of the perspective.
89) If a view displays a resource tree, consider using the window input as the root of visible information in the view.
90) Use the view pulldown menu for presentation commands, not selection-oriented commands.
91) Use the standard order of commands for view pulldown menus.
92) Put only the most commonly used commands on the toolbar. Any command on a toolbar must also appear in a menu, either the context menu or the view menu.
93) Fill the context menu with selection oriented actions, not presentation actions.
94) "Use the standard order of commands for view context menus.
95) Fill the context menu with a fixed set of commands for each selection type, and then enable or disable each to reflect the selection state."
96) If an object appears in more than one view, it should have the same context menu in

each.
97) Register all context menus in the view with the platform
98) Implement a Command Filter for each object type in the view.
99) If a view has support for Cut, Copy, Paste, or any of the global commands, these commands must be executable from the same commands in the window menu bar and toolbar.
100) Persist the state of each view between sessions.
101) Navigation views should support "Link with Editor" on the view menu

8. Perspectives

102) Create a new perspective type for long lived tasks, which involve the performance of smaller, non-modal tasks.
103) If you just want to expose a single view, or two, extend an existing perspective type.
104) The size and position of each view in a perspective should be defined in a reasonable manner, such that the user can resize or move a view if they desire it. When defining the initial layout, it is important to consider the overall flow between the views (and editors) in the perspective.
105) If a perspective has just one part, it may be better suited as a view or editor.
106) If it is undesirable to have an editor area in a perspective, hide it. Do not resize the editor area to the point where it is no longer visible
107) Populate the window menu bar with commands and command sets which are appropriate to the task orientation of the perspective, and any larger workflow.
108) A new perspective should be opened only if the user explicitly states a desire to do so. In making this statement, the user agrees to leave their old context, and create a new one.
109) If a new perspective is opened as a side effect of another command, the user should be able to turn this behavior off.
110) If a new perspective is opened, it should be opened within the current window, or in a new window, depending on the user preference.
111) The list of shortcuts added to the New, Open Perspective, and Show View menus should be at most 7 plus / minus 2 items.

9. Windows

112) Use an Action Set to contribute actions to the window menu bar and toolbar.
113) Follow the platform lead when distributing actions within an Action Set.
114) Contribute actions to the window menu bar first, and then to the window toolbar if they will be frequently used.
115) Define each action set with a specific task in mind.
116) An action set should contain the smallest possible semantic chunking of actions. Avoid the temptation to provide only one action set for an entire plug-in.
117) Use an action set to share a set of actions which are useful in two or more views or editors.
118) Let the user control the visible action sets. Don't try to control it for them.
119) Open Object actions must appear in the Navigate pulldown menu of the window.
120) Always use the global status bar to display status related messages.

10. Properties

121) Use the Properties view to edit the properties of an object when quick access is

important, and you will switch quickly from object to object.
122) Use a Properties Dialog to edit the properties of an object which are expensive to calculate.
123) Use a Properties Dialog to edit the properties of an object which contain complex relationships to one another.
124) Properties Dialog should contain the superset of items shown in the Properties view.

11. Widget

125) For Tree and Table widgets that have a checkbox associated with a cell item, changing the current selection should not automatically change the check state of the selected item. However, the current selection should be set to a given item when its check state is changed.

12. Standart components

126) If appropriate, add actions to standard components of Eclipse using the plug-in registry.
127) If you subclass or copy any of the standard components, always carry over the standard components' characteristics.

13. The navigator view

128) Add actions to the Navigator View menu, toolbar, and context menu using the plug-in registry.
129) Use the attributes defined in IResourceActionFilter.java and IProjectActionFilter.java to control the visibility of context menu actions in the Navigator.
130) Use a "Navigate -> Show In Navigator" command in each view, to link resources back to the Navigator.

14. The task view

131) Add markers (tasks, errors and warnings) to the Tasks view using the Marker Manager services from the Core Resources Management plugin.
132) The description text of each marker should be short and concise, so that it will fit in the status line of Eclipse.
133) Add actions to the Tasks view menu, toolbar, and context menu using the plug-in registry.
134) Use the attributes defined in IMarkerActionFilter.java to control the visibility of context menu actions in the Tasks view.
135) Support F1 keyboard command and link it to an infopop that gives a detailed description of the selected item in the Task view.

15. The preference dialog

136) Global options should be exposed within the Preferences Dialog.
137) Expose the preferences for a particular view, editor or window in the view itself, via a menu or tool item.
138) Start out with a single preference page. Then evolve to more if you need to.
139) If you create a preference group, use the root page for frequently used preferences, or those preferences which have wide spread effect. Specialize within the sub pages. The root preference page should not be blank.
140) Attempt to integrate plug-in preferences, wizards, and views into existing categories for

a new plug-in first, before considering the creation of a new category.

16. Flat look design

141) Use Flat Look design for user scenarios that involve extensive property and configuration editing.
142) Have the core sections on the overview page expanded, and provide a "Home" icon on other pages to take users back to the overview page.
143) Use grouping elements corresponding to tabs in the Flat Look content editor for the organization of the tree view in outline view.

17. The taoof resource

144) Expose the resource for resource equivalent model objects using an IContributorResourceAdapter.

18. Accessibility

145) All of the features provided by a tool should be accessible using a mouse or the keyboard.

3 Mobile

3.1 Liste de vérification du W3C

Basic guidelines

1) [THEMATIC_CONSISTENCY] Ensure that content provided by accessing a URI yields a thematically coherent experience when accessed from different devices.
2) [CAPABILITIES] Exploit device capabilities to provide an enhanced user experience.
3) [DEFICIENCIES] Take reasonable steps to work around deficient implementations.
4) [TESTING] Carry out testing on actual devices as well as emulators.
5) [URIS] Keep the URIs of site entry points short.
6) [NAVBAR] Provide only minimal navigation at the top of the page.
7) [BALANCE] Take into account the trade-off between having too many links on a page and asking the user to follow too many links to reach what they are looking for.
8) [NAVIGATION] Provide consistent navigation mechanisms.
9) [ACCESS_KEYS] Assign access keys to links in navigational menus and frequently accessed functionality.
10) [LINK_TARGET_ID] Clearly identify the target of each link.
11) [LINK_TARGET_FORMAT] Note the target file's format unless you know the device supports it.
12) [IMAGE_MAPS] Do not use image maps unless you know the device supports them effectively.
13) [POP_UPS] Do not cause pop-ups or other windows to appear and do not change the current window without informing the user.
14) [AUTO_REFRESH] Do not create periodically auto-refreshing pages, unless you have informed the user and provided a means of stopping it.
15) [REDIRECTION] Do not use markup to redirect pages automatically. Instead, configure the server to perform redirects by means of HTTP 3xx codes.
16) [EXTERNAL_RESOURCES] Keep the number of externally linked resources to a minimum.
17) [SUITABLE] Ensure that content is suitable for use in a mobile context.
18) [CLARITY] Use clear and simple language.
19) [LIMITED] Limit content to what the user has requested.
20) [PAGE_SIZE_USABLE] Divide pages into usable but limited size portions.
21) [PAGE_SIZE_LIMIT] Ensure that the overall size of page is appropriate to the memory limitations of the device.
22) [SCROLLING] Limit scrolling to one direction, unless secondary scrolling cannot be avoided.
23) [CENTRAL_MEANING] Ensure that material that is central to the meaning of the page precedes material that is not.
24) [GRAPHICS_FOR_SPACING] Do not use graphics for spacing.
25) [LARGE_GRAPHICS] Do not use images that cannot be rendered by the device. Avoid large or high resolution images except where critical information would otherwise be lost.
26) [USE_OF_COLOR] Ensure that information conveyed with color is also available without color.
27) [COLOR_CONTRAST] Ensure that foreground and background color combinations

provide sufficient contrast.

28) [BACKGROUND_IMAGE_READABILITY] When using background images make sure that content remains readable on the device.
29) [PAGE_TITLE] Provide a short but descriptive page title.
30) [NO_FRAMES] Do not use frames.
31) [STRUCTURE] Use features of the markup language to indicate logical document structure.
32) [TABLES_SUPPORT] Do not use tables unless the device is known to support them.
33) [TABLES_NESTED] Do not use nested tables.
34) [TABLES_LAYOUT] Do not use tables for layout.
35) [TABLES_ALTERNATIVES] Where possible, use an alternative to tabular presentation.
36) [NON-TEXT_ALTERNATIVES] Provide a text equivalent for every non-text element.
37) [OBJECTS_OR_SCRIPT] Do not rely on embedded objects or script.
38) [IMAGES_SPECIFY_SIZE] Specify the size of images in markup, if they have an intrinsic size.
39) [IMAGES_RESIZING] Resize images at the server, if they have an intrinsic size.
40) [VALID_MARKUP] Create documents that validate to published formal grammars.
41) [MEASURES] Do not use pixel measures and do not use absolute units in markup language attribute values and style sheet property values.
42) [STYLE_SHEETS_USE] Use style sheets to control layout and presentation, unless the device is known not to support them.
43) [STYLE_SHEETS_SUPPORT] Organize documents so that if necessary they may be read without style sheets.
44) [STYLE_SHEETS_SIZE] Keep style sheets small.
45) [MINIMIZE] Use terse, efficient markup.
46) [CONTENT_FORMAT_SUPPORT] Send content in a format that is known to be supported by the device.
47) [CONTENT_FORMAT_PREFERRED] Where possible, send content in a preferred format.
48) [CHARACTER_ENCODING_SUPPORT] Ensure that content is encoded using a character encoding that is known to be supported by the device.
49) [CHARACTER_ENCODING_USE] Indicate in the response the character encoding being used.
50) [ERROR_MESSAGES] Provide informative error messages and a means of navigating away from an error message back to useful information.
51) [COOKIES] Do not rely on cookies being available.
52) [CACHING] Provide caching information in HTTP responses.
53) [FONTS] Do not rely on support of font related styling.
54) [MINIMIZE_KEYSTROKES] Keep the number of keystrokes to a minimum.
55) [AVOID_FREE_TEXT] Avoid free text entry where possible.
56) [PROVIDE_DEFAULTS] Provide pre-selected default values where possible.
57) [DEFAULT_INPUT_MODE] Specify a default text entry mode, language and/or input format, if the device is known to support it.
58) [TAB_ORDER] Create a logical order through links, form controls and objects.
59) [CONTROL_LABELLING] Label all form controls appropriately and explicitly associate labels with form controls.

60) [CONTROL_POSITION] Position labels so they lay out properly in relation to the form controls they refer to.