A quality-centered approach for web application engineering

Tuan Anh Do

le cnam

École Doctorale d'Informatique, Télécommunications et Electronique

Centre d'étude et de recherche en informatique et communications

THÈSE DE DOCTORAT

présentée par : **Tuan Anh DO**

soutenue le : **18 décembre 2018**

pour obtenir le grade de : **Docteur du Conservatoire National des Arts et Métiers**

Discipline / Spécialité : **Informatique**

A QUALITY-CENTERED APPROACH FOR WEB APPLICATION ENGINEERING

THÈSE DIRIGÉE PAR

Mme. COMYN-WATTIAU Isabelle	*Professeur, CNAM*
Mme. SISAID-CHERFI Samira	*Professeur, CNAM*

RAPPORTEURS

Mme. MARCAL DE OLIVEIRA Kathia	*MCF HDR, Université de Valenciennes*
Mme. KEDAD Zoubida	*MCF HDR, Université de Versailles*

EXAMINATEURS

M. AKOKA Jacky	*Président du jury, Professeur, CNAM & IMT-BS*
M. MAURICE-DEMOURIOUX Michel	*Directeur d'études, IMT-BS*

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Isabelle Comyn-Wattiau. It is through her own perseverance that I could finish this thesis. I would like to thank her for her patience, motivation, and help. I would also like to thank Prof. Samira Si-said Cherfi who co-advised my PhD for her support, and her immense knowledge. Their guidance helped me in all the time of research and writing of this thesis.

Besides my advisors, I would like to thank Dr. Káthia Marçal de Oliveira, Dr. Zoubida Kedad, who accepted to serve as reviewers of my thesis and provided me with useful advice.

A special warm thank to Prof. Jacky Akoka who was the head of the ISID research team when I started my PhD. He was always helpful. Moreover he contributed to my work by rereading the whole manuscript. Thank you for his encouragement and insightful comments.

I also want to thank Dr. Michel Maurice-Demourioux, who made me the honor to participate in my PhD jury.

My sincere thanks also go to all members of ISID team of CEDRIC laboratory, CNAM for their helps in the time I worked here, especially Dr. Nadira Lammari, who heads ISID team. She always had words of encouragement for me.

I am also grateful to the Vietnam Ministry of Education and Training for their financial support; the Department of Geography Information System, Institute of Information Technology, Vietnam Academic Science and Technology for their allowance to realize my thesis.

In addition, I would like to thank my fellow doctoral students for their feedback, cooperation, and friendship. I also thank my friends for their moral support.

Last but not least, I would like to thank my family: my parents, my wife and my children for supporting me spiritually throughout this thesis and my life in general.

Résumé

Avec le développement d'Internet, les applications web sont de plus en plus nombreuses et importantes. De nombreux standards de qualité, des modèles de qualité, des méthodes d'ingénierie web ont été proposés, mais la qualité des applications web n'est pas toujours au niveau souhaité.

Dans cette thèse, nous proposons une approche pour tenter de résoudre ce problème. Elle comporte trois phases itératives: définition, mesure et amélioration de la qualité des applications web. Dans la première phase, nous proposons une définition plus complète et plus riche de la qualité des applications web. La qualité d'une application web n'est pas uniquement perçue comme la qualité d'un logiciel, mais également comme la qualité des informations qu'elle met à disposition. Enfin, elle comprend des éléments de qualité spécifiques à ces applications qui contribuent notamment au succès et à la réputation de l'organisation. Dans la seconde phase, nous construisons une taxonomie de métriques pour mesurer la qualité des applications web. Cette taxonomie est fondée sur le standard ISO25010. Dans la troisième phase, nous avons collecté et adapté les « guidelines » de la littérature pour les mettre à la disposition des concepteurs-développeurs d'applications web. A cet effet, nous avons proposé un méta-modèle de guideline, une grammaire et un outil pour les gérer.

Mots clés : Application web, qualité des applications web, amélioration continue, métrique de qualité, guideline.

Abstract

With the development of Internet, web applications are more and more important. Many quality standards, models, web engineering methods were proposed but the quality of many web applications is not yetat the desired level.

In this thesis, we propose an approach contributing to this area. Our approachcontains three iterative phases for, respectively, defining, measuring, and improving quality of web applications. In the first phase, we define a more complete, richer definition of quality of web applications. The latter is not only seen as quality of software, but also as quality of information, and quality of specific web features. In the second phase we build a taxonomy of metrics for measuring quality of web applications. This taxonomy is based on the ISO25010 quality model. In the third phase we collect and adapt guidelines for improving quality of web applications and providing web applications developers with useful advice. Our contribution consists of a guideline meta-model, a grammar, and a tool for managing guidelines.

Keywords : Web application, web application quality, continuous improvement, quality metrics, guidelines.

Résumé substantiel

Introduction

Internet a été créé à la fin des années 1960. Son prédécesseur était ARPANET, un réseau du département américain de la Défense. Mais jusqu'au début des années 1990, lors de la création du World Wide Web, Internet était encore inconnu dans le monde entier, alors il se développait à grande vitesse. Maintenant, Internet devient l'outil de communication le plus puissant. Il concerne tous les aspects de la vie et constitue le support privilégié pour les communications de tous les jours. Dans presque tout ce que nous faisons, nous utilisons Internet.

Le nombre d'internautes n'était que de 25 millions en 1994 [Le Journal du Net 2016]. Mais vers le 30 juin 2017, il y avait près de 3,9 milliards d'internautes dans le monde. Cela représente 51,7% de la population mondiale [Internet World Stats]. En 1995, il était inférieur à 1 % [Internet Live Stats]. Le nombre d'utilisateurs d'Internet augmente régulièrement. Ce nombre a été multiplié par dix entre 1999 et 2013. Le premier milliard a été atteint en 2005, le deuxième en 2010 et le troisième en 2014 [Internet Live Stats]. Une autre statistique issue d'une source française présente le nombre d'utilisateurs du début de 1990 à nos jours (Fig. 1).

Le nombre d'internautes n'était que de 25 millions en 1994 [Le Journal du Net 2016]. Mais vers le 30 juin 2017, il y avait près de 3,9 milliards d'internautes dans le monde. Cela représente 51,7% de la population mondiale [Internet World Stats]. En 1995, il était inférieur à 1 % [Internet Live Stats]. Une autre statistique source française présente le nombre d'utilisateurs du début de 1990 à nos jours (Fig 1). Le nombre d'utilisateurs d'Internet augmente régulièrement. Ce nombre a été multiplié par dix entre 1999 et 2013.

Le premier milliard a été atteint en 2005, le deuxième en 2010 et le troisième en 2014 [Internet Live Stats].

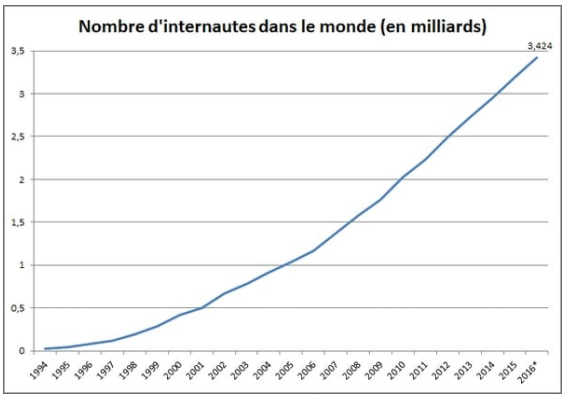

Figure 1: Nombre d'internautes de 1994 à 2016 (source: [Le Journal du Net 2016]

Le développement d'Internet est également marqué par le nombre de sites Web. Nous avons à ce jour plus de 1,8 milliard de sites [Netcraft 2012, cited January 2018], représentant 47,5 milliards de pages Web indexées par Google.

Une autre mesure de l'importance du phénomène est à mentionner, c'est le trafic Internet, c'est-à-dire le flux de données sur Internet. Le volume de données transféré par Internet est énorme. Le trafic mondial de données est de 96 EB (1 exaoctet = 1 milliard de gigaoctets) par mois [Cisco Systems 2017].

Les statistiques citées ci-dessus montrent qu'Internet est de plus en plus important pour la vie humaine en général.

L'importance du domaine de l'ingénierie Web

Le développement du World Wide Web a également conduit à un développement rapide de ses composants. Le développement web est l'un des métiers à la croissance la plus rapide au monde. Cela a créé beaucoup de travail pour les entreprises qui ont besoin d'une présence sur le Web. Dans le domaine Web, il existe également de nombreux postes, tels que développeurs Web, concepteurs Web, administrateurs Web, etc.

Une multitude de logiciels permettent cette activité. Dans une enquête réalisée en 2017 sur le site StackOverflow, 72,6% des développeurs travaillent en tant que développeurs Web

[Stack Overflow 2017, cited January 2018]. Le développement d'applications Web exige un développement « front-end » et un développement « back-end ».

Le développement « front-end » concerne les parties extérieures d'un site web ou d'une application web. Il s'appuie généralement le langage HTML, les feuilles de style en cascade (CSS) et JavaScript. Fondamentalement, les développeurs « front-end » construisent l'apparence extérieure, c'est-à-dire les pages des sites Web telles que les utilisateurs les voient. De plus, ce code dit client fonctionne sur le poste client, dans la plupart des cas le navigateur Web.

Le développement web « back-end » est ce qui se passe dans les coulisses. La partie arrière ne peut pas être vue par l'utilisateur final, mais c'est l'élément le plus fondamental d'une application Web. L'arrière-plan s'exécute sur le serveur. Contrairement au développement « front-end » (qui utilise principalement HTML, CSS et JavaScript), le développement « back-end » peut s'appuyer sur une gamme de langages et de « frameworks ».Les langages les plus populaires pour le développement « back-end » incluent: PHP, Python, ASP et Ruby. Pour les sites web de grande taille et les applications web, plus d'un langage « back-end » et plusieurs« frameworks » sont nécessaires. Toutes les informations d'un site web ou d'une application doivent être stockées quelque part. C'est là que les bases de données entrent en jeu. Les développeurs du « back-end » se chargent de leur gestion. Les systèmes de gestion de bases de données relationnelles populaires incluent SQL Server, MySQL et PostgreSQL.

Un autre type de développement web est appelé « pile complète ». C'est ainsi qu'on désigne l'ensemble de tous les constituants d'une application web. Dans les petites entreprises et startups, une seule personne sera souvent responsable de tous les aspects du développement Web. En revanche, dans les grandes entreprises, les personnes travaillent en équipe et ont des rôles plus spécialisés. Cela signifie que la pile complète n'est présente que dans les petites entreprises ou les projets.

Un navigateur Web est un logiciel qui permet à un utilisateur de localiser, d'accéder et d'afficher des pages Web sur Internet. Les cinq navigateurs web les plus populaires sont Google Chrome, Internet Explorer, Safari, Firefox et Opera.

Problèmes de qualité

Le site Web et les applications web d'une entreprise sont importants pour son image. C'est le visage d'une entreprise que tout le monde peut voir. Il est exposé à la société d'une nouvelle manière. Une enquête réalisée en 2010 a répertorié les problèmes les plus fréquemment cités concernant les aspects techniques des sites Web [Gelbmann 2010]. Le premier est "Aucune page Web trouvée sur une URL non-www" (5,3% des sites). Cela signifie que le serveur Web est configuré sur www.example.com, mais pas sur example.com. La seconde est "Déclaration du titre avant la déclaration de codage de caractères" (4.3% sites). Cela peut être un problème, car il n'est pas possible de lire la page sans connaître l'encodage. Le troisième est "Heure serveur incorrecte" (4.1% sites). Il est assez surprenant que 4% des serveurs aient ce paramètre incorrect, parfois avec un décalage supérieur à un jour. Plusieurs fonctionnalités de HTTP (HyperText Transfer Protocol) reposent sur l'échange d'horodatages, par exemple la mise en cache de pages et d'éléments de page, et l'expiration de certaines informations telles que les cookies. La quatrième est "Spécifications de codage de caractères contradictoires" (3,4% de sites). Le codage de caractères d'une page peut être défini de plusieurs manières: dans l'en-tête HTTP, dans l'en-tête XML (Extensible MarkupLanguage) et dans une balise méta HTML. Si quelqu'un définit le codage sur la page elle-même, cela peut provoquer une contradiction. Ces quatre problèmes sont faciles à éviter. Nous savons donc par cette enquête que ces problèmes révèlent une qualité médiocre des applications web et que nous devrions disposer d'un outil pour les éviter. Certaines entreprises ont également eu des problèmes légaux ou économiques en raison d'un mauvais site Web.

Objectif de la thèse

La qualité des applications web est un problème important et non résolu. Cette thèse a pour objectif de fournir quelques artefacts afin d'évaluer et d'améliorer la qualité des applications web. Tout d'abord, nous examinons la littérature et collectons des mesures de qualité à partir d'articles, de livres et d'autres sources. Nous classons les métriques de qualité sur la base du modèle de qualité ISO 9126 et de son successeur ISO 25010.

Nous pouvons évaluer la qualité des applications web grâce à un modèle construit avec ces métriques. Deuxièmement, nous rassemblons des « guidelines » issus de la littérature. Nous pouvons améliorer la qualité des applications web en appliquant ces guidelines pour remédier aux carences constatées dans le processus d'évaluation. Troisièmement, notre approche est mise en œuvre dans un outil nous permettant de gérer et de faire évoluer les guidelines.

Problèmes de recherche et solutions proposées

La partie Introduction présentait une vue globale des problèmes de qualité des applications web. Dans cette partie, nous discuterons plus en profondeur des problèmes abordés dans cette thèse et proposerons un aperçu de notre solution.

Difficultés de création des applications web

À partir des années 1990, Internet a été disponible pour le public. Depuis ce moment, Internet s'est développé rapidement et largement. Plus de 3,9 milliards de personnes avaient utilisé les services d'Internet en juin 2017 [Internet World Stats]. La taille du World Wide Web est estimée à environ 14 milliards de pages Web [Size]. Outre les sites Web, des applications web sont également développées sur Internet. Elles deviennent populaires. Les utilisateurspeuvent utiliser des applications web à la place des applications traditionnelles dans certains cas. Cependant, la qualité des applications web ne croît pas aussi rapidement que leur développement rapide. De nombreux développeurs ne réalisaient pas que les applications web avaient des caractéristiques et des exigences spécifiques, très différentes de celles des logiciels traditionnels. Les conséquences sont que près de 25% des projets Web ont échoué [Krigsman 2008].

Selon Krigsman, les trois raisons principales expliquant le taux d'échec des applications web sont les suivantes: (i) exigences changeantes, (ii) demandes incohérentes des parties prenantes et (iii) temps ou budget insuffisant. De nombreux sites Web sont créés chaque jour. Certains s'adaptent aux besoins des utilisateurs et proposent un contenu diversifié. Cependant, certains sites et applications web sont produits par des amateurs. Un

inconvénient qui en résulte est leur déficit de qualité.

Ces produits sont difficiles à maintenir et à développer dans le futur. Un moyen efficace d'évaluer la qualité des applications web est nécessaire.

La construction d'applications web n'est pas une tâche facile, pour certaines raisons. En énumérant les principales différences entre les applications web et les logiciels classiques, [Deshpande et al. 2002] a expliqué les difficultés suivantes pour la création d'applications web. Ce sont : des programmes de développement très concentrés dans le temps, une évolution constante avec des cycles de révision raccourcis, des spécifications insuffisantes, une absence de processus de test suffisants, un support de gestion minimal, une criticité de la performance attendue, des normes en évolution, auxquelles les applications web doivent se conformer, en fonction des circonstances spécifiques, une très grande variété de profils de développeurs, un environnement de mise en œuvre en évolution rapide, englobant diverses plateformes. Les méthodes classiques de génie logiciel ne sont pas suffisantes pour améliorer leur qualité car les applications web sont des logiciels spécifiques. Les méthodes spécifiques d'ingénierie des applications web sont également insuffisantes pour garantir un bon niveau de qualité. Nous proposons de fournir aux concepteurs de sites web un ensemble d'artefacts afin de définir, de mesurer et d'améliorer la qualité de leurs applications web. À cette fin, nous proposons une approche basée sur un cycle itératif d'amélioration continue.

Approche d'amélioration continue

L'ingénierie de la qualité a déjà une longue histoire. Cela a commencé avec la mise en place des contrôles qualité, puis l'amélioration de la qualité et ensuite le développement de systèmes de gestion de la qualité. Après une période où la gestion de la qualité totale était l'objectif principal de tous les spécialistes de la qualité, les chercheurs et les praticiens ont tous convergé vers le principe de l'amélioration continue qui a été adoptée comme le meilleur moyen de résoudre les problèmes de qualité [Davenport 1993].

Dans cette veine, la roue PDCA (Plan - Do - Check - Act) est l'approche dominante dans le cycle de vie d'un projet [Dale 2015]. Notre approche s'appuie sur un processus itératif adapté de PDCA. Nous rappelons d'abord les principaux concepts de PDCA avant de décrire les principales étapes de notre approche.

Le cycle PDCA est connu non seulement des spécialistes de la qualité, mais également d'un grand nombre de cadres [Deming 1993]. Ce cycle est représenté par un diagramme destiné à aider l'apprenant et à favoriser l'amélioration des produits ou des processus.

Inspirés par PDCA, nous proposons une approche en trois phases: (i) définition, (ii) mesure et (iii) amélioration de la qualité des applications web. Le processus s'inscrit dans un cycle itératif.

Premièrement, nous définissons la qualité des applications web en définissant des facteurs et des objets de qualité au cours de la première phase. Deuxièmement, nous mesurons la qualité des applications web. Troisièmement, nous fournissons des instructions pour aider les utilisateurs à améliorer leurs applications web (Fig. 2).

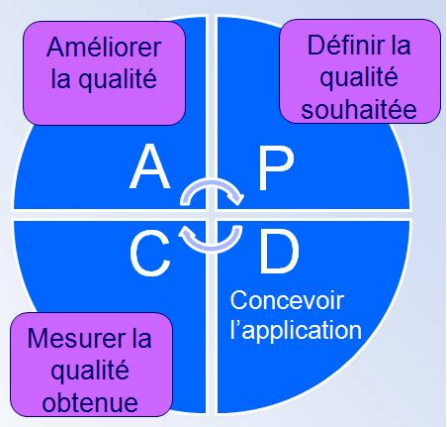

Figure 2: Trois phases de notre approche inspirée par PCDA

Définition des facteurs de qualité

Dans cette étape de notre méthode, nous élaborons un plan d'amélioration de la qualité d'une application web et définissons les facteurs de cette qualité. De nombreux facteurs peuvent influer sur la qualité d'une application. Chaque application peut être vue selon plusieurs aspects. Chaque vue présente des défauts ou des inconvénients. Nous nous concentrons donc sur les caractéristiques que nous souhaitons améliorer. Nous choisissons les métriques en fonction de ces caractéristiques.

Mesure de la qualité des applications web

A l'issue de la première phase de conception d'une application, on peut mettre en place les métriques nécessaires pour mesurer la qualité de cette application. Notre travail se concentre sur l'évaluation de la qualité des applications web, car la construction d'applications web n'entre pas dans le cadre de notre thèse. Après avoir réalisé cette phase, nous obtenons un résultat contenant des aspects ou des caractéristiques de l'application nécessitant des améliorations.

Améliorer la qualité des applications web

Dans cette phase, nous proposons d'utiliser des « guidelines » afin de remédier aux défauts que nous avons identifiés lors de la deuxième phase. Nous identifions des guidelines qui pourraient aider à résoudre les problèmes spécifiques de qualité. Si les « guidelines » peuvent résoudre les problèmes identifiés, nous pouvons mettre fin au processus. Sinon, cela signifie que l'application des guidelines n'est pas suffisante pour améliorer la qualité. Dans ce cas, il faut entamer un nouveau cycle. Nous répétons les trois phases de manière itérative jusqu'à ce que toutes les exigences sont satisfaites.

Un cadre de référence pour la qualité des applications web

Cette thèse vise à proposer une approche pour améliorer la qualité d'une application web. L'application web est un artefact complexe. C'est un logiciel. Il fournit également des informations. Enfin, il présente certaines caractéristiques spécifiques utilisées pour évaluer sa qualité, par exemple sa réputation. Dans cette thèse, nous proposons une nouvelle définition de la qualité des applications web englobant toutes ces dimensions.

Il existe de nombreuses publications ciblant la qualité des applications web. Elles diffèrent selon le point de vue des auteurs et l'objectif. Par souci de généralité, nous n'avons pas souhaité adopter un point de vue particulier, par exemple celui du développeur, ni aborder un type spécifique d'application web, par exemple un marché électronique. Au contraire, nous avons effectué une comparaison deux à deux des approches publiées pour pouvoir ensuite construire une approche les combinant au mieux.

Ainsi, nous décrivons notre approche de recherche qui comprend: 1) l'identification des approches antérieures pertinentes, 2) la sélection des approches les plus significatives, 3) leur analyse comparative, 4) la classification de ces approches et 5) la proposition d'un cadre résumant ceux proposés dans la littérature.

Nous avons effectué une recherche par mot-clé via différents moteurs de recherche (Google Scholar, IEEE Xplore, Business Source Complete, ScienceDirect). Les mots-clésutilisésétaient: "quality framework of a website", "quality assessment framework for a website", "quality framework of a web application", "quality assessment framework for an application", "quality assessment approach for a website", "approach to evaluating the quality of a web application". Ensuite, en utilisant des techniques de chaînage avant et arrière, nous avons rassemblé une série d'articles. Nous avons examiné ces documents et sélectionné ceux qui fournissent un cadre permettant de définir et d'évaluer la qualité d'une application web, qu'elle soit générique ou spécifique. Nous avons finalement obtenu quatorze approches. Notre objectif était de sélectionner un sous-ensemble d'entre elles pouvant être utilisé efficacement pour rassembler tous les points de vue de la qualité des applications web.

Les modèles de qualité que nous avons collectés sont résumés. Leur intérêt commun est de proposer un cadre de référence étendu constitué de plusieurs niveaux (au moins deux). Ils ont un nombre limité d'axes (entre 2 et 6). De plus, nous résumons les caractéristiques des cadres de référence proposés: le nombre d'axes (caractéristiques au premier niveau), le nombre de niveaux, le nombre de caractéristiques au deuxième niveau. Excepté [Kotian and Meshram 2017], tous les « frameworks » sont purement hiérarchiques. Dans le cadre [Kotian and Meshram 2017], plusieurs axes partagent certaines caractéristiques.

Similarité des dimensions

De nombreux cadres de référence ont déjà été proposés pour définir et évaluer la qualité d'une application web. À notre connaissance, il n'existe pas de cadre de référence standard permettant à différentes parties prenantes (concepteurs, développeurs, sponsors, etc.) de partager un point de vue commun et de faciliter ainsi leurs échanges. Pour avancer dans la définition d'une telle norme, nous avons comparé les cadres de référence existants afin d'en

déduire un cadre de référence commun. La question de recherche abordée dans ce chapitre est donc la suivante: est-il possible d'unifier tous les cadres de référence existants afin de les enrichir mutuellement et de produire un cadre pouvant être accepté par tous?

Comme l'a montré la revue de littérature précédente, tous les cadres de référence sont, à une exception près, construits de manière hiérarchique. Ils contiennent deux à trois niveaux. Ils sont toujours décrits de manière descendante, en justifiant le premier niveau à l'aide de points de vue ou de perspectives. De nombreuses différences existent puisque chaque article se concentre sur des éléments différents, enrichissant ainsi chaque perspective. Chaque axe ou premier niveau est ensuite décrit à l'aide de nombreuses caractéristiques dont la dénomination ne fait pas non plus l'objet d'un consensus.

Afin de construire un cadre unificateur, nous avons défini une mesure de similarité entre tous ces axes, ce qui a conduit à une matrice stockant toutes ces similitudes. Dans cette section, nous décrivons ce processus de comparaison. La revue de la littérature nous a conduits à la sélection de treize dimensions (Table 1).

Afin de comparer ces dimensions deux à deux, nous avons défini cinq niveaux de similarité comme suit:

1. deux dimensions sont totalement différentes
2. deux dimensions partagent très peu de sous-caractéristiques
3. deux dimensions partagent certaines sous-caractéristiques
4. deux dimensions partagent beaucoup de sous-caractéristiques
5. deux dimensions sont identiques.

Nous construisons une matrice des similitudes ainsi obtenues. Bien entendu, cette matrice est symétrique et sa diagonale est composée de valeurs toutes égales à 1. Les valeurs de 0.25, 0.5 et 0.75 ne reflètent pas la proportionnalité mais définissent un ordre total sur toutes les similitudes. En étudiant cette matrice, nous pouvons faire apparaître manuellement trois ou quatre dimensions principales. Par exemple, la qualité du système, la perspective du développeur et la qualité de la source sont très similaires. Cependant, afin

Table 1: Axes de qualité sélectionnés

Numéro	Dimension de qualité	Références
1	System quality	[Cao et al. 2005] [Orehovački et al. 2013] [Kotian and Meshram 2017]
2	Developer perspective	[Nabil et al. 2011]
3	Source quality	[Zhao and Zhu 2014]
4	Technical Adequacy	[Aladwani and Palvia 2002]
5	Design / User friendly quality	[Hasan and Abuelrub 2011]
6	Information / Content quality	[Cao et al. 2005] [Hasan and Abuelrub 2011] [Orehovački et al. 2013] [Zhao and Zhu 2014] [Kotian and Meshram 2017]
7	Visitor perspective	[Nabil et al. 2011]
8	Web content	[Aladwani and Palvia 2002]
9	Web appearance	[Aladwani and Palvia 2002]
10	Service quality	[Cao et al. 2005] [Orehovački et al. 2013] [Kotian and Meshram 2017]
11	Application specific quality	[Zhao and Zhu 2014]
12	Owner perspective	[Nabil et al. 2011] [Yuhana et al. 2014]
13	Organization quality	[Hasan and Abuelrub 2011]

de proposer un ensemble de dimensions plus robuste fondé sur cette matrice, nous avons effectué plusieurs tentatives de regroupement. Ils sont décrits dans la section suivante.

Classification automatique des dimensions de qualité

Basée sur les données ci-dessus, nous avons effectué différents essais de classifications automatiques. Nous avons utilisé l'outil CIMminer [Genomics and Pharmacology Facility, Center for Cancer Research, National Cancer Institute cited January 2018], un outil capable de générer des cartes d'images en grappes codées par couleur (cartes thermiques) afin de représenter des ensembles de données de grande dimension.

Nous avons d'abord effectué des classifications en utilisant alternativement trois algorithmes différents de classification hiérarchique (Single Linkage, Complete Linkage et Average Linkage), trois fonctions d'agrégation des distances ((Euclidean, Manhattan et Cor-

relation) et trois ensembles de valeurs de similitude (normale, haute et basse). L'objectif de toutes ces expérimentations était de réduire l'arbitraire induit par le choix des valeurs de similarité.

La valeur dessimilitudes initiales est la collection de 1, 0.75, 0.5, 0.25, 0 définie ci-dessus.

Ce choix des valeurs est subjectif. Cela ne fait que refléter un ordre entre similitudes. Ainsi, nous avons également utilisé deux autres ensembles de valeurs de similitude : haute et basse pour observer l'impact de ces choix sur les résultats de la classification. Pour exécuter des algorithmes de classification non hiérarchiques, nous avons utilisé le langage R, un langage de programmation et un environnement logiciel libre pour le calcul statistique et la fouille de données. Nous avons donc 27 résultats d'algorithmes hiérarchiques et 15 résultats d'algorithmes non hiérarchiques. Nous remarquons que les 42 expériences successives nous ont conduits à seulement 20 configurations différentes. Nous constatons que la configuration la plus fréquente est la configuration qui contient les quatre groupes suivants :

A. System quality, Developer perspective et Source quality

B. Information / Content quality, Visitor perspective et Web content

C. Web appearance, Application specific quality, Owner perspective et Organization quality

D. Technical adequacy, Design / User friendly quality et Service quality

Discussion et conclusion

Comme expliqué ci-dessus, notre processus de classification nous a permis de dégager quatre axes ou groupes assez robustes.

Le groupe A contient System quality, Developer perspective et Source quality. Il rassemble toutes les caractéristiques analysant les applications web en tant que logiciels. Ils contiennent des caractéristiques très similaires, telles que la réactivité et les délais, la traçabilité et la testabilité ou la modularité, la personnalisation et l'adaptabilité, etc.

Le groupe B contient Information/Content quality, Visitor perspective et Web content. Cela permet généralement aux auditeurs d'évaluer une application web en tant que

fournisseur d'informations. Sept cadres incluent cette dimension. De nombreuses caractéristiques communes rendent ces trois axes très similaires: précision (dans les trois dimensions), pertinence, etc. Cette dimension est assez facile à obtenir. Elle devrait bénéficier des efforts de normalisation tels que proposés par ISO 8000.

Le groupe D présente des caractéristiques qui traitent de la qualité du logiciel, mais pas du point de vue du développeur. Ils traitent en particulier de la sécurité, de la disponibilité, de la convivialité, de la facilité d'accès et de la fiabilité.

Enfin, le groupe C traite principalement des aspects spécifiques du produit qu'est une application web. Ainsi, il contient la popularité ou l'attractivité, la cohérence de présentation ou de couleur, l'identité, l'innovation, l'utilisation correcte des couleurs, la langue / les styles, etc.

Métriques pour la qualité des applications web

La partie précédente était consacrée à la définition de la qualité d'une application web. C'était la première étape de notre approche. La deuxième étape vise à mesurer cette qualité. À cette fin, dans cette partie, nous passons en revue et classons les mesures proposées dans la littérature pour traiter ce problème d'évaluation de la qualité.

Le but de la recherche décrite ci-dessous est de collecter et de caractériser les indicateurs de qualité des applications web. La partie précédente nous a permis de révéler le grand nombre de cadres de référence proposés pour structurer les différentes facettes de la qualité des applications web, en fonction de la perspective. D'autres publications, liées à ces cadres de référence ou indépendantes d'eux, ont proposé des dizaines de métriques permettant aux parties prenantes de mesurer différentes facettes. Dans cette partie, nous présentons le résultat de nos recherches visant à établir un lien entre les caractéristiques de qualité des applications web et de telles mesures. Nous avons choisi de traiter les caractéristiques contenues dans le standard ISO 25010. Ce dernier est largement adopté. De plus, à notre connaissance, il n'existe pas de recherche antérieure qui effectue une mise en correspondance des caractéristiques et sous-caractéristiques ISO avec ces métriques.

Méthodologie de recherche

Les mesures analysées dans cette partie ont été recueillies dans la littérature. Nous avons dû définir une approche systématique afin de les sélectionner dans les nombreux articles de ce domaine. À cette fin, nous avons effectué les deux étapes décrites ci-dessous:

1. Étude exploratoire

   En 2013, à partir de [Calero et al. 2005], la plus récente revue de littérature détaillée sur le sujet, nous avons mis à jour leur étude avec les objectifs suivants:

   - Sélection des métriques de qualité et par conséquent, élimination de toutes les métriques descriptives, par exemple les métriques de taille,
   - Identification des nouvelles publications pertinentes, proposant des nouvelles métriques,
   - Mise en correspondance de l'ensemble des métriques résultantes avec les caractéristiques et sous-caractéristiques de la norme ISO 9126. Le résultat de cette première étape a été publié dans [Cherfi et al. 2013].

2. Deuxième étape

   Afin d'obtenir un ensemble de mesures représentatif plus fiable, nous avons procédé à une revue de littérature plus systématique. À cette fin, nous avons défini quels documents devraient être les nouvelles entrées de notre processus. Ceci est défini à l'aide de critères d'inclusion et d'exclusion des articles.

   La distribution des métriques Web par caractéristique est représentée par le diagramme à secteurs de la Figure 3. Il montre en particulier que deux caractéristiques, à savoir la maintenabilité et l'utilisabilité, totalisent près de 60% des métriques. De plus, la caractéristique de fiabilité attire moins de 10% des mesures recensées, ainsi que l'efficacité.

   La précédente étude réalisée par Calero et ses co-auteurs dans [Calero et al. 2005] met en évidence une situation différente (Figure 4). Rappelons-nous que c'était en 2005. Il est donc intéressant d'analyser l'évolution de la situation. Il y a un chevauchement

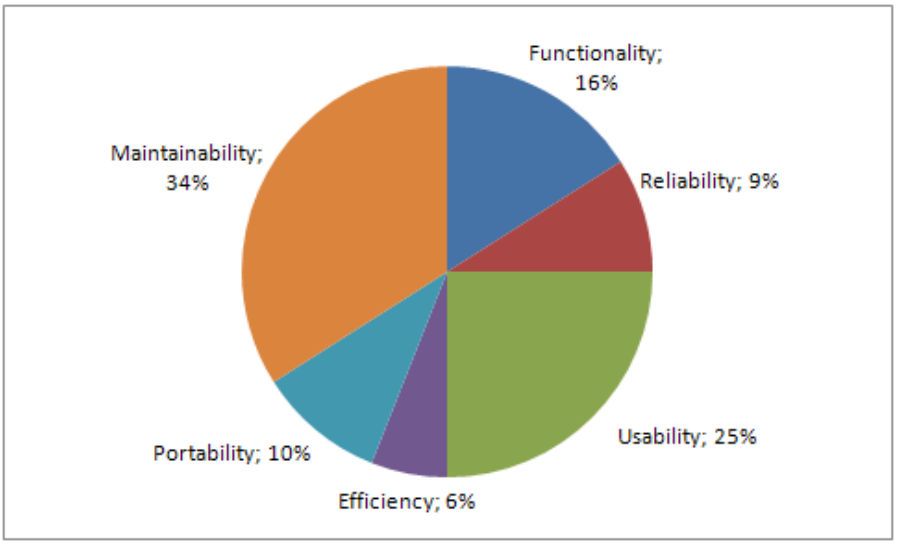

Figure 3: Vue d'ensemble des métriques par caractéristique

entre les deux sous-ensembles de métriques étudiés, mais notre analyse intègre des travaux récents, postérieurs aux recherches de Calero.

La première observation est que les métriques de maintenabilité semblent susciter un intérêt croissant avec 34% du total des métriques. En effet, grâce à leur attractivité, les applications web deviennent plus populaires pour les particuliers et même pour les entreprises. Dans le même temps, ils ont tendance à être plus complexes, générant ainsi des coûts de maintenance élevés. Cette complexité est inhérente à leurs architectures et technologies sous-jacentes. C'est aussi une conséquence de leur évolution rapide due à leur attractivité et à la pression du marché. Les approches préventives de la qualité logicielle, basées sur des métriques d'évaluation, ont permis d'envisager la qualité plus tôt dans le processus de développement. Cela a conduit à une réduction des coûts de maintenance. Nous pouvons en déduire que le même phénomène devrait être observé dans le développement d'applications web.

La convivialité attire 25% des métriques, ce qui est inférieur à la valeur observée dans [Calero et al. 2005]. Toutefois, cela ne veut pas dire que c'était plus important en 2005. En effet, nous avons ici des pourcentages, ce qui signifie que d'autres caractéristiques, telles que la maintenabilité, ont gagné un intérêt relativement grand. D'une part, les applications web sont la plupart du temps utilisées par des utilisateurs

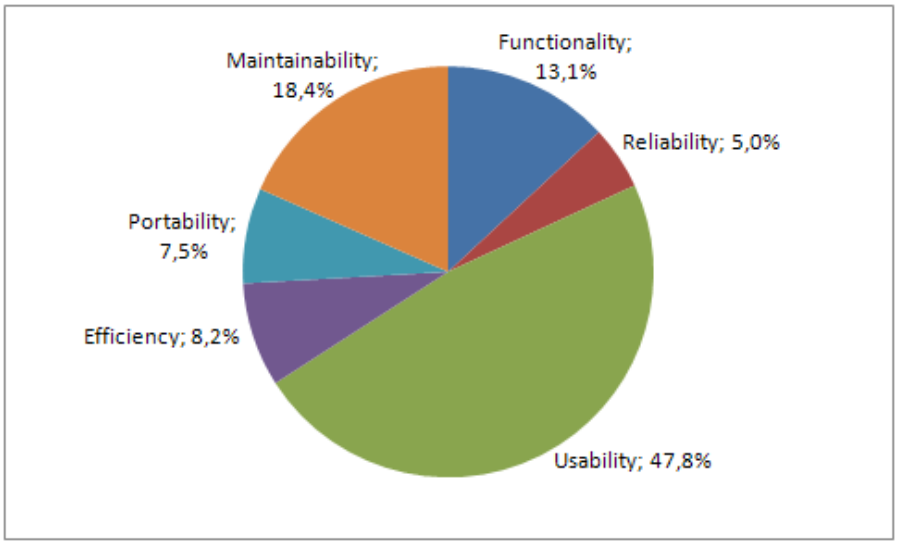

Figure 4: Distribution des métriques de qualité web selon [Calero et al. 2005]

finaux n'ayant pas de compétences spécifiques dans les technologies informatiques. D'autre part, le succès de ces applications dépend de leur acceptation par ces personnes non qualifiées. Cela montre l'importance de la convivialité, et plus précisément de la compréhensibilité. La portabilité a également évolué. Cela est dû à la diversité et à l'hétérogénéité des technologies utilisées. L'efficacité et la fiabilité suscitent encore peu d'intérêt. Cela est probablement dû à une relativement meilleure gestion des problèmes sous-jacents liés au matériel (utilisation de serveurs de secours, configuration de procédures de récupération, etc.). Des solutions préventives, basées sur des métriques, pourraient toutefois fournir de bonnes solutions complémentaires en ciblant bien les problèmes. Cependant, nous voudrions souligner les limites de notre étude, qui n'a probablement pas pris en compte l'abondant travail lié aux mesures de la qualité Web, car il faudrait beaucoup plus de temps et de moyens. Néanmoins, il s'agit d'un bon point de départ pour une étude plus vaste.

Étude complémentaire de la littérature

La deuxième étape de nos recherches avait deux objectifs : a) Mise à jour de notre première étude, b) Assurer un degré d'exhaustivité en effectuant une recherche plus systématique des métriques de qualité web.

Protocole

Notre objectif était de créer un ensemble assez complet de mesures de la qualité Web nous permettant de fournir aux développeurs d'applications web un outil pratique d'évaluation des applications web. En particulier, nous avons cherché à mapper ces métriques aux composants du cadre proposé par ISO25010 (SQUARE).

Nous avons défini les termes de recherche suivants: «Web quality», «qualitymetrics» et «web metrics», puis nous avons sélectionné les articles à l'aide de la liste suivante de critères d'inclusion et d'exclusion:

Critère d'inclusion 1: le document décrit une recherche consacrée aux métriques de qualité Web.

Critère d'inclusion 2: le document a été publié après 2000.

Critère d'inclusion 3: le document est rédigé en anglais ou en français.

Critère d'inclusion 4: le document est une œuvre originale. Cela signifie que les métriques proposées sont publiées pour la première fois.

Critère d'exclusion 1: le document est dédié aux métriques de qualité web, mais se concentre uniquement sur les métriques des précédents et n'en propose pas de nouvelle.

Critère d'exclusion 2: le document étudie la qualité du logiciel globalement et pas spécifiquement la qualité des applications web.

En effectuant ce processus de revue de littérature, nous avons ainsi rassemblé 7 articles supplémentaires à ajouter à notre liste. Tous ces articles ont été publiés de 2013 à 2017, à l'exception d'un article de Vaucher (publiée en 2009) et qui avait échappé à notre première étude.

Résultats

Nous avons analysé les 167 métriques de la littérature et les avons classés en fonction des huit caractéristiques de qualité d'ISO25010 et de leurs sous-caractéristiques. Nous présentons une analyse globale.

Mesure de l'Adéquation fonctionnelle

Nous avons trouvé 47 mesures qui correspondent le mieux à une ou plusieurs sous-caractéristiques. Par exemple, la présence d'une carte du site web est liée à la complétude fonctionnelle, alors que la taille de l'image caractérise la pertinence fonctionnelle. Un autre exemple est le défilement horizontal qui caractérise à la fois la correction fonctionnelle et la pertinence.

Mesure de la Performance

La plupart des métriques sont dédiés à la mesure de l'utilisation des ressources. Ce résultat est similaire à l'analyse précédente. Cela reflète peut-être uniquement le fait qu'il existe plus de moyens de mesurer l'utilisation des ressources que le comportement temporel, évalué de manière assez classique grâce au temps de téléchargement, au temps de réponse et au trafic.

Mesure de la Facilité d'utilisation

116 métriques (sur 166) caractérisent, d'une manière ou d'une autre, la facilité d'utilisation des applications web. Leur répartition est assez homogène parmi les six sous-caractéristiques.

Mesure de la Fiabilité

Vingt-quatre métriques peuvent être associées à une ou plusieurs sous-caractéristiques de fiabilité. La plupart d'entre elles peuvent être associées à la Maturité, c'est-à-dire «la mesure dans laquelle un système, un produit ou un composant répond aux besoins de fiabilité dans des conditions de fonctionnement normales».

Mesure de la Maintenabilité

Nous avons obtenu cinquante-cinq paramètres pour l'évaluation de la maintenabilité. Notons que quelques métriques sont des métriques spécifiques proposées pour la modularité ou la réutilisabilité. Les développeurs d'applications web devraient ainsi s'appuyer sur les références de la programmation orientée objet qui a conduit à la définition de nombreuses métriques pouvant être adaptées au contexte du développement web.

Mesure de la Portabilité

La dimension de Portabilité a un périmètre réduit dans la nouvelle norme car la fonctionnalité de coexistence a été transférée vers la nouvelle dimension de compatibilité. Par

conséquent, seules trois sous-caractéristiques décrivent la portabilité d'un logiciel. Nous avons trouvé 22 mesures alignées sur cette dimension. La plupart d'entre elles mesurent l'adaptabilité.

Mesure de la Sécurité

Les documents traitant des métriques d'application web ne traitent pas de la dimension de sécurité au même niveau. C'est la raison pour laquelle nous n'avons pas trouvé beaucoup de métriques mesurant les sous-caractéristiques de sécurité. En raison de l'importance de ce sujet, il faudrait une étude spécifique que nous n'avons pas pu mener faute de temps.

Mesure de la Compatibilité

Vingt-cinq métriques décrivent cette dimension. La compatibilité revêt une importance particulière dans les applications web qui doivent communiquer de manière dynamique ensemble. Le principal objectif de cette communication est l'échange d'informations. Les deux sous-caractéristiques sont mesurées par de nombreux paramètres.

La principale contribution de notre recherche, décrite dans cette partie, est la mise en correspondance des métriques et des sous-caractéristiques de qualité. Elle enrichit la littérature en fournissant une association fine entre l'ISO 9126, ainsi que l'ISO 25010, et les principales métriques décrites dans la littérature.

De nombreuses métriques sont définies, testées et proposées pour aider les développeurs lors de l'évaluation de leurs applications web. Cependant, toutes les métriques ne peuvent pas être facilement implémentées. De plus, de nombreuses métriques peuvent ne pas être estimées de manière significative avant la mise en exploitation réelle de l'application web. C'est une justification encore plus grande de notre approche proposant une définition cyclique de la qualité des applications web.

Guidage des applications web

Les entreprises développent et gèrent des sites Web complexes qui leur permettent de communiquer facilement et de manière dynamique avec leurs clients, fournisseurs, partenaires, etc. En 2008, 24% des projets Web n'avaient pas été livrés dans les limites du budget et 5% étaient incapables de respecter le budget prévu pour leur développement.

En outre, 21% n'ont pas répondu aux exigences des parties prenantes et près du tiers des projets Web (31%) n'ont pas été livrés dans les délais convenus [Krigsman 2008]. Plus récemment, une étude portant sur plus de 5400 projets informatiques a conclu que 45% des grands projets dépassaient les prévisions budgétaires, 7% dans le temps et 56% offraient une valeur inférieure à celle prévue [Bloch et al. 2013]. Les raisons varient : objectifs peu clairs, manque d'alignement avec les activités (objectif manquant), exigences changeantes, complexité technique (problèmes de contenu), équipe inadaptée, manque de compétences (problèmes de compétences), calendrier peu réaliste, planification réactive (problèmes d'exécution) [Bloch et al. 2013], demandes incohérentes des parties prenantes et manque de temps ou de budget [Krigsman 2008].

Cependant et malgré les recherches et les efforts d'outillage, très peu de développeurs adoptent les méthodes et beaucoup continuent d'appliquer des pratiques ad hoc. La raison principale est que ces approches souffrent d'un manque d'intérêt. Même si les concepteurs d'applications web se réfèrent à ces approches, ils ne disposent pas de connaissances suffisantes leur permettant de les implémenter efficacement. En conséquence, les applications résultantes ne sont ni conviviales ni faciles à gérer.

Nous soutenons que les approches actuelles sont bien structurées. Cependant, elles doivent être enrichies de guidelines aidant les concepteurs dans les nombreuses décisions qu'ils doivent prendre lors du développement d'applications web. Par conséquent, nous avons rassemblé les différents ensembles de guidelines proposés dans la littérature et les avons organisés selon différentes dimensions. Cette structure nous permet notamment de lier les recommandations aux objectifs de qualité (maintenabilité, performances, fonctionnalité, sécurité, etc.) et aux étapes pertinentes de la conception d'une application web (conception du contenu, de la navigation et de la présentation).

Un test d'utilisation des guidelines

Avant de définir la question de recherche abordée dans cette partie, nous avons dressé un rapide inventaire pour déterminer dans quelle mesure les meilleures pratiques et guidelines en matière de conception Web sont suivies par les sites Web existants. L'objectif était i) d'analyser si les guidelines existants sont utilisés et ii) d'identifier comment faciliter leur

adoption et d'éviter ainsi les approches ad hoc. Ainsi, nous avons d'abord collecté 475 recommandations provenant de plusieurs sources.

Collecte des guidelines

Identification des sources pertinentes

Pour collecter efficacement les guidelines de la littérature, nous avons effectué une recherche par mots clés, tels que «website guideline», «guideline for website», «guideline securityweb application» dans le titre et le contenu du document, à partir des principales bibliothèques électroniques et bases de données bibliographiques de la recherche informatique : IEEE Xplore, Springer, ScienceDirect, ACM et DBLP. Par exemple, sur la base des mots-clés «web» et «guideline», nous avions 1273 résultats deIEEE, 273 résultats de ScienceDirect et 168 résultats de DBLP. Avec Springer et ACM, nous avons beaucoup plus de résultats dans de nombreux domaines. Nous avons donc dû affiner les résultats et choisir des résultats très pertinents (calculés par les moteurs de recherche). Nous avons ensuite défini des critères d'inclusion pour sélectionner les sources (études primaires) et rejeter les autres. Les critères d'inclusion sont présentés dans le tableau ci-dessous (Table 2).

Table 2: Critères d'inclusion

Critère Description	
C1	L'étude porte sur la définition de guidelines pour les sites web
C2	L'étude mentionne les caractéristiques de qualité des sites web
C3	Le document est récent, c'est-à-dire publié depuis 2000
C4	Le papier propose des guidelines originaux (ne mentionne pas seulement les guidelines issus d'autres études)

Nous avons trouvé 14 sources avec 475 guidages. Dans certains cas, nous avons découpé en plusieurs certains guidelines, de sorte que le nombre de guidelines finalement obetnu peut être supérieur au nombre de guidelines proposé dans ces documents.

Capitalisation des guidelines: une approche guidée par un modèle

Dans la littérature, nous trouvons différentes façons de décrire les guidages: dans [Chiuchi et al. 2011], elles sont représentées par trois attributs: Category, Name et Contain. Dans [Ekberg et al. 2010], un guideline comprend trois parties: solutions de conception

/ application, objectif et description. Cette information descriptive n'est pas suffisante pour faciliter la réutilisation des guidelines par les concepteurs d'applications web. En particulier, ces derniers doivent trouver facilement les guidelines en utilisant des critères différents. Par exemple, dans le cas de la conception d'une application web pour les aveugles: quelles recommandations doivent-ils prendre en compte ? Si les développeurs souhaitent principalement faciliter la maintenabilité de l'application web, quels guidelines visent cet objectif ? Etc. Nous proposons d'abord un modèle permettant de capitaliser et de structurer les guidelines. Le méta-modèle est décrit à la Figure 5.

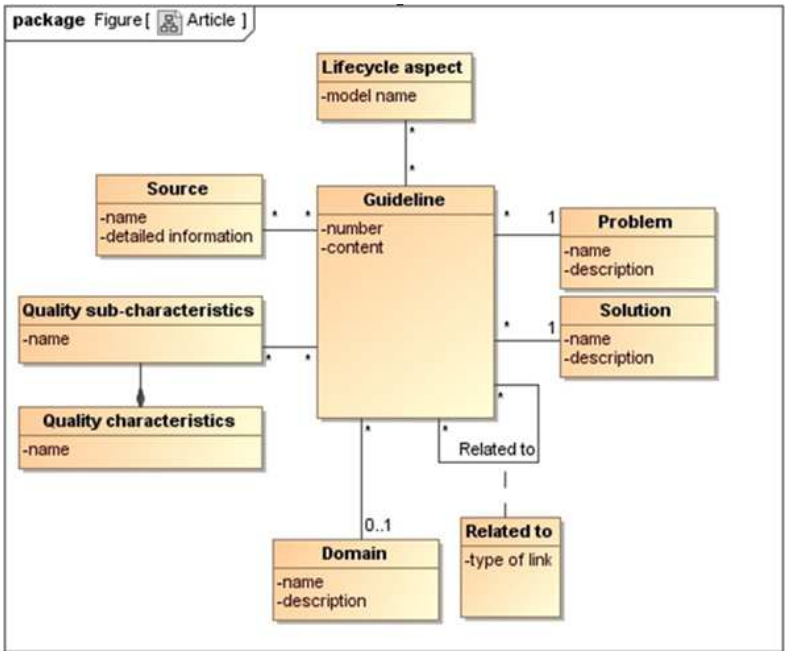

Figure 5: Le méta-modèle de guideline

Après la description générale des modèles pour les processus de décision [Harrison et al. 2007], nous proposons de lier chaque guideline aux catégories suivantes:

- la source où le guideline a été trouvé,
- les caractéristiques de qualité et sous-caractéristiques abordées dans le guideline,
- le problème qu'il vise à résoudre,
- la solution proposée,

- le domaine particulier concerné, le cas échéant,
- l'aspect du cycle de vie, c'est-à-dire le modèle d'application web (modèle de contenu, modèle de navigation, modèle de présentation) concerné.

Cette structure constituera une base de connaissances pour une réutilisation automatique via un outil de conception d'applications web. Le méta-modèle est représenté sous la forme d'un diagramme de classes UML à la Figure 5. La relation « related to » entre les guidelines nous permet de représenter les liens potentiels. Ainsi, l'attribut « type of link » peut prendre les valeurs "in contradiction with", "specializes" ou "similar to". Chaque guideline résout un problème. Cependant, plusieurs guidelines peuvent traiter le même problème. La solution du guideline décrit les règles à appliquer. Comme expliqué ci-dessus, dans notre processus, nous avons décomposé certains guidelines afin que chaque guideline résultant recommande une et une seule solution. Le domaine peut être général ou spécifique. Les caractéristiques de qualité (adéquation fonctionnelle, performance / efficacité, compatibilité, facilité d'utilisation, fiabilité, sécurité, maintenabilité, portabilité) et les sous-caractéristiques font référence à la norme ISO 25010 pour la qualité logicielle. Certains guidelines sont communs à plusieurs sources, d'où la multiplicité de la relation. Enfin, l'aspect cycle de vie comprend trois éléments: le contenu, la navigation et la présentation.

Grammaire de description des guidelines

Les sections précédentes ont capitalisé sur les guidelines trouvés dans la littérature. Afin de faciliter leur acquisition et de les enrichir, nous proposons de structurerchaque guideline sous la forme d'une phrase en langage naturel contrôlé. Ces phrases doivent être faciles à comprendre en se référant uniquement à des structures simples. Pour définir de telles structures, nous proposons une grammaire.

Notre grammaire s'appuie sur les quatre règles de Pohl [Pohl 2010], qui permettent aux concepteurs de documenter des scénarios:

- Règle 1: Utilisez le temps présent
- Règle 2: Utiliser la voix active

- Règle 3: Utiliser la structure de phrase sujet-prédicat-objet (SPO)
- Règle 4: Évitez les verbes modaux.

Cependant, la règle 4 est appropriée pour les scénarios mais pas pour les guidelines qui doivent en réalité contenir différentes modalités définies à l'aide de verbes modaux. Ainsi, nous n'avons appliqué que les trois premières règles.

Notation Backus-Naur de grammaire de guideline

```
<guideline> ::= <first part> <main part> <complement part>
<first part> ::= <modal verb> | <modal verb> ‘not’ | ‘do not’ | ∅
<modal verb> ::= ‘should’| ‘must’ | ‘have to’
<main part> ::= <verb> <main part complement>
<main part complement> ::= <main part complement> <comma> |
  <adjective>* <noun phrase> <adverb>*
<complement part> ::= <preposition> <body of complement> | ∅
<comma> ::= ‘,’
<noun phrase> ::= <determiner> <pre-modifier> <noun>
  <complement of noun phrase> |
  <determiner> <pre-modifier> <noun> <post-modifier>
<body of complement> ::= <clause> | <gerund phrase>
<clause> ::= <noun phrase> <verb phrase>
<gerund phrase> ::= <gerund> <complement of gerund phrase>
<complement of gerund phrase> ::= <noun> | <pronoun> | <adverb>
<gerund> ::= <verb>’-ing’
<determiner> ::= ‘a’|’an’|’the’
<pre-modifier> ::= <adjective> | <noun> | ∅
<post-modifier> ::= <adverb> | <prepositional phrase> | <clause>
<complement of noun phrase> ::=  <prepositional phrase> | <clause>
<verb phrase> ::= <verb> | <auxiliary verb> <gerund> |
  <auxiliary verb> <past participle verb> |<modal verb> <verb>
<prepositional phrase> ::= <preposition> <noun> | <preposition> <pronoun>
```

Figure 6: Description BNF de la grammaire de guideline

Sur la base de ces trois règles, nous avons examiné l'ensemble des guidelines de la littérature et construit une grammaire en utilisant un processus inductif. Cette grammaire est présentée avec la forme Backus-Naur. La notation Backus-Naur (plus communément appelée BNF ou Backus-Naur Form) est un moyen formel de décrire un langage, qui a

été développé par John Backus [Marcotty and Ledgard 2012]. Il est utilisé pour définir formellement la grammaire d'une langue ou d'un langage. Ainsi, un guideline est décrit par une phrase constituée de trois composants (Figure 6): la première partie, la partie principale et la partie complémentaire.

La première partie est un verbe modal (should, must, have to) en fonction du niveau de la recommandation. C'est optionnel. Le guideline peut être exprimé sous forme de phrase négative. La partie principale de la phrase est composée d'un verbe et d'un complément. Le complément de la partie principale peut être composé de plusieurs parties avec des adjectifs, des phrases nominales et des adverbes. Enfin, la phrase peut contenir une partie complémentaire. Le verbe peut être n'importe quel verbe du dictionnaire. Une liste des verbes déjà utilisés est proposée, mais c'est une liste ouverte. De la même manière, la phrase peut contenir des prépositions, des adjectifs, des noms, des adverbes, des pronoms, des verbes auxiliaires et des verbes participe passés.

Pré-traitement des guidelines bruts

Lors de la collecte de guidelines dans la littérature, nous avons effectué un pré-traitement des guidelines qui ne satisfaisaient pas la grammaire proposée. Nous avons décomposé les guidelines longs en plusieurs guidelines plus courts. Nous avons transformé certains guidelines, par exemple la position relative des éléments de clauses afin de respecter les règles de la grammaire tout en préservant leur signification.

Ainsi, nous avons harmonisé les guidelines extraits de la littérature afin de faciliter leur compréhension et leur appropriation par les concepteurs d'applications web. Nous décrivons l'outil permettant de mettre ces guidelines à la disposition des concepteurs-développeurs d'applications web.

Description du prototype

Nous proposons de rendre les guidelines disponibles via un outil web permettant aux concepteurs-développeurs d'applications web d'ajouter, d'interroger et de vérifier les guidelines. Le prototype de cet outil est décrit ci-dessous. Il contient trois modules permettant d'ajouter, de vérifier et d'interroger des guidelines.

Conclusion et perspectives

Dans cette thèse, nous avons apporté plusieurs contributions au domaine de la qualité des applications web:

1. Un cadre de référence pour la définition de la qualité,
2. Une structuration des métriques de la littérature basée sur la norme ISO 25010 /SQUARE qui fait référence dans le domaine de la qualité des logiciels,
3. Un méta-modèle de guidelines de qualité,
4. Une grammaire pour la définition des guidelines,
5. Un prototype pour gérer les guidelines (insertion, recherche, modification).

Ces contributions ont fait l'objet de plusieurs publications [Cherfi et al. 2013], [Do et al. 2016a], [Do et al. 2016b] qui nous ont permis de vérifier leur pertinence et leur intérêt.

Plusieurs pistes de recherche sont devant nous pour consolider ou étendre nos résultats. Premièrement, la validation du cadre de référence (chapitre 4) nécessite une étude plus approfondie des quatre dimensions proposées pour définir les sous-caractéristiques nécessaires et suffisantes. D'autres études statistiques pourraient compléter utilement la classification automatique décrite dans le chapitre 4, y compris une analyse en composantes principales. Les efforts de rapprochement avec d'autres normes ISO, notamment ISO 8000, faciliteraient la consolidation de l'axe qualité de l'information. De plus, les deux autres axes doivent être renforcés en trouvant des normes ISO similaires ou en menant des efforts de normalisation.

Deuxièmement, la profusion de métriques illustrée par le chapitre 5 doit amener le chercheur à utiliser un ensemble complet mais aussi minimal de métriques pour la qualité des applications web. Une validation doit être effectuée pour vérifier la pertinence et la faisabilité de ces mesures. Enfin, la procédure de mise à disposition des référentiels doit être couplée à une approche méthodologique couvrant l'ensemble du cycle de vie de l'application web afin que, à chaque étape et à chaque itération, le concepteur, le développeur ou l'ingénieur en charge de la maintenance se voit proposer les guidelines

pertinents pour cette étape et adaptés au contexte (type d'application web, utilisation, etc.).

Table of Contents

List of Tables

List of Figures

Part I

Chapter 1

Introduction

1.1 The importance of the Internet and the World Wide Web

Internet was created in the late 1960s. Its predecessor was ARPANET, a network of the US Department of Defense. But until the beginning of 1990s, when World Wide Web was created, Internet was still unknown around the world, since it was developed rapidly at a high speed. Now Internet becomes the most powerful communication tool. It impacts every aspect of life, being the preferred media for everyday communications. In almost everything we do, we use the Internet. For example, before the Internet, if you wanted to follow the news, you had to walk down to the newsstand. But today a click or two is enough to read your local paper and any new sources from anywhere in the world, updated up to the minute [Dentzel 2014].

The number of Internet users was only 25 millions in 1994 [Le Journal du Net 2016]. But around June 30, 2017, there were nearly 3.9 billion Internet users in the world. That means 51.7% of the world's population [Internet World Stats]. In 1995, it was less than 1% [Internet Live Stats]. Another French source statistic presents the number of users from the beginning of 1990 to the present day (Fig 1.1). The number of Internet users increases regularly and steadily. This number has increased tenfold from 1999 to 2013. The first billion was reached in 2005, the second billion in 2010 and the third billion in 2014 [Internet Live Stats].

The development of Internet is also marked by the number of websites. From the appearance of the first website on December 20, 1990, until today we have a vast number

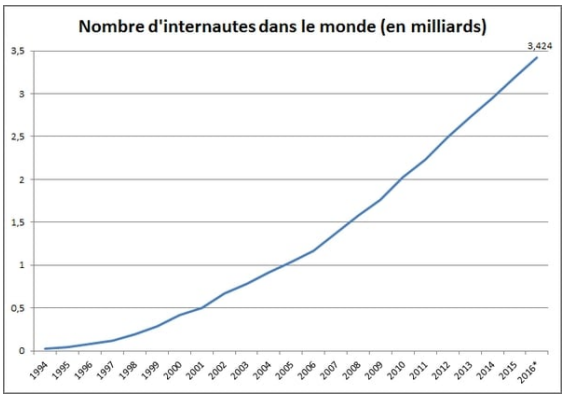

Figure 1.1: Number of Internet users from 1994 to 2016 (source: [Le Journal du Net 2016]

of websites: over 1.8 billion sites [Netcraft 2012, cited January 2018], with 47.5 billion webpages indexed by Google, the leader of search engines in the Internet.

Another thing that should be mentioned is the Internet traffic, the flow of data across the Internet. The volume of data which is transfered by Internet is enormous. The global data traffic is 96 EB (1 exabyte = 1 billion gigabytes) per month [Cisco Systems 2017].

These statistics quoted above show that Internet is more and more important for human life in general.

1.2 The importance of the Web engineering field

The development of the World Wide Web has led to a fast development of its components also. Web development is one of the fastest growing trades in the world. It has created a lot of work for companies that have and need a web presence. In the web field, there are also many job positions such as web developers, web designers, web administrators...

In a survey of the Stack Overflow site in 2017, 72.6% of developers work as web developers [Stack Overflow 2017, cited January 2018].

Developing web applications requires front-end development and back-end development. Front-end development deals with the outer-facing parts of a website or of a web application. It uses HyperText Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript. HTML is the key structural component of all websites in the Internet. CSS

adds style to HTML. JavaScript has been evolving over the last several years. In relation to front-end development, JavaScript helps make web pages interactive.

Basically, front-end developers construct the outward appearance, which are the website pages that users see. Moreover, the front end runs on the client system, in most cases, the web browser.

Back-end web development is what goes on behind the scenes. The back end part cannot be seen by the end user, but it is the most fundamental element of a web application. The back end runs on the server, or, as it's often called, "server-side".

Unlike the front-end development (which primarily uses HTML, CSS, and JavaScript), back-end web development can rely on a range of languages and frameworks. A few popular languages used in the back end include: PHP, Python, ASP, Ruby... For large-scale websites and web applications, more than a back-end language and a framework are needed. All the information of a website or of an application must be stored somewhere. This is where databases come in. Back-end developers handle these as well. Popular relational database management systems include SQL Server, MySQL, PostgreSQL...

Another type of web development is full stack. The latter is the combination of both the front end and the back end. In smaller companies/startups, a single person would more likely be responsible for all sides of the web development spectrum. However, in larger companies, people work in teams and have specialized roles. It means that full stack is present only in small size companies or projects.

A web browser is a software program that allows a user to locate, access, and display web pages in the Internet. The five most popular Web browsers are Google Chrome, Internet Explorer, Safari, Firefox and Opera.

1.3 Problems of quality

Website and web applications of a company are important for its image. It is like the face of a company that everyone can see. It exposes a company in a new way.

A survey in 2010 listed the most frequently mentioned problems in technical aspects of websites [Gelbmann 2010]. The first is "No web page found at non-www url" (5.3% sites).

It means that a web server is configured at www.example.com, but not at example.com. If the users missed the www, they can not reach the website. The second is "Title declaration before character encoding declaration" (4.3% sites). This may be a problem, because in principle, it is not possible to read the page without knowing the encoding. Most modern browsers and search engines are smart enough to work around such problems, but one has to keep in mind that websites are not only processed by modern browsers. The third is "Incorrect server time" (4.1% sites). It is quite surprising that 4% of the servers have an incorrect setting, sometimes by more than a day. Several features of HyperText Transfer Protocol (HTTP) rely on exchanging timestamps, for example caching pages and page elements, and expiration of certain information such as cookies. The fourth is "Contradictory character encoding specifications" (3.4% sites). The character encoding of a page can be defined in several ways: in the HTTP header, the Extensible Markup Language (XML) header and in an HTML meta tag. If someone defines the encoding on the page itself, it may cause the contradiction.

These four problems are easy to avoid. So from this survey we know that these problems reveal a low quality of web applications and we should have a tool to avoid them.

Some companies also had problems because of a bad website. One example is Penny Juice [PennyJuice]. It is a company selling juices for children. Their site was colorful, with color from a rainbow, and the text, the important information, was hard to read. It seems that the owners of this site chose a wrong way to approach their clients. Their site is dedicated to adult clients but it is targeted at children. Fortunately, the owners just realized that a bad website had bad effect on their business, so they redesigned their site just at the beginning of 2018.

1.4 Objective of thesis

Quality of web applications is an important and unsolved problem. This thesis aims to propose a few artifacts in order to evaluate and improve the quality of web applications.

First, we survey the research literature and collect quality metrics from articles, books and other sources. We categorize quality metrics based on the ISO 9126 quality model

and its successor ISO 25010. We can evaluate the quality of web applications through a model built with these metrics. Second, we collect guidelines from the literature, select those which help us to satisfy our goal and build a conceptual model representing these guidelines. We can improve the quality of web applications by applying these guidelines to overcome the shortcomings found in the evaluating process. Third, our approach is implemented in a tool enabling us to manage the guidelines.

1.5 Contributions of Thesis

This thesis delivers the following contributions:

1. *A taxonomy of quality metrics for web applications*

   We survey research literature about web applications in general and quality of web applications in particular. We collect quality metrics from them and categorize them into groups.

2. *A mapping of these metrics with the sub-characteristics of the quality model ISO 9126*

   We choose ISO 9126 model to organize the classification quality metrics. Collected quality metrics are divided according to relevant sub-characteristics. A metric may appear in more than one sub-characteristic.

3. *A selection of guidelines for designers and developers of web applications*

   We select guidelines from other researches, both for general and specific purposes. More than 400 guidelines are selected to cover all aspects of quality of web applications. We adapt them to satisfy our goal. If a guideline is too long, it is divided into several smaller guidelines. If a guideline is hard to understand, we restructure it in a simpler form.

4. *A multidimensional conceptual model which is a representation of these guidelines*

   The guidelines are expressed in natural language, so they must be formalized. We represent them by building a multidimensional conceptual model.

5. *A grammar for expression of guidelines*

By generalizing the guidelines, we obtain a grammar.

6. *A guideline management tool (input, update, search criteria...)*

   We built a tool for managing guidelines. Users can add new guidelines, update content of guidelines, remove unused guidelines, search required guidelines based on criteria, etc. This tool helps users to manage guidelines more effectively and faster.

1.6 Plan of thesis

Chapter 2 is the state of the art. We synthesize the whole literature related to web application quality and quality engineering. Chapter 3 presents a global view of problems in web application quality. Our contributions are in three chapters 4, 5 and 6. We propose a new definition of web application quality encompassing all these dimensions in Chapter 4. In Chapter 5, we review and categorize metrics proposed in the literature addressing this measurement problem. Chapter 6 is dedicated to improving quality by structuring the guidelines available to web application designers. The conclusion and perspective are in Chapter 7.

Chapter 2

Web application quality: a state of the art

In this chapter we synthesize the whole literature related to web application quality and quality engineering. First, we define the main concepts embedded in our topic: web applications, quality, quality engineering. Then we present a state of the art regarding web application quality. Finally we describe the literature on web application engineering.

2.1 Definitions

The object of our research is web application. So in the first chapter we discuss web applications and related issues. First of all, we give some common examples of web applications. We consider three types of web applications. One example is webmail. The latter is a system in which a user can access his emails via a browser on any computer or device that is connected to the Internet. The following sections are definitions of (i) web application and (ii) quality.

2.1.1 Web application

With the development of the Internet, applications also appear in the web platforms. Several definitions of web application can be found in the literature:

Microsoft Dictionary [Microsoft Corporation 2002] defines a web application as a set of clients and servers that cooperate to provide a solution to a problem.

In the Dictionary of Computer Science published in 2016 [Butterfield and Ngondi 2016], a web application is a client/server application where the client is a web browser and the two communicate using HTTP. In practical terms, the server must produce output in the form of dynamic web pages and accept input in the form of HTTP requests.

In ISO 9241 [ISO 2010], a web application is defined as an application providing functionalities to the user through a browser or other type of agents using Web formats and protocols. Meanwhile, a website is a coherent collection of interlinked Web resources (for example, Web pages or Web services) that are located on one or several computers connected to the Internet, and that can usually be accessed through the same domain specification part of a URL.

In dictionary.com [dictionary.com 2017], web application or the abbreviate form, web-bapp, is defined simply as a software program that provides interactive functionalities and is accessed through a web browser and a URL.

A web application is any application that uses a web browser as a client. It is also a collection of servlets, HTML pages, classes, etc.

From these definitions of web applications, we can view that the definition of web applications changed as time passed. In the beginning of 2000s, web application is simply described as a set providing a solution to a problem.

In the definition of Oracle [Oracle], a web site is a related collection of files available on the web that is managed by a single entity and contains information in hypertext for its users. A web site often includes hypertext links to other web sites.

We can therefore see that a Web application is much more complicated than simple HTML web pages, and consists of more than just the front-end graphical user interfaces that users see.

In the next paragraph, we introduce various types of web applications.

2.1.1.1 Type of web applications

According to Mavromoustakos [Mavromoustakos and Andreou 2007], web applications can be divided into the following ten overlapping types:

1. *Informational* – It provides useful information to users (e.g., online newspapers, electronic books, newsletters);
2. *Interactive* – It allows the interaction between the user and the application through a graphical user interface (e.g., registration forms, customized presentations, online games);
3. *Transactional* – It includes a transaction mechanism (e.g., e-commerce, online banking);
4. *Service oriented* – It provides an online service (e.g., estimating a mortgage payment);
5. *Downloadable* – It provides information available for downloading by the user;
6. *Customizable* – It contains content that can be customized based on the users' preferences;
7. *Interactive* – It offers the communication among users via chat rooms, bulletin boards or instant messaging;
8. *Web portal* – It offers access to a great number of other web applications according to a variety of thematic contents (e.g., online intermediaries, electronic shopping malls);
9. *Database access* – It queries a database and retrieves information;
10. *Data warehousing* – It queries a collection of large databases and provides information.

Chopra [Chopra 2016] classified Web applications into five categories depending on their functionalities:

1. *Document centric websites*: these websites are very simple. They consist of only a set of web pages which are stored on the web server. This category contains static websites and also interactive web applications;
2. *Transactional web applications*: this category is more complex than document centric websites. It involves databases to store customer web data. Examples of this category are online shopping mall, online banking, etc;

3. *Workflow-based web applications*: these applications allows easy handling or workflows within or between different organizations. Examples of this category are business-to-business (B2B) solutions in e-commerce, e-government applications or Web-based supports of patient workflows. The social Web today is also in this category;

4. *Portal-oriented Web applications*: portals are the central hubs that act as a point of access to the Web. Some general portals are Yahoo, Netscape, etc;

5. *Ubiquitous Web applications*: they provide services as per the customers' demand. They may be small applications, e.g. displaying the temperature on the users' screen or menu displays of the day.

We can show that, in these two lists of categories, some types are similar. The *Informational* and *Interactive* of Mavromoustakos are in *Document centric websites* of Chopra. *Transactional* and *Web portal* are the same in the two lists. *Service oriented* of Mavromoustakos is part of *Workflow-based web applications* of Chopra. Because the work of Mavromoustakos is older than Chopra's, some types in that list are not still refined (such as Downloadable or Database). Let us notice that some sorts of web applications, as Social networks, are not in the list of Mavromoustakos.

2.1.2 Quality

In this section, we will define the main concepts underlying quality. Quality issues are important for web application and the development of web system.

2.1.2.1 Definition

Because this work targets quality-centred approach, we first cite some definitions of quality. Quality is defined in ISO 8402 standard [ISO/IEC 1994] as a set of characteristics of an element conferred upon it by the aptitude to meet explicit and implicit needs.

Deming [Deming 1993] also proposed a definition of quality. Key aspects of quality for the customer include:

1. Good design – looks and style;

2. Good functionality – it does the job well;

3. Reliable - acceptable level of breakdowns or failure;

4. Consistency;

5. Durable - lasts as long as it should;

6. Good after sales service;

7. Value for money.

2.1.2.2 Quality engineering

Improving quality of software in general and of web applications in particular is the goal of software engineering.

Quality engineering is the management, development, operation and maintenance of Information Technology (IT) systems and enterprise architectures with a high quality standard. The term "quality engineering" stresses the end-to-end aspect of software quality management.

Another definition of quality engineering is "a discipline that deals with the analysis of a manufacturing system at all stages, to improve the quality of the production process and of its output" [businessdictionary.com cited January 2018].

2.1.2.3 Process quality, product quality and quality in use

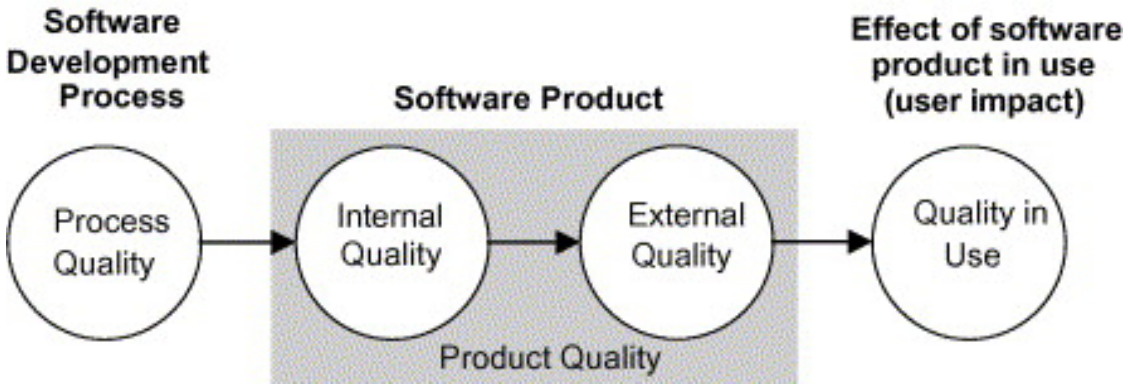

Figure 2.1: Quality categories [ISO 2001].

As in ISO/IEC 9126 [ISO 2001] and its successor ISO/IEC 25000:2005 [ISO/IEC 2014], software quality is decomposed into (i) process quality, (ii) product quality, and (iii) quality in use (Fig. 2.1).

Software processes implement best practices of software engineering in an organizational context. Process quality expresses the degree to which defined processes were followed and completed. Software products are the output of software processes. Product quality is determined by the degree to which the developed software meets the defined requirements. A product that perfectly matches defined requirements does not guarantee to be useful in the hands of a user when the implemented requirements do not reflect the intended use. Quality in use addresses the degree to which a product is fit for purpose when exposed to a particular context of use.

2.2 State of the art on web application quality

We synthesize in this section the main approaches dedicated to web application quality. The quality aspects which are mentioned are (i) standards and characteristics of quality; (ii) website quality models; and (iii) quality evaluation methods and tools.

2.2.1 Standards and characteristics of quality

In this section, we propose a literature review on web application quality. More precisely, we aim to provide an aggregate view of standards, models, methods, and tools proposed to evaluate and ensure web application quality.

2.2.1.1 ISO

There is no specific standard for web application quality. However, web applications are specific software applications. Therefore quality standards of software can be applied in web applications.

The norm ISO 9126 [ISO 2001] is the international standard that provides important parameters to assure the quality of applications and evaluate software quality. It was first introduced in 1991. The standard is divided into four parts which address, respectively, the

following subjects: (i) quality model; (ii) external metrics; (iii) internal metrics; and (iv) quality in use metrics. It is an extension of previous work performed by McCall [McCall et al. 1977], Boehm [Boehm et al. 1978], and other authors. ISO 9126 Part one, referred to as ISO 9126-1, is a quality model which defines a set of software quality characteristics.

The ISO 9126-1 software quality model identifies 6 main quality characteristics. Each of these main characteristics is decomposed into sub features (Fig. 2.2). The latest version of the standard contains 27 sub-characteristics for the internal and external quality as follows [Calero et al. 2005]:

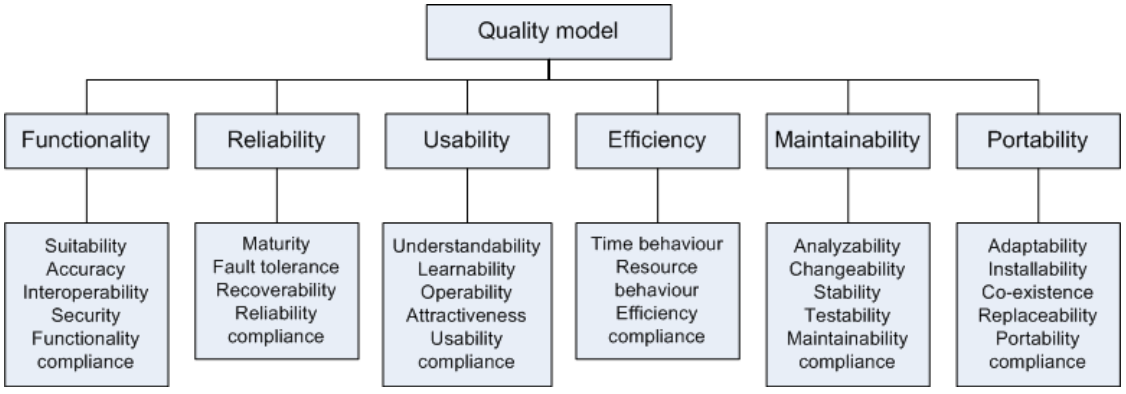

Figure 2.2: Quality model in ISO 9126.

- **Functionality**. It is a set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs. It answers *"Are the required functions available in the software?"*
  - ***Suitability***: attribute of software that bears on the presence and appropriateness of a set of functions for specified tasks;
  - ***Accuracy***: attribute of software that bears on the provision of right or agreed results or effects;
  - ***Interoperability***: attribute of software that bears on its ability to interact with specified systems;
  - ***Security***: attribute of software that bears on its ability to prevent unauthorized access, whether accidental or deliberate, to programs or data;

  - ***Functionality compliance***: attributes of software that make the software adhere to application related standards or conventions or regulations in laws and similar prescriptions. A compliance subcharacteristic is defined for each characteristic.

- **Reliability**: It is a set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time. It answers *"How reliable is the software?"*.

  - ***Maturity***: attribute of software that bears on the frequency of failure by faults in the software;
  - ***Fault tolerance***: attribute of software that bears on its ability to maintain a specified level of performance in cases of software faults or of infringements of its specified interface;
  - ***Recoverability***: attribute of software that bears on the capability to re-establish its level of performance and recover the data directly affected in case of a failure, and on the time and effort needed for it;
  - ***Reliability compliance***.

- **Usability**: is a set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users. It answers *"Is the software easy to use?"*.

  - ***Understandability***: attribute of software that bears on the users' effort for recognizing the logical concept and its applicability;
  - ***Learnability***: attribute of software that bears on the users' effort for learning its application (for example, control, input, output);
  - ***Operability***: attribute of software that bears on the users' effort for the operation and the operation control;
  - ***Attractiveness***: attribute of software that bears on the satisfaction of latent user desires and preferences, through services, behavior and presentation beyond actual demand;

  - ***Usability compliance***.

- **Efficiency**: is a set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions. It answers *"How efficient is the software?"*

  - ***Time behaviour***: attribute of software that bears on response and processing times and on throughput rates in performing its function;
  - ***Resource behaviour***: attribute of software that bears on the amount of resources used and the duration of such use in performing its function;
  - ***Efficiency compliance***.

- **Maintainability**: is a set of attributes that bear on the effort needed to make specified modifications. It answers *"How easy is can the the software be modified?"*

  - ***Analyzability***: attributes of software that bear on the effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified
  - ***Changeability***: attributes of software that bear on the effort needed for modification, fault removal or for environmental change
  - ***Stability***: attributes of software that bear on the risk of unexpected effect of modifications
  - ***Testability***: attributes of software that bear on the effort needed for validating the (modified) software
  - ***Maintainability compliance***.

- **Portability** is a set of attributes that bear on the ability of the software to be transformed from one environment to another. It answers *"How easy is to transfer the software to another environment?"*

  - ***Adaptability***: attributes of software that bear on the opportunity for its adaptation to different specified environments without applying other actions or means than those provided for this purpose for the software considered

- ***Installability***: attributes of software that bear on the amount of resources used and the duration of such use in performing its function
- ***Co-existence***: the capability of the software to co-exist with other independent software in a common environment sharing common resources
- ***Replaceability***: attributes of software that bear on the opportunity and effort of using it in the place of specified other software in the environment of that software
- ***Portability compliance***.

Each quality sub-characteristic is further divided into attributes. An attribute in this domain is an entity which can be verified or measured in the software product. However, attributes are not defined in the standard, due to the fact that they vary between different software products. The characteristics are manifested externally when the software is used as a consequence of internal software attributes. The attributes are measured by means of internal metrics. As an example, the maturity, which is a sub-characteristic of reliability, may be measured by metrics, such as lack of cohesion in methods, or tight class cohesion.

ISO 9126-1 is a generic model for quality assessment and it is necessary to adjust it to the type of software product which is estimated. It has been referenced and completed by numerous quality models, such as Quint2 (Fig. 2.3). Quint or extended ISO 9126 is an extension of the ISO 9126 standard for product quality. The model from [van Zeist et al. 1996] elaborates work from the Quint project and is therefore also known as Quint2. It adds 10 sub-characteristics to the 21 of ISO 9126 that are most appropriate for web products and are used in daily practice with their means were defined in [Calero et al. 2005]:

- **Functionality**
  - ***Traceability***: attributes of software that bear on the effort needed to verify correctness of data processing on required points
- **Reliability**

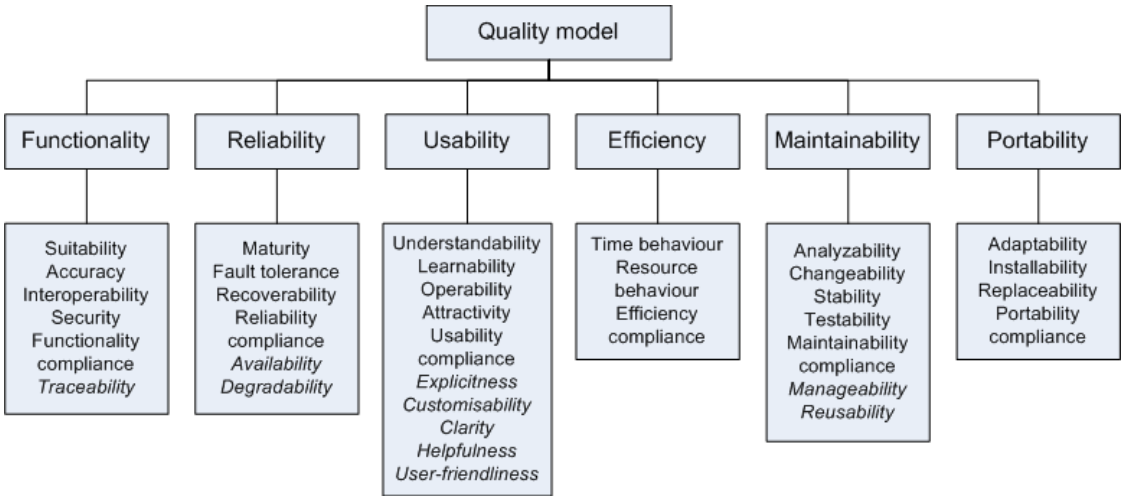

Figure 2.3: Quint-2 model, with 10 sub-characteristics added (in italics) and one sub-characteristic removed from ISO 9126

  - ***Availability***: attributes of software that bear on the amount of time the product is available to the user at the time it is needed
  - ***Degradability***: attributes of software that bear on the effort needed to re-establish the essential functionality after a breakdown

- **Usability**
  - ***Explicitness***: attributes of software that bear on the software product with regard to its status (progression bars, etc.)
  - ***Customisability***: attributes of software that enable the software to be customized by the user to reduce the effort required for use and increase satisfaction with the software
  - ***Clarity***: attributes of software that bear on the clarity of making the user aware of the functions it can perform
  - ***Helpfulness***: attributes of software that bear on the availability of instructions for the user on how to interact with it
  - ***User-friendliness***: attributes of software that bear on the users' satisfaction

- **Maintainability**

- ***Manageability***: attributes of software that bear on the effort needed to (re)establish its running status
- ***Reusability***: attributes of software that bear on its potential for complete or partial reuse in another software product

ISO 9126 was issued first time in 1991 and revised in 2001. So it is now obsolete. The need of making a new standard lead to the creation of ISO/IEC 25010. ISO/IEC later worked on SQuaRE (Software product Quality Requirements and Evaluation), a more extensive series of standards to replace ISO/IEC 9126, corresponding to the family ISO/IEC 250xx. After several years of development, a group of the ISO released in 2011 a reworked software product quality model standard named ISO/IEC 25010 [ISO/IEC 2011]. It is still strongly influenced by its predecessor ISO 9126 but restructures and adds several parts of the quality models. This standard is likely to become the most well-known type of software quality models [Wagner 2013].

ISO 25010 has eight main product quality characteristics (in contrast to six of ISO 9126) and 31 sub-characteristics (Fig. 2.4). In comparison with ISO 9126-1, some sub-characteristics such as *Learnability* or *Replaceability* among others have remained while new ones such as *Accessibility*, *Availability*, *Helpfulness* were added. *Security* has been added as a separate characteristic, rather than as a sub-characteristic of functionality in the former, while other names have changed slightly to enhance descriptiveness [Lew et al. 2010]. *Compatibility* is also a new characteristic with two sub-characteristics retrieved from other old characteristics. Some sub-characteristics were combined as one sub-characteristic like *Changeability* and *Stability* were merged to compose to *Modifiability*.

2.2.1.2 Other standards

Besides ISO, the community of quality researchers and practitioners also has developed other standards for other characteristics. In this section we discuss two of them: Usability and Accessibility.

Standard of Accessibility

For Accessibility, the most famous standard is Web Content Accessibility Guidelines

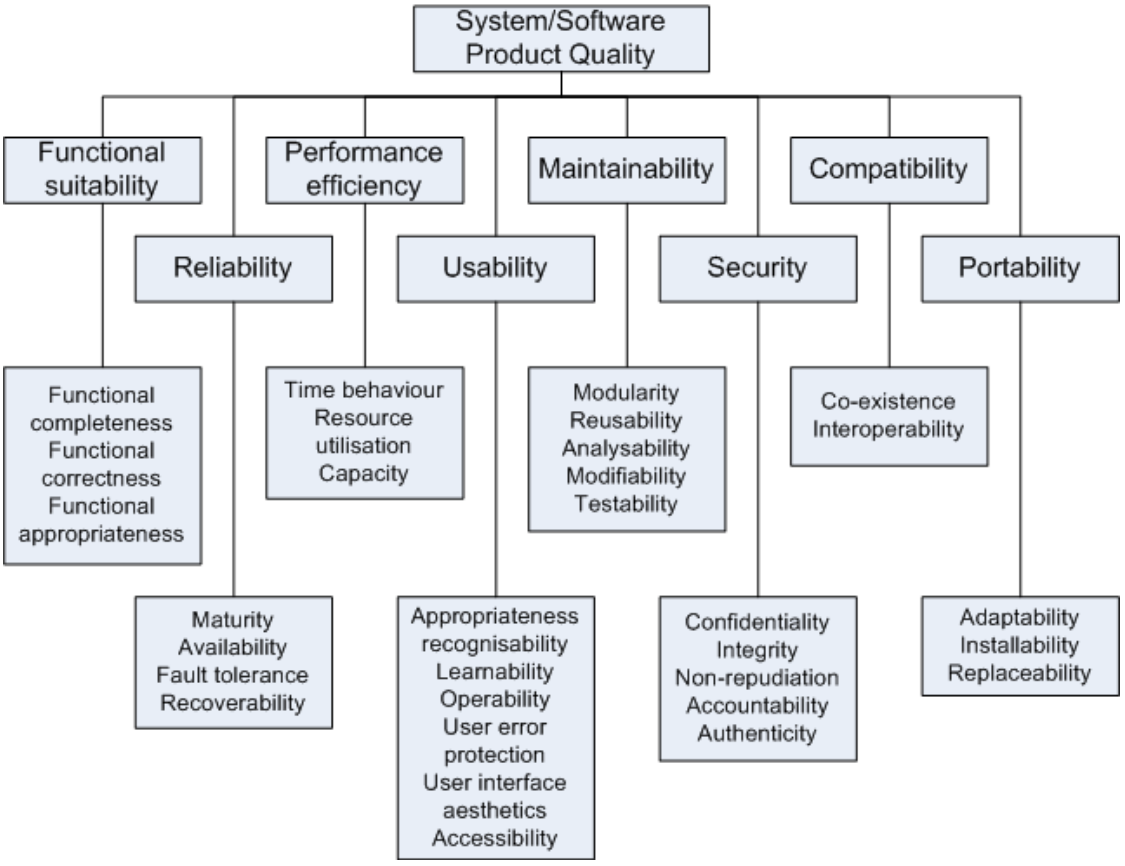

Figure 2.4: Quality model in ISO 25010

(WCAG) of W3C [WCAG]. It is a collection of guidelines of Accessibility, a guideline of the World Wide Web Consortium (W3C) organization. The current version WCAG 2.0 consists of 14 guidelines which are general principles of accessible design in 4 categories: Perceivable, Operable, Understandable and Robust. Each guideline consists of smaller guidelines. WCAG was published in December 2008 and became an ISO standard: ISO/IEC 40500:2012 in October 2012.

The conformance of WCAG is defined according to three levels: A, AA and AAA; corresponding to three levels of priority: must, should and may. The higher the level, the more restraining it becomes on design. Website is evaluated based on these three levels. If a website does not satisfy level A, it will be impossible for one or more groups to access the Web content; if it does not satisfy level AA, some groups will find it difficult to access the Web content; if it satisfies level AAA, it is easier for some groups to access the Web

content.

Standard of Usability

Another standard from ISO addresses usability in term of Human-Computer Interaction (HCI). As explained by [Bevan 2001], standards related to usability can be categorized as primarily concerned with the following.

- Use of the product (effectiveness, efficiency and satisfaction in a particular context of use).
- User interface and interaction.
- Process used to develop the product.
- Capability of an organization to apply user-centred design.

Other standards of Web usability are ISO 9241-151 and guidelines of US Department of Health and Human Services (HHS). HHS contains 207 guidelines for effective web design and usability for information-oriented sites, while ISO 9241-151 has 141 recommendations for user-centered design of web user interfaces [Bevan and Spinhof 2007].

ISO 9126 standard and WCAG standard have tools for demonstrating and applicating them. These tools will be described in Section 2.2.3.

2.2.2 Website quality models

According to [Kappel et al. 2004], a web application is a software system based on technologies and standards of the World Wide Web Consortium (W3C) that provides web specific resources such as content and services through a user interface, the web browser. Web applications have application-related, usage-related, development-related and evolution-related characteristics. In application-related characteristics, when developing web applications, one has to consider not only functionality but equally address content, hypertext and presentation aspects. We present below the main web quality models, which are (i) WQM; (ii) 2QCV3G; (iii) Signore's model; (iv) WebQEM; (v) Web QModel; and (vi) WAQE; and we do a brief synthesis and comparison of these models.

2.2.2.1 WQM

[Ruiz et al. 2003] proposed a Web Quality Model (WQM) which structures the characteristics according to three dimensions: features, quality characteristics, and life cycle processes. Quality characteristics have all 6 characteristics of ISO 9126 and also extended characteristics of Quint2 model. Life cycle processes, which are based on the ISO 12207-1 standard, contains development, exploitation and maintenance. Features have functions, content and infrastructure & environment. The model can be used for classifying existing metrics, characterizing research works and assessing web quality. It is the previous work of [Calero et al. 2005]. Some details in the model are smoothed in [Calero et al. 2005]. For example "Features" were changed to "Web Features" and Life-cycle processes are added more phases.

2.2.2.2 2QCV3Q

[Mich et al. 2003] proposed a conceptual model called 2QCV3Q to evaluate website quality based on seven dimensions: who, what, why, where, when, how, and with what means and devices. These dimensions correspond to Identity, Content, Services, Location, Management, Usability and Feasibility. The objective of Identity is to increase the users' trust in the site owner. Evaluating Content means how well the site covers its domain in terms of site owner and user requirements. Evaluating Services is also evaluating site functions from both owner and user viewpoints. Besides the adequacy of the functions, their correctness and security and the secure use of personal information are evaluated. Location concerns both site's reachability and user's ability to interact with the host and other users. Website management involves updating the information it provides. Usability concerns all aspects that enable site use in terms of cost, time or cognitive effort. The last dimension, Feasibility, is essential to website development (project management).

2.2.2.3 Signore's model

A comprehensive quality model is proposed by [Signore 2005]. It considers five dimensions. They are correctness, presentation, content, navigation, and interaction. Each dimension has smaller features. We can show that this model also has three main web

features: presentation, content and navigation. Moreover, they have correctness, a merely technical aspect and interaction, an aspect showing the relation between users and system.

2.2.2.4 WebQEM

Let us mention also QEM [Olsina et al. 2001], WebQEM [Santos 1999], Web-QModel [Cimino and Micali 2008] and WAQE [Mavromoustakos and Andreou 2007], etc. This list is not exhaustive. It only illustrates the proliferation of such models, due to the recent interest of researchers in this specific field of software quality.

WebQEM [Santos 1999] has six major steps in its methodology: selecting a site or a set of competitive sites to evaluate or compare; specifying goals and the user viewpoint; defining the website quality characteristics and attribute requirement tree; defining criterion function for each attribute and applying attribute measurement; aggregating elementary preferences to yield the global quality preference; analyzing, assessing, and comparing partial and global quality preferences.

2.2.2.5 Web QModel

Web-QModel [Cimino and Micali 2008] proposes two steps: at first the authors collect all attributes of all models, then they separate them in six groups based on their semantic meanings, assembled as Interface Communication, Content, Navigation, Management and Accessibility, Interactivity, and Accessibility for people with disabilities.

Attributes in groups are classified in three levels: Basic (Q), Normal (QQ) and Exciting (QQQ). It distinguishes attributes based on their importance in a good quality website design.

2.2.2.6 WAQE

WAQE [Mavromoustakos and Andreou 2007] proposed a model which has two axons: an axon for user and another for developer or expert. They also base on ISO 9126 for building their models. Five characteristics of ISO were used: functionality, efficiency, usability, reliability and maintainability (they missed portability). They use a numerical value from 1 to 5 to evaluate each factor and also the importance of each factor. Each

factor is calculated based on users' review and experts' review.

2.2.2.7 Synthesis and comparison

We can show main domains of all models (Table 2.1). This table contains details of models and compares their functions. We can also view the existence of automatic / semi-automatic tools supporting the models.

As we see in table 2.1, web features: content, navigation and presentation are presented in four models. Two models do not have all these features (WebQEM and WAQE). However WAQE also has Navigability as one of its characteristics. ISO 9126 is also frequently used. However these models do not use all of its characteristics. WQM uses all six characteristics, while WebQEM and WAQE use respectively four and five characteristics.

For automatic and semi-automatic tools, two cases exist: specific tool and other tool. Specific tools are developed by authors for illustrating their approaches. Other tools are used by authors for evaluating quality from their models. Only WebQEM has a specific tool. Three other models use other tools such as Web site watchers and validators: Astra SiteManager, Linkbot, Bobby [Mich et al. 2003] or some automated tools which are presented in the next section. WAQE uses opinions from users and developers, so they do not use tools.

We presented comprehensive set of web quality models. However, they present some limitations because they focus mainly on the application of software quality to the specific artifact of web application.

Table 2.1: Web model comparisons

Models / Functions	WebQEM	WQM	2QCV3Q	Signore	WAQE	WebQModel
Content		x	x	x		x
Navigation		x		x		x
Presentation		x		x		x
Identity			x			
Services			x			
Location			x			
Management			x			x
Feasibility			x			
Correctness			x	x		
Interaction				x		x
Accessibility						x
ISO 9126						
Usability	x	x	x		x	
Functionality	x	x			x	
Reliability	x	x			x	
Efficiency	x	x			x	
Maintainability		x			x	
Portability		x				
Auto / semi-auto tool						
Specific tool	x					
Other tool		x	x	x		

2.2.3 Quality evaluation methods and tools

In this section, we summarize the literature on methods and tools for website quality evaluation. The contents of this section are (i) website quality evaluation methods; and (ii) web quality evaluation tools.

2.2.3.1 Website quality evaluation methods

Distinctions are made between user-focused and expert-focused evaluation methods. Some methods are focused on certain quality characteristics, such as [Blas et al. 2002] which addresses the usability. It combines the inspection by an expert and empirical testing through panels of users. [Elling 2012] presents five studies comparing user-focused evaluation methods. It compares the methods according to the role of users in the evalu-

ation (non-use or in-use), the context of the evaluation (real-life or laboratory). They can use very different techniques, such as questionnaire, eye-tracking methods, etc.

Questionnaires are also used by other authors. The study described in [Mavromoustakos and Andreou 2007] gives users and developers or experts a questionnaire for evaluating from 1 to 5. [Dragulanescu 2002] gives users questions about 8 evaluation criteria: accuracy, authority, coverage, currentness, density, interactivity, objectivity and promptness. However how we can achieve the website quality is not shown clearly in this study.

Some authors proposed interesting approaches for evaluating quality. The study of Dominic [Dominic and Jati 2011] is based on Analytical Hierarchy Process (AHP) aggregating website quality metrics value. AHP is a popular model to aggregate multiple criteria for decision-making. In this study they chose 11 criteria to evaluate the quality of a website, namely: load time, response time, page rank, frequency of update, traffic, design optimization, size, number of items, accessibility error, markup validation and broken link. For each criterion, the best result of the sites that they reviewed is 1, and the worst result is 0, other results lie between 0 and 1. Rank of a site is achieved from a weighted score of that site.

[Rekik and Kallel 2011] used fuzzy sets to evaluate website quality. It is also based on Multiple Criteria Decision Making. The method has three steps: first, a user selects and evaluates criteria for a website with the evaluation tools; second, the measured criteria values are inputs of the fuzzy system to perform fuzzy computation; third, the website was ranked. A probabilistic approach was proposed by [Malak et al. 2010].

2.2.3.2 Web quality evaluation tools

Besides methods, authors also developed tools for demonstrating and operationalizing their approaches. Various tools for evaluating the quality of web applications have been developed. For example the tool described in [Malak et al. 2010] is composed of several modules: a measurement module using static and dynamic analysis, a probability function generation, and a Bayes network edition. [Fernandes et al. 2012] also developed a tool named QualWeb Evaluator for evaluating the accessibility criterion.

A tool evaluating the usability was proposed in [Ivory 2001a]. It combines a HTML parser, a browser emulator, a site crawler tool, a metrics computation tool, and an analysis tool. [Saba et al. 2006] described a tool for testing reliability of web applications. It emulates an unlimited number of users accessing a web application for testing its fault tolerance. An open source tool was proposed by [Guillemot and König 2006] to automate web applications testing.

The tool of [Rekik and Kallel 2011] is named Fuzz-Web. It is based on Matlab fuzzy logic toolbox, a tool for the design of intelligent systems.

Some other tools can be found in the Internet. Accessibility is the sub-characteristic which is best covered by these tools. [TAW] is one of these tools. The outputs of TAW are Problems and Warnings classified according to four categories: Perceivable, Operable, Understandable and Robust. [WebAIM] provides the user with reports using icons, structures and texts, making it easy to find errors in a website. However these reports are not as much detailed as TAW. [PowerMapper] provides a collection of tools for measuring the quality of websites, as broken links, spelling errors, browser compatibility, accessibility, web standards validation, navigation. It measures accessibility based on WCAG standards (Web Content Accessibility, W3C organization). Recently the latter became an ISO standard: ISO/IEC 40500:2012. [LinkPopularity] measures link popularity of a site, i.e. the total number of websites that link to this site. In fact, the free version of this tool uses results from three large search engines, i.e. Google, Bing and Yahoo. Another tool supporting evaluation website quality is Xenu. This freeware checks websites for broken hyperlinks. Based on <a> tags, it follows links to other pages and checks whether they live. It has support for Secure Socket Layer (SSL) websites. Let us mention also [WebQual] which assesses the usability, information, and service interaction quality of Internet web-sites, particularly those offering e-commerce facilities.

This is only a partial list of all tools available on website quality evaluation. However, let us notice that all these tools measure external quality characteristics, when web applications are in use. To the best of our knowledge, there is no specific tool to measure external quality of websites during their design and development, nor can we find tools allowing us to predict website quality during the early phases of their life cycle.

2.2.3.3 Synthesis and comparison

The tools are dedicated to the evaluation of how web applications perform and to the detection of defects. It shows that not all tools can cover all requirements of users and developers.

Through a survey of the methods and tools of the previous authors, we found that they mainly (i) refer to the usability of web applications; (ii) measure the performance of the web application. However, the quality of web applications as the impact of design process and development process has not been demonstrated. That is the task that needs to be handled. In the next chapters, we will propose an approach to fill this gap.

2.3 Web engineering: a state of the art

In this section, we will review some web engineering methods.

Companies need web applications which are delivered on time, within budget, have a high level of quality and are easy to maintain. To develop such applications, web development teams need to use sound methodologies, systematic techniques, quality assurance, rigorous, disciplined and repeatable processes, better tools, and baselines. Web engineering aims to meet such needs [Ginige and Murugesan 2001]. Web Engineering involves the use of scientific, engineering, and management principles and systematic approaches with the aim of successfully developing, deploying, and maintaining high quality web-based systems and applications [Murugesan and Deshpande 2001].

In the domain of web engineering, the researchers proposed many methods and approaches for constructing and representing web applications. When developing a web application, it is necessary to specify not only functionality but equally address content, navigation and presentation aspects [Grünbacher et al. 2004].

Methods of web engineering can be divided in two categories: traditional and model driven.

2.3.1 Traditional methods

The first web engineering methods were derived from traditional software engineering methods such as, in chronological order, Object-Oriented Hypermedia Design Method (OOHDM), Website Design Method (WSDM), Web Modeling Language (WebML), Object Oriented Hypermedia (OO-H) and UWL-based Web Engineering (UWE).

2.3.1.1 Website Design Method

Website Design Method (WSDM) was introduced by De Troyer and Leune in 1998 [Troyer and Leune 1998]. This method consists of four phases: User Modeling, Conceptual Design, Implementation Design and actual Implementation. User Modeling phase consists of two sub-phases: User Classification and User Class Description. The Conceptual Design phase also consists of two sub-phases: Object Modeling and Navigational Design (Fig. 2.5).

In the first sub-phase of User Modeling, they identify and classify users of the website. They create user classes which are subsets of all the potential users and are collections of users which have same information requirements. The same person may be in different user classes depending on the different roles he/she plays in the system. In the second sub-phase of User Modeling, user classes are analyzed. If within a user class, users have different characteristics with different usability requirements, the user class is divided in two so-called perspectives.

During the sub-phase Object Modeling of the phase Conceptual Design, the information requirements of the different user classes and their perspectives are formally described. In the Navigational Design sub-phase, designers describe how the different users can navigate through the website. They also build a Conceptual Navigational Model in this sub-phase. To arrive at a Navigational Model, a navigation track is constructed for each perspective. Each navigation track consists of three layers: context layer for connecting the different navigation tracks, navigation layer for providing different ways to access the information and information layer.

In the Implementation phase the "look and feel" of the website is designed. The aim is

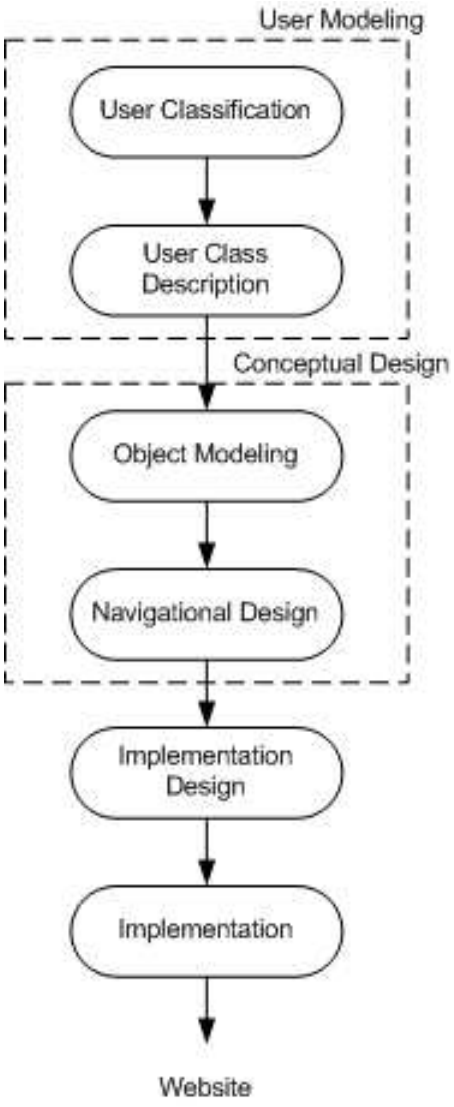

Figure 2.5: Overview of the WSDM phases [Troyer and Leune 1998]

to create a consistent, pleasant and efficient look and feel for the conceptual design made in the previous phase. The result of this phase is an Implementation Model.

The last phase is the actual development of the website using the chosen implementation. This phase can be largely automated using available tools and environments for assisting in HTML implementations.

2.3.1.2 Web Modeling Language

Web Modeling Language (WebML) [Ceri et al. 2000] is a high level specification language for designing data-intensive Web applications at the conceptual level. This method presents a web application in four models:

- *Structural model* expresses the data content of the site in terms of the relevant entities and relationships. It is compatible with classical notations like E/R model or UML class diagrams.
- *Hypertext model* describes hypertext that can be published in the site. Each different hypertext defines a site-view which consists of two sub-models:
  - *Composition model* specifies which pages compose the hypertext and which content units make up a page.
  - *Navigation model* expresses how pages and content units are linked to form the hypertext.
- *Presentation model* expresses the layout and graphic appearance of pages, independently of the output device and of the rendition language.
- *Personalization model* models explicitly users and user groups as predefined entities called User and Group.

2.3.1.3 Object Oriented Hypermedia

The Object Oriented Hypermedia (OO-H) Method [Gómez et al. 2000] is an extension of the OO-Method conceptual modeling approach to address the particularities associated with the design of web interfaces. It is based on the OO-Method class diagram, which captures the statics of the system. This method adds several navigation and interface constructs to the OO-Method conceptual model, which defines the semantics suitable for capturing the specific functionality of web application interfaces. For each type of users, it captures the information which each type of user can access and the navigation paths

from one information view to another. An interface execution model is provided in order to determine the way the conceptual model is implemented in a given developmental environment. It also defines how interface and application modules have to communicate with each other.

The navigation model is captured by means of the Navigation Access Diagram (NAD). The main components of the NAD are navigation classes, navigation targets, navigation links and collections. Each of these constructs addresses the navigation model from a different dimension.

Navigation classes are based on the classes identified during the conceptual modeling phase. They are represented in three areas: name of the class, attributes relevant to the considered user and view, services capable of being invoked by the actual user of the NAD. A navigation target is a set of navigation classes which together provide the user with a coherent view of the system. A navigation target is associated to each user's navigation requirement. The definition of navigation target implicitly captures the 'navigation context pattern': the same nodes might appear in several navigation targets as long as they do not represent the same information requirement, and in each navigation target its layout and navigation mode might be different. Navigation links have four types: requirement link, service link, internal link and traversal link. The navigation links are always directed. It means that if there is a need to navigate in both senses, two links must be explicitly or implicitly specified. Collection is a structure, hierarchical or not, which abstracts some concepts regarding both external and internal navigation. It is useful for limiting the interaction options between user and application, thus improving the usability of the system.

The execution model provides the method with the representation details of the interface conceptual model for a target development environment. OO-H Method is centered on defining how to implement the interface level information associated to web environments. All the concepts represented in the NAD are stored in an object repository and, from there, a default presentation diagram can be generated.

2.3.1.4 UWL-based Web Engineering

UWL-based Web Engineering (UWE) [Koch and Kraus 2002, 2003] methodology covers the whole life-cycle of a web application development, proposing an object-oriented and iterative approach based on the Unified Software Development Process. The main focus of the UWE approach is the systematic design followed by a semi-automatic generation of web applications. The notation used for design is a "lightweight" UML profile [Koch and Kraus 2002]. A UML profile is a UML extension based on the extension mechanisms defined by the UML itself with the advantage of using a standard notation that can be easily supported by tools and that does not impact the interchange formats. The UWE profile includes stereotypes and tagged values defined for the modeling elements needed to model the different aspects of web applications, such as navigation, presentation, user, task and adaptation aspects.

Requirements can be specified in UWE with use cases. UML use case diagrams are built with two main UML modeling elements, namely use cases and actors and use case relationships between these elements, such as associations between an actor and a use case and dependencies "'includes"' and "'extends"' between use cases.

UWE defines navigation and presentation models which are supplemented by other UML diagrams and UML modeling elements. Navigation modeling of web applications comprises the construction of two navigation models, the navigation space model and the navigation structure model. The former specifies which objects can be visited by navigation through the application. It is a model at the analysis level. The latter defines how these objects are reached. It is a model at the design level. The navigation models are represented by stereotyped class diagrams.

The presentation model is based on a particular form of a class diagram. The presentation model describes where and how navigation objects and access primitives will be presented to the user. Presentation design supports the transformation of the navigation structure model in a set of models that show the static location of the objects visible to the user.

2.3.2 Model driven methods

In classical software engineering, the model-driven paradigm was proposed to facilitate the handling of the whole design and development process. Software development consists in producing several models going from assisstant to concrete. Web engineering is a specific domain in which Model-Driven Engineering (MDE) can be usefully applied [Vallecillo et al. 2007]. The application of MDE in Web engineering is called Model-Driven Web Engineering (MDWE). MDWE is the application of the model-driven paradigm to the domain of Web software development where it is particularly helpful because of the continuous evolution of Web technologies and platforms [Kraus et al. 2007].

This section presents some model driven web engineering methods: Web Software Architecture (WebSA), UML-based Web Engineering (UWE), Web Modeling Language (WebML) and WebRatio, Object-Oriented Hypermedia Method for RIA (OOH4RIA), Navigational Development Techniques (NDT). Some methods in this list are derived from traditional methods.

2.3.2.1 Web Software Architecture

Web Software Architecture (WebSA) [Meliá et al. 2003; Beigbeder and Cachero 2004] is a web model-driven approach that is based on the standard Model Driven Architecture (MDA). Its main target is to cover all the phases of the web application development and to contribute to cover the gap existing between traditional web design models and their implementation. It defines an instance of the MDA Development Process for the web application domain. In order to define a web application system, they propose a web application view model that is made up of eight views, grouped into three viewpoints: requirements, functional and architectural viewpoints.

The requirements viewpoint consists of two views: Functional Requirements and Non-Functional Requirements. Functional ViewPoint, based on functional requirements, consists of four views: Conceptual, Process, Navigational and Presentation View. The Architectural Viewpoint, built on non-functional requirements, is a main contribution of WebSA. This viewpoint includes a logical architectural view that gathers the set of logical compo-

nents (subsystems, modules and/or software components) and the relationships among them. It also includes a physical architecture view that describes the physical components that integrate the lower level specification of the application (clients, servers, networks, etc.)

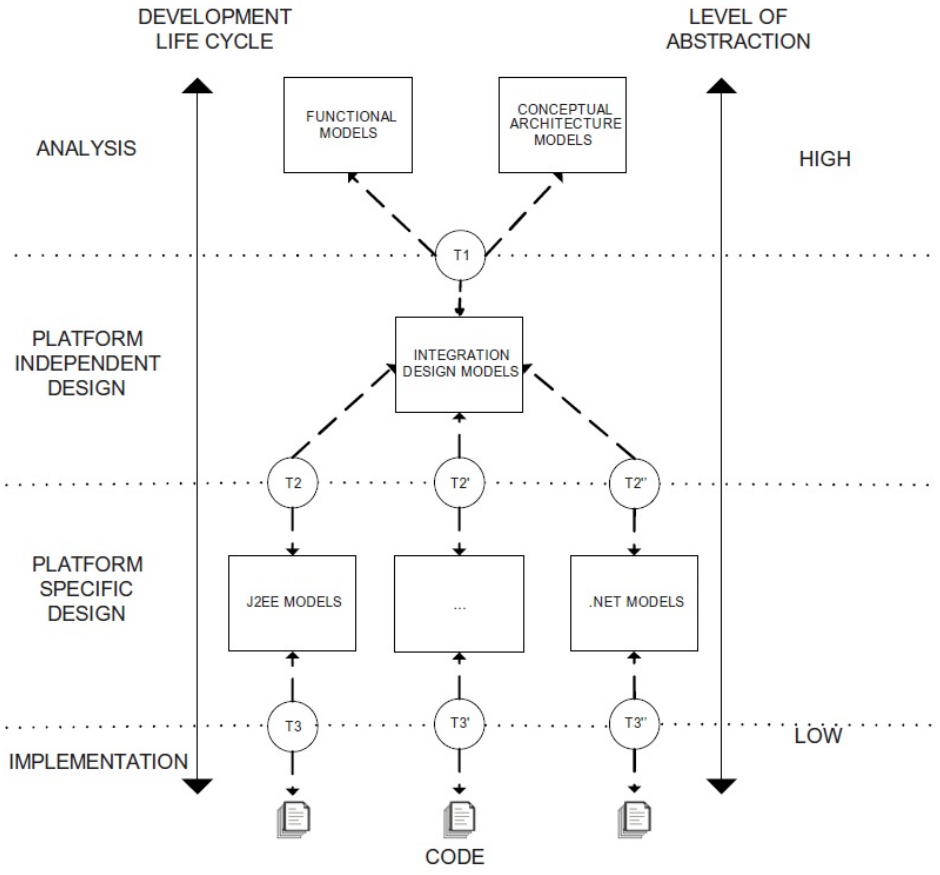

Figure 2.6: WebSA Web development process [Beigbeder and Cachero 2004]

WebSA Development Process is based on the MDA Development Process (Fig. 2.6). It establishes a correspondence between its web-related artifacts and the MDA artifacts. This Process has four phases of the development life cycle: analysis; platform independent design, in which platform independent models are built; platform specific design, in which platform specific models are built; implementation.

In the analysis phase, the Web application specification is divided horizontally into two viewpoints. The functional-perspective models reflect the functional analysis, while the ar-

chitectural models define the system architecture. Both models are platform independent models (PIM) in the context of an MDA framework. The PIM-to-PIM transformation of these models provides a set of artifacts in which the conceptual elements of the analysis phase are mapped to concrete elements where the information about functionality and architecture is integrated. Output models of this phase are transformed into Platform Specific Models (PSM) by PIM-to-PSM transformations, becoming the specification of web application for a given form. In the last phase (implementation) a PSM-to-code transformation which be implemented by means of templates is executed. WebSA formalizes the three transformations (PIM-to-PIM, PIM-to-PSM and PSM-to-code) by means of QVT (Query View Transformation).

2.3.2.2 UWL-based Web Engineering

The authors of **UWE** also develop a model driven version [Kraus et al. 2007]. The UWE approach separates the concerns (content, navigation, presentation, process...). Applying the MDA principles, the UWE approach proposes to build a set of Computation Independent Models (CIM), PIMs, and PSMs as results of the analysis, design and implementation phases of the model-driven process. It uses UML extension (UML profile) for web specific notation.

The aim of the analysis phase is to gather a stable set of requirements. The functional requirements are captured by means of the requirements model. The requirements model comprises specialized use cases and a class model for the Web application.

The design phase consists of constructing series of models for the content, navigation, process, presentation and adaptivity aspects at a platform independent level. Transformations implement the systematic construction of dependent models by generating default models, which then can be refined by the designer. Finally, the design models are transformed to the platform specific implementation.

UWE presents five models:

- Requirements model
- Content model

- Navigation model
- Presentation model
- Process model

2.3.2.3 WebML

WebML [Brambilla et al. 2008] is a method and also a visual language for specifying the content structure of a web application and the organization and presentation of such content in a hypertext.

WebML development process consists of different phases (Fig. 2.7). As other normal processes, it has also Requirement Analysis, Conceptual Modeling, Implementation, Testing and Evaluation, Deployment, Maintenance and Evolution.

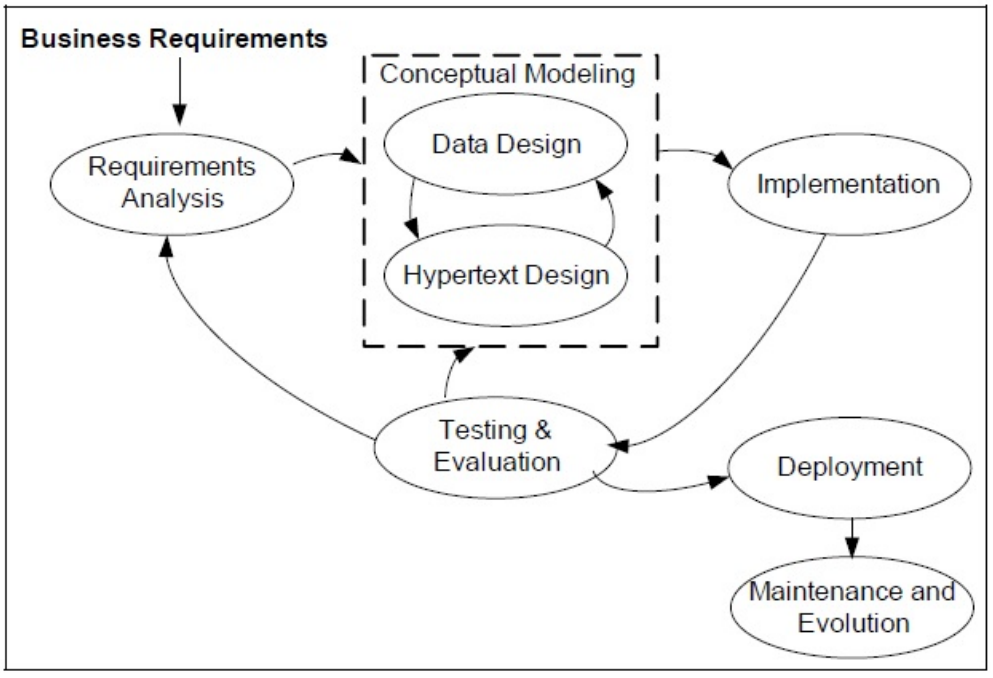

Figure 2.7: Phases in the WebML development process [Brambilla et al. 2008]

Requirement analysis focuses on collecting information about the application domain and the expected functions. The main results of this phase are the identification of the groups of users; the specification of functional requirements, the identification of core

information objects, the decomposition of the Web application into site views.

According to the WebML approach, conceptual modeling consists of data design and hypertext design. Two designs provide two corresponding models: data model and hypertext model.

The phases following conceptual modeling consist of implementing the application, testing and evaluating it in order to improve its internal and external quality, deploying it on top of a selected architecture, and maintaining and possibly evolving the application once it has been deployed. They are well supported by WebML. The model-driven approach benefits the systematic testing of applications, thanks to availability of the conceptual model and the model transformation approach to code generation. Model-driven development also fosters innovative techniques for quality assessment. In a model-driven process, maintenance and evolution also benefit from the existence of a conceptual model of the application.

WebRatio is an integrated development environment supporting the modeling of applications with WebML and their implementation with model-driven code generators. It is a commercial tool for designing and implementing web applications. The architecture of WebRatio consists of two layers: a design layer and a run-time layer. The design layer includes a graphical user interface for data and hypertext design.

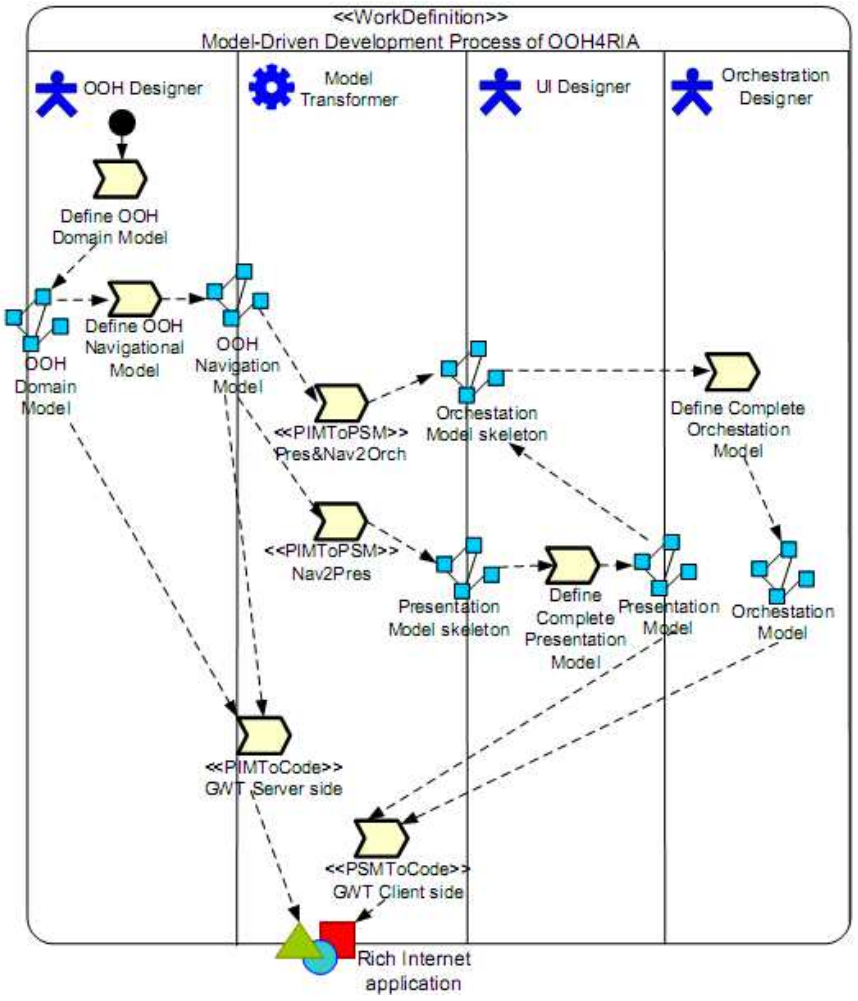

Figure 2.8: Development process of OOH4RIA [Meliá et al. 2008]

2.3.2.4 Object-Oriented Hypermedia for rich Internet Applications

Object-Oriented Hypermedia Method for Rich Internet Applications (OOH4RIA) [Meliá et al. 2008] is an approach for developing Rich Internet Application (RIA). It is based on the MDE paradigm that proposes a complete development process based on a set of models and transformations allowing to obtain the implementation of RIAs (Fig. 2.8).

The OOH method is based on the object-oriented paradigm which provides the designer with the semantics and notations necessary for the development of the traditional Web applications. OOH defines a set of models: the domain model, the navigation model, and the presentation model. Its authors developed a support tool as an Eclipse plugin.

2.3.2.5 Navigational Development Techniques

Navigational Development Techniques (NDT) [Cuaresma and Aragón 2008] is a methodological approach which deals with requirements in web environments. NDT was proposed in order to support the requirements engineering and the analysis phase of web systems and is based on the MDE paradigm (Fig. 2.9). NDT consists of mainly only two phases: requirement and analysis. NDT development process can be defined as a bottom-up process. After specifying requirements, this process defines three models: content model, navigational model and abstract interface model. Content model which expresses the static view of the system is a class diagram. Navigational model is a model which shows how users can navigate through the systems.

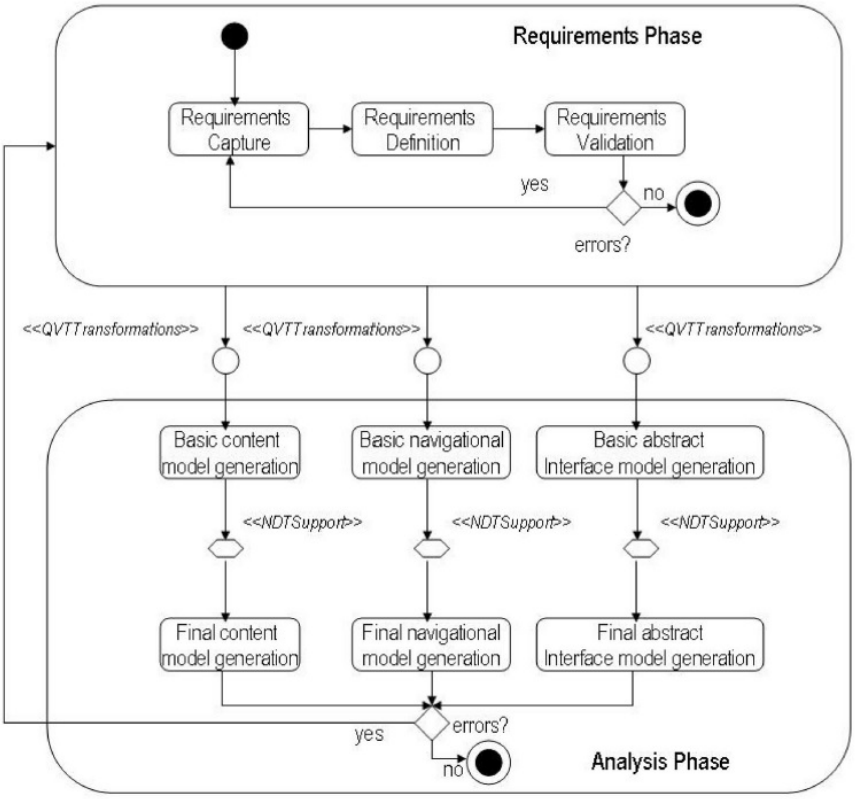

Figure 2.9: NDT development process [Cuaresma and Aragón 2008]

NDT executes model-driven transformations by using QVT Transformations. These transformations can be grouped in three transformations:

- *Requirements2Content*: This transformation allows the generation of the basic content model from the requirements model.
- *Requirements2Navigational*: This transformation allows the generation of the basic navigational model from the requirements model
- *Requirements2Prototypes*: This transformation allows the generation of the basic abstract interface model from the requirements definition.

NDT and its definition based on metamodels and transformations can be translated into a tool, named NDT-Tool. This tool not only supports the definition of models or results, like other tools in web engineering, but also the complete life cycle of NDT, it lets patterns and model be defined, and it guarantees the traceability using model-driven support. This tool is developed with J2EE using the Model-View-Controller pattern.

We summarize the main activities covered by the methods reviewed above in tables 2.2 and 2.3.

In table 2.2, we list the phases of the life cycle which are supported by this method. As we can see in this table, except NDT which supports only requirement analysis, the other methods have design and implementation phases. Except WebSA, the methods have support tools, based on one platform such as MagicUWE of UWE based on MagicDraw or OOH4RIA tool based on Eclipse IDE.

Table 2.2: Phases of method

Methods / Phases	WebSA	UWE	WebML	OOH 4RIA	NDT
Requirement analysis	x	x	x	x	x
Design	x	x	x	x	
Implementation	x	x	x	x	
Test			x		
Support tool		x	x	x	x

In table 2.3 we list models which are supported by methods. As we can see in this table, except webSA which has different models, the other methods support basic models of Web Application: Content (or Domain), Navigation and Presentation models. Some specific

models are Process model of UWE, Personalization model of WebML, Orchestration model of OOH4RIA.

Table 2.3: functionality web metrics

Methods / Models	WebSA	UWE	WebML	OOH 4RIA	NDT
Requirements		x			x
Content/ Domain		x		x	x
Data			x		
Navigation		x	x	x	x
Presentation		x	x	x	
Process		x			
Interface					x
Orchestration				x	
Personalization			x		
Subsystem	x				
Web Component Configuration	x				
Web Component Integration	x				

There are many methods of web engineering: traditional methods and model-driven methods. They support users in building web applications from the beginning to the end. But they do not ensure the quality of their products (web applications) and the processes are not guided by quality. This thesis is a step forward in defining web application quality and providing designers with guidelines and tools.

This chapter presents the definitions related to thesis topic. We also present an overview of the quality of web applications and an overview of the methods and tools of web engineering. From surveying the works of the preceding authors, we define tasks for the following chapters.

Chapter 3

Problem statements and Solution proposals

Chapter 1 presented a global view of problems in web application quality. In this chapter, we will discuss more deeply the problems addressed in this thesis and propose an overview of our solution.

3.1 Difficulties in building web applications

From 1990s, Internet was introduced to the public. Since that moment, Internet has developed rapidly and largely. More than 3.9 billion people have used the services of the Internet as of June 2017 [Internet World Stats]. World Wide Web or the Web in abbreviation is a system of hypertext documents accessed via Internet. These hypertext documents are assembled to websites. Number of websites exist and have been created in World Wide Web now. The size of World Wide Web is estimated around 14 billion web pages [Size]. Besides websites, web applications are also developed in the Internet. A web application is an application which uses a web browser as a client. It is stored on web servers and uses tools to deliver experiences beyond the standard web pages or web web form. Due to the development of Internet now, web applications become popular. Users can use web applications instead of traditional applications in some cases. For example, users now use Google Docs for composing and editing documents, cloud services for storing data, instead of saving it in the hard disk of their computers.

However, the quality of web applications does not grow as well as their rapid development. Many developers did not realize that web applications have specific characteristics and requirements, considerably different from that of traditional software. The consequences are that nearly 25% of web projects failed [Krigsman 2008].

According to Krigsman, the three principal reasons for failure rate of web applications are: (i) Changing requirements, (ii) Inconsistent stakeholder demands, and (iii) Insufficient time or budget.

Many websites are created each day. Some advantages are that they adapt users' requirements, their contents are diversified. However, some websites and web applications are produced by amateurs. One disadvantage is their quality. These products are hard to maintain and develop in the future. An efficient way to evaluate the quality of websites and web applications is required.

However, building web applications is not an easy task, for some reasons. When listing major differences between web applications and conventional software, [Deshpande et al. 2002] explained the following difficulties of building web applications.

1. *Compressed development schedules*: the time available for completing the project is not as long as for similar software projects. There is time pressure in the evolution of web applications. In many cases, it is not possible to fully specify the requirements before implementing;

2. *Constant evolution with shortened revision cycles*: web application development needs to meet requirements of constant evolution and revision cycles is shortened;

3. *Insufficient requirement specifications*;

4. *Lack of accepted testing processes*;

5. *Minimal management support*;

6. *Criticality of performance*;

7. *Evolving standards to which Web applications should or must comply, depending on the specific circumstances*;

8. *Variety of backgrounds of developers*: Early Web developers came from non-software engineering background;

9. *Rapidly evolving implementation environment, encompassing various hardward platforms.*

Web application quality is an important topic. Classical software engineering methods are not sufficient to improve their quality since web applications are specific softwares. Specific web application engineering methods also are insufficient to guarantee a good level of quality. In this thesis, we propose to provide web designers with a set of artifacts in order to define, measure, and improve their web application quality. To this end, we propose an approach based on an iterative cycle.

3.2 Iterative cycle approach

Quality engineering has already a long history. It started with quality control, quality improvement, and quality management system. After a period where total quality management was the main objective of all quality specialists, researchers and practitioners all agreed on the fact that continuous improvement was the best way to address quality issues [Davenport 1993].

In this stream, the PDCA (Plan - Do - Check - Act) Deming wheel is the dominant approach in project life cycle [Dale 2015]. Our approach is an iterative process adapted from PDCA. We first recall the main concepts of PDCA before describing the main steps of our approach.

PDCA or Deming cycle/wheel is an iterative four-step management method. It was proposed in the 1950s by Edward Deming in order to perform continuous quality management. Just as a circle has no end, the PDCA cycle should be repeated again and again for continuous improvement. The model is both widely applicable and easy to learn and to use. At each stage of this cycle, we use the experience to improve the next iteration.

The PDCA cycle is known not only by quality specialists but also by a large number of executives [Deming 1993]. This cycle is reprensented by a diagram to help the learner,

and to drive product or process improvement.

The changes are made below clockwise in the Do step. The results are examined in the Check step. And the change is either adopted or abandoned in the Act step. This leads to the start step, i.e., next Plan for change.

The four steps of PDCA are presented below (Fig. 3.1).

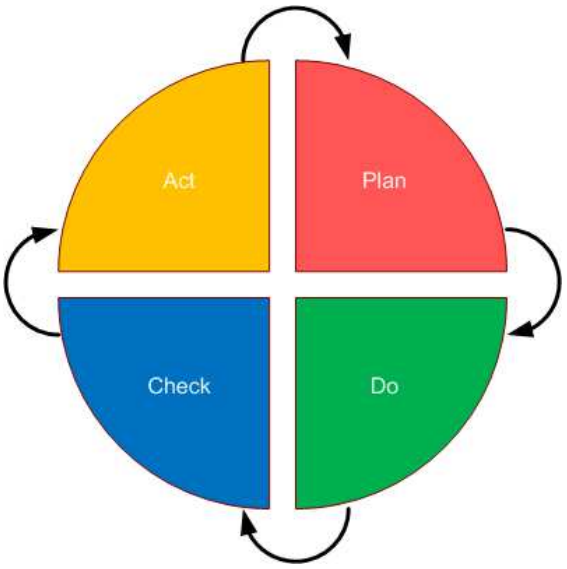

Figure 3.1: Four phases of PDCA (redrawn from Deming)

3.2.1 Plan

The first step is Plan. When we have an idea to improve a product or a process, we make a plan for a change in the first step. In this step we establish the objectives and processes necessary to deliver results in accordance with the expected output, target or goal. This is the initial state resulting in the preparation of a test. The complete cycle is based on this step. The precipitated start of a project may create unnecessary costs and frustrations. It tends to shorten this stage and to quickly move to the second step.

To prepare a plan, we can start by choosing among several suggestions. What will we check? What can be the result? We compare the different choices. What is the suggestion that seems most interesting to learn something or make a profit? The problem is how to achieve a realistic goal.

3.2.2 Do

The second step is Do. It means testing, preferably at a small scale, according to the choices taken in the first stage. We carry out the plan, document problems and unexpected observations, collect data for analyzing them.

3.2.3 Check

The third step is Check. We complete the analysis of the data, compare data to predictions and summarize what was learned. We study the results which were measured and evaluate whether they meet expectations defined in the Plan. If not, why were the desired results not obtained?

We look for deviation in implementation from the plan and also look for the appropriateness and completeness of the plan. Charting data can make this much easier to see trends over several PDCA cycles and to convert the collected data into information. Information is what we need for the next step "Act".

3.2.4 Act

The fourth step is Act. We see what changes are to be made and decide to go to next cycle or not. If the Check shows that the Plan which was implemented in Do is an improvement to the prior standard, then that becomes the new standard for how the organization should Act going forward. If the Check shows that the Plan that was implemented in Do is not an improvement, then the existing standard will remain in place. In either case, if the Check showed something different than expected (whether better or worse), then there is some more learning to be done... and that will suggest potential future PDCA cycles. The Act also involves making adjustments or corrective actions, but generally it would be against the philosophy of PDCA to propose and decide upon alternative changes without using a proper Plan phase, or to make them the new standard without going through Do and Check steps.

3.3 Solution overview

Inspired by PDCA, we propose a three-phase approach: (i) Defining, (ii) Measuring and (iii) Improving quality of web applications. The process is also an iterative cycle.

First, we define quality of web applications by defining factors and objects of quality in the first phase. Second we measure quality of web applications. Third, we provide guidelines for helping users improving their web applications (Fig. 3.2).

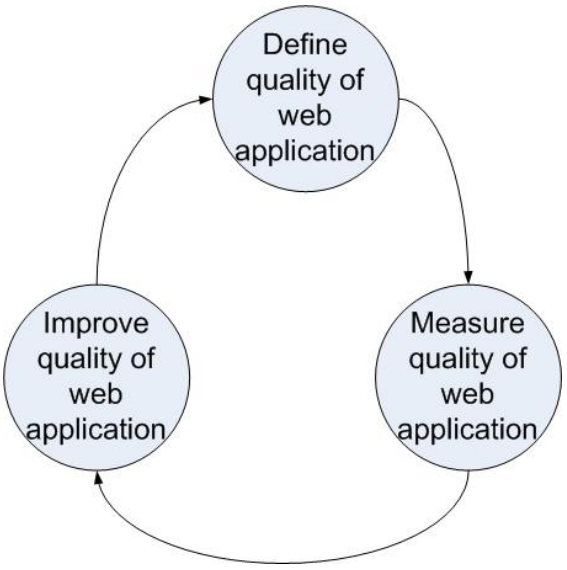

Figure 3.2: The three phases of our approach

3.3.1 Defining factors of quality

In this step of our method, we make a plan on how to improve the quality of a web application, and define factors of its quality. There are many factors which can influence the quality of an application. So we choose factors which satisfy the specific requirements of the problem at hand.

We determine the priority of factors and objects of quality. What is the most important factor?

Each application can be seen according to several aspects. Each view has some defects or disadvantages. So we focus on the characteristics we want to improve. We choose the metrics according to these characteristics.

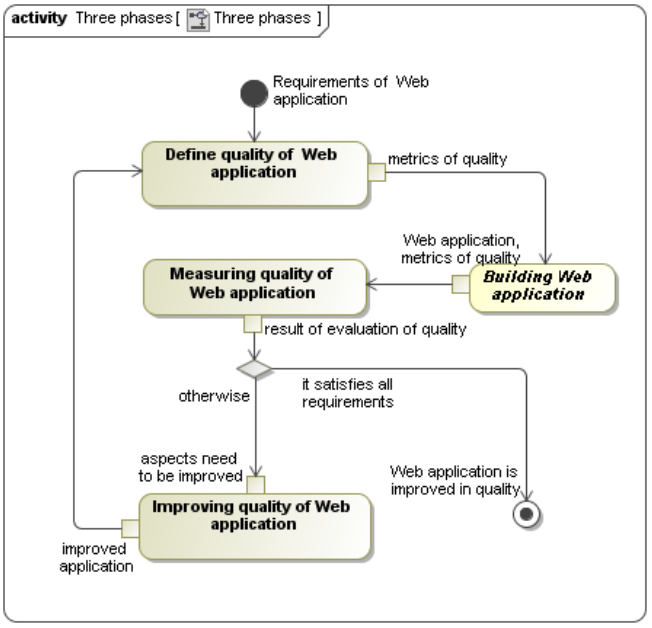

Figure 3.3: Process of our approach

3.3.2 Measuring the quality of web applications

The input of this phase is the design of an application or the application in itself. After the first phase, we have the required metrics for measuring web application quality. It is a measure of quality defined by the user (developer, customer...).

This phase contains two steps: building or updating the web application and evaluating its quality. Our work focuses on the second step: evaluating web application quality, because building web applications is not in the scope of our thesis.

After realizing this phase, we have a result which contains aspects or characteristics needing improvements.

3.3.3 Improving quality of web applications

In this phase, we propose to use guidelines in order to address the defects which we found in second phase. We identify guidelines that could help solving the specific issues of quality. We determine whether we continue. If the guidelines can solve problems identified, we can terminate the process. If not, it means that the guidelines are not enough for improving

quality as we expected, so we have to begin a new cycle. We repeat the three phases in an iterative way until all requirements are satisfied (Fig. 3.3).

3.4 Contribution of thesis

The contribution of our thesis is related to the three main phases of the process mentioned in Section 3.3.

- *A more complete, richer definition of quality of web application.* Quality of web application is not only seen as quality of software, but seen as quality of data and quality of specific web features. This approach assesses quality of web applications more globally, so it can measure quality more accurately. This work will be presented in Chapter 4.
- *A taxonomy of metrics for measuring quality of web applications.* We build a taxonomy of metrics which serves for evaluating quality. This taxonomy is also based on the three aspects of quality, so it can cover up most of quality of web applications in reality. This work will be presented in Chapter 5.
- *Guidelines for improving quality of web application.* We collected guidelines from the literature, adapted them for satisfying our requirements. After evaluating quality, the web designer chooses appropriate guidelines with the purpose of improving its quality. This work will be presented in Chapter 6.

In conclusion, in this chapter, we described our approach to define, measure and improve the quality of web applications in brief. It contains three phases corresponding to targets of our approach. In the next chapters, our contribution to each phase of our approach is described in detail.

Part II

Chapter 4

A framework for web application quality

4.1 Introduction

This thesis aims at proposing an approach to improve the quality of a web application. Web application is a complex artifact. It is a software. It also delivers information. Finally it presents some specific features that are used to assess its quality, for example its reputation. In this chapter, based on these characteristics and others, we propose a new definition of web application quality encompassing all these dimensions.

In the state of the art synthesized in Chapter 2, we have reviewed the different definitions of quality in general and of quality of web applications in particular. In this chapter, the goal is to provide a framework for evaluating the quality of a web application. This framework will allow the implementation of the first step of our approach, namely the definition of one or more objectives for the quality of an application. In the next chapter, we will study how this quality can be measured. Finally, in Chapter 6, we will see how it can be improved, resulting in a cyclical approach composed of three phases: defining the quality of a web application (Chapter 4), assessing quality (Chapter 5), and improving quality (Chapter 6).

There are many publications targeting the quality of web applications. They differ according to the authors' point of view and the objective. For the sake of generality, we did not wish to adopt a particular point of view, for example that of the developer, or

to address a specific type of web application, for example an electronic market place. On the contrary, we performed a two-by-two comparison of past approaches to finally classify them hierarchically.

Thus, in this chapter, we describe our research approach which consisted of: 1) the identification of relevant past approaches, 2) the selection of the most significant approaches, 3) their comparative analysis, 4) the classification of these approaches, and 5) the proposal for a framework summarizing those proposed in the literature.

In the next section, we provide a state of the art of frameworks for defining the quality of web applications. Then, we describe our method for comparing these approaches. Finally, we propose a hierarchical classification of these approaches and analyze the result of this classification by trying to identify the clusters obtained.

4.2 Survey of Literature

In this section, we synthesize the literature on frameworks for web application quality. There is no standard for web application quality. Existing standards are devoted to software quality or information quality or quality in general. Therefore, this chapter is a step forward for defining such a framework.

To identify all relevant past approaches, we conducted a keyword search via various search engines (Google Scholar, IEEE Xplore, Business Source Complete, ScienceDirect). The keywords used were: quality framework of a website, quality assessment framework for a website, quality framework of a web application, quality assessment framework for an application, quality assessment approach for a website, approach to evaluating the quality of a web application. Then, using forward and backward snowballing techniques, we collected a set of articles. We have gone through these papers and selected those that actually provide a framework for defining and evaluating the quality of a web application, whether generic or specific. We finally obtained fourteen approaches (Table 4.1). We describe briefly below these approaches. Our objective is to select a subset of them that can effectively be used to gather all viewpoints of web application quality.

Table 4.1: Fourteen works of quality web application

N	Title	Reference
1	Measuring Web Application Quality with WebQEM	[Olsina and Rossi 2002]
2	Developing and validating an instrument for measuring user-perceived web quality	[Aladwani and Palvia 2002]
3	B2C e-commerce web site quality: an empirical examination	[Cao et al. 2005]
4	WebQual: An Instrument for Consumer Evaluation of Web Sites	[Loiacono et al. 2007]
5	Modeling web quality using a probabilistic approach: An empirical validation	[Malak et al. 2010]
6	Assessing the quality of web sites	[Hasan and Abuelrub 2011]
7	Web-Based Applications quality factors: A survey and a proposed conceptual model	[Nabil et al. 2011]
8	Web site quality evaluation in Higher Education Institutions	[Carlos and Rodrigues 2012]
9	Evaluating the perceived and estimated quality in use of Web 2.0 applications	[Orehovački et al. 2013]
10	Automated evaluation of website navigability: an empirical validation of multilevel quality models	[Vaucher et al. 2013]
11	Academic Information System Quality Measurement Using Quality Instrument: A Proposed Model	[Yuhana et al. 2014]
12	Formalizing and validating the web quality model for web source quality evaluation	[Zhao and Zhu 2014]
13	Metrics for Quality Assurance of Web based Applications	[Zia 2015]
14	A Framework for Quality Management of E-commerce Websites	[Kotian and Meshram 2017]

[Aladwani and Palvia 2002] is one of the oldest publications. They designed a quality model with three dimensions: technical adequacy, web content and web appearance, with a total of 55 characteristics (called items in this work).

[Cao et al. 2005] examined and integrated four sets of factors that capture e-commerce web site quality using the technology acceptance model (TAM) [DeLone and McLean 2003]: system quality, information quality, service quality, and attractiveness. A questionnaire survey was conducted to verify the measures of web site quality. Based on TAM, a framework was developed relating web site quality to customers' beliefs (perceived usefulness

and ease of use), attitudes (preferences for the site), and intentions (to revisit the site).

In the work of [Nabil et al. 2011] the authors proposed a conceptual quality model to organize web applications quality factors in terms of its sub factors. In addition, they proposed conceptual quality model that effectively reflects the main views (visitor, owner, end user) of web applications based on the opinion of highly skilled professionals. The main goal of this work is identifying, categorizing, and modeling web applications quality factors. They collected and categorized the quality sub-characteristics into three aspects, based on perspectives: developer perspective, visitor perspective and owner perspective.

The work of [Hasan and Abuelrub 2011] reviewed the evaluation criteria methods which were used in different e-business services. Furthermore, it proposed general criteria for evaluating the quality of any website regardless of the type of service that it offers. The proposed framework has four dimensions which are content quality, design quality, organization quality, and user-friendly quality. These dimensions together with their comprehensive indicators and check lists can be used by web designers and developers to create quality websites to improve the electronic service and then the image of any organization on the Internet.

The model proposed in [Orehovački et al. 2013] contains six axes: System Quality, Service Quality, Content Quality, Performance, Effort and Acceptability, but we chose only the three main axes for building our proposed model, since the last three may be considered as sub components of three main axes.

The web quality model proposed in [Zhao and Zhu 2014] is formalized with ISO/IEC Z language and was built with the Structural Equation Modeling approach. A web source quality evaluation process based on this quality model is implemented and verified. The weights of quality criteria are automatically produced in the validation procedure, which avoids the subjective weight assignment in some classical assessment approaches. The web quality model has three aspects: web source quality, web information quality and web application-specific quality composed of 13 sub-characteristics.

[Yuhana et al. 2014] introduced a framework for measuring the quality of web based Academic Information Systems (AIS) using visitors, developers and institutions perspec-

tives. The AIS quality instruments are built from the combination of other quality models such as ISO/IEC 9126, ISO/IEC 25010:2011, Web Based Application Quality Model (WBAQM), and COBIT 4.1. This framework was expected to produce a more accurate measurement of academic quality web-based information systems and provide detailed recommendations in order to produce a better system, especially to support the business processes of AIS. WBAQM was proposed by [Nabil et al. 2011], so there are many common sub-characteristics between the two frameworks.

[Kotian and Meshram 2017] proposed a model which has two axes: external quality (for end-users) and internal quality (for developers). In each axis they also have quality metrics (as quality characteristics) and quality attributes (as quality sub-characteristics), but metrics and attributes may be assigned to both axes.

For the six other works, there are some reasons for which we did finally not choose them. For example they do not propose an organization of their characteristics. In the next paragraph we summarize these works.

[Olsina and Rossi 2002] based their research on ISO 9126 in order to propose their model. The latter contained four of the six characteristics of ISO 9126, but they did not categorize them as quality dimensions. [Loiacono et al. 2007] focused on the evaluation of consumers of web sites. Their proposed framework contains four dimensions: ease of use, usefulness in gathering information, usefulness in carrying out transactions, and entertainment value. They are not in the scope of our research, being merely dedicated to usability dimension. [Malak et al. 2010] proposed a model based on a probabilistic approach, but they only showed six characteristics of ISO 9126 and did not show sub-characteristics in lower levels. [Carlos and Rodrigues 2012] evaluated quality of web site in Higher Education Institutions. They used the framework of Aladwani [Aladwani and Palvia 2002]. They did not propose a new model, so we did not select them. [Vaucher et al. 2013] also proposed a model, but they only focus on website navigability. [Zia 2015] presented the distinguishable metrics for the Quality Assurance processes involved in web applications and scrutinized the major problem that has been persistent in Quality Assurance related to web applications; i.e. the lack of standards, and development models for the web applications. However they only proposed some metrics and not a comprehensive framework

for quality of web applications.

The quality models we collected are summarized in Table 4.2. Their common interest is that they propose an extensive framework consisting of several levels (at least two). They have a limited number of axes (between 2 and 6). Table 4.2 mentions the title of the paper, the authors, the year of publication, the number of citations as reflected by Google Scholar (as of 2018). Moreover, we summarize the characteristics of the proposed frameworks: the number of axes (characteristics at the first level), the number of levels, the number of characteristics at the second level. Except [Kotian and Meshram 2017], all frameworks are purely hierarchical. In [Kotian and Meshram 2017] framework, several axes share some characteristics. Table 4.2 also mentions if the paper proposes metrics or refers to metrics at the last level of the framework.

Figure 4.1 shows the reference relations between the eight papers. If a paper references another previous paper, they are connected by an arrow, the top of arrow is the referenced work and the bottom of arrow is the referencing work. It shows that [Aladwani and Palvia 2002] is a seminal paper in the domain. [Aladwani and Palvia 2002], [Nabil et al. 2011] and [Hasan and Abuelrub 2011] are the most referenced papers. They are also the three frameworks addressing general web applications (see column Usage of Table 4.2).

4.3 Similarity of dimensions

Many frameworks have been proposed previously to define and evaluate the quality of a web application. To our knowledge, there is no standard framework that allows different stakeholders (designers, developers, sponsors, etc.) tyto share a common view and thus to facilitate their exchanges. To advance in the definition of such a standard, we have compared existing frameworks in order to derive a common framework. Thus, the research question we address in this chapter is: is it possible to unify all the existing frameworks in order to enrich each other and produce a framework that could be accepted by all?

As the previous literature review showed, all frameworks are, with one exception, built hierarchically. They contain two to three levels. They are always described in a descending way, justifying the first level with the help of points of view or perspectives. Many differ-

Table 4.2: Collected works of quality models

Article title	Authors	Years	Usage	Citation	Axis	Levels	N. of chars	Met rics	Pure hier.
Developing and validating an instrument for measuring user-perceived web quality	Aladwani Palvia	2002	general	1444	3	2	55	Y	Y
B2C e-commerce web site quality: an empirical examination	Cao Zhang Seydel	2005	e-commerce	514	4	2	7	Y	Y
Assessing the quality of web sites	Hasan Abuelrub	2011	general	171	4	2	23	N	Y
Web-Based Applications quality factors: A survey and a proposed conceptual model	Nabil Mosad Hefny	2011	general	25	3	3	21	N	Y
Evaluating the perceived and estimated quality in use of Web 2.0 applications	Orehovacki Granic Kermeka	2013	Web 2.0 quality in use	44	6	2	19	Y	Y
Academic Information System Quality Measurement Using Quality Instrument: A Proposed Model	Yuhana Raharjo Rochimah	2014	academic	4	3	3	29	Y	Y
Formalizing and validating the web quality model for web source quality evaluation	Zhao Zhu	2014	web source	7	3	2	13	Y	Y
A Framework for Quality Management of E-commerce Websites	Kotian Meshram	2017	e-commerce	1	2	3	29	Y	N

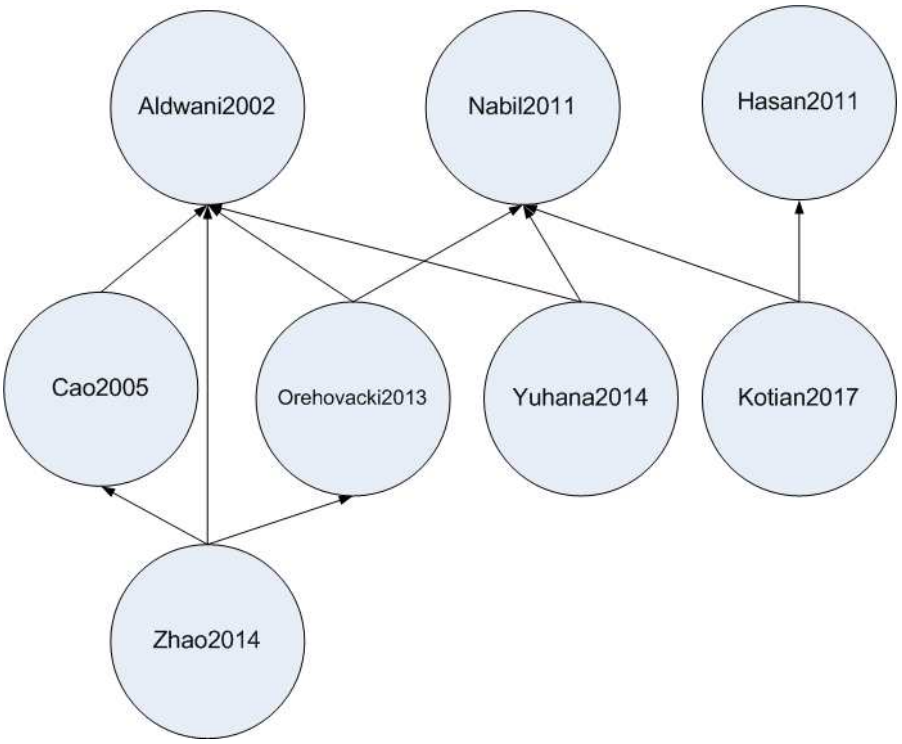

Figure 4.1: The reference relations between the eight papers.

ences exist since each paper focuses differently and thus enriches each perspective. Each axis or first level is then described using many characteristics whose denomination is not the subject of a consensus either.

In order to build a unifying framework, we defined a measure of similarity between all these axes, leading to a matrix storing all these similarities. In this section, we describe this comparison process.

The literature review led us to the elicitation of thirteen dimensions (Table 4.3).

In order to compare these dimensions two by two, we defined five similarity levels as follows:

1. two dimensions are totally different

2. two dimensions share very few sub-characteristics

3. two dimensions share some sub-characteristics

Table 4.3: Selected quality axes

Number	Quality dimenson	References
1	System quality	[Cao et al. 2005] [Orehovački et al. 2013] [Kotian and Meshram 2017]
2	Developer perspective	[Nabil et al. 2011]
3	Source quality	[Zhao and Zhu 2014]
4	Technical Adequacy	[Aladwani and Palvia 2002]
5	Design / User friendly quality	[Hasan and Abuelrub 2011]
6	Information / Content quality	[Cao et al. 2005] [Hasan and Abuelrub 2011] [Orehovački et al. 2013] [Zhao and Zhu 2014] [Kotian and Meshram 2017]
7	Visitor perspective	[Nabil et al. 2011]
8	Web content	[Aladwani and Palvia 2002]
9	Web appearance	[Aladwani and Palvia 2002]
10	Service quality	[Cao et al. 2005] [Orehovački et al. 2013] [Kotian and Meshram 2017]
11	Application specific quality	[Zhao and Zhu 2014]
12	Owner perspective	[Nabil et al. 2011] [Yuhana et al. 2014]
13	Organization quality	[Hasan and Abuelrub 2011]

4. two dimensions share many sub-characteristics

5. two dimensions are identical.

We propose to measure and analyze these similarities using a matrix with the following values:

1. 1 for comparing a dimension with itself (identical dimensions)

2. 0.75 they have many similar sub-characteristics

3. 0.5 they have some similar sub-characteristics

4. 0.25 they have one or two similar sub-characteristics

5. 0 they do not have any similar sub-characteristics (totally different dimensions).

Table 4.4: Matrix of 13 dimensions

	System quality	Developer perspective	Source quality	Technical adequacy	Design/ User friendly quality	Information/ Content quality	Visitor perspective	Web content	Web appearance	Service quality	Application specific quality	Owner perspective	Organization quality
System quality	1	0.75	0.5	0.5	0.75	0.25	0.25	0	0.25	0	0	0	0.5
Developer perspective	0.75	1	0.5	0	0	0.5	0.25	0	0	0	0	0	0
Source quality	0.5	0.5	1	0.5	0.25	0.25	0.25	0	0	0.25	0	0	0
Technical adequacy	0.5	0	0.5	1	0.75	0.25	0.25	0	0	0.5	0	0	0
Design/User friendly quality	0.75	0	0.25	0.75	1	0.25	0.25	0	0	0.5	0	0	0
Information/ Content quality	0.25	0.5	0.25	0.25	0.25	1	0.75	0.5	0.5	0.25	0.5	0	0
Visitor perspective	0.25	0.25	0.25	0.25	0.25	0.75	1	0.5	0.25	0.25	0	0.25	0.25
Web content	0	0	0	0	0	0.5	0.5	1	0	0	0	0	0
Web appearance	0.25	0	0	0	0	0.5	0.25	0	1	0	0	0	0.25
Service quality	0	0	0.25	0.5	0.5	0.25	0.25	0	0	1	0	0	0
Application specific quality	0	0	0	0	0	0.5	0	0	0	0	1	0	0
Owner perspective	0	0	0	0	0	0	0.25	0	0	0	0	1	0
Organization quality	0.5	0	0	0	0	0	0.25	0	0.25	0	0	0	1

Table 4.4 presents the resulting similarity matrix. Of course, this matrix is symmetrical and its diagonal is composed of values all equal to 1. The values of 0.25, 0.5, and 0.75 do not reflect proportionality but they define a total order on all the similarities. We could manually elicit three or four main dimensions by studying this matrix. As an example, system quality, developer perspective and source quality look very similar. However, in order to propose a more robust set of dimensions based on this matrix, we performed several clustering efforts. They are described in the following section.

4.4 Automatic clustering of the quality dimensions

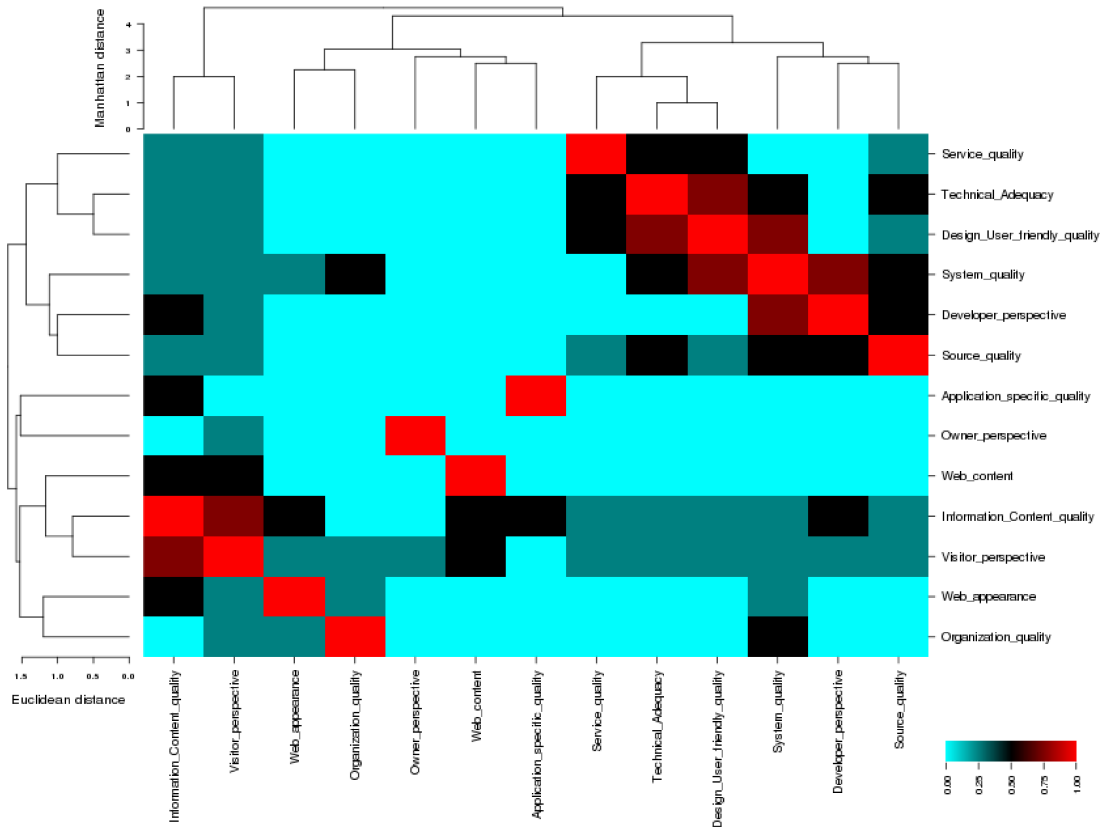

Figure 4.2: Result of Euclidean and Manhattan distance with average linkage cluster

Based on the data above, we performed different automatic clusterings. We used the CIMminer tool [Genomics and Pharmacology Facility, Center for Cancer Research, National Cancer Institute cited January 2018], a tool that can generate color-coded Clustered

Image Maps (heat maps) to represent high-dimensional data sets, for this purpose. As an example, we got different results such as illustrated at Figure 4.2.

This result is performed using the Euclidean distance and Manhattan distance, with Average Linkage cluster algorithm, the weights being 0, 0.25, 0.5 and 0.75.

All the algorithms require the definition of a targeted number of clusters. By examining Figure 4.2 and other similar results, we observed that it was easy to obtain two clusters. However, the result is not interesting leading to, on the one hand, software quality, and, on the other hand, all other aspects. We then tried to obtain three meaningful clusters and we reproduced the framework of [Nabil et al. 2011]. Thus we decided to target a four-cluster breakdown. We describe below the different experiments and results.

First we performed the clustering using 3 hierarchical clustering algorithms (Single Linkage, Complete Linkage and Average Linkage), 3 distances (Euclidean, Manhattan and Correlation) and 3 sets of distance values (normal, extreme high and extreme low). The objective was to reduce the arbitrariness of the similarity values.

The initial distance value is the collection of {1, 0.75, 0.5, 0.25, 0} that we defined above. Maybe the choice of values is subjective. It only reflects an order between similarities. Thus we also used two other distance value sets: extreme high and extreme low for observing the impact on clustering results. In the extreme high value, we increase 0.75 to 0.9 and 0.5 to 0.75, and keep the value 0.25. In the extreme low value, we decrease 0.5 to 0.25 and 0.25 to 0.1 and keep the value 0.75.

The results are shown in Table 4.5.

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
1	Single Linkage	Euclidean	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 13} {11} {12}
2	Single Linkage	Euclidean	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 13} {11} {12}

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
3	Single Linkage	Euclidean	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 13} {11} {12}
4	Single Linkage	Manhattan	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
5	Single Linkage	Manhattan	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7} {8, 9, 11, 13} {12}
6	Single Linkage	Manhattan	{1, 0.75, 0.25, 0.1, 0}	{1, 2} {3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
7	Single Linkage	Correlation	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}
8	Single Linkage	Correlation	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}
9	Single Linkage	Correlation	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}
10	Complete Linkage	Euclidean	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
11	Complete Linkage	Euclidean	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
12	Complete Linkage	Euclidean	{1, 0.75, 0.25, 0.1, 0}	{1, 2} {3, 4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
13	Complete Linkage	Manhattan	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7} {8, 9, 11, 12, 13}

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
14	Complete Linkage	Manhattan	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
15	Complete Linkage	Manhattan	{1, 0.75, 0.25, 0.1, 0}	{1, 2} {3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
16	Complete Linkage	Correlation	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
17	Complete Linkage	Correlation	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8, 11} {9, 12, 13}
18	Complete Linkage	Correlation	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8, 11} {9, 12, 13}
19	Average Linkage	Euclidean	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8} {9, 13} {11, 12}
20	Average Linkage	Euclidean	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
21	Average Linkage	Euclidean	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
22	Average Linkage	Manhattan	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
23	Average Linkage	Manhattan	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7} {8, 9, 11, 12, 13}

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
24	Average Linkage	Manhattan	{1, 0.75, 0.25, 0.1, 0}	{1, 2} {3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
25	Average Linkage	Correlation	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 11} {9, 13} {12}
26	Average Linkage	Correlation	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}
27	Average Linkage	Correlation	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 11} {9, 13} {12}

Table 4.5: Result of 27 clusterings with hierarchical clustering algorithms

The analysis of Table 4.5 allows us to check that some clusters (e.g. {1, 2, 3} or {4, 5, 10}) are very strong since they are present in many results whatever algorithms, distances, similarity values. However, other clusters are not so stable. This finding led us to perform more experiments with non-hierarchical clustering algorithms. We used two common algorithms: k-means and k-medoids. The three distances and three similarity values remain unchanged. Since k-medoids algorithm is not appropriate for Correlation distance, we have 15 test cases in this part.

For executing non-hierarchical clustering algorithms, we used R language, a programming language and a free software environment for statistical computing and data mining. The results are shown in Table 4.6.

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
28	k-means	Euclidean	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
29	k-means	Euclidean	{1, 0.9, 0.75, 0.25, 0}	{1, 13} {2, 3, 12} {4, 5, 10} {6, 7, 8, 9, 11}
30	k-means	Euclidean	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3, 5} {4, 9, 13} {6, 7, 8, 11} {10, 12}
31	k-means	Manhattan	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3, 4, 5, 13} {6, 7, 9, 10} {8} {11, 12}
32	k-means	Manhattan	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3, 6, 7} {4, 5} {8, 9, 11, 12, 13} {10}
33	k-means	Manhattan	{1, 0.75, 0.25, 0.1, 0}	{1, 5} {2, 3, 6} {4, 7, 10} {8, 9, 11, 12, 13}
34	k-means	Correlation	{1, 0.75, 0.5, 0.25, 0}	{1, 3, 7} {2, 6} {4, 5, 10} {8, 9, 11, 12, 13}
35	k-means	Correlation	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}
36	k-means	Correlation	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 5} {3, 4, 10} {6, 7, 8, 9, 11} {12, 13}
37	k-medoids	Euclidean	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
38	k-medoids	Euclidean	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}

N	Algorithm	Distance	Set of similarity values	Resulting set of clusters
39	k-medoids	Euclidean	{1, 0.75, 0.25, 0.1, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8, 11} {9, 12, 13}
40	k-medoids	Manhattan	{1, 0.75, 0.5, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7} {8, 9, 11, 12, 13}
41	k-medoids	Manhattan	{1, 0.9, 0.75, 0.25, 0}	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}
42	k-medoids	Manhattan	{1, 0.75, 0.25, 0.1, 0}	{1, 2} {3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}

Table 4.6: Result of 15 clusterings with non-hierarchical clustering algorithms

Thus we have 27 results of hierarchical algorithms and 15 results of non-hierarchical algorithms. In Table 4.7, we summarized the resulting configurations in order to analyze their frequency among the 42 clustering experiments. We notice that the 42 successive experiments lead us to 20 different configurations.

Config	Content of clusters	No of cases	Frequency
1	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 13} {11} {12}	1 2 3	3
2	{1, 2, 3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}	4	1
3	{1, 2, 3, 4, 5, 10} {6, 7} {8, 9, 11, 13} {12}	5	1
4	{1, 2} {3, 4, 5, 10} {6, 7} {8, 9, 11, 12, 13}	6 15 24 *42*	4

Config	Content of clusters	No of cases	Frequency
5	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}	7 8 9	3
6	{1, 2, 3} {4, 5, 10} {6, 7, 8} {9, 11, 12, 13}	10 11 16 20 21 *28 37 38 41*	**9**
7	{1, 2} {3, 4, 5, 10} {6, 7, 8} {9, 11, 12, 13}	12	1
8	{1, 2, 3} {4, 5, 10} {6, 7} {8, 9, 11, 12, 13}	13 14 22 23 *40*	5
9	{1, 2, 3} {4, 5, 10} {6, 7, 8, 11} {9, 12, 13}	17 18 *39*	3
10	{1, 2, 3, 4, 5, 10} {6, 7, 8} {9, 13} {11, 12}	19	1
11	{1, 2, 3, 4, 5, 10} {6, 7, 8, 11} {9, 13} {12}	25 27	2
12	{1, 2, 3, 4, 5, 10} {6, 7, 8, 9, 11} {12} {13}	26	1
13	{1, 13} {2, 3, 12} {4, 5, 10} {6, 7, 8, 9, 11}	*29*	1
14	{1, 2, 3, 5} {4, 9, 13} {6, 7, 8, 11} {10, 12}	*30*	1
15	{1, 2, 3, 4, 5, 13} {6, 7, 9, 10} {8} {11, 12}	*31*	1

Config	Content of clusters	No of cases	Frequency
16	{1, 2, 3, 6, 7} {4, 5} {8, 9, 11, 12, 13} {10}	*32*	1
17	{1, 5} {2, 3, 6} {4, 7, 10} {8, 9, 11, 12, 13}	*33*	1
18	{1, 3, 7} {2, 6} {4, 5, 10} {8, 9, 11, 12, 13}	*34*	1
19	{1, 3, 4, 5, 10} {2, 9, 11, 12, 13} {6} {7, 8}	*35*	1
20	{1, 2, 5} {3, 4, 10} {6, 7, 8, 9, 11} {12, 13}	*36*	1

Table 4.7: Number of 20 configurations

In Table 4.7, the configurations of non-hierarchical algorithms are in italics for distinguishing from configurations of hierarchical algorithms. We find that the most frequent configuration is configuration 6. The latter contains 4 groups: A. System quality, Developer perspective and Source quality ({1, 2, 3}) B. Information / Content quality, Visitor perspective and Web content ({6, 7, 8}) C. Web appearance, Application specific quality, Owner perspective and Organization quality ({9, 11, 12, 13}) D. Technical adequacy, Design / User friendly quality and Service quality ({4, 5, 10})

Configuration 6 appears 9 times in total, including 5 times in executions of hierarchical algorithms and 4 times in executions of non-hierarchical algorithms. Besides, its groups also appear in other configurations. In detail, group A composed of {1, 2, 3} appears 17 times, group D ({4, 5, 10}) 18 times, group B ({6, 7, 8}) 11 times and group C ({9, 11, 12, 13}) 10 times.

4.5 Discussion and conclusion

As explained above, our classification process allowed us to elicit four rather robust axes, we will now provide details regarding these four groups.

Group A contains System quality, Developer perspective and Source quality. Together they encompass six of the eight frameworks that we selected (Table 4.8). Group A puts together all the characteristics analyzing web applications as softwares. They contain very similar characteristics, such as responsiveness and timelines, traceability and testability or modularity, customizability and adaptability, etc.

Group B contains Information/Content quality, Visitor perspective and Web content. Typically it allows auditors to evaluate a web application as an information provider. Seven frameworks include this dimension. Many common characteristics make these three axes very similar: accuracy (in the three dimensions), relevance or suitability, accuracy, etc. This dimension is rather easy to elicit. It should benefit from standardization efforts such as ISO 8000 tends to propose.

Group D puts together characteristics that deal with software quality but not from the developer viewpoint. In particular, they address security, availability, usability, ease of access, and reliability.

Finally, Group C mainly addresses the specific aspects of the product "web application". Thus it contains popularity or attractiveness, presentation or color consistency, identity, innovation, proper use of colors, language/styles, etc

In this chapter, we presented a detailed study of eight frameworks for web application quality. We analyzed their similarities and conducted classification efforts in order to elicit four main axes describing web application quality. We used a bottom-up approach consisting in comparing two by two the different dimensions of the eight frameworks and grouping them together with the help of a clustering approach. We consolidated our results by experimenting several algorithms, several similarity sets, and several distances. The four main axes subsume the main dimensions of the eight frameworks.

Future research is necessary to reinforce this framework, especially by defining the

lower levels of the proposed hierarchy, but also by verifying the strength of the resulting framework with specific measures.

In the next chapter, we will concentrate our efforts on quality metrics, constituting the lowest level of web quality models.

Chapter 5

Metrics for web application quality

Chapter 4 was dedicated to the definition of web application quality. It was the first step of our approach. The second step aims at measuring this quality. To this end, in this chapter, we review and categorize metrics proposed in the literature addressing this measurement problem.

5.1 Introduction

Chapter 2 and 4 described many viewpoints of web application quality. They allowed us to show how the concept of quality in general, and of web application quality in particular, is composed of many features. As a consequence, evaluating this quality relies also on many metrics.

5.1.1 The importance of measurement

As Lord Kelvin, an Irish physicist, wrote: "When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind" [Thomson 1883]. This claim shows the importance of measurement in physics in particular and natural sciences in general. It is also true in computer science.

Successful organizations use measurement as part of their basic activities [McGarry et al. 2001]. Software measurement is a key component of all software management processes. It contributes to software process improvement [Barcellos et al. 2010]. Barcellos

et al. propose an ontology as a basis for guiding software engineers in their measurement processes. Measurable entities may be quantified with either basic or derived measures.

In the quality domain, the measurement helps users to evaluate the quality and to compare different qualities. Measuring quality requires that the stakeholders share a common understanding of the relations “same quality as” and “better quality as” [Jorgensen 1999].

In order to effectively ensure web application quality, it is necessary to assess it and thus to measure it. The benefits obtained from quality measurement in the software engineering field led to its adoption in many other fields.

5.1.2 Definition of metric

In the Oxford Dictionary, a metric is defined simply as a system or standard of measurement [Press cited January 2017].

The Dictionary of Computer Science [Butterfield and Ngondi 2016] proposes a more explicit definition: "a metric is a number representing the degree to which a software, or an entity related to software, possesses some notional property, as observed or perceived by an individual or a group."

In the domain of web quality, a metric is defined as a measurement method associating a value for a measurable quality attribute such as understandability or maintainability. In order to objectively evaluate the quality of web applications, suitable quality metrics have to be defined.

5.1.3 Scope of Chapter 5

The goal of the research described below is to collect and characterize the quality metrics of web applications. Chapter 4 allowed us to reveal the high number of frameworks proposed to structure the different facets of web application quality, depending on the perspective. Other approaches, related to these frameworks or independent of them, proposed dozens of metrics that allow stakeholders to measure different facets. In this chapter, we present the result of our research aiming at linking web application quality characteristics to such metrics. We chose to address the characteristics contained in ISO25010. The latter

is largely adopted but, to the best of our knowledge, there is no previous research mapping ISO characteristics and sub-characteristics to these metrics.

5.1.4 Research methodology

The metrics analyzed in this chapter were collected from the literature. We had to define a systematic approach in order to select them in the numerous papers of this domain. To this end, we performed the two steps described below:

1. Exploratory study

   In 2013, starting from [Calero et al. 2005], which was the most recent detailed literature review on the topic, we updated their study with these objectives:

   - Selecting quality metrics and thus discard all the metrics that only describe the web applications, for example size metrics,
   - Finding new papers adding material on the subject, mainly metrics,
   - Mapping the whole set of resulting metrics to the characteristics and sub-characteristics of ISO9126 standard.

   The result of this first step was published in [Cherfi et al. 2013] and is described in more details in Section 5.2.

2. Second step

   In order to obtain a more reliable representative set of metrics, we conducted a complementary literature review borrowing from systematic literature review (SLR) principles. To this end, we defined which papers should be the new inputs of our process. This is defined through the definition of inclusion and exclusion criteria for selecting papers and, among these papers, inclusion and exclusion criteria for selection metrics. The result of this second step is described in more details in Section 5.3.

5.2 Exploratory study

The objective of the first step was to lay the groundwork for our measurement problem. To this end, we have analyzed 108 metrics found in the literature and classified them

according to the six quality characteristics and their sub-characteristics from ISO 9126.

Our starting point was the paper of Calero [Calero et al. 2005]. They had elicited 385 web metrics. However not all are quality metrics. As an illustration, they have size metrics (page code size, total number of photos, etc.). Others are descriptive (page language, page interface). For these 385 metrics, they proposed a categorization based on the three dimensions of WQM (this model is described in Chapter 2): features, life-cycle processes and quality aspects. Features are the three classical models describing web applications: content, presentation and navigation. Life-cycle processes follow the ISO 12207 standard (ISO/IEC, 1995; ISO, 2002). They include: the development process, the operation process (encompassing the operative support for users), the maintenance process (both curative and evolving), the project-management process, and the reuse program-management process. Finally, quality is based on the six main characteristics of ISO/IEC 9126. Since we are interested in quality measurement, we have focused on this quality dimension and proposed to map the metrics on the six characteristics but also on their sub-characteristics. We present in the following sub sections an analysis of each characteristic.

5.2.1 Measuring quality characteristics

In this first study, we concentrated on ISO/IEC 9216 composed of six main characteristics: functionality, efficiency, usability, reliability, maintainability, and portability.

5.2.1.1 Measuring Functionality

Functionality helps verifying whether the web site provides its intended functionalities. If we consider its sub-characteristics, we notice that some of them are more prone to automatic evaluation than others. For example, suitability is more likely to be assessed by surveys and questionnaires. We could however consider that adequacy of image size or the possibility of horizontal scrolling could increase suitability and thus define an automatic way to measure this sub-characteristic. Moreover, we found no metric taking into account user profiles (children, elderly persons, students, scientists etc.). Literature about web metrics, in general, lacks considering the specific domain of web site usage in its evaluation, except perhaps e-commerce web sites that have benefited from specific contributions such as the

research presented in [Stefani and Xenos 2009]. As we can see in Table 5.1, accessibility has attracted many researchers especially during recent years. Surprisingly, security lacks metrics definitions. Table 5.1 presents our classification of web application functionality metrics.

Table 5.1: An excerpt of functionality web metrics

	Sui.	**Acu.**	**Int.**	**Sec.**	**Acc.**	References
Image size	x				x	[Signore 2005]
Presence of site name in title		x			x	[Rio and Abreu 2010]
Link title to extra information		x			x	[Olsina and Rossi 2002]
Broken links					x	[Olsina and Rossi 2002]
Number of syllables per word	x	x				[Signore 2005]
Number of words per sentence	x	x				[Signore 2005]
Horizontal scrolling	x	x				[Olsina and Rossi 2002]
Lack of cohesion in methods			x			[Chae et al. 2007]
Response time					x	[Dominic and Jati 2011]
Frequency of update				x		[Olsina and Rossi 2002]
Abbreviation: Sui. = Suitability; Acu. = Accuracy; Int. = Interoperability; Sec. = Security; Acc. = Accessibility;						

We notice that all sub-characteristics are not equally covered. Accessibility and Accuracy are rather well covered. However, very few efforts are done in terms of security metrics definition. The distribution of metrics sketched in Figure 5.1 reveals that among 58 metrics measuring Functionality, only 5 representing less than 9% are dedicated to security. One reason is that security is more considered as a non-functional requirement tested once code is produced. Practices are more oriented toward error detection than prevention. Considering security concerns early in the development process will probably increase the need for security measurement in the very first steps of the web application life cycle.

5.2.1.2 Measuring Efficiency

Efficiency measurement is related to time and resource consuming. Web page interaction requires programs execution (calculation, displaying texts, images etc.) and data transfer. It is rather easy to define metrics to evaluate the three sub-characteristics.

Concerning Efficiency, developers are more accustomed with run time measurement than with static analysis. This is probably the reason why few metrics have been defined.

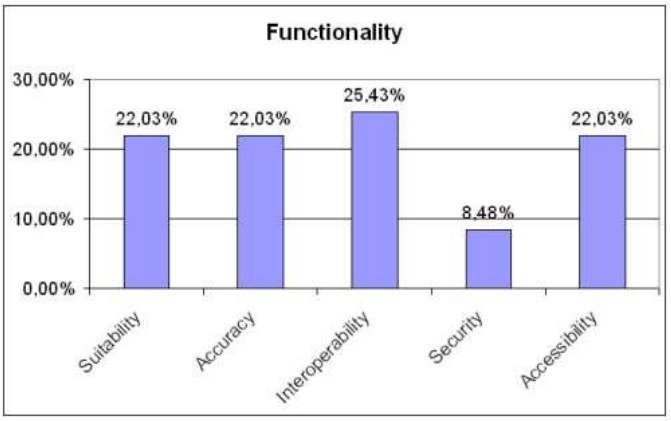

Figure 5.1: Distribution of functionality web quality metrics

Table 5.2: An excerpt of efficiency web metrics

Metrics' name	**TBe.**	**Res.**	References
Number of internal links		x	[Alves and Ponti 2001]
Download time	x		[Bajaj and Krishnan 1999]
Response time	x		[Dominic and Jati 2011]
Traffic	x	x	[Dominic and Jati 2011]
Non-frame version		x	[Olsina and Rossi 2002]
Abbreviation: TBe. = Time behaviour; Res. = Resource utilisation			

This last remark is especially true for time behavior. This is also due to the fact that many web metrics can be used to evaluate several web quality characteristics. The case of security seems to be particular. Researchers and web developers should probably define specific metrics based on their experiences and related programming knowledge. They are probably able to gather information on web application behavior regarding time and resource consuming enabling the definition of accurate and suitable metrics for efficacy that could be evaluated early in the development process.

5.2.1.3 Measuring Usability

Web application and their underlying technology are more and more complex. This makes the Usability quality characteristic very important as it has a direct impact on their acceptability and success. This quality characteristic addresses quality from the user point of view leading generally to a subjective evaluation. Objective evaluations based on metrics rely on assumptions about user perception. For example, if we consider the

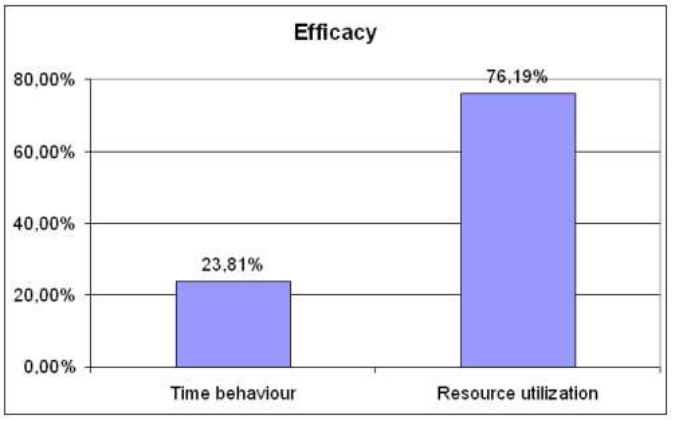

Figure 5.2: Distribution of efficiency web quality metrics

'Emphasized body words' metric from Table 5.3, we assume that it has impact on user perception. However, some users will consider that the emphasis helps them detecting the important parts of the page while others prefer having their own opinion about the relevance of the information. Thus this second category of users considers this emphasis as impacting negatively their perception.

Measuring Usability implies concentrating on application properties enhancing efficiency and effectiveness of navigation. This explains the variety of metrics exploiting within or inter pages links. It also relies on visual perception and acceptation of pages explaining the importance of properties such as font's variety, existence of images and tables etc. It also encompasses ease of understanding of the content of pages. Such characteristic requires semantic-based measurements such as local consistency of the page or global consistency of the overall web site. Table 5.3 presents some of these metrics.

We also analyzed the distribution of metrics among the sub characteristics of usability (Figure 5.3). We notice that, among the four sub-characteristics, Understandability totalizes nearly 40% of the metrics. As any quality characteristic, measuring Understandability could rely on user perception (external quality) or on product properties (internal quality). Most of the measures are related to external quality ('number of title words', 'different fonts colors' etc.).

Internal quality is however less considered as it is more difficult to handle and it requires validation. Indeed, internal quality measurement requires making assumptions on inter-

Table 5.3: An excerpt of usability web metrics

Metrics' name	**Und.**	**Lea.**	**Ope.**	**Att.**	References
Emphasized body word count			x		[Ivory 2001a]
Page title word count	x	x		x	[Ivory 2001a]
Number of different text fonts in CSS	x				[Ivory 2001a]
Word count	x		x		[Ivory 2001a]
Images count				x	[Olsina and Rossi 2002]
Email contact presence				x	[Olsina and Rossi 2002]
Broken links	x	x		x	[Olsina and Rossi 2002]
Paragraph size			x		[Signore 2005]
Page size			x		[Rio and Abreu 2010]
Number of script files per page			x		[Reifer 2000]
Abbreviation: Und. = Understandability; Lea. = Learnability; Ope. = Operability; Att. = Attractiveness					

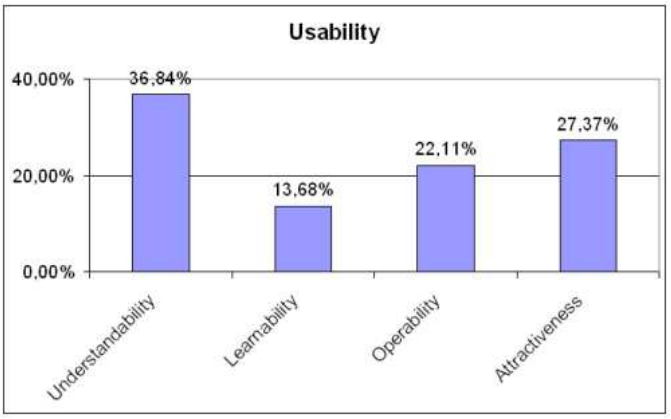

Figure 5.3: Distribution of usability web quality metrics

dependencies between some internal properties of the product (size, complexity etc.) and the quality characteristic it evaluates. Once the metrics derived from these assumptions are defined, they have to be validated by experiments in which the objective is to validate the values calculated by the metrics with the observations or the judgment of the persons participating to the experiments. Some measures such as 'local coherence' or those related to navigability, however, fall into this last category – internal quality - and are reused from software quality findings.

5.2.2 Measuring Reliability

Reliability is often defined as "The probability of failure-free operation for a specified time in a specified environment for a specified purpose" [Sommerville 1995]. Among metrics measuring reliability there are those relying on run time testing, such as 'Mean time to failure' or 'Mean time to repair'. We have not collected such metrics in our study since we are more interested in measuring quality during the analysis and the design of applications.

Table 5.4: An excerpt of reliability web metrics

Metrics' name	**Mat.**	**FTo.**	**Rec.**	References
Response time		x	x	[Dominic and Jati 2011]
Design optimisation	x			[Dominic and Jati 2011]
Percent of dead-end web pages	x			[Olsina et al. 2001]
Number of orphan pages	x			[Olsina and Rossi 2002]
Presence of ALT attribute in image			x	[Olsina and Rossi 2002]
Frequency of update	x	x	x	[Olsina and Rossi 2002]
Number of different broken links	x	x	x	[Olsina and Rossi 2002]
HTML warnings per page	x			[Rio and Abreu 2010]
Abbreviation: Mat. = Maturity; FTo. = Fault tolerance; Rec. = Recoverability				

At Table 5.4, we notice that most of the metrics defined for reliability are specific to web applications. For example existence of textual description for images that could help rebuilding the application in case of failure having altered the image is very specific to web sites. It's also the case for the number of links. However, web developers should also rely on metrics from software engineering that hold for web applications. For example, several findings from architecture-based software reliability metrics could be reused. We can find in [Rosenberg et al. 1998] a variety of metrics that could be applied in the context of web applications reliability. As an example, experiences showed that modules with high complexity and high size are less reliable and more fault prone. More generally all the metrics measuring modularity, cohesion, coupling and reuse could be used for reliability measurement of web applications.

The analysis of the distribution of metrics among the sub-characteristics leads to the conclusion that recoverability is less covered. This is first due to the complexity of web applications architectures with their layered architectures and the variety of co-existing

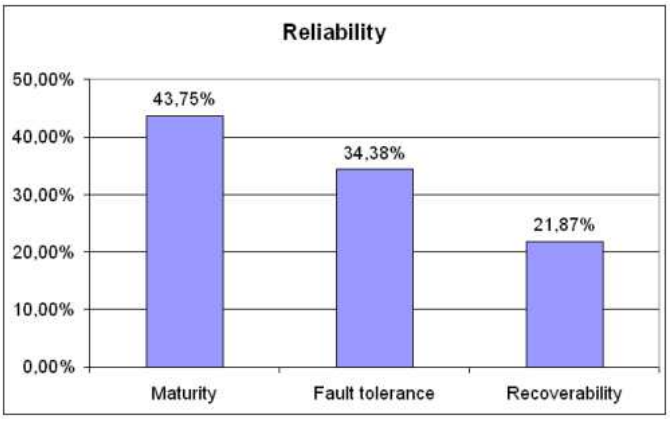

Figure 5.4: Distribution of reliability web quality metrics

technologies. However, recovery of web applications could be even more critical than for traditional software. Indeed, many companies' commercial activities rely on their e-commerce and web sites. And the survey of literature on the subject shows that there is a lack of contributions on web metrics for reliability leading to insufficient reliability prevention.

5.2.3 Measuring Maintainability

Maintainability aims to reduce time and effort devoted to the maintenance of the final product.

An overview of the selected maintainability metrics which constitute a representative sample from this category is provided at Table 5.5. We notice that they are compliant with the principles generally adopted in software engineering. Indeed, they are to some extent related to size (script size, number of web pages, etc.), complexity (design optimization, lack of cohesion, etc.), and coupling (data abstraction coupling, responding methods, etc.).

Concerning their distribution among sub-characteristics, we notice that the percentages varying from 23% to 26% are very close. The metrics proposed to assess these sub-characteristics are based on the same internal properties of the application. Thus, the papers describing these metrics address in a balanced way the four categories. If a web page lacks cohesion this could be due to heterogeneity of presentation styles or heterogeneity of implementation choices for the objects within the page or semantic heterogeneity of

Table 5.5: An excerpt of maintainability web metrics

Metrics' name	**An.**	**Ch.**	**St.**	**Te.**	References
Lack of cohesion in methods	x	x			[Chae et al. 2007]
Data abstraction coupling		x			[Chae et al. 2007]
Script size per page		x	x	x	[Di Lucca et al. 2004]
Markup validation				x	[Dominic and Jati 2011]
Total number of server pages		x	x		[Ghosheh et al. 2008]
Total number of client pages		x	x		[Ghosheh et al. 2008]
Abbreviation: An. = Analysability; Ch. = Changeability; St. = Stability; Te. = Testability					

the content. In all these situations, the web page will be difficult to analyze, difficult to change and also difficult to test as it will require several competencies in addition to the problem of co-existence of different implementation techniques.

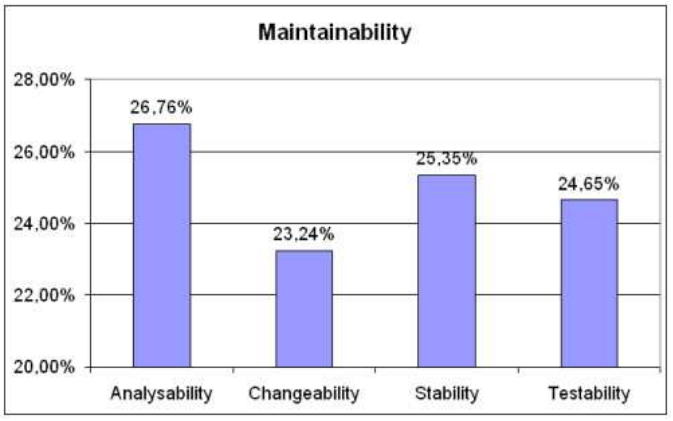

Figure 5.5: Distribution of maintainability web quality metrics

5.2.4 Measuring Portability

Portability of web application is a fundamental property. Indeed, when we consider a web page we assume that it can be displayed using any web browser or any version of a web browser. However, error messages and alerts, informing you that you are not using the right browser or the right version of browser, are frequent.

Portability quality metrics should measure the effort needed to transfer an application from an environment (hardware and software) to an other. It requires measuring: (1) the adaptation effort of the application code, (2) the installation effort on a given platform

Table 5.6: An excerpt of portability web metrics

Metrics' name	**Ada.**	**Ins.**	**Rep.**	**CoE.**	References
Total link count	x	x	x		[Olsina and Rossi 2002]
Number of panes regarding frames	x	x	x	x	[Olsina and Rossi 2002]
Image size	x				[Signore 2005]
CSS size per page	x				[Rio and Abreu 2010]
Average font size in pixel in CSS	x	x	x		[Rio and Abreu 2010]
Abbreviation: *Ada. = Adaptability; Ins. = Installability; Rpe. = Replaceability; CoE. = Co-existence;*					

using a given operating system such as mobile devices, (3) replacing parts or modules from the application with taking into account that (4) several, heterogeneous and sometimes incompatible, implementation techniques must be managed. All these four aspects are more complex in web development than in traditional software developments because of the variety of underlying technologies (HTML definitions, Java script code, style files, script functions etc.). In addition to that, many of web developers are not always software developers. These reasons explain partly the extensive use of tools in web developments. The consequence is thus the difficulty to predict the effort needed to perform the changes as the expertise of developers is partly embedded in the tools they use.

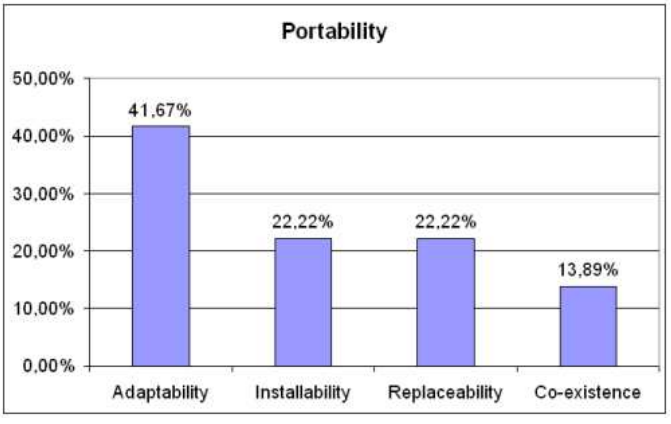

Figure 5.6: Distribution of portability web quality metrics

As a conclusion, this section allowed us to explain that we could map the metrics not only to the characteristics of ISO/IEC 9126 but also to the sub-characteristics. Moreover, some metrics may be associated with several sub-characteristics. In the following section,

we synthesize the conclusions of a general analysis of these 108 metrics.

5.2.5 General analysis

The distribution of web metrics at the characteristics level is given by the pie chart of Figure 5.7. It shows that two characteristics, namely Maintainability and Usability totalize nearly 60% of the metrics. Moreover, Reliability characteristic attracted less than 10% of the metrics analyzed as well as Efficacy.

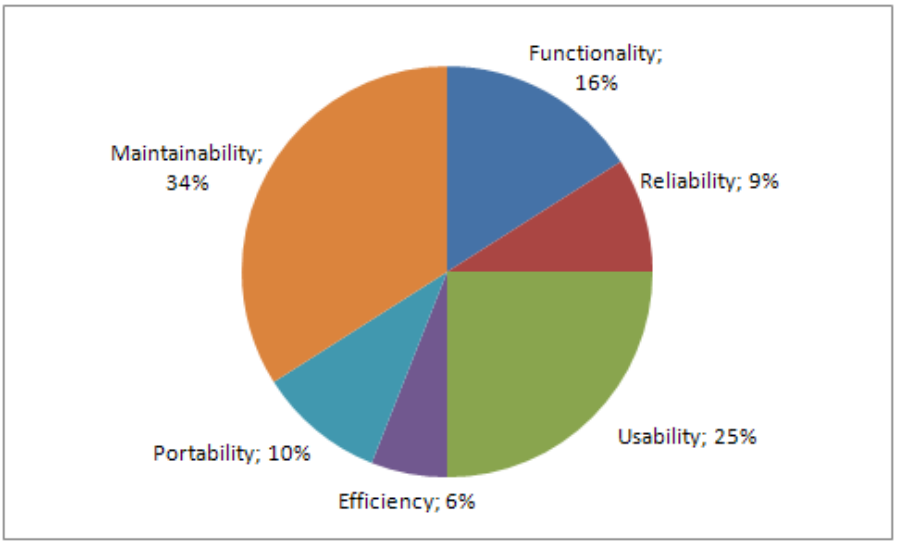

Figure 5.7: Global overview of metrics per characteristic

The previous study performed by Calero and co-authors in [Calero et al. 2005] highlights a different situation (see Figure 5.8). Let us keep in mind that it was in 2005. Thus it is interesting to analyze how the situation evolved. Even if there is an overlap between the two sub sets of metrics studied, our analysis incorporates recent work, subsequent to the research of Calero.

The first observation is that maintainability metrics seem to capture a growing interest with 34% of the total of metrics. Indeed, thanks to their attractiveness, web applications become more popular for individuals and even for companies. In the same time, they tend to be more complex, thus generating high maintenance cost and time. This complexity is inherent to their underlying architectures and technologies. It is also a consequence of their rapid evolution due to their attractiveness and the pressure of the market. Preventive

approaches in software quality, based on evaluation metrics, allowed considering quality earlier in the development process. This led to a reduction of maintenance costs. We can deduce that the same phenomenon may be observed in web applications development.

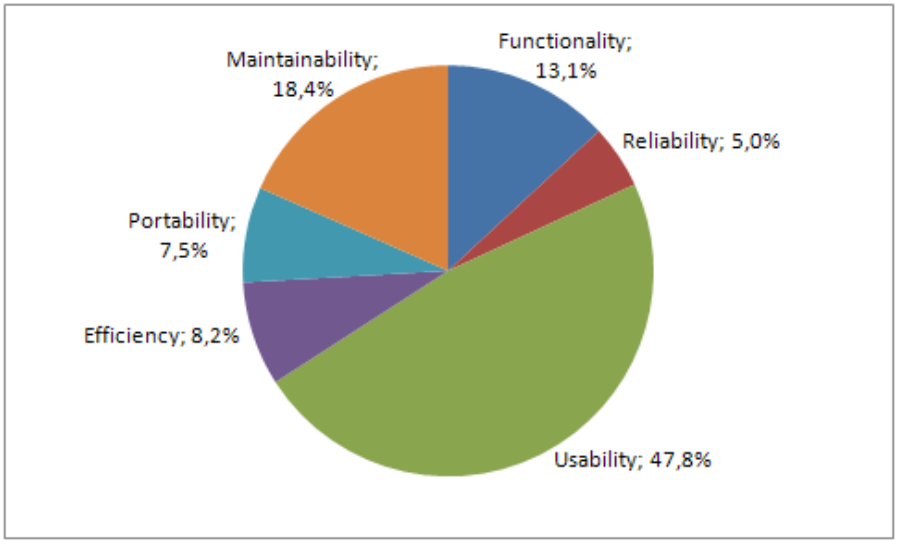

Figure 5.8: Web quality metrics distribution according to [Calero et al. 2005]

Usability attracts 25% of metrics, which is lower than the value observed in [Calero et al. 2005]. However, this does not mean that it was more important in 2005. Indeed, we have here percentages meaning that other characteristics, such as Maintainability, relatively gained in interest. On the one hand, web applications are most of the time used by end-users having no specific skills or competencies in computer based technologies. On the other hand, the success of these applications depends on their acceptance by these non-skilled persons. This shows the importance of usability, and more precisely Understandability as shown at Figure 5.3.

Portability also evolved. This is due to the diversity and heterogeneity of the technologies used. Efficacy and reliability still attract little interest. This is probably due to a relatively better handling of underlying problems on hardware (relying on standby servers, setting up recovery procedures etc.). Preventive solutions, based on metrics, could however provide good complementary solutions by well targeting the problems.

However, we would like to highlight the limitations of our study that has probably not considered all the abundant work related to web quality metrics as it would require

substantial time and effort. Nevertheless, it is a good starting point for a larger study.

5.2.6 Conclusion

The literature on the topic of web quality contains papers dedicated to quality criteria definition, and others on quality metrics without an explicit link with a quality factor. Our literature analysis also revealed that current approaches evaluate external web site quality, when the application is already online. For example, [Fernandez et al. 2011] describes a systematic mapping study of the usability evaluation methods for the web covering 206 papers selected among 2703 published over 14 years. Finally, some quality factors, such as security, have received very little attention. Accessibility was largely studied and led to many recommendations; however, few metrics were proposed to assess this factor. This analysis convinced us that there is a necessity to propose a web site design method encompassing the evaluation and improvement of quality based on a complete quality model, covering most ISO proposals.

5.3 The complementary literature review

The second step of our research consisted in: a) Updating our first study, b) Ensuring a degree of completeness by conducting a more systematic search of web quality metrics.

5.3.1 Protocol

Our objective was to build a comprehensive set of web quality metrics allowing us to provide web application developers with a practical tool for evaluating web applications. In particular, we aimed at mapping these metrics to the components of the framework proposed by ISO25010 (SQUARE).

We defined the following search terms “web quality”, “quality metrics” and “web metrics” and then scanned the papers using the following list of inclusion and exclusion criteria:

Inclusion criterion 1: the paper describes a research dedicated to web quality metrics.

Inclusion criterion 2: the paper was published after 2000.

Inclusion criterion 3: the paper is written in English or in French.

Inclusion criterion 4: the paper is an original work. It means that the proposed metrics are published for the first time.

Exclusion criterion 1: the paper is dedicated to web quality metrics but only focuses on metrics from previous papers. One example is the work of Thangammal [Thangammal and Pethalakshmi 2015], because it only reuses the metrics from the research of Vaucher [Vaucher et al. 2009].

Exclusion criterion 2: the paper encompasses software quality and not specifically web application quality.

Performing this process, we have collected 7 more articles to add to our list (Table 5.7). All these articles were published from 2013 to 2017, except a work of Vaucher (published in 2009). That paper, containing interesting metrics, should have already been considered in our first study.

Table 5.7: Collected articles

Article name	Published date	References
Designing Highly Usable Web Applications	2014	[Abrahão et al. 2014]
A Practical Approach for Measuring Quality of Website	2014	[Gautam and Sharma 2014]
Evaluating Web Accessibility Metrics for Jordanian Universities	2016	[Kamal et al. 2016]
Measuring Web Site Usability Quality Complexity Metrics for Navigability	2015	[Kumar et al. 2015]
Evaluation of Web Metrics	2017	[Mittal 2017]
Prism Based Quality Evaluation and Prediction of Web Applications	2013	[Sethuraman et al. 2013]
Recommending Improvements to Web Applications Using Quality-Driven Heuristic Search	2009	[Vaucher et al. 2009]

As a result, we obtained a table of 166 metrics. 108 metrics were collected before 2013 and analyzed in the previous section of this chapter. 54 metrics were added in the second phase of this work. Moreover, we mapped the 166 metrics to the eight quality factors/characteristics of ISO25011/2017 SQUARE. The ISO standard does not provide indications on how to measure or appreciate the characteristics. This explains the growing interest in defining and using metrics to evaluate these factors.

5.3.2 Results

We have analyzed the 166 metrics found in the literature and classified them according to the above eight quality characteristics and their sub-characteristics. We present in the following sub sections an analysis of each characteristic. Then we present a more global analysis.

5.3.2.1 Measuring Functional Suitability

This characteristic replaces the Functionality of ISO/IEC 9126. It is defined as "the degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions". It is composed of three sub-characteristics: functional completeness, which is a new feature missing in the previous standard; functional correctness, instead of accuracy; and functional appropriateness for suitability. The Interoperability sub-characteristic is no more linked to this characteristic since it moves to Compatibility. Finally the sub-characteristic Security becomes a characteristic.

Table 5.8: Functional suitability metrics

Metrics' name	**Com.**	**Cor.**	**App.**	References
Exclamation point count		x		[Ivory 2001b]
Images count		x		[Olsina and Rossi 2002]
Total embedded links			x	[Olsina and Rossi 2002]
Number of panes regarding frames			x	[Olsina and Rossi 2002]
CSS size per page			x	[Rio and Abreu 2010]
Image size			x	[Signore 2005]
Script size per page		x		[Di Lucca et al. 2004]
Presence of name's author	x			[Charland et al. 2007]
Presence of logo	x			[Charland et al. 2007]
Presence of site name in title	x			[Rio and Abreu 2010]
Presence of navigation / menu bar	x	x	x	[Signore 2005]
Presence of breadcrumbs	x	x	x	[Signore 2005]
Presence of page title in link	x	x		[Rio and Abreu 2010]
Number of script files per page		x	x	[Reifer 2000]
Number of CSS files per page				[Rio and Abreu 2010]
Presence of specific CSS to device		x	x	[Signore 2005]
Number of div tags			x	[Signore 2005]
Link image count			x	[Alves and Ponti 2001]
Image number per page			x	[Olsina and Rossi 2002]
Number of syllables per word		x	x	[Signore 2005]

Table 5.8: Functional suitability metrics

Metrics' name	**Com.**	**Cor.**	**App.**	References
Number of words per sentence		x	x	[Signore 2005]
Horizontal scrolling		x	x	[Olsina and Rossi 2002]
Response for classes		x		[Chae et al. 2007]
Response time	x			[Dominic and Jati 2011]
Frequency of update	x			[Olsina and Rossi 2002]
Design optimisation		x		[Dominic and Jati 2011]
Markup validation		x		[Dominic and Jati 2011]
The harmony between using font and the background			x	[Kamal et al. 2016]
Presence of a search engine	x			[Vaucher et al. 2009]
Presence of a site map	x			[Vaucher et al. 2009]
Proportion of titles chosen suitably for each icon/title		x		[Abrahão et al. 2014]
Proportion of meaningful messages (error, advise, and warning messages)		x		[Abrahão et al. 2014]
Breadth of the internavigation			x	[Abrahão et al. 2014]
Depth of the navigation			x	[Abrahão et al. 2014]
Presence of about info	x			[Sethuraman et al. 2013]
Not presence of auto refresh option			x	[Sethuraman et al. 2013]
Safe color is used			x	[Gautam and Sharma 2014]
Use color for blind people			x	[Gautam and Sharma 2014]
Underlined text is used			x	[Gautam and Sharma 2014]
CSS attributes	x	x		[Gautam and Sharma 2014]
Description of meta	x			[Gautam and Sharma 2014]
Using thumbnails	x	x	x	[Gautam and Sharma 2014]
Presence of bulletin board	x			[Gautam and Sharma 2014]
Presence of information guide	x			[Gautam and Sharma 2014]
Presence of customer feedback	x			[Gautam and Sharma 2014]
Presence of domain name	x			[Gautam and Sharma 2014]
Presence of information publicity				[Gautam and Sharma 2014]
Abbreviation: Com. = Functional completeness; Cor. = Functional correctness; App. = Functional appropriateness				

Sticking to this new definition of Functionality, we found 47 metrics that best map to one or several sub-characteristics (Table 5.8). As an example, the presence of a site map is related to functional completeness whereas the image size characterizes functional appropriateness. Another example is horizontal scrolling that characterizes both functional correctness and appropriateness. Figure 5.9 presents the distribution of metrics according

to the three sub-characteristics.

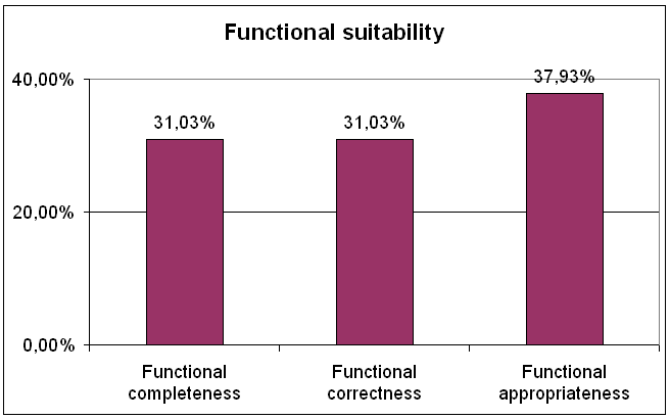

Figure 5.9: Distribution of functional suitability metrics

5.3.2.2 Measuring Performance Efficiency

This is the new term used for the Efficacy characteristic. It is composed of three sub-characteristics: time behavior and resource utilization, also contained in ISO9126, and a new one called Capacity. The latter is defined as the "degree to which the maximum limits of product or system parameters meet requirements". We found 22 metrics when matching our set with performance efficiency sub-characteristics (Table 5.9).

Table 5.9: Performance efficiency metrics

Metrics' name	**TBe.**	**Res.**	**Cap.**	References
Body text words		x		[Singh et al. 2011]
Word count per page		x		[Ivory 2001b]
Page size		x	x	[Rio and Abreu 2010]
Number of panes regarding frames		x		[Olsina and Rossi 2002]
CSS size per page		x		[Rio and Abreu 2010]
Download time of home page	x			[Bajaj and Krishnan 1999]
Image size		x		[Signore 2005]
Script size per page		x		[Di Lucca et al. 2004]
Download time	x			[Bajaj and Krishnan 1999]
Download time of all pages	x			[Bajaj and Krishnan 1999]
Number of script files per page		x		[Reifer 2000]
Number of CSS files per page		x		[Rio and Abreu 2010]
Number of links to other sites		x		[Alves and Ponti 2001]
Number of internal links		x		[Alves and Ponti 2001]
Image title		x		[Olsina and Rossi 2002]

Table 5.9: Performance efficiency metrics

Metrics' name	**TBe.**	**Res.**	**Cap.**	References
Word count		x		[Ivory 2001b]
Non-frame version		x		[Olsina and Rossi 2002]
Response time	x			[Dominic and Jati 2011]
Traffic	x	x	x	[Dominic and Jati 2011]
Number of items		x		[Dominic and Jati 2011]
Time of read and use content	x			[Kamal et al. 2016]
Abbreviation: TBe. = Time Behaviour; Res. = Resource utilization; Cap. = Capacity				

Most of them are dedicated to resource utilization measurement (Figure 5.10). This result is similar to the previous analysis. It perhaps reflects only the fact that there are more various ways to measure resource utilization than time behavior which is quite classically evaluated thanks to download time, response time and traffic.

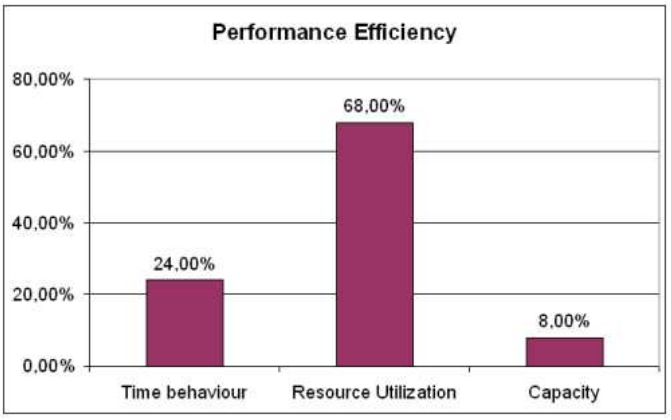

Figure 5.10: Distribution of performance efficiency metrics

5.3.2.3 Measuring Usability

The Usability characteristic is now composed of six sub-characteristics: appropriateness recognizability instead of understandability, learnability (unchanged), operability (unchanged), user interface aesthetics (instead of attractiveness), and two new sub-characteristics: user error protection and accessibility.

User error protection denotes the "degree to which a system protects users against making errors". This is a very common concept in databases where integrity constraints

are defined mainly to improve data quality and implemented through input control rules.

Accessibility is also a very important feature. Its main role led to the definition of specific recommendations. The Web Content Accessibility Guidelines (WCAG) define how to make web content more accessible to people with disabilities.

Table 5.10: Usability metrics

Metrics' name	**Ap**	**L**	**O**	**UE**	**UI**	**Ac**	References
Total number of web pages			x				[Mendes et al. 2001]
Page title word count	x						[Ivory 2001a]
Body text words			x				[Singh et al. 2011]
Word count per page			x				[Ivory 2001a]
Total link count	x		x				[Olsina and Rossi 2002]
Page link count			x				[Ivory 2001a]
Exclamation point count	x						[Ivory 2001a]
Images count					x		[Olsina and Rossi 2002]
Page size			x				[Rio and Abreu 2010]
Total embedded links	x	x					[Olsina and Rossi 2002]
Number of lists					x		[Singh et al. 2011]
Number of panes regarding frames					x		[Olsina and Rossi 2002]
Table count					x		[Ivory 2001a]
Within page links			x				[Singh et al. 2011]
Emphasized body word count					x		[Ivory 2001a]
Total emphasized text					x		[Singh et al. 2011]
Display word count			x				[Ivory 2001a]
Wrapped links					x	x	[Singh et al. 2011]
Download time of home page			x				[Bajaj and Krishnan 1999]
Image size			x				[Signore 2005]
Download time			x				[Bajaj and Krishnan 1999]
Download time of all pages			x				[Bajaj and Krishnan 1999]
Presence of contacts/info form			x				[Olsina and Rossi 2002]
Number of label tags	x						[Rio and Abreu 2010]
Presence of name's author					x		[Charland et al. 2007]
Presence of logo			x				[Charland et al. 2007]
Presence of site name in title						x	[Rio and Abreu 2010]
Presence of navigation / menu bar	x			x			[Signore 2005]
Presence of breadcrumbs	x			x			[Signore 2005]
Presence of page title in link	x						[Rio and Abreu 2010]
Number of script files per page			x			x	[Reifer 2000]
Number of CSS files per page			x			x	[Rio and Abreu 2010]
Number of tables per page					x		[Ivory 2001a]
Presence of specific CSS to device						x	[Signore 2005]
Use of HTML notation in formatting	x	x					[Rio and Abreu 2010]

Table 5.10: Usability metrics

Metrics' name	**Ap**	**L**	**O**	**UE**	**UI**	**Ac**	References
Number of div tags	x					x	[Signore 2005]
Presence of tables inside tables						x	[Rio and Abreu 2010]
Average link words			x				[Ivory 2001a]
Link title (with explanatory help)	x			x		x	[Olsina and Rossi 2002]
Broken links	x	x				x	[Olsina and Rossi 2002]
Number of broken links to other sites						x	[Signore 2005]
Number of links to other sites			x				[Alves and Ponti 2001]
Link image count					x		[Alves and Ponti 2001]
Graphics link count						x	[Stefani and Xenos 2009]
Text link count			x		x		[Ivory 2001a]
Number of internal broken links						x	[Signore 2005]
Number of internal links			x				[Alves and Ponti 2001]
Presence of ALT attribute in image	x	x					[Olsina and Rossi 2002]
Image title	x	x					[Olsina and Rossi 2002]
Accessibility issues per page						x	[Signore 2005]
Image number per page	x				x		[Olsina and Rossi 2002]
Average of font size in em (percentage) in CSS					x		[Rio and Abreu 2010]
Average font size in pixel in CSS					x		[Rio and Abreu 2010]
Maximum font size in em in CSS					x		[Rio and Abreu 2010]
Maximum font size in pixel in CSS					x		[Rio and Abreu 2010]
Minimum font size in em in CSS					x		[Rio and Abreu 2010]
Minimum font size in pixel in CSS					x		[Rio and Abreu 2010]
Average heading length	x				x		[Signore 2005]
Number of italic text bigger than 20 characters					x	x	[Rio and Abreu 2010]
Number of different text colors in CSS	x				x		[Rio and Abreu 2010]
Number of different text fonts in CSS	x				x		[Ivory 2001a]
Number of sentences per paragraph	x				x		[Signore 2005]
Number of sub-headings per heading	x				x		[Signore 2005]
Number of words in metatag description			x				[Rio and Abreu 2010]
Number of words in metatag keywords			x				[Rio and Abreu 2010]
Maximum size of paragraph			x		x		[Signore 2005]
Paragraph size			x				[Signore 2005]
Sub-heading length	x						[Signore 2005]
Total number of newlines	x						[Signore 2005]

Table 5.10: Usability metrics

Metrics' name	**Ap**	**L**	**O**	**UE**	**UI**	**Ac**	References
Total sentences	x						[Signore 2005]
Total syllables	x						[Signore 2005]
Word count	x		x				[Ivory 2001a]
Number of uppercase sentences	x				x		[Rio and Abreu 2010]
Response time			x				[Dominic and Jati 2011]
Page rank						x	[Dominic and Jati 2011]
Frequency of update			x				[Olsina and Rossi 2002]
Accessibility error						x	[Dominic and Jati 2011]
Markup validation						x	[Dominic and Jati 2011]
Number of metatags	x						[Mittal 2017]
Minimum meta keyword length			x				[Mittal 2017]
Maximum meta keyword length			x				[Mittal 2017]
Words in ALT images	x	x					[Mittal 2017]
Number of headings	x						[Mittal 2017]
Number of headings as link	x						[Mittal 2017]
The harmony between using font and the background					x		[Kamal et al. 2016]
Number of anchor tags	x						[Kamal et al. 2016]
Number of empty links (links without anchor text)						x	[Kamal et al. 2016]
Ratio of links with titles						x	[Vaucher et al. 2009]
Ratio of links with text						x	[Vaucher et al. 2009]
Presence of a search engine	x						[Vaucher et al. 2009]
Presence of a site map	x			x			[Vaucher et al. 2009]
Indication of location in site				x			[Vaucher et al. 2009]
Visited links change color	x	x		x		x	[Vaucher et al. 2009]
Link to home page	x						[Vaucher et al. 2009]
Support of back button	x			x			[Vaucher et al. 2009]
Proportion of titles chosen suitably for each icon/title		x					[Abrahão et al. 2014]
Proportion of meaningful messages (error, advise, and warning messages)	x						[Abrahão et al. 2014]
Breadth of the internavigation	x						[Abrahão et al. 2014]
Depth of the navigation	x						[Abrahão et al. 2014]
Number of colors in a page	x				x		[Sethuraman et al. 2013]
Presence of RGB color system	x				x		[Sethuraman et al. 2013]
The use of red and green colors for framing titles, fonts	x				x		[Sethuraman et al. 2013]
Underlined text is only for hyperlink				x			[Sethuraman et al. 2013]
Presence of tabbed buttons in each page			x				[Sethuraman et al. 2013]

Table 5.10: Usability metrics

Metrics' name	**Ap**	**L**	**O**	**UE**	**UI**	**Ac**	References
Presence of about info	x						[Sethuraman et al. 2013]
Not presence of auto refresh option				x	x		[Sethuraman et al. 2013]
Page resolution					x		[Gautam and Sharma 2014]
Safe color is used			x		x		[Gautam and Sharma 2014]
Use color for blind people		x		x	x	x	[Gautam and Sharma 2014]
Underlined text is used					x		[Gautam and Sharma 2014]
Label and caption for link, table and form			x				[Gautam and Sharma 2014]
Description of meta			x				[Gautam and Sharma 2014]
Plug-in support			x				[Gautam and Sharma 2014]
Using thumbnails			x				[Gautam and Sharma 2014]
Web terrific			x				[Gautam and Sharma 2014]
Abbreviation: Ap. = Appropriateness recognizability; L. = Learnability; O. = Operability; UE. = User error protection; UI. = User interface aethestics; Ac. = Accessibility							

116 metrics (out of 166) characterize, in one way or another, web application usability (Table 5.10). There is a pretty homogeneous distribution of them among the six sub-characteristics (Figure 5.11).

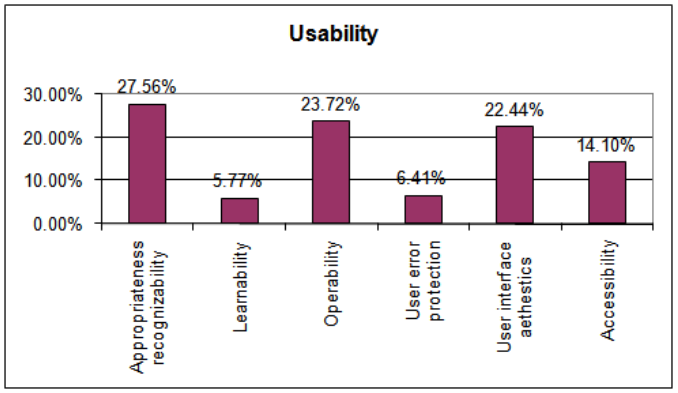

Figure 5.11: Distribution of usability metrics

5.3.2.4 Measuring Reliability

The Reliability characteristic is now composed of four sub-characteristics including availability. The latter is defined as "the degree to which a system, product or component is operational and accessible when required for use". It is a very important feature describing

reliability in all systems. Availability is part of security dimensions in some certification repositories. However, it is also part of operational safety or reliability.

Twenty-four metrics may be mapped to one or several reliability sub-characteristics (Table 5.11).

Table 5.11: Reliability metrics

Metrics' name	**Mat.**	**Ava.**	**FTo.**	**Rec.**	References
Download time of home page		x			[Bajaj and Krishnan 1999]
Download time of all pages		x			[Bajaj and Krishnan 1999]
Presence of contacts/info form	x				[Olsina and Rossi 2002]
Broken links	x		x	x	[Olsina and Rossi 2002]
Number of broken links to other sites	x		x	x	[Signore 2005]
Graphics link count	x				[Stefani and Xenos 2009]
Number of internal broken links	x		x	x	[Signore 2005]
Number of different broken links	x		x		[Olsina and Rossi 2002]
Unimplemented link count	x		x		[Olsina and Rossi 2002]
Number of orphan pages	x				[Olsina and Rossi 2002]
Quick access page = link count / page count				x	[Stefani and Xenos 2009]
Percent of dead-end web pages	x				[Olsina et al. 2001]
HTML errors per page	x				[Signore 2005]
HTML warnings per page	x				[Rio and Abreu 2010]
Presence of ALT attribute in image			x	x	[Olsina and Rossi 2002]
Non-frame version		x			[Olsina and Rossi 2002]
Response for classes	x				[Chae et al. 2007]
Response time		x	x	x	[Dominic and Jati 2011]
Frequency of update	x	x	x	x	[Olsina and Rossi 2002]
Design optimisation	x				[Dominic and Jati 2011]
Words in ALT images			x	x	[Mittal 2017]
Use color for blind people		x			[Gautam and Sharma 2014]
Abbreviation: Mat. = Maturity; Ava. = Availability; FTo. = Fault tolerance Rec. = Recoverability					

Most of them may be associated with Maturity, which is the "degree to which a system, product or component meets needs for reliability under normal operation" (Figure 5.12).

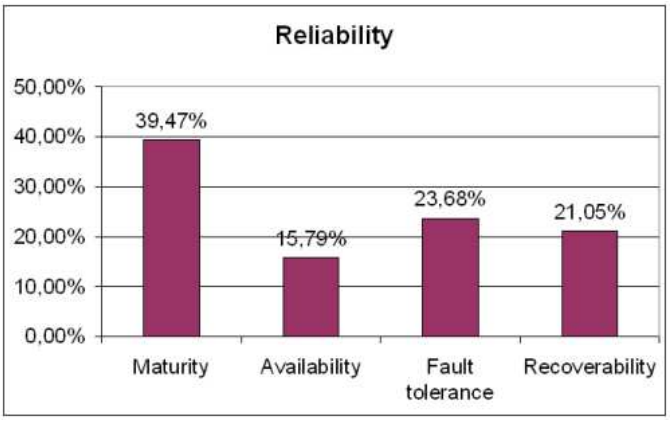

Figure 5.12: Distribution of reliability metrics

5.3.2.5 Measuring Maintainability

Maintainability is defined through five sub-characteristics whereas four were contained in ISO9126 standard. Two of them, changeability and stability, were merged to constitute Modifiability sub-characteristic. Moreover, two new sub-characteristics were adopted: Modularity defined as "the degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components"; and Reusability generally defined as "the degree to which an asset can be used in more than one system, or in building other assets".

Table 5.12: Maintainability metrics

Metrics' name	**Md.**	**Re.**	**An.**	**Mo.**	**Te.**	References
Total number of web pages				x		[Mendes et al. 2001]
Page title word count				x	x	[Ivory 2001a]
Body text words				x		[Singh et al. 2011]
Total link count				x		[Olsina and Rossi 2002]
Images count				x		[Olsina and Rossi 2002]
Page size				x		[Rio and Abreu 2010]
Total embedded links				x		[Olsina and Rossi 2002]
Number of panes regarding frames				x		[Olsina and Rossi 2002]
Within page links				x		[Singh et al. 2011]
CSS size per page				x	x	[Rio and Abreu 2010]
Image size				x		[Signore 2005]
Script size per page				x	x	[Di Lucca et al. 2004]
Number of label tags			x			[Rio and Abreu 2010]
Number of script files per page				x	x	[Reifer 2000]

Table 5.12: Maintainability metrics

Metrics' name	**Md.**	**Re.**	**An.**	**Mo.**	**Te.**	References
Number of CSS files per page				x	x	[Rio and Abreu 2010]
Presence of specific CSS to device			x	x		[Signore 2005]
Broken links			x	x	x	[Olsina and Rossi 2002]
Number of broken links to other sites			x	x	x	[Signore 2005]
Graphics link count			x	x		[Stefani and Xenos 2009]
Number of internal broken links			x	x	x	[Signore 2005]
Number of different broken links				x		[Olsina and Rossi 2002]
Number of orphan pages			x	x		[Olsina and Rossi 2002]
Percent of dead-end web pages			x	x		[Olsina et al. 2001]
HTML errors per page			x	x	x	[Signore 2005]
HTML warnings per page			x	x	x	[Rio and Abreu 2010]
Presence of ALT attribute in image			x	x		[Olsina and Rossi 2002]
Image title			x	x		[Olsina and Rossi 2002]
Image number per page				x		[Olsina and Rossi 2002]
Total number of server pages				x		[Ghosheh et al. 2008]
Total number of client pages				x		[Ghosheh et al. 2008]
Total number of form pages				x	x	[Ghosheh et al. 2008]
Total number of form elements				x	x	[Ghosheh et al. 2008]
Total number of client components (style sheet and JavaScript)				x		[Ghosheh et al. 2008]
Total number of link relationships				x		[Ghosheh et al. 2008]
Total number of submit relationships				x		[Ghosheh et al. 2008]
Total number of build relationships				x		[Ghosheh et al. 2008]
Total number of forward relationships				x		[Ghosheh et al. 2008]
Total number of include relationships				x		[Ghosheh et al. 2008]
Total number of use tag relationships				x		[Ghosheh et al. 2008]
Number of relationships over number of web pages				x		[Ghosheh et al. 2008]
Number of data exchanged over number of server pages				x		[Ghosheh et al. 2008]
Number of include relationships over number of web pages				x		[Ghosheh et al. 2008]
Lack of cohesion in methods				x		[Chae et al. 2007]
Data abstraction coupling				x		[Chae et al. 2007]
Response for classes				x		[Chae et al. 2007]
Markup validation					x	[Dominic and Jati 2011]
Number of metatags			x			[Mittal 2017]
Words in ALT images			x	x		[Mittal 2017]
Ratio of links with titles				x		[Vaucher et al. 2009]
Ratio of links with text				x		[Vaucher et al. 2009]

Table 5.12: Maintainability metrics

Metrics' name	**Md.**	**Re.**	**An.**	**Mo.**	**Te.**	References
Web terrific	x					[Gautam and Sharma 2014]
Number of links on web page of roadmap tree	x	x		x		[Kumar et al. 2015]
Cyclomatic complexity	x	x		x		[Kumar et al. 2015]
Abbreviation: Md. = Modularity; Re. = Reusability; An. = Analysability Mo. = Modifiability; Te. = Testability						

We elicited fifty-five metrics (Table 5.12) for maintainability assessment. Let's note that a few metrics are proposed specific metrics for modularity or reusability (Figure 5.13). Web application developers should refer to object-oriented programming which led to the definition of many metrics that could be adapted to the context of web development.

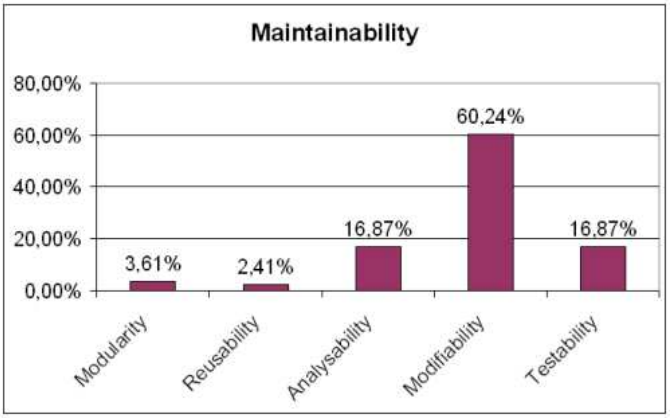

Figure 5.13: Distribution of maintainability metrics

5.3.2.6 Measuring Portability

Table 5.13: Portability metrics

Metrics' name	**Ada.**	**Ins.**	**Rep.**	References
Total number of web pages	x	x	x	[Mendes et al. 2001]
Body text words	x			[Singh et al. 2011]
Total link count	x	x	x	[Olsina and Rossi 2002]
Number of lists	x			[Singh et al. 2011]
Number of panes regarding frames	x	x	x	[Olsina and Rossi 2002]
CSS size per page	x			[Rio and Abreu 2010]
Image size	x			[Signore 2005]

Table 5.13: Portability metrics

Metrics' name	**Ada.**	**Ins.**	**Rep.**	References
Script size per page	x		x	[Di Lucca et al. 2004]
Presence of specific CSS to device	x			[Signore 2005]
Use of HTML notation in formatting	x	x	x	[Rio and Abreu 2010]
Number of div tags	x	x	x	[Signore 2005]
Average of font size in em (percentage) in CSS	x	x	x	[Rio and Abreu 2010]
Average font size in pixel in CSS	x	x	x	[Rio and Abreu 2010]
Maximum font size in em in CSS	x	x	x	[Rio and Abreu 2010]
Maximum font size in pixel in CSS	x	x	x	[Rio and Abreu 2010]
Minimum font size in em in CSS	x	x	x	[Rio and Abreu 2010]
Minimum font size in pixel in CSS	x	x	x	[Rio and Abreu 2010]
Number of items	x	x		[Dominic and Jati 2011]
The harmony between using font and the background	x			[Kamal et al. 2016]
CSS attributes	x			[Gautam and Sharma 2014]
One media in one page	x		x	[Gautam and Sharma 2014]
Abbreviation: *Ada. = Adaptability; Ins. = Installability; Rep. = Replaceability*				

The scope of the Portability dimension is reduced in the new standard since the Co-existence feature moves to the new Compatibility dimension. Hence, only three sub-characteristics describe the portability of a software. We found 22 metrics aligned with this dimension (Table 5.13). Most of them measure the adaptability (Figure 5.14).

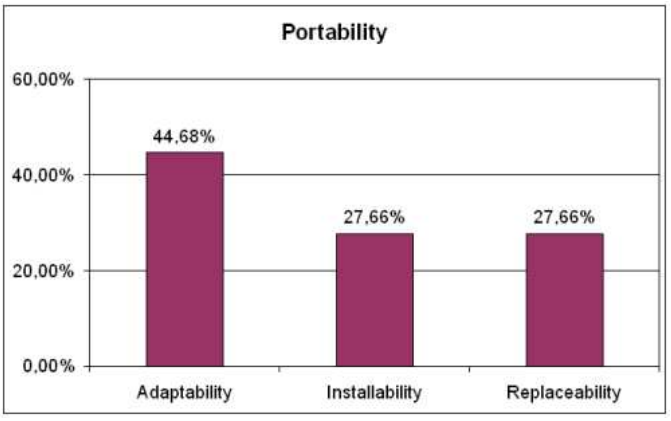

Figure 5.14: Distribution of portability metrics

5.3.2.7 Measuring Security

Security was a sub-characteristic of software quality in ISO/IEC 9126. It is promoted as a characteristic in ISO25010. In the domain of web applications, it is completely relevant since the development of B2B applications requires electronic payment that raises many security challenges. The problem is not only the integrity of the data and the process but also the authenticity and the non-repudiation of the transaction. Following the ISO security recommendations and other certification repositories, the security dimension is composed of five sub-characteristics: confidentiality, integrity, non-repudiation, accountability and authenticity. Confidentiality deals with the ability to control that only authorized users may access to information whereas integrity focuses on protecting the information from undue changes. Non-repudiation is of main concern when an electronic transaction is concluded. Accountability is defined as the "state of being answerable for the actions and decisions that have been assigned". Finally, an entity "is authentic if it is what it claims to be".

The papers dealing with web application metrics do not address the security dimension at the same level. This is the reason why we did not find many metrics measuring the security sub-characteristics. Due to the importance of this topic, it would require a specific attention and a dedicated study that we could not conduct for time reasons.

5.3.2.8 Measuring Compatibility

The new Compatibility dimension is composed of two sub-characteristics: co-existence and interoperability that come respectively from portability and functionality dimensions of ISO/IEC 9126.

Table 5.14: Compatibility metrics

Metrics' name	**CoE.**	**Int.**	References
Number of panes regarding frames	x		[Olsina and Rossi 2002]
Script size per page	x		[Di Lucca et al. 2004]
Number of script files per page		x	[Reifer 2000]
Number of CSS files per page		x	[Rio and Abreu 2010]
Presence of specific CSS to device		x	[Signore 2005]
Presence of tables inside tables		x	[Rio and Abreu 2010]

Table 5.14: Compatibility metrics

Metrics' name	**CoE.**	**Int.**	References
Number of links to other sites		x	[Alves and Ponti 2001]
Graphics link count		x	[Stefani and Xenos 2009]
Average of font size in em (percentage) in CSS	x		[Rio and Abreu 2010]
Average font size in pixel in CSS	x		[Rio and Abreu 2010]
Maximum font size in em in CSS	x		[Rio and Abreu 2010]
Maximum font size in pixel in CSS	x		[Rio and Abreu 2010]
Minimum font size in em in CSS	x		[Rio and Abreu 2010]
Minimum font size in pixel in CSS	x		[Rio and Abreu 2010]
Total number of link relationships		x	[Ghosheh et al. 2008]
Total number of submit relationships		x	[Ghosheh et al. 2008]
Total number of build relationships		x	[Ghosheh et al. 2008]
Total number of forward relationships		x	[Ghosheh et al. 2008]
Total number of include relationships		x	[Ghosheh et al. 2008]
Total number of use tag relationships		x	[Ghosheh et al. 2008]
Number of relationships over number of web pages		x	[Ghosheh et al. 2008]
Number of data exchanged over number of server pages		x	[Ghosheh et al. 2008]
Number of include relationships over number of web pages		x	[Ghosheh et al. 2008]
Lack of cohesion in methods		x	[Chae et al. 2007]
Data abstraction coupling		x	[Chae et al. 2007]
Abbreviation: *CoE. = Co-existence; Int. = Interoperability*			

Twenty-five metrics describe this dimension (Table 5.14). Compatibility is of particular importance in web applications that have to communicate dynamically together. The main objective of this communication is information exchange. Both sub-characteristics are measured by many metrics (Figure 5.15).

5.4 Conclusion

The main contribution of our research, described in this chapter, is the mapping between metrics and quality sub-characteristics. It enriches the literature by providing a fine-grained association between ISO/IEC 9126, as well as ISO25010, and the main metrics described in the literature.

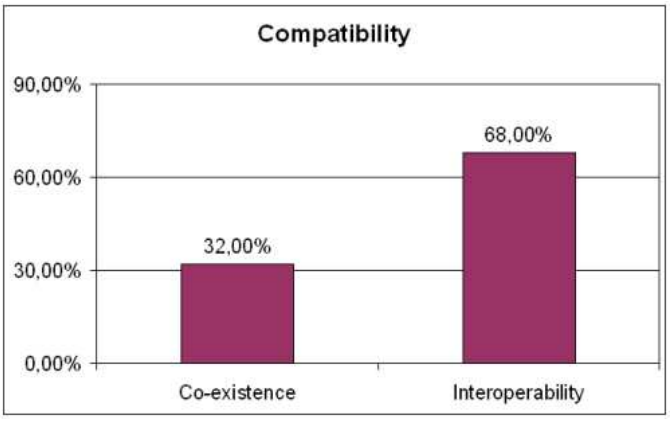

Figure 5.15: Distribution of compatibility metrics

At the end of this chapter, the reader is aware of the richness of the literature on the subject. Many metrics are defined, tested and proposed to help developers when evaluating their web applications. However, not all metrics may be easily implemented. Moreover, many metrics may not be meaningfully estimated before real-life. This is an even greater justification of our approach proposing a cyclic definition of web application quality.

Two research axes will conduct our future work. First studying security metrics, especially those dedicated to web applications. Moreover, mapping all the metrics described in this chapter to the new framework described in Chapter 4 must be performed. Finally, enriching the framework with metrics for the other axes should lead to a more largely acceptable and adoptable quality model.

Chapter 6

Guidelines for web application

6.1 Introduction

Companies develop and maintain complex web sites that allow them to communicate easily and dynamically with their customers, suppliers, partners, etc. In 2008, according to Krigsman, 24% web projects failed to be delivered within budget and 5% were unable to confirm the final cost of their web development project. Moreover, 21% failed to meet stakeholder requirements and nearly a third of web based projects (31%) were not delivered within the agreed timescales [Krigsman 2008]. More recently, a research, conducted by McKinsey and the University of Oxford on more than 5400 IT projects, concluded that 45% of large projects are over budget, 7% are over time and 56% delivered less value than predicted [Bloch et al. 2013]. The reasons vary: unclear objectives, lack of business focus (missing focus), shifting requirements, technical complexity (content issues), unaligned team, lack of skills (skill issues), unrealistic schedule, reactive planning (execution issues) [Bloch et al. 2013], inconsistent stakeholder demands, and insufficient time or budget [Krigsman 2008].

Web sites and web applications are in fact software applications. In this sense, the classical application methodologies may be used manually or with the help of computer aided software engineering (CASE) tools. However, the very specific nature of these applications led to the proposition of more dedicated approaches. Indeed, during the two last decades, research in Web Engineering brought a rich contribution composed of methods and techniques to support Web applications development. These methods such as UWE [Koch and

Kraus 2002], WebML [Ceri et al. 2000], or others are generally founded on a model-driven development paradigm, and provide models and transformation rules to handle several web applications' aspects such as data, navigation, interaction, and presentation.

However and despite the research and the tooling efforts, very few developers adopt these methods and many continue to apply ad-hoc practices. The main reason is that these approaches suffer from a lack of guidance. Even if web application designers refer to these approaches, they do not have sufficient knowledge and help in implementing them efficiently. As a consequence, the resulting applications are neither user-friendly nor easy to maintain.

We argue that the current approaches are well structured. However they need to be enriched with guidelines helping designers in the numerous decisions they have to make during the web application development. Therefore, we have collected the different sets of guidelines proposed in the literature and organized them along different dimensions. In particular, this structure allows us to link the guidelines with the quality objectives (maintainability, performance, functionality, security, etc.) and with the relevant steps of the web application design (content design, navigation design and presentation design).

This chapter is organized as follows. Section 6.2 is dedicated to related works on guidelines. Section 6.3 describes how we collected and selected the guidelines, and a short experiment we conducted on how methods and guidelines are followed in websites construction. Based on the survey conclusions, Section 6.4 motivates and describes the research question we address in this chapter. Section 6.5 describes the meta-model we propose in order to represent the guidelines in a useful way. Section 6.6 analyzes the set of resulting guidelines. Section 6.7 is dedicated to the grammar we propose for guideline descriptions. Section 6.8 sketches the prototype we developed for guideline management. Finally, the last section concludes and sketches future research directions.

6.2 Related Works

In this section, we synthesize the literature on guidelines for web site design. We organized this state of the art in two categories: first the approaches which propose guidelines

and, second, the approaches which involve such guidelines.

6.2.1 Design for Guidelines

One of the most famous works delivering guidelines for web site design is Web Accessibility Initiative (WAI) of W3C [Web Accessibility Initiative WAI]. It is a collection of standards, guidelines, and techniques for making accessible products in four categories: websites, authoring tools, browsers, and web applications. Each category has a bunch of guidelines for constructing web design and for improving accessibility. Other sources of guidelines enrich considerably the W3C recommendations [AgeLight LCC 2001][U.S. Dept. of Health and Human Services 2006].

Khlaisang is an example of research illustrating how these guidelines may be either validated or elicited [Khlaisang 2015]. The author developed user interface guidelines and a prototype for evaluating educational service websites. Based on source sites of Thailand Cyber University Project (TCU), he studied the use of sites, the website structure, the user interface design and conducted usability tests of the site. Resulting from these experiments, he presented a model of suitable website for TCU service. Starting from this website model, he designed and developed a prototype of site. The paper also mentions similar approaches.

6.2.2 Design by Guidelines

Besides works creating guidelines, other works used existing guidelines for proposing ways to improve quality of websites.

Leuthold et al. [Leuthold et al. 2008] designed enhanced text user interfaces for blind Internet users. Starting from the guidelines of web content accessibility guidelines (WCAG), they proposed enhanced text user interface (ETI) helping blind users in spending less time to complete tasks, making fewer mistakes and expressing greater satisfaction when surfing the website. This system contains nine guidelines. For blind users, this system is more usable than normal graphical user interface (GUI).

Another work building on WCAG guidelines is reported in [Sloan et al. 2006]. Using e-learning as an example, they propose a framework that guides web authors and policy makers in addressing accessibility at a higher level, by defining the context in which a Web

resource will be used and considering how new alternatives may be combined to enhance the accessibility of the web site.

After a brief description of the 14 guidelines of WCAG (version 1), Radosav et al. discussed the choice of colours for adjusted web design [Radosav et al. 2011]. They classified colours into several groups and concluded that colours, which cannot be differentiated by people with colour discrimination disability, should not be placed next to each other.

As a conclusion of this brief state of the art, research in this field is prolific and aims at i) proposing guidelines for web site designers, ii) enriching existing ones, iii) implementing guidelines into more comprehensive approaches, iv) evaluating guidelines through experiments. To the best of our knowledge, we did not find any paper proposing a meta-model, a grammar, and a tool allowing web application designers to put together the different guidelines as a first step for their reuse in an automatic way.

6.3 An experiment on guideline usage

Before defining the research question addressed in this chapter, we performed a quick inventory on how well web design best practices and guidelines are followed by existing websites. The objective was i) to analyze whether existing practices and guidelines are used and ii) identify how to facilitate their adoption and hence avoid ad hoc approaches. Thus, we first collected 475 guidelines from several sources and confronted them with three websites: the web site of our university department (deptinfo.cnam.fr), the website of a French newspaper (lemonde.fr) and a well-known e-commerce web site (amazon.fr). We first describe briefly the collected guidelines and then their verification on the three websites.

6.3.1 Collecting the Guidelines

World Wide Web Consortium (W3C) is the main international standards organization for the World Wide Web. This consortium gathers around 400 organizations. They developed Web Content Accessibility Guidelines (WCAG) with the goal of proposing a single shared standard for web content accessibility that meets the needs of individuals, orga-

nizations, and governments (Web Accessibility Initiative). Two versions of WCAG were published until now. The first one was introduced in 1999. It contains 14 large guidelines. Each main guideline is composed of atomic guidelines addressing the same topic. The second version was published in 2008. It contains 12 guidelines organized into four categories, targeting four desirable characteristics of websites: perceivable, operable, understandable, and robust.

WCAG defines three levels of conformance, respectively A, AA and AAA. Some of the related guidelines can be automatically checked whereas others require manual checking. Authors in [Trulock and Hetherington 2008] conducted a case study on Irish websites showing that web designers are aware of web accessibility but they concentrate their efforts on ensuring validation of automatically controlled checkpoints and ignore those requiring additional manual testing.The guidelines of WCAG focus only on accessibility. Thus, we collected other guidelines which address all the characteristics of web site quality. The literature contains guidelines for specific web sites (for children for instance) as well as rules available for all sites.

6.3.1.1 Identifying the Relevant Sources

For collecting guidelines from literature effectively, we use some keywords when searching, such as "website guideline", "guideline for website", "guideline security web application" in title and content of document, from main electronic libraries and databases in computer science: IEEE Xplore, Springer, ScienceDirect, ACM, and DBLP. As an example, based on the keywords "web" and "guideline", we have 1273 results from IEEE, 273 results from ScienceDirect and 168 results from DBLP. With Springer and ACM, we have much more results in many domains, so we had to refine the results and choose results with high relevance (as computed by the search engines). Then we defined inclusion criteria for selecting sources (primary studies) and rejecting the other ones. The inclusion criteria are presented in the table below (Table 6.1).

We found several guideline lists published since 2000. However, these documents are sparse and address many domains. One objective is to gather them, categorize, and model guidelines. Thus they will be more usable for supporting web application developers.

Table 6.1: Inclusion criteria

Criterion	Description
C1	The study focuses on guideline definition for web sites
C2	The study mentions quality characteristics of web sites
C3	The paper is recent, i.e. published since 2000
C4	The paper proposes original guidelines (does not only mention guidelines from other studies)

Some guidelines are general and others are dedicated to specific domains: education, international, or for particular ages (children or seniors). As an illustration, the guidelines of AgeLight Company are divided in six categories: layout and style, color, text, general usability testing, accessibility and disabilities, user customization [AgeLight LCC 2001]. Web sites for old people are the research object of a number of studies [Xie et al. 2011][Sun and Zhao 2010]. Meloncon et al., in contrast, concentrated on guidelines for children [Meloncon et al. 2010]. Maguire focused on e-commerce international sites [Maguire 2011]. Some papers focused on the characteristics of quality directly, such as [Chiuchi et al. 2011] which targeted portability and efficiency. [Radosav et al. 2011] capitalizes on the 14 guidelines from WCAG, so we did not collect them. Finally, we took into account fourteen sources. Their analysis is described below.

6.3.1.2 Extracting the Appropriate Guidelines

Our systematic search followed by a scan of sources allowed us to exhibit fourteen papers containing relevant guidelines. The next step consisted in studying all the guidelines and selecting the helpful ones. In each source of guidelines, we found some obsolete guidelines or some recommendations which were out of our scope. For example, in [U.S. Dept. of Health and Human Services 2006], guidelines in the last part (part 18), such as "Use an iterative design approach" or "Solicit test participants' comments" were not selected, since they are too general or dedicated to testing. So we eliminated them from the list.

We found 14 sources with 475 guidelines. The number of guidelines of each source is presented in Table 6.2. In some cases, we split some guidelines, hence the number of selected guidelines may be higher than the number of guidelines proposed in these papers.

Some sources propose general guidelines. Others are more specific. For example

Table 6.2: Source, number and scope of guidelines

Source	Proposed guidelines	Selected guidelines	Scope
[AgeLight LCC 2001]	53	35	General
[Bargas-Avila et al. 2010]	20	20	General but concentrating on web forms
[Chiuchi et al. 2011]	17	15	General / focusing portability and efficiency
[Carnegie Mellon University cited January 2017]	7	8	University
[U.S. Dept. of Health and Human Services 2006]	196	209	General
[Leuthold et al. 2008]	9	9	Blind people
[Lokman et al. 2009]	13	14	General
[Maguire 2011]	20	8	International site
[Meloncon et al. 2010]	21	11	Children
[Microsoft Developer Network cited January 2017]	50	49	General
[Ministry of Community and Social Services of Ontario 2012]	11	11	General
[Ozok and Salvendy 2004]	20	20	General
[Sun and Zhao 2010]	31	31	Old people
[Xie et al. 2011]	7	10	Old people / medical information

[Bargas-Avila et al. 2010] concentrates on web forms or [Chiuchi et al. 2011] focuses on portability and efficiency.

Some guidelines are too complex, so we had to divide them into two parts or more. For example the guideline for images in [Chiuchi et al. 2011] is separated into two atomic guidelines: "The preferred use of JPEG and GIF images" and "The resolution of image should be set correctly inside the tags".

6.3.2 Analyzing the Guidelines Usage

To analyze how well the guidelines are applied in practice, we defined four levels, namely: Yes, No, Partial and NN. Yes means that the site satisfies completely the guideline, No means that this site does not satisfy it, Partial means that this site partially meets the guideline and NN means that "We don't know", since either the guideline cannot be applied

to the site or we don't have enough information.

The result is synthesized at Figure 6.1. We confronted all guidelines to the three websites, in order to evaluate how many guidelines were followed by the three web application developers. Thus, 206 guidelines are verified on the three selected sites. 33 guidelines are completely observed by two web sites and partially by the third. 46 guidelines are observed by two web sites. 47 guidelines are not respected on the three selected sites. But let us remind that 3 guidelines are dedicated to international or children sites, and thus are not required in the three tested web sites. Besides them, there are 83 guidelines that we could not verify, since we don't have access to the administration of the web sites. Regarding this last category, many guidelines are related to the security aspects. To check if they are fulfilled, we require the admin authority, so we cannot conclude about these guidelines.

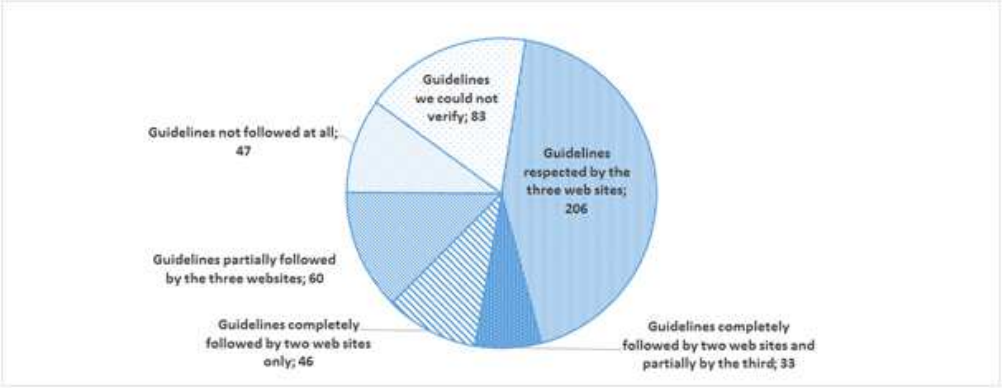

Figure 6.1: Confrontation of guidelines to three web sites

As an illustration, the guideline G115 "considering both levels: 'high' and 'low' of cultural context for satisfying both viewpoints" or G176 "Limit navigational topics" are not relevant for the three web sites. Others may be irrelevant, such as G217 "Inform users of long download times" or G247 "Limit homepage length" since we had high speed connection for our tests.

Figure 6.2 compares the scores obtained by the three websites if we consider the rule: the more guidelines the web site complies with, the better score it obtains. deptinfo.cnam.fr fully respects 246 guidelines while lemonde.fr respects 268 guidelines. Finally, amazon.fr is the best one since it respects completely 284 guidelines (Figure 6.2).

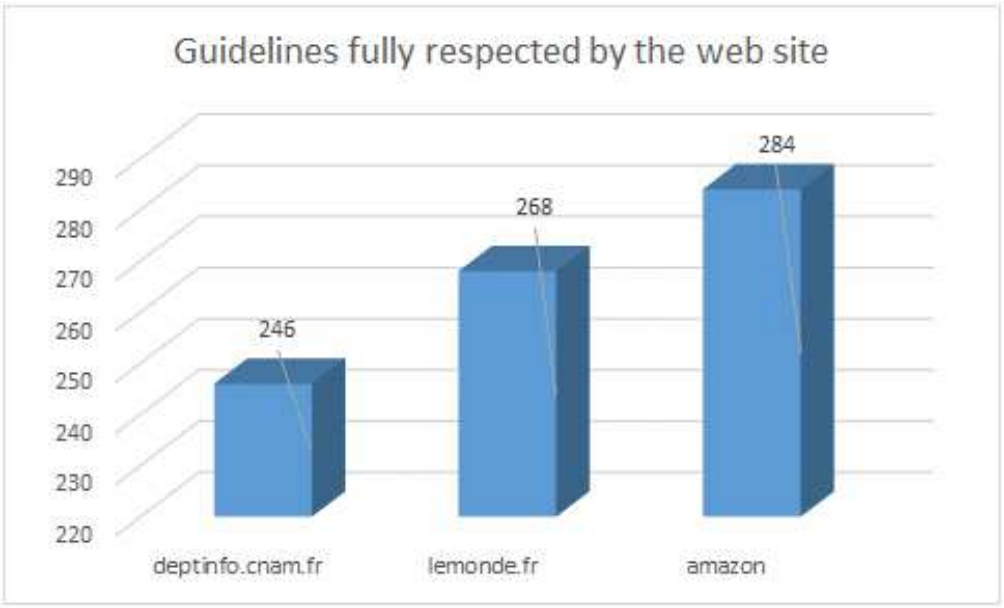

Figure 6.2: Guidelines fully respected by the three web sites

If we also consider the guidelines partially observed by the web sites, we obtain the scores of Figure 6.3. deptinfo.cnam.fr is aligned totally or partially with 283 guidelines whereas lemonde.fr is in accordance with 306 guidelines. Finally, amazon complies totally or partially with 317 guidelines.

These figures show that either these guidelines are not considered as references or these websites still face quality problems. As an example, let us mention G345 "Provide auto-tabbing functionality" for increasing users' convenience and G362 "Using photographs of people" for increasing users' reliability. The three websites are not aligned with these two guidelines. That means perhaps that these guidelines which were validated through complex processes are not sufficiently known by web site designers.

6.4 Research Questions

From the middle of 1990s, methods and approaches have been created for helping developers to build web applications more easily and constructively. The Object-Oriented Hypermedia Design Method (OOHDM) was one of the first methods proposing a rigorous process from requirements elicitation to implementation including navigational and inter-

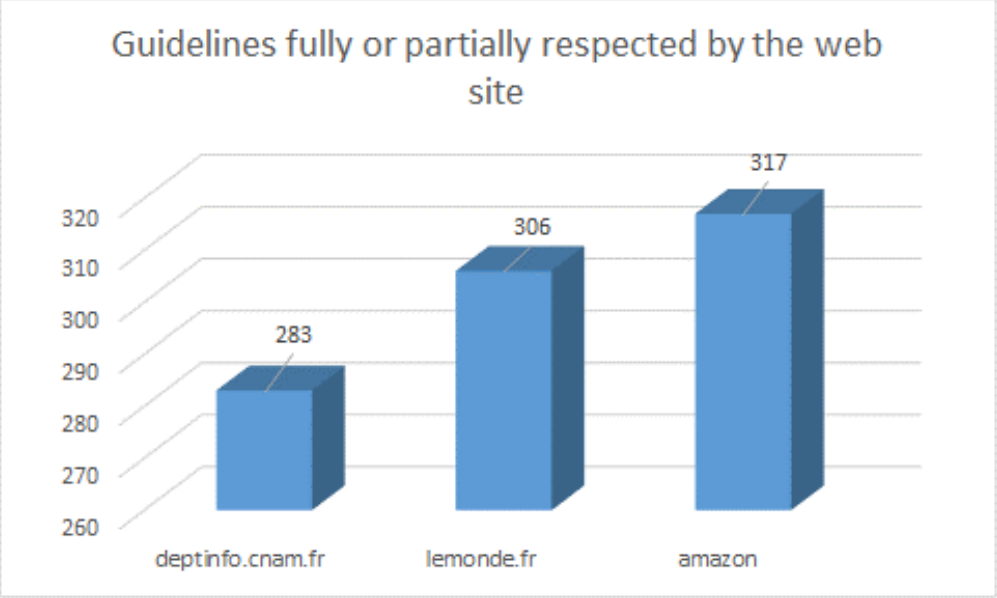

Figure 6.3: Guidelines respected by the three web sites

face design [Schwabe and Rossi 1995]. The method relies on object-oriented principle and proposes notation mainly derived from UML. The transition from models to specification is not supported and thus requires a considerable effort.

The Web Modelling Language (WebML) is a model driven web engineering method dedicated to data-intensive web applications [Ceri et al. 2000]. WebML is one of the most used web engineering methodologies. It is supported by a development framework, Ratio5 [Acerbis et al. 2007] that is fully integrated to the Eclipse framework. Several extensions of the first version have been proposed offering a rich modeling approach for developers. However, the method relying very few on standards, it led to a proliferation of proprietary notations increasing the method complexity.

The UML-based Web Engineering (UWE) methodology [Hennicker and Koch 2000] is a model-driven Web Engineering approach. It relies heavily on UML and is extensively related to standards. The model driven orientation allows generating platform specific implementation through dedicated transformation rules. Model driven approaches are based on four levels of abstraction: the computer independent model (CIM), the platform independent model (PIM), the platform specific model (PSM), and the code. Some methods address only the CIM level, other methods focus on the PIM level. In the same way, some

methods deal with the transformation of CIM to PIM (e.g. NDT, OOWS), others address the transformation of PIM to PSM (e.g. WebML, UWE) and others incorporate the transformation of PSM to code (e.g. OOHDM, UWE) [Aragon et al. 2012]. Even if these methods offer a real support, they are still not used by practitioners probably since they are complex and they do not provide designers with sufficient guidance.

We argue that most methods do not provide their users with relevant guidance in the design and development process. Either in the same approaches or in other sources, researchers propose many guidelines in order to help designers and developers. These guidelines may be very helpful to support them and to convince designers to use the methods that embed them.

Thus the research question we address in this chapter may be defined as follows: "How to structure all the existing guidelines helping website designers to understand and apply them?" To answer this question the experiment presented in Section 6.3 helped us to elicit the main characteristics of these guidelines. We then defined a meta-model allowing us to represent this knowledge. Finally we categorized the selected guidelines based on our meta-model. This categorization aims to facilitate their reuse. Then we defined a grammar enabling to model all these guidelines and serving as a basis for our guideline management prototype. This prototype, described below, is a first answer to our second research question: "Can we help the web application designers by providing them with a tool for managing literature guidelines enriched with their guidelines?"

6.5 Guideline Capitalization: A Model-Based Approach

In the literature, we find different ways to describe guidelines: in [Chiuchi et al. 2011], they are represented by three attributes: Category, Name and Content. Meanwhile in [Ekberg et al. 2010], a guideline has three parts: design/application solutions, objective and description. We argue that this descriptive information is not sufficient to facilitate the reuse of guidelines by web application designers. In particular, the latter must find easily the guidelines using different criteria. For example, in case of designing a web application for blind people: which recommendations do they have to take into account? If developers

want mainly to facilitate the maintainability of the web application: which guidelines aim at this objective? Etc.

We first propose a model helping capitalizing and structuring the guidelines. The meta-model is depicted at Figure 6.4.

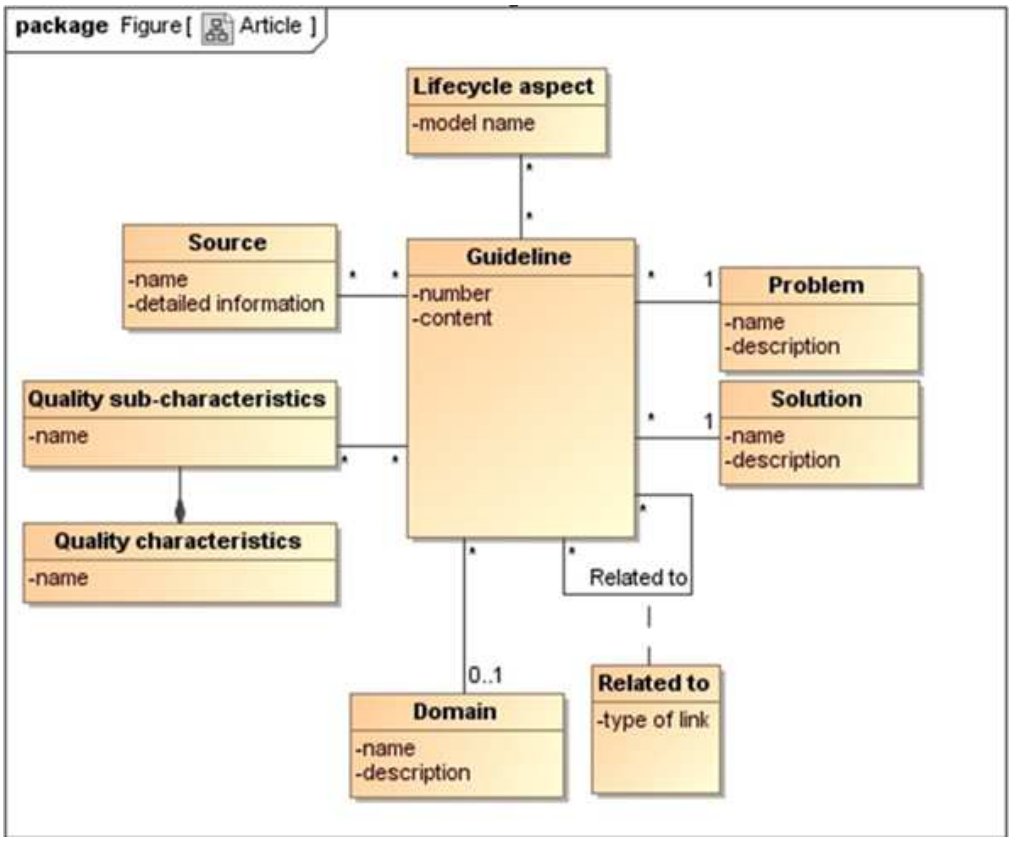

Figure 6.4: The meta-model of guidelines

Following the general description of patterns for decision processes [Harrison et al. 2007], we propose to link each guideline with the following categories:

- The source where the guideline was found,
- The quality characteristics and sub-characteristics that the guideline addresses,
- The problem it aims to solve,
- The solution proposed,
- The particular domain concerned if any,

- The lifecycle aspect, meaning which web application model (content model, navigation model, presentation model) it deals with.

This structure will constitute a knowledge base for automatic reuse through a web application design tool. The meta-model is represented as a UML class diagram at Figure 6.4. The *related to* relation between guidelines allows us to represent potential links between guidelines. Thus the attribute *type of link* may take the values "in contradiction with", "specializes" or "similar to".

Each guideline solves a problem; however several guidelines may tackle the same problem. The solution of the guideline describes the rules to be applied. As explained above, in our process, we split some guidelines such that each resulting guideline recommends one and only one solution. The domain may be general or it may be a specific one. The quality characteristics (functional suitability, performance/efficiency, compatibility, usability, reliability, security, maintainability, portability) and sub-characteristics refer to ISO 25010 for software quality. For space reasons we do not list all of them. Some guidelines are common to several sources, hence the multiplicity of the relation here is many-to-many. Finally, the lifecycle aspect consists of three elements: Content, Navigation, and Presentation.

In order to illustrate, let us describe the guideline G37: "For body copy, the recommended faces for the web, in order of preference, are Verdana, Arial and Helvetica. The browser should use Verdana first; if it is not available, use Arial and then Helvetica. If none are available, use another Sans serif font" (Figure 6.5).

Number: #37
Content: For body copy, the recommended faces for the web, in order of preference, are Verdana, Arial and Helvetica. The browser should use Verdana first; if it is not available, use Arial and then Helvetica. If none are available, use another Sans serif font.
Problem: Choosing appropriate font for a website
Domain: web for university (even if it can also apply to other types of site)
Lifecycle aspect: Presentation
Quality sub-characteristics: User interface aesthetics
Quality characteristic: Usability
Solution: Choose Sans serif font, namely Verdana, Arial and Helvetica.
Source: (Carnegie Mellon University)

Figure 6.5: Example of guideline

6.6 Guidelines analysis

In this section, we provide the reader with an analysis of the guidelines according to the different dimensions of our meta-model. Let us remind that our selection process led to the constitution of a set of 475 guidelines (the guidelines can be found at Annex B).

If we analyze them from the lifecycle dimension (Content/ Navigation/ Presentation), we counted 203 guidelines for Presentation, 291 guidelines for Content and only 40 guidelines for Navigation. Some guidelines address more than one model. Hence the total exceeds 475 (Figure 6.6).

The 475 guidelines were mapped with quality sub-characteristics. Some guidelines are mapped with several sub-characteristics. The characteristic Usability, with sub-characteristics Operability and User interface aesthetics is the most involved one. It is easy to explain since many papers address interface aspects (User interface aesthetics) and aim to build easy-to-use interfaces (Operability).

Many guidelines are about font (G37, G42, G49, G50, etc.) and color (G6, G8, G39, G41, G86, G185, G186, etc.) of websites. White is the color which is not recommended (G9, G39, G189, etc.).

We can detect some contradictory guidelines, since some guidelines aim at different goals. In the guidelines of a university (Carnegie Mellon University) the documents should be opened in new windows (G35), probably for legal responsibilities. It is opposite to guideline G101 [AgeLight LCC 2001] which recommends not to open external links in new windows, since it can cause user distracting.

Guideline G37 recommends using only Sans-serif font, but meanwhile G85 accepts serif font in web site for printing.

Some guidelines are dedicated to different types of users, but finally they have same contents. As an illustration, Sun et al. [Sun and Zhao 2010] focused on website for old people; meanwhile Meloncon et al. [Meloncon et al. 2010] concentrated on web applications for children. Old people and children are two types of users who have some specific characteristics in comparison with others (e.g. not being able to understand complex con-

tent).

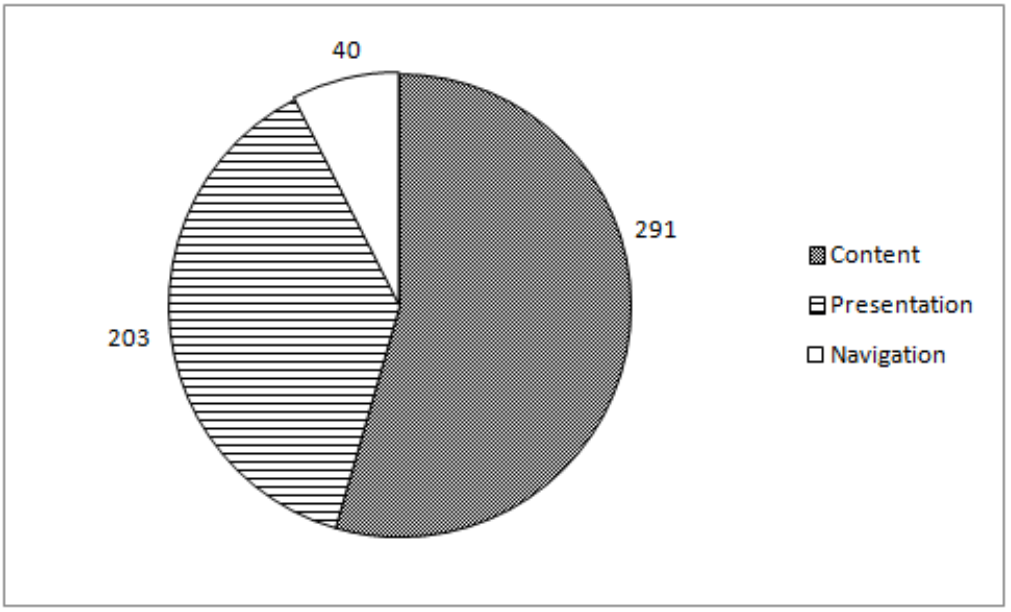

Figure 6.6: Percentage of guidelines per lifecycle aspect

The guideline about Security of web applications in MSDN of Microsoft (Microsoft Developer Network) contains about 50 sections. Many of them address Integrity (prevent unauthorized access) (in 38 sections) and Confidentiality (data are accessible only to those authorized) (in 18 sections). This is due to the fact that Integrity and Confidentiality are important for web applications which are designed for many kinds of users and also are the targets of attacks.

Among the eight quality characteristics, Compatibility is not mentioned at all, since guidelines focus on the site itself, and not on the relation of the site with other sites or other applications (scope of Compatibility).

6.7 Guideline Description Grammar

The previous sections of the chapter capitalized on guidelines found in the literature. In order to facilitate their acquisition and to enrich them, we propose to structure each guideline as a sentence. These sentences must use natural language (English here) but they must be easy to understand by referring only to simple structures. To define such

structures, we propose a grammar in this section. We based our grammar on Pohl's four rules [Pohl 2010] which allow designers to document scenarios:

- Rule 1: Use the present tense
- Rule 2: Use the active voice
- Rule 3: Use the subject-predicate-object (SPO) sentence structure
- Rule 4: Avoid modal verbs

However, rule 4 is adequate for scenarios but not for guidelines which actually have to contain different modalities defined using modal verbs. Thus, we applied only the first three rules.

6.7.1 Guideline Grammar Backus-Naur notation

Based on these three rules, we screened the whole literature guidelines and built a grammar using an inductive process. This grammar is presented with Backus-Naur Form. Backus-Naur notation (more commonly known as BNF or Backus-Naur Form) is a formal way to describe a language, which was developed by John Backus [Marcotty and Ledgard 2012]. It is used to define the grammar of a language formally, so we can use it for describing our grammar of guidelines. A guideline is composed of three components (Figure 6.7): the first part, the main part, and the complement part.

```
<guideline> ::= <first part> <main part> <complement part>
<first part> ::= <modal verb> | <modal verb> ‘not’ | ‘do not’ | ∅
<modal verb> ::= ‘should’| ‘must’ | ‘have to’
<main part> ::= <verb> <main part complement>
<main part complement> ::= <main part complement> <comma> |
  <adjective>* <noun phrase> <adverb>*
<complement part> ::= <preposition> <body of complement> | ∅
<comma> ::= ‘,’
<noun phrase> ::= <determiner> <pre-modifier> <noun>
  <complement of noun phrase> |
  <determiner> <pre-modifier> <noun> <post-modifier>
<body of complement> ::= <clause> | <gerund phrase>
<clause> ::= <noun phrase> <verb phrase>
<gerund phrase> ::= <gerund> <complement of gerund phrase>
<complement of gerund phrase> ::= <noun> | <pronoun> | <adverb>
<gerund> ::= <verb>’-ing’
<determiner> ::= ‘a’|’an’|’the’
<pre-modifier> ::= <adjective> | <noun> | ∅
<post-modifier> ::= <adverb> | <prepositional phrase> | <clause>
<complement of noun phrase> ::=  <prepositional phrase> | <clause>
<verb phrase> ::= <verb> | <auxiliary verb> <gerund> |
  <auxiliary verb> <past participle verb> |<modal verb> <verb>
<prepositional phrase> ::= <preposition> <noun> | <preposition> <pronoun>
```

Figure 6.7: BNF description of the guideline grammar

The first part is a modal verb (must, have to, should) depending on the level of the recommendation. It is optional. The guideline may be expressed as a negative sentence. The main part of the sentence is composed of a verb and a complement. The main part complement may be composed of several parts with adjectives, noun phrases and adverbs. Finally, the sentence may contain a complement part. The verb may be any verb of the dictionary. A closed list of already used verbs is proposed, but it is an open list. In the same way, the sentence may contain prepositions, adjectives, nouns, adverbs, pronouns, auxiliary verbs, and past participle verbs.

Table 6.3 illustrates some examples of guidelines which fit the proposed grammar.

Table 6.3: Some examples of guidelines corresponding to grammar

Number	Guideline	Grammar
G16	Avoid / pull down / menu	Verb + pre-modifier + noun
G5	Provide / a / text / equivalent for / images	Verb + determiner + noun + preposition + noun
G66	Should / use / longer / pages / for / content pages	Modal verb + verb + pre-modifier + noun + preposition + noun
G325	Do / not / make / user-entered / codes / case sensitive	Do + not + verb + pre-modifier + noun + post-modifier

6.7.2 Pre-processing of raw guidelines

When collecting guidelines from literature, we performed a pre-processing of guidelines which did not satisfy the grammar we proposed. We divided long guidelines into several shorter guidelines. We transformed some guidelines, for example the relative position of elements of clauses in order to follow the rules of the grammar while preserving their meanings.

The simplest form of guidelines is Verb + Noun. An example is guideline G20: "Provide a site-map". A more sophisticated form is guideline G17: "Do not use a deep hierarchy and group information into meaning categories".

The guideline "Left justified text, text line should not be long" was split into two guidelines: "Justify left text" and "Do not use long text line".

Thus, we harmonized the guidelines extracted from the literature in order to facilitate their understanding and their appropriation by web application designers. In the following section, we describe the tool making these guidelines available.

6.8 Prototype description

We propose to make the guidelines available through a web tool allowing web application designers to add, query, and verify guidelines. The prototype of this tool is described below. It contains three modules for respectively adding, verifying, and querying guidelines.

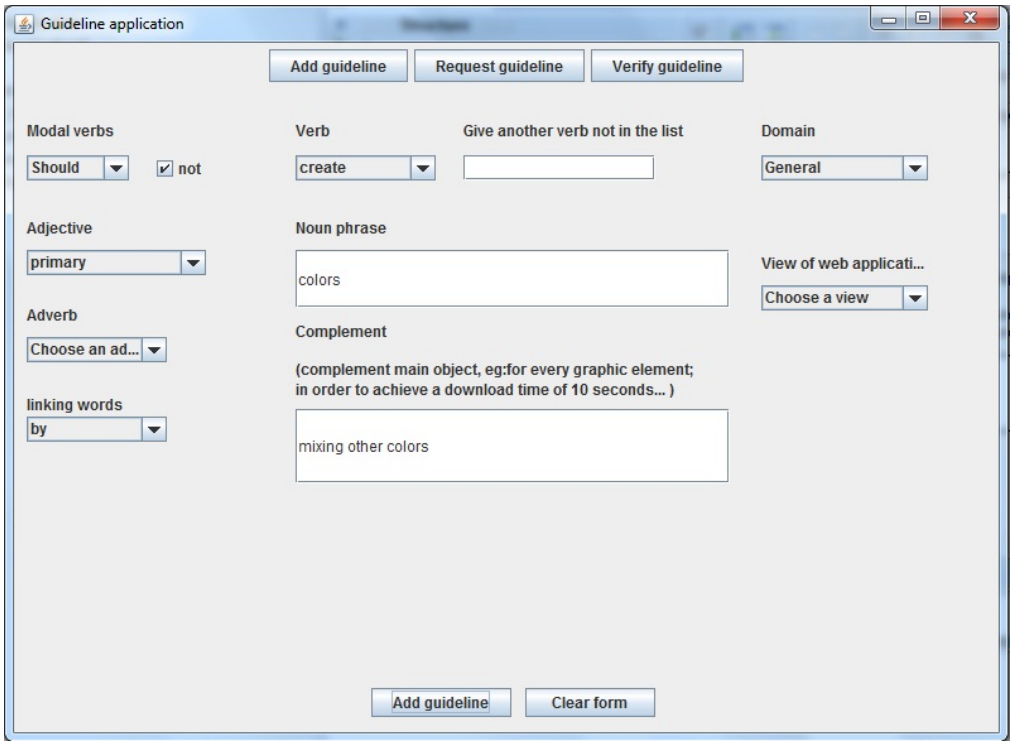

Figure 6.8: Adding a new guideline

6.8.1 Add guidelines

The first module allows the user to enter new guidelines. The basic syntax of the sentence is made available through a screen form (Figure 6.8). One example is adding guideline G73: "Should not create primary colors by mixing other colors". We choose "Should" from modal verbs, tick "not", choose "create" in verb list (or can add new verbs not in the list), choose "primary" from adjective list, add "colors" in the Noun phrase box. There is no adverb in this guideline. The complement is "by mixing other colors", so we choose "by" from "linking words" and "mixing other colors" in the complement box.

We choose the button "Add guideline" and this guideline is added into the pending list. We turn into the "Verify guideline" part (Figure 6.9).

6.8.2 Verify guidelines

In the first version of the prototype, the verification process is limited to finding existing similar and/or contradictory guidelines and presenting them to the user. The similar

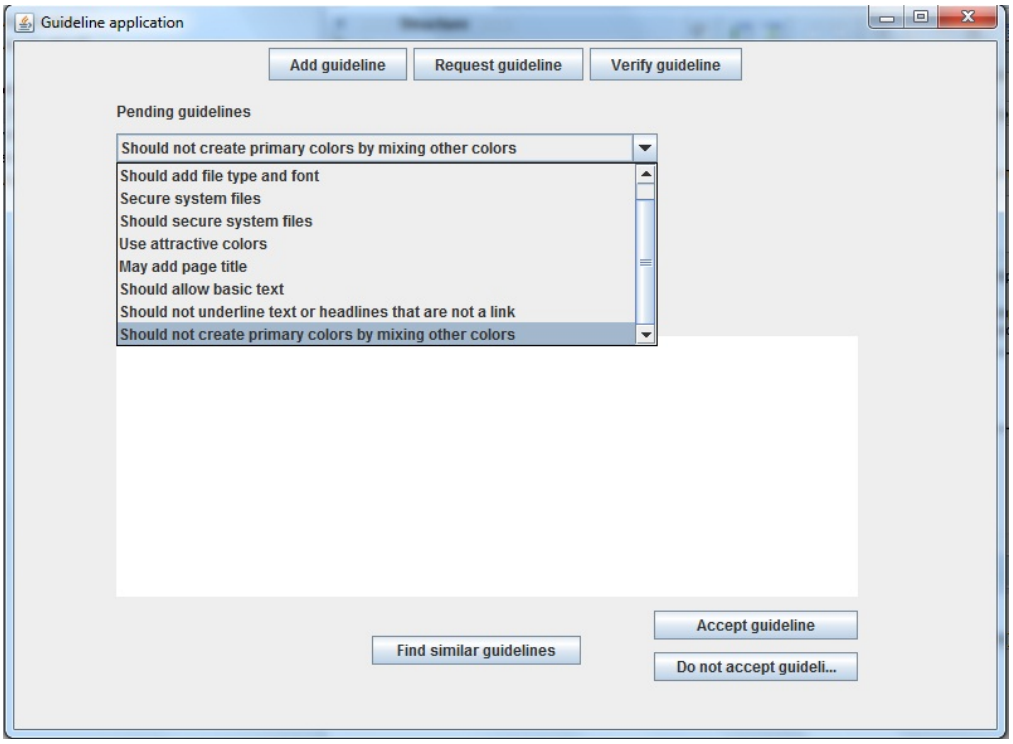

Figure 6.9: Check guidelines

and/or contradictory guidelines are extracted by comparing the different components of the sentence based on Levenshtein distance [Yujian and Bo 2007]. The prototype lists all existing guidelines whose distance's value passing defined threshold. Here, we chose a threshold equal to 0.5. The guideline is in the pending list and we select the button "Find similar guidelines". The result appears in the below box: the only similar guideline is itself: this guideline is new and we can accept it (Figure 6.10).

As an illustration, we can add another guideline which is similar to this guideline. It is "Should not create secondary colors by mixing many colors". The result box lists two results: the first is the guideline G73 that we have just added before with the distance 0.81 and the second is the guideline to be added. After comparing it with guideline G73, we can accept the new guideline (Figure 6.11).

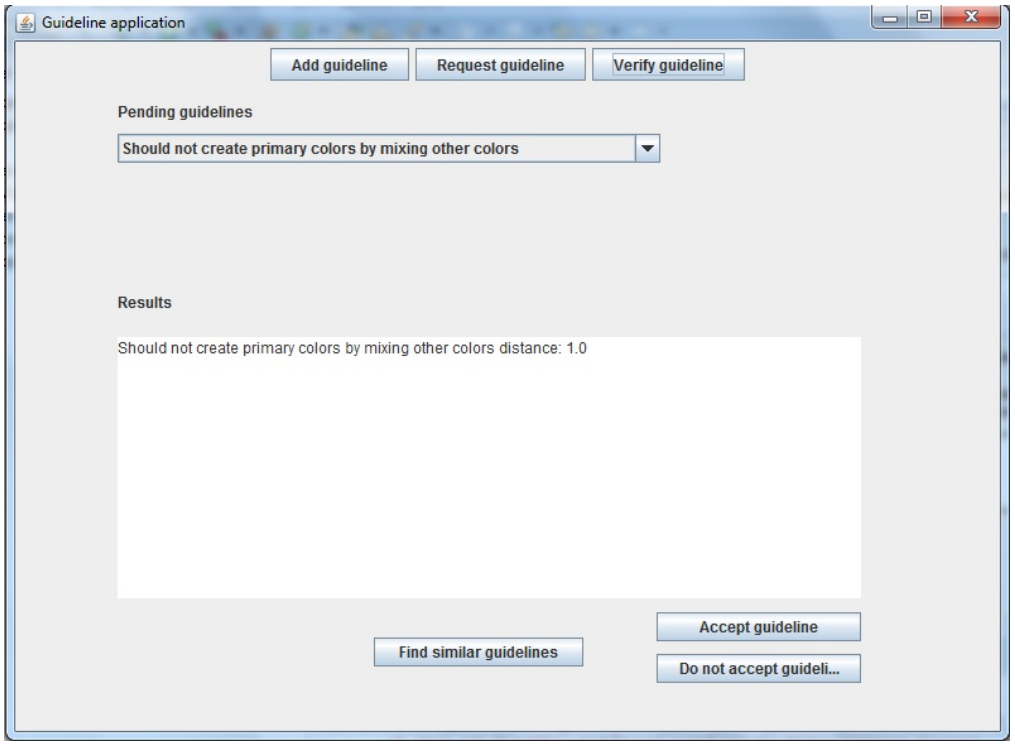

Figure 6.10: Verifying guidelines

6.8.3 Request guideline

With the "Request guideline" function, we can query the guideline base using between one and three criteria, which are the domain (general, children, etc.), the view (presentation, content, navigation), and a keyword (a word contained in the guideline). Figure 6.12 shows the result when only the keyword color is entered.

6.9 Conclusion and future research

Companies grasp the importance of having usable and efficient web applications. Thus, their development and maintenance is of high importance. The academic literature on the subject contains hundreds of guidelines aiming at helping web site designers. The research question we addressed in this chapter may be expressed as follows: How to structure the existing guidelines helping website designers in order to facilitate their application? As a first contribution, we defined a meta-model allowing us to describe each guideline with

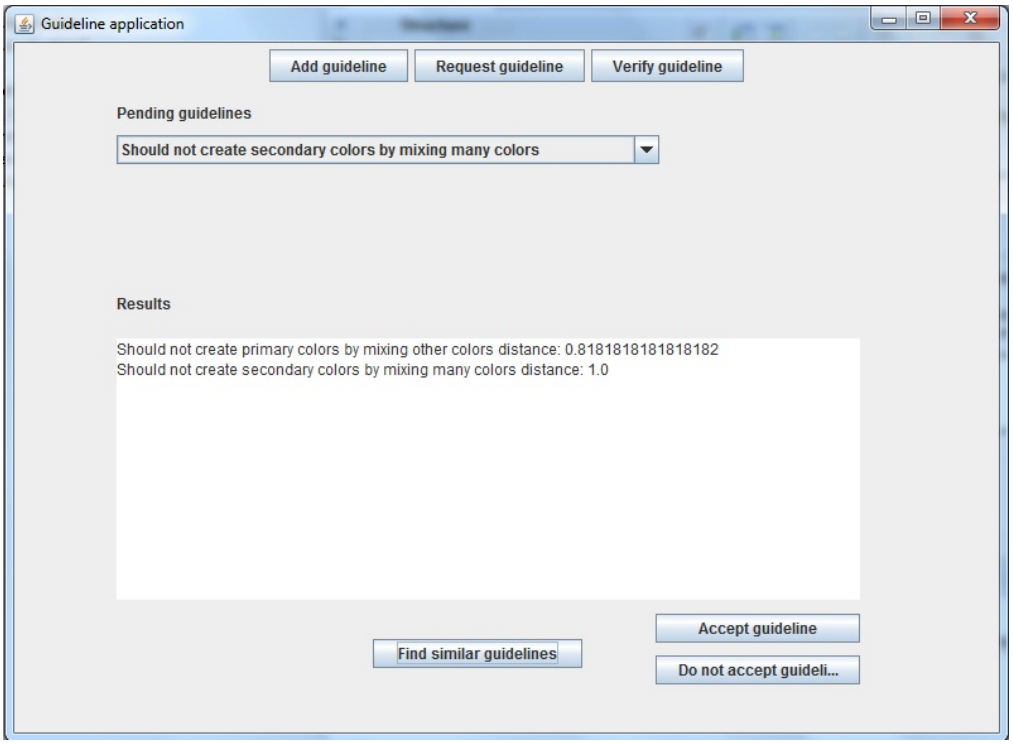

Figure 6.11: Verifying guidelines: another example

six dimensions: the problem it addresses, the solution it proposes, the lifecycle aspect it deals with, the target quality characteristics, the source it comes from, the potential links (similarity, contradiction, specialization) with other guidelines. Our search and selection process allowed us to define 475 such guidelines and to feed our meta-model with them. This required the mapping of them with the relevant quality sub-characteristics. As a first evaluation of these guidelines, we checked whether they were compliant with three very different web sites. Second, we proposed a grammar for homogenizing the guideline description. Finally, we developed a prototype allowing us to store such guidelines and providing users with simple access to the guidelines as well as the possibility to enrich them with new guidelines.

This research presents some limitations. It is rather easy to check the contradiction between guidelines attached to the same quality characteristics and/or sub characteristics. However, contradictions may also occur between guidelines associated with different quality characteristics. Moreover, some guidelines may become obsolete due to new technical

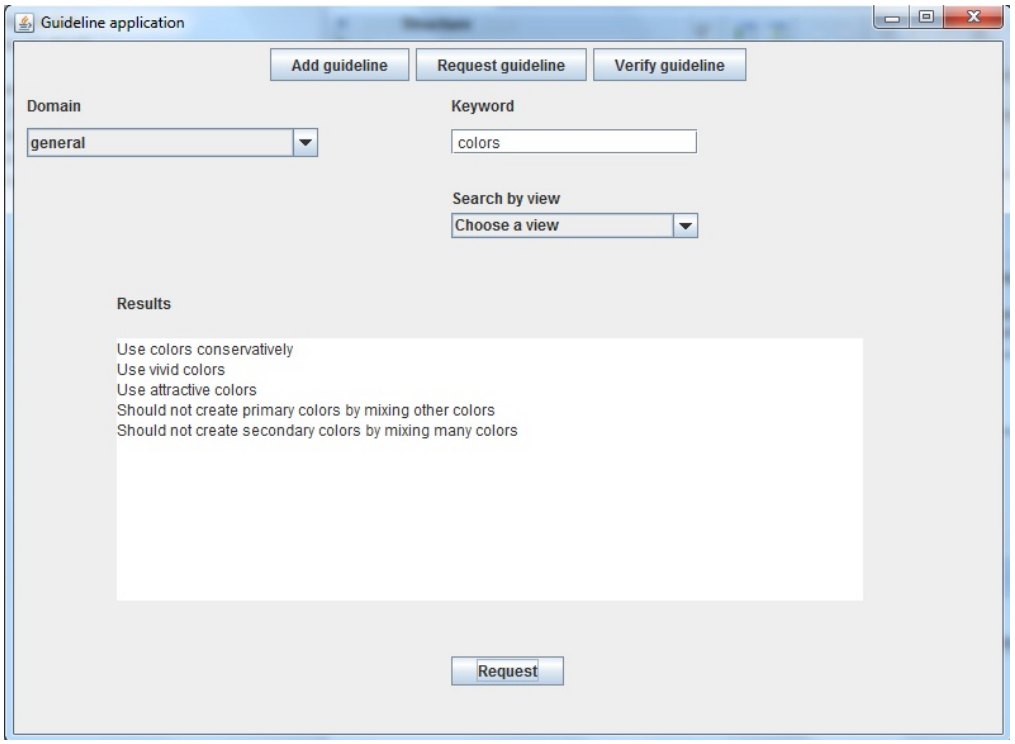

Figure 6.12: Querying the guideline database

opportunities. It is not easy to ensure an easy update of guidelines.

Future research will explore three directions: first, the implementation of these guidelines in a CASE tool implementing UWE web application design method; second, a validation of the approach through an experiment with web site designers, in order to evaluate how the guidelines help them when using the CASE tool; third, we would like to build an audit tool for automatically checking the quality of web applications thanks to our guideline database.

Chapter 7

Conclusion and perspectives

It's hard to browse a newspaper or magazine without finding an article about the digital revolution in business and society. In reality, this evolution is progressive. In particular, the Internet is now several decades old and gradually all businesses and organizations have one or more websites. Depending on the area of activity and ambition, these sites, from simple static showcase sites, become web applications that require a significant and constant investment to ensure quality. It is precisely this subject of the quality of web applications that we have addressed in this thesis.

After recalling its importance and context in the introduction (Chapter 1), we proposed a state of the art in Chapter 2, taking stock of the concepts, approaches, models, and methods proposed in the literature to provide designers and developers of web applications with structuring foundations likely to improve the quality of their applications. As in all contemporary quality approaches, it is not a goal achieved once and for all, but rather a continuous effort to improve the design, development and maintenance process. This is why we have proposed an iterative cyclical approach described in Chapter 3. Our approach is composed of three main steps: i) define web application quality, ii) measure web application quality, and iii) improve web application. Chapter 4 details the first phase of this cycle by proposing a new framework for defining the quality of a web application. In Chapter 5, we focus on the most operational level of this framework by studying a large number of metrics to measure the quality factors of web applications. Finally, Chapter 6 is dedicated to improving quality by structuring the guidelines available to web application designers.

In this thesis, we have made several contributions to the field of web application quality:

1. A reference framework for the definition of quality,
2. A structuring of the metrics of the literature based on the ISO25010/SQUARE standard which makes reference in the field of software product quality,
3. A meta model of web application guidelines,
4. A grammar for the definition of the guidelines,
5. A prototype to manage the guidelines (insertion, search, modification).

These contributions have been the subject of several publications [Cherfi et al. 2013], [Do et al. 2016a], [Do et al. 2016b] which allowed us to verify their relevance and interest. In particular, [Do et al. 2016a] has been awarded the Best Paper Certificate in the Area of Information Systems Analysis and Specification.

Several research avenues are before us to consolidate or extend our results. First, the validation of the reference framework (Chapter 4) requires a refinement of the four proposed dimensions to define the necessary and sufficient sub-characteristics. Other statistical studies could usefully complement the automatic classification described in Chapter 4, including a principal component analysis. Reconciliation efforts with other ISO standards, notably ISO8000, would facilitate consolidation of the quality of information axis. Moreover, the two other axes must be reinforced by finding similar ISO standards or by conducting standardization efforts. Secondly, the profusion of metrics that Chapter 5 has illustrated has to lead the researcher to propose a better organization so as to propose a complete and minimal range. Validation must be conducted to verify the relevance and feasibility of measurements based on these metrics. Finally, the procedure for making the guidelines available must be coupled with a methodological approach covering the entire life cycle of the web application so that, at each stage and at each iteration, the designer, the developer or the engineer in charge of the maintenance are offered the relevant guidelines for this stage and appropriate to the context (type of web application, use, etc.).

Our publications

- Samira Si-Said Cherfi, Tuan Anh Do, and Isabelle Comyn-Wattiau. An exploratory study on websites quality assessment. In *Advances in Conceptual Modeling - ER 2013 Workshops, LSAWM, MoBiD, RIGiM, SeCoGIS, WISM, DaSeM, SCME, and PhD Symposium, Hong Kong, China, November 11-13, 2013, Revised Selected Papers,* pages 170–179, 2013.

- Tuan Anh Do, Isabelle Comyn-Wattiau, and Samira Si-Said Cherfi. Structuring guidelines for web application designers - A meta-model. In *ICEIS 2016 - Proceedings of the 18th International Conference on Enterprise Information Systems, Volume 1, Rome, Italy, April 25-28, 2016,* pages 327–335, 2016. (Best paper award of IS Analysis and Specification area)

- Tuan Anh Do, Isabelle Comyn-Wattiau, and Samira Si-Said Cherfi. Guidelines for web application designers: A meta-model, a grammar, and a tool. In *Enterprise Information Systems - 18th International Conference, ICEIS 2016, Rome, Italy, April 25-28, 2016, Revised Selected Papers,* pages 242–260, 2016.

Bibliography

Silvia Mara Abrahão, Emilio Insfrán, and Adrian Fernandez. Designing highly usable web applications. In *Computing Handbook, Third Edition: Information Systems and Information Technology*, pages 36: 1–14. CRC Press, 2014. 138, 140, 145, 208

Roberto Acerbis, Aldo Bongio, Marco Brambilla, and Stefano Butti. Webratio 5: An eclipse-based case tool for engineering web applications. In *Web Engineering*, pages 501–505, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. 164

AgeLight LCC. Interface design guidelines for users of all ages. Technical report, AgeLight LCC, 2001. 157, 160, 161, 168, 218, 219, 220, 221, 222

Adel M. Aladwani and Prashant C. Palvia. Developing and validating an instrument for measuring user-perceived web quality. *Information & Management*, 39(6):467–476, 2002. 19, 103, 105, 106, 109

A de Silva Alves and R de Mattos Fortes Ponti. Web quality metrics: an analysis using machine learning systems. In *International Conference on Information Systems, Analysis and Synthesis, World Multiconference on Systemics, Cybernetics and Informatics, Information Systems Technology, SCI*, 2001. 128, 139, 141, 144, 153, 205

Gustavo Aragon, M.J. Escalona, Michael Lang, and Jose Hilera. An analysis of model-driven web engineering methodologies. 8:1–10, 01 2012. 165

A. Bajaj and R. Krishnan. CMU-WEB: a conceptual model for designing usable web applications. *Journal of Database Management*, 10(4):33–43, 1999. 128, 141, 143, 147, 204

Monalessa Perini Barcellos, Ricardo de Almeida Falbo, and Rodrigo Dal Moro. A well-founded software measurement ontology. In *Formal Ontology in Information Systems, Proceedings of the Sixth International Conference, FOIS 2010, Toronto, Canada, May 11-14, 2010*, pages 213–226, 2010. 123

J. A. Bargas-Avila, O. Brenzikofer, S. P. Roth, A. N. Tuch, S. Orsini, and K. Opwis. Simple but Crucial User Interfaces in the World Wide Web: Introducing 20 Guidelines for Usable Web Form Design. In Rita Matrai, editor, *User Interfaces*. Intech, 2010. 161, 254, 255, 256

Santiago Meliá Beigbeder and Cristina Cachero. An MDA Approach for the Development of Web Applications. In *Web Engineering - 4th International Conference, ICWE 2004, Munich, Germany, July 26-30, Proceedings*, pages 300–305, 2004. 43, 81, 82

Nigel Bevan. International standards for HCI and usability. *Int. J. Hum.-Comput. Stud.*, 55(4):533–552, 2001. 68

Nigel Bevan and Lonneke Spinhof. Are guidelines and standards for web usability comprehensive? In *Human-Computer Interaction. Interaction Design and Usability, 12th International Conference, HCI International 2007, Beijing, China, July 22-27, 2007, Proceedings, Part I*, pages 407–419, 2007. 68

Nicoletta Di Blas, Maria Pia Guermand, Carolina Orsini, and Paolo Paolini. Evaluating The Features Of Museum Websites: (The Bologna Report). In *Museums and the Web*, 2002. 72

Michael Bloch, Sven Blumberg, and Jurgen Laartz. Delivering large-scale it projects on time, on budget, and on value. *McKinsey on Finance*, Winter(45):28–35, 2013. 28, 155

Barry W. Boehm, John R. Brown, Hans Kaspar, Myron Lipow, Gordon J. Macleod, and Michael J. Merrit. *Characteristics of Software Quality. Vol. 1.* TRW series of software technology. Elsevier, Amsterdam, Lausanne, New York, 1978. 61

Marco Brambilla, Sara Comai, Piero Fraternali, and Maristella Matera. Designing web applications with WebML and WebRatio. In *Web Engineering: Modelling and Implementing Web Applications*, pages 221–261. Springer, 2008. 43, 84

businessdictionary.com. Definition of quality engineering. Online in http://www.businessdictionary.com/definition/quality-engineering.html, cited January 2018. 59

Andrew Butterfield and Gerard Ekembe Ngondi. *A Dictionary of Computer Science (7 ed.)*. Oxford University Press, 2016. 56, 124

Coral Calero, Julián Ruiz, and Mario Piattini. Classifying web metrics using the web quality model. *Online Information Review*, 29(3):227–248, 2005. 22, 23, 24, 43, 44, 61, 64, 69, 125, 126, 135, 136

Mei Cao, Qingyu Zhang, and John Seydel. B2C e-commerce web site quality: an empirical examination. *Industrial Management and Data Systems*, 105(5):645–661, 2005. 19, 103, 109

Vera Silva Carlos and Ricardo Gouveia Rodrigues. Web site quality evaluation in higher education institutions. *Procedia Technology*, 5:273–282, 2012. 103, 105

Carnegie Mellon University. Web guidelines. `http://www.cmu.edu/marcom/brand-guidelines/print-web-products/web/index.html`, cited January 2017. 161, 216, 217

Stefano Ceri, Piero Fraternali, and Aldo Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. *Computer Networks*, 33(1-6):137–157, 2000. 78, 156, 164

Heung Seok Chae, Tae Yeon Kim, Woo-Sung Jung, and Joon-Sang Lee. Using Metrics for Estimating Maintainability of Web Applications: An Empirical Study. In *6th Annual IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007), 11-13 July, Melbourne, Australia*, pages 1053–1059. IEEE Computer Society, 2007. 127, 133, 140, 147, 149, 153, 207

François Charland, Linda Badri, and Ghazwa Malak. WEBQUALITY : Towards a Tool Supporting the Assessment of Web-Based Applications Quality. In *Proceedings of the*

ISCA 20th International Conference on Computer Applications in Industry and Engineering, CAINE 2007, November 7-9, San Francisco, California, USA, pages 115–121. ISCA, 2007. 139, 143, 204

Samira Si-Said Cherfi, Anh Do Tuan, and Isabelle Comyn-Wattiau. An exploratory study on websites quality assessment. In *Advances in Conceptual Modeling - ER 2013 Workshops, LSAWM, MoBiD, RIGiM, SeCoGIS, WISM, DaSeM, SCME, and PhD Symposium, Hong Kong, China, November 11-13, 2013, Revised Selected Papers*, pages 170–179, 2013. 22, 34, 125, 180

Cleriston Araujo Chiuchi, Rogéria Cristiane Gratão de Souza, Adriana Barbosa Santos, and Carlos Roberto Valêncio. Efficiency and portability: guidelines to develop websites. In *Proceedings of the 23rd International Conference on Software Engineering & Knowledge Engineering (SEKE'2011), Eden Roc Renaissance, Miami Beach, USA, July 7-9, 2011*, pages 37–41, 2011. 29, 160, 161, 165, 257, 258

Rajiv Chopra. *Web Engineering*. Prentice-Hall of India Pvt.Ltd, 2016. 57

Serena Cimino and Francesco Micali. Web Q-Model: a new approach to the quality. In *International conference on computer Human Interaction, CHI 2008, April 5 – April 10*, 2008. 70

Cisco Systems. The zettabyte era: Trends and analysis. `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html`, June 2017. 10, 50

María José Escalona Cuaresma and Gustavo Aragón. Ndt. a model-driven approach for web requirements. *IEEE Trans. Software Eng.*, 34(3):377–390, 2008. 43, 87

Barrie Dale. *Total quality management*. Wiley Online Library, 2015. 14, 93

Thomas H Davenport. Need radical innovation and continuous improvement? Integrate process reengineering and TQM. *Strategy & Leadership*, 21(3):6, 1993. 14, 93

William DeLone and Ephraim McLean. The DeLone and McLean model of information

systems success: a ten-year update. *Journal of Management Information Systems*, 19 (4):9–30, 2003. 103

W. Edwards Deming. *The New Economics for Industry, Government, Education*. MIT Center for Advanced Engineering Study, 1993. 15, 58, 93

Zaryn Dentzel. How the internet has changed everyday life. Online in https://www.bbvaopenmind.com/en/article/internet-changed-everyday-life/, 2014. 49

Yogesh Deshpande, San Murugesan, Athula Ginige, Steve Hansen, Daniel Schwabe, Martin Gaedke, and Bebo White. Web engineering. *J. Web Eng.*, 1(1):3–17, 2002. 14, 92

G. A. Di Lucca, A. R. Fasolino, P. Tramontata, and C. A Visaggio. Towards the Definition of a Maintainability Model for Web Applications. In *8th European Conference on Software Maintenance and Re-engineering*, pages 279–287, 2004. 133, 139, 141, 148, 151, 152, 204

dictionary.com. Dictionary.com unabridged, Sep 2017. URL `http://www.dictionary.com/browse/webapp`. 56

Tuan Anh Do, Isabelle Comyn-Wattiau, and Samira Si-Said Cherfi. Structuring guidelines for web application designers - A meta-model. In *ICEIS 2016 - Proceedings of the 18th International Conference on Enterprise Information Systems, Volume 1, Rome, Italy, April 25-28, 2016*, pages 327–335, 2016a. 34, 180

Tuan Anh Do, Isabelle Comyn-Wattiau, and Samira Si-Said Cherfi. Guidelines for web application designers: A meta-model, a grammar, and a tool. In *Enterprise Information Systems - 18th International Conference, ICEIS 2016, Rome, Italy, April 25-28, 2016, Revised Selected Papers*, pages 242–260, 2016b. 34, 180

P.D.D. Dominic and Handaru Jati. A comparison of Asian airlines websites quality: using a non-parametric test. *International Journal of Business Innovation and Research*, 5(5): 499–523, September 2011. 73, 127, 128, 131, 133, 140, 142, 145, 147, 149, 151, 207

Nicolae-George Dragulanescu. Website Quality Evaluations: Criteria and Tools. *The International Information & Library Review*, 34(3):247–254, September 2002. 73

Joakim Ekberg, Leni Ericson, Toomas Timpka, Henrik Eriksson, Sam Nordfeldt, Lena Hanberger, and Johnny Ludvigsson. Web 2.0 systems supporting childhood chronic disease management: design guidelines based on information behaviour and social learning theories. *Journal of medical systems*, 34(2):107–117, 2010. 29, 165

Sanne Elling. Evaluating website quality: Five studies on user-focused evaluation methods. LOT Dissertations Series, Vol: 308, 2012. 72

Nádia Fernandes, Daniel Costa, Carlos Duarte, and Luís Carriço. Evaluating the Accessibility of Web Applications. In *Proceedings of the 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion, DSAI 2012, Douro Region, Portugal, July 19-22*, volume 14 of *Procedia Computer Science*, pages 28–35. Elsevier, 2012. 73

Adrian Fernandez, Emilio Insfran, and Silvia Abrahão. Usability evaluation methods for the web: A systematic mapping study. *Information and Software Technology*, 53(8): 789–817, 2011. 137

Anju Gautam and Vivek Kumar Sharma. A Practical Approach for Measuring Quality of Website. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, 3(3):258–262, 2014. 138, 140, 146, 147, 150, 151, 208, 209

Matthias Gelbmann. The 3 most common technical website quality problems. Online in https://w3techs.com/, 2010. 51

Genomics and Pharmacology Facility, Center for Cancer Research, National Cancer Institute. Cimminer. `https://discover.nci.nih.gov/cimminer/home.do`, cited January 2018. 19, 111

Emad Ghosheh, Sue Black, and Jihad Qaddour. Design metrics for web application maintainability measurement. In *The 6th ACS/IEEE International Conference on Computer Systems and Applications, AICCSA 2008, Doha, Qatar, March 31 - April 4*, pages 778–784. IEEE, 2008. 133, 149, 153, 206

Athula Ginige and San Murugesan. Guest Editors' Introduction: Web Engineering - An Introduction. *IEEE MultiMedia*, 8(1):14–18, 2001. 75

Jaime Gómez, Cristina Cachero, and Oscar Pastor. Extending a Conceptual Modelling Approach to Web Application Design. In *Advanced Information Systems Engineering, 12th International Conference CAiSE 2000, Stockholm, Sweden, June 5-9, 2000, Proceedings*, pages 79–93, 2000. 78

Paul Grünbacher, Rudolf Ramler, Werner Retschitzegger, and Wieland Schwinger. Making Quality a First-Class Citizen in Web Engineering. In *2nd Workshop on Software Quality, ICSE 04, Edinburgh, UK, May, 2004*, pages 24–29, 2004. 75

Marc Guillemot and Dierk König. Web testing made easy. In *Companion to the 21th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2006, October 22-26, Portland, Oregon, USA*, pages 692–693, 2006. 74

Neil B Harrison, Paris Avgeriou, and Uwe Zdun. Using patterns to capture architectural decisions. *IEEE software*, 24(4), 2007. 30, 166

Layla Hasan and Emad Abuelrub. Assessing the quality of web sites. *Applied Computing and Informatics*, 9(1):11–29, 2011. 19, 103, 104, 106, 109

Rolf Hennicker and Nora Koch. A uml-based methodology for hypermedia design. In *UML 2000 — The Unified Modeling Language*, pages 410–424, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. 164

Internet Live Stats. Number of Internet Users (2016). Online in http://www.internetlivestats.com/internet-users/. 9, 10, 49

Internet World Stats. http://www.internetworldstats.com. 9, 13, 49, 91

ISO. *ISO/IEC 9126. Software engineering – Product quality.* International Organization for Standardization, 2001. 43, 59, 60

ISO. *ISO 9241-210:2010 Ergonomics of human-system interaction.* International Organization for Standardization, 2010. 56

ISO/IEC. *ISO 8402:1994. Quality management and quality assurance – Vocabulary.* ISO/IEC, 1994. 58

ISO/IEC. *ISO/IEC 25010:2011. Systems and software engineering – systems and software quality requirements and evaluation (SQuaRE) – system and software quality models.* ISO/IEC, 2011. 66

ISO/IEC. *ISO/IEC 25000:2014. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE.* ISO/IEC, 2014. 60

Melody Yvette Ivory. *An Empirical Foundation for Automated Web Interface Evaluation.* PhD thesis, University of California, Berkeley, 2001a. 74, 130, 143, 144, 145, 148

Melody Yvette Ivory. *An Empirical Foundation for Automated Web Interface Evaluation.* PhD thesis, 2001b. 139, 141, 142, 204, 205, 206

M. Jorgensen. Software quality measurement. *Advances in Engineering Software*, 30(12): 907–912, 12 1999. 124

Israa Wahbi Kamal, Izzat M. Alsmadi, Heider A. Wahsheh, and Mohammed N. Al-Kabi. Evaluating Web Accessibility Metrics for Jordanian Universities. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 7(7):113–122, 2016. 138, 140, 142, 145, 151, 207

Gerti Kappel, Elke Michlmayr, Birgit Pröll, Siegfried Reich, and Werner Retschitzegger. Web engineering - old wine in new bottles? In *Web Engineering - 4th International Conference, ICWE 2004, Munich, Germany, July 26-30, 2004, Proceedings*, pages 6–12, 2004. 68

Jintavee Khlaisang. Research-based Guidelines for Evaluating Educational Service Website: Case Study of Thailand Cyber University Project. *Journal of Technology Research*, Procedia - Social and Behavioral Sciences:751–758, 2015. 157

Nora Koch and Andreas Kraus. The Expressive Power of UML-based Web Engineering. In *Second International Workshop on Web-oriented System Technologies (IWWOST'02), Malaga, Spain, May 2002*, pages 105–120, 2002. 80, 155

Nora Koch and Andreas Kraus. Towards a Common Metamodel for the Development of Web Applications. In *Web Engineering, International Conference, ICWE 2003, Oviedo, Spain, July 14-18, Proceedings*, pages 497–506. Springer, 2003. 80

Harshita Kotian and BB Meshram. A framework for quality management of e-commerce websites. In *Nascent Technologies in Engineering (ICNTE), 2017 International Conference on*, pages 1–6, 2017. 17, 19, 103, 105, 106, 109

Andreas Kraus, Alexander Knapp, and Nora Koch. Model-Driven Generation of Web Applications in UWE. In *Proceedings of the 3rd International Workshop on Model-Driven Web Engineering MDWE 2007, Como, Italy, July 17, 2007*, 2007. 81, 83

Michael Krigsman. Research: 25 percent of web projects fail. Online in http://www.zdnet.com/article/research-25-percent-of-web-projects-fail/, May 2008. 13, 28, 92, 155

Sandeep Kumar, Santosh Kumar Swain, and Rajib Mall. Measuring Web Site Usability Quality Complexity Metrics for Navigability. In *Intelligent Computing, Communication and Devices*, Advances in Intelligent Systems and Computing, pages 393–401. Springer, 2015. 138, 150, 209

Le Journal du Net. Nombre d'internautes dans le monde. Online in http://www.journaldunet.com/ebusiness/le-net/1071539-nombre-d-internautes-dans-le-monde/, May 2016. 9, 10, 43, 49, 50

Stefan Leuthold, Javier A. Bargas-Avila, and Klaus Opwis. Beyond web content accessibility guidelines: Design of enhanced text user interfaces for blind internet users. *Int. J. Hum.-Comput. Stud.*, 66(4):257–270, 2008. 157, 161, 253, 254

Philip Lew, Luis Olsina, and Li Zhang. Quality, Quality in Use, Actual Usability and User Experience as Key Drivers for Web Application Evaluation. In *Web Engineering, 10th International Conference, ICWE 2010, Vienna, Austria, July 5-9*, pages 218–232, 2010. 66

LinkPopularity. http://www.linkpopularity.com/. 74

Eleanor T Loiacono, Richard T Watson, and Dale L Goodhue. Webqual: An instrument for consumer evaluation of web sites. *International Journal of Electronic Commerce*, 11 (3):51–87, 2007. 103, 105

Anitawati Mohd Lokman, Nor Laila Md. Noor, and Mitsuo Nagamachi. Expertkanseiweb: A tool to design kansei website. In *Enterprise Information Systems, 11th International Conference, ICEIS 2009, Milan, Italy, May 6-10, 2009. Proceedings*, pages 894–905, 2009. 161, 217, 218

Martin C. Maguire. Guidelines on website design and colour selection for international acceptance. In *Design, User Experience, and Usability. Theory, Methods, Tools and Practice - First International Conference, DUXU 2011, Held as Part of HCI International 2011, Orlando, FL, USA, July 9-14, 2011, Proceedings, Part I*, pages 162–171, 2011. 160, 161, 222, 223

Ghazwa Malak, Houari Sahraoui, Linda Badri, and Mourad Badri. Modeling web quality using a probabilistic approach: An empirical validation. *ACM Transactions on the Web (TWEB)*, 4(3):9:1–9:31, July 2010. 73, 103, 105

Michael Marcotty and Henry Ledgard. *The world of programming languages*. Springer Science & Business Media, 2012. 33, 170

Stephanos Mavromoustakos and Andreas S. Andreou. WAQE: a Web Application Quality Evaluation model. *Int. J. Web Eng. Technol.*, 3(1):96–120, 2007. 56, 70, 73

Jim A. McCall, Paul K. Richards, and Gene F. Walters. Factors in software quality. volume i. concepts and definitions of software quality. Technical report, General Electric, 11 1977. 61

John McGarry, David Card, Cheryl Jones, Beth Layman, Elizabeth Clark, Joseph Dean, and Fred Hall. *Practical Software Measurement: Objective Information for Decision Makers*. Addison-Wesley Professional, Boston, USA, 2001. 123

Santiago Meliá, Cristina Cachero, and Jaime Gómez. Using MDA in Web Software Architectures. In *Proc. 2nd OOPSLA Wsh. Generative Techniques in the Context of Model Driven Architecture, Anaheim*, pages 1–6, 2003. 81

Santiago Meliá, Jaime Gómez, Sandy Pérez, and Oscar Díaz. A Model-Driven Development for GWT-Based Rich Internet Applications with OOH4RIA. In *Proceedings of the Eighth International Conference on Web Engineering, ICWE 2008, 14-18 July, Yorktown Heights, New York, USA*, pages 13–23, 2008. 43, 86

Lisa Meloncon, Erin Haynes, Megan Varelmann, and Lisa Groh. Building a playground: General guidelines for creating educational web sites for children. *Technical Communication*, 57(4):398–416, 2010. 160, 161, 168, 228, 229, 230

E. Mendes, N. Mosley, and S. Counsell. Estimating design and authoring effort. *IEEE MultiMedia*, pages 50–57, Special Issue, January-March 2001. 143, 148, 150, 204

Luisa Mich, Mariangela Franch, and Loris Gaio. Evaluating and Designing the Quality of Web Sites. *IEEE MultiMedia*, 10(1):34–43, 2003. 69, 71

Microsoft Corporation. *Microsoft Computer Dictionary Fifth Edition.* Microsoft Press, Redmond, Washington, 2002. 55

Microsoft Developer Network. Chapter 4 - design guidelines for secure web applications. `https://msdn.microsoft.com/en-us/library/ff648647.aspx`, cited January 2017. 161, 223, 224, 225, 226, 227, 228

Ministry of Community and Social Services of Ontario. Making your website more accessible. `http://www.mcss.gov.on.ca/en/mcss/publications/accessON/accessible_websites/toc.aspx`, 2012. 161, 251, 252

Pooja Mittal. Evaluation of web metrics. *International Journal of Engineering Science and Computing (IJESC)*, 7(7):14325–14333, 2017. 138, 145, 147, 149, 207

San Murugesan and Yogesh Deshpande, editors. *Web Engineering, Managing Diversity and Complexity of Web Application Development*, volume 2016 of *Lecture Notes in Computer Science*, 2001. Springer. 75

Doaa Nabil, Abeer Mosad, and Hesham A Hefny. Web-based applications quality factors: A survey and a proposed conceptual model. *Egyptian Informatics Journal*, 12(3):211–217, 2011. 19, 103, 104, 105, 106, 109, 112

Netcraft. November 2017 web server survey. Online in https://news.netcraft.com/archives/2017/11/21/november-2017-web-server-survey.html, 2012, cited January 2018. 10, 50

Luis Olsina and Gustavo Rossi. Measuring Web Application Quality with WebQEM. *IEEE MultiMedia*, 9(4):20–29, 2002. 103, 105, 127, 128, 130, 131, 134, 139, 140, 141, 142, 143, 144, 145, 147, 148, 149, 150, 152, 204, 205, 206, 207

Luis Olsina, Guillermo Lafuente, and Gustavo Rossi. Specifying Quality Characteristics and Attributes for Websites. In *Web Engineering, Software Engineering and Web Application Development*, pages 266–278, 2001. 70, 131, 147, 149, 205

Oracle. Oracle Argus Affiliate User's Guide. `https://docs.oracle.com/cd/E60959_01/doc.80/e54660/glossary.htm`. Online; accessed 25 September 2017. 56

Tihomir Orehovački, Andrina Granić, and Dragutin Kermek. Evaluating the perceived and estimated quality in use of web 2.0 applications. *Journal of Systems and Software*, 86 (12):3039–3059, 2013. 19, 103, 104, 109

A. Ant Ozok and Gavriel Salvendy. Twenty guidelines for the design of web-based interfaces with consistent language. *Computers in Human Behavior*, 20(2):149–161, 2004. 161, 252, 253

PennyJuice. http://www.pennyjuice.com. 52

Klaus Pohl. *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated, 2010. 31, 170

PowerMapper. http://www.powermapper.com/. 74

Oxford University Press. Oxford dictionary. `https://en.oxforddictionaries.com/`, cited January 2017. 124

D. Radosav, D. Karuovic, B. Markoski, and Z. Ivankovic. Guidelines on accessible web portal design. In *2011 IEEE 12th International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 297–302, Nov 2011. 158, 160

Donald J Reifer. Web development: estimating quick-to-market software. *IEEE software*, 17(6):57–64, 2000. 130, 139, 141, 143, 148, 152, 204

Rim Rekik and Ilhem Kallel. Fuzzy reduced method for evaluating the quality of institutional web sites. In *7th International Conference on Next Generation Web Services Practices (NWeSP)*, page 296–301, 2011. 73, 74

Americo Rio and Fernando Brito e Abreu. Websites Quality: Does It Depend on the Application Domain? In *Quality of Information and Communications Technology, 7th International Conference on the Quality of Information and Communications Technology, QUATIC 2010, Porto, Portugal, 29 September - 2 October, Proceedings*, pages 493–498. IEEE Computer Society, 2010. 127, 130, 131, 134, 139, 141, 143, 144, 145, 147, 148, 149, 150, 151, 152, 153, 204, 205, 206

L. Rosenberg, T. Hammer, and J Shaw. Software Metrics and Reliability. In *Proceedings of IEEE International Symposium on Software Reliability Engineering*, 1998. 131

Julián Ruiz, Coral Calero, and Mario Piattini. A Three Dimensional Web Quality Model. In *Web Engineering, International Conference, ICWE 2003, Oviedo, Spain, July 14-18, 2003, Proceedings*, pages 384–385. Springer, 2003. 69

Hugo Saba, Eduardo Manuel de Freitas Jorge, Victor Franco Costa, and Hernane Borges de Barros Pereira. WEBTESTE: A Stress Test Tool. In *WEBIST 2006, Proceedings of the Second International Conference on Web Information Systems and Technologies: Internet Technology / Web Interface and Applications, Setúbal, Portugal, April 11-13*, pages 246–249, 2006. 74

Luis Olsina Santos. Web-site Quality Evaluation Method: a Case Study on Museums. In *ICSE 99 - 2nd Workshop on Software Engineering over the Internet*, 1999. 70

Daniel Schwabe and Gustavo Rossi. The object-oriented hypermedia design model. 38: 45–46, 08 1995. 164

J. Sethuraman, V. Vaithiyanathan, and K.R. Sekar. Prism based quality evaluation and prediction of web applications. *International Journal of Applied Engineering Research*, 8(20):2639–2647, 2013. 138, 140, 145, 146, 208

Oreste Signore. A Comprehensive Model for Web Sites Quality. In *Seventh IEEE International Workshop on Web Site Evolution (WSE 2005), 26 September 2005, Budapest, Hungary*, pages 30–38. IEEE Computer Society, 2005. 69, 127, 130, 134, 139, 140, 141, 143, 144, 145, 147, 148, 149, 150, 151, 152, 204, 205, 206

Yogesh Singh, Ruchika Malhotra, and Poonam Gupta. Empirical Validation of Web Metrics for Improving the Quality of Web Page. *International Journal of Advanced Computer Science and Applications*, 2(5):22–28, 2011. 141, 143, 148, 150, 204

World Wide Web Size. http://www.worldwidewebsize.com. 13, 91

David Sloan, Andy Heath, Fraser Hamilton, Brian Kelly, Helen Petrie, and Lawrie Phipps. Contextual web accessibility - maximizing the benefit of accessibility guidelines. In *Proceedings of the 2006 International Cross-disciplinary Workshop on Web Accessibility (W4A): Building the Mobile Web: Rediscovering Accessibility?*, W4A '06, pages 121–131, New York, NY, USA, 2006. ACM. 157

Ian Sommerville. *Software Engineering*. Addison Wesley Longman Publishing Co. Inc., Redwood City, CA, USA, 1995. 131

Stack Overflow. Developer survey results 2017. Online in https://insights.stackoverflow.com/survey/2017, 2017, cited January 2018. 11, 50

Antonia Stefani and Michalis Nik Xenos. Meta-metric Evaluation of E-Commerce-related Metrics. *Electronic Notes in Theoretical Computer Science*, 233:59–72, 2009. 127, 144, 147, 149, 153, 205

Z. Sun and Y. Zhao. The preliminary construction of accessibility design guidelines of learning website for old people. In *2010 Second International Workshop on Education Technology and Computer Science*, volume 2, pages 612–615, March 2010. 160, 161, 168, 214, 215, 216

TAW. http://www.tawdis.net/. 74

N Thangammal and A Pethalakshmi. Improving the navigability of web pages using genetic algorithm. *International Journal of Computational Intelligence and Informatics*, 5(2): 132–136, 2015. 138

William Thomson. Lecture on "electrical units of measurement", May 1883. 123

O. M. F. De Troyer and C. J. Leune. WSDM: A User Centered Design Method for Web Sites. In *Computer Networks and ISDN Systems*, pages 85–94, 1998. 43, 76, 77

Vivienne Trulock and Richard Hetherington. Assessing the progress of implementing web accessibility - an irish case study. In *ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems, Volume HCI, Barcelona, Spain, June 12-16, 2008*, pages 105–111, 2008. 159

U.S. Dept. of Health and Human Services. Research-based web design & usability guidelines, 2006. 157, 160, 161, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250

Antonio Vallecillo, Nora Koch, Cristina Cachero, Sara Comai, Piero Fraternali, Irene Garrigós, Jaime Gómez, Gerti Kappel, Alexander Knapp, Maristella Matera, Santiago Meliá, Nathalie Moreno, Birgit Pröll, Thomas Reiter, Werner Retschitzegger, José Eduardo Rivera, Andrea Schauerhuber, Wieland Schwinger, Manuel Wimmer, and Gefei Zhang. MDWEnet: A Practical Approach to Achieving Interoperability of Model-Driven Web Engineering Methods. In *Proceedings of the 3rd International Workshop on Model-Driven Web Engineering MDWE 2007, Como, Italy, July 17*, 2007. 81

B van Zeist, P Hendriks, R Paulussen, and J Trienekens. *Quality of software products - Experiences with a quality model (in Dutch).* Kluwer Bedrijfswetenschappen, 1996. 64

Stephane Vaucher, Samuel Boclinville, Houari Sahraoui, and Naji Habra. Recommending Improvements to Web Applications Using Quality-Driven Heuristic Search. In *Proceedings of the 10th International Conference on Web Information Systems Engineering, WISE 2009, Poznen, Poland, October 5-7*, 2009. 138, 140, 145, 149, 207, 208

Stéphane Vaucher, Antoine Moulart, Houari Sahraoui, and Naji Habra. Automated evaluation of website navigability: an empirical validation of multilevel quality models. *Journal of Software: Evolution and Process*, 25(8):815–839, 2013. 103, 105

S. Wagner. *Software Product Quality Control*. Springer-Verlag New York Incorporated, 2013. ISBN 9783642385704. 66

Web Content Accessibility Guidelines 2.0 (WCAG). http://www.w3.org/tr/wcag20/. 67

Web Content Accessibility Guidelines (WCAG) Overview Web Accessibility Initiative (WAI). http://www.w3.org/wai/intro/wcag/. 157

WebAIM. http://webaim.org/. 74

WebQual. http://www.webqual.co.uk/. 74

Bo Xie, Ivan Watkins, and Man Huang. Making web-based multimedia health tutorials senior-friendly: design and training guidelines. In *iConference 2011, Inspiration, Integrity, and Intrepidity, Seattle, Washington, USA, February 8-11, 2011*, pages 230–237, 2011. 160, 161, 258, 259

Umi Laili Yuhana, Agus Budi Raharjo, and Siti Rochimah. Academic information system quality measurement using quality instrument: A proposed model. In *Data and Software Engineering (ICODSE), 2014 International Conference on*, pages 1–6, 2014. 19, 103, 104, 109

Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1091–1095, 2007. 174

Bo Zhao and Yan Zhu. Formalizing and validating the web quality model for web source quality evaluation. *Expert Systems with Applications*, 41(7):3306–3312, 2014. 19, 103, 104, 109

Qasim Zia. Metrics for quality assurance of web based applications. *Global Journal of Computer Science and Technology: E Network, Web & Security*, 15(7):25–32, 2015. 103, 105

Annexes

This part is annex of the thesis. The annex contains three parts. The first part is the list of metrics, the second part is the list of guidelines and the last part is the source code of tool.

Appendix A

Table of metrics

This annex belongs to Chapter 5. It contains 166 metrics, with 31 corresponding sub-characteristics of ISO 25010, the corresponding web features (Content / Navigation / Presentation), whether the metric can be measured automatically, their type and their source.

		Characteristics																															Web features			Auto	Type	Source
		Fun.		Sui.		Perf.		Eff.		Comp.		Usability						Reliability					Security					Maintenability					Portability			Con	Pre	Nav			
No	Metrics	Cm	Co	Ap	TB	Re	Ca	CE	In	AR	Le	Op	UE	UI	Ac	Ma	Av	FT	Rc	Cn	Ig	NR	At	Au	Md	Ru	An	Mo	Te	Ad	Is	Rp						
1	Page count / Total number of web pages											x																x		x	x	x	x		x	x	Number	Mendes et al. [2001]
2	Page title word count									x																		x	x				x	x		x	Number	Ivory [2001b]
3	Body text words					x						x																x		x			x	x		x	Number	Singh et al. [2011]
4	Word count per page					x						x																					x	x		x	Number	Ivory [2001b]
5	Total link count									x		x																x		x	x	x			x	x	Number	Olsina and Rossi [2002]
6	Page link count											x																							x	x	Number	Ivory [2001b]
7	Exclamation point count		x							x																							x			x	Number	Ivory [2001b]
8	Images count		x											x														x					x	x		x	Number	Olsina and Rossi [2002]
9	Page size					x	x					x																x					x	x		x	Number	Rio and Abreu [2010]
10	Total embedded links			x						x	x																	x							x	x	Number	Olsina and Rossi [2002]
11	Number of lists													x																x			x	x		x	Number	Singh et al. [2011]
12	Number of panes regarding frames			x		x		x						x														x		x	x	x		x		x	Number	Olsina and Rossi [2002]
13	Table count													x																			x	x		x	Number	Ivory [2001b]
14	Within page links											x																x							x	x	Number	Singh et al. [2011]
15	Emphasized body word count													x																				x		x	Number	Ivory [2001b]
16	Total emphasized text													x																				x		x	Number	Singh et al. [2011]
17	Display word count											x																						x		x	Number	Ivory [2001b]
18	Wrapped links													x	x																		x	x		x	Number	Singh et al. [2011]
19	CSS size per page			x		x																						x	x	x			x			x	Real	Rio and Abreu [2010]
20	Download time of home page				x							x					x																x			x	Real	Bajaj and Krishnan [1999]
21	Image size			x		x						x																x		x			x	x		x	Real	Signore [2005]
22	Script size per page		x			x		x																				x	x	x		x	x			x	Real	Di Lucca et al. [2004]
23	Download time	x			x							x																						x		x	Real	Bajaj and Krishnan [1999]
24	Download time of all pages				x							x					x																x			x	Real	Bajaj and Krishnan [1999]
25	Presence of contacts/info form											x				x																	x			x	Binary	Olsina and Rossi [2002]
26	Number of label tags									x																	x						x			x	Number	Rio and Abreu [2010]
27	Presence of name's author	x												x																			x	x		x	Binary	Charland et al. [2007]
28	Presence of logo	x										x																					x	x		x	Binary	Charland et al. [2007]
29	Presence of site name in title	x													x																			x		x	Binary	Rio and Abreu [2010]
30	Presence of navigation / menu bar	x	x	x						x			x																					x	x	x	Binary	Signore [2005]
31	Presence of breadcrumbs	x	x	x						x			x																					x	x	x	Binary	Signore [2005]
32	Presence of page title in link	x	x							x																								x	x	x	Binary	Rio and Abreu [2010]
33	Number of script files per page		x	x		x			x			x			x						x	x	x	x				x	x				x		x	Number	Reifer [2000]
34	Number of CSS files per page		x	x		x			x			x			x													x	x				x		x	Number	Rio and Abreu [2010]
35	Number of tables per page													x																			x	x		x	Number	Ivory [2001b]
36	Presence of specific CSS to device		x	x					x						x												x	x		x			x	x		x	Binary	Signore [2005]

		Characteristics																																Web features			Auto	Type	Source
		Fun.		Sui.		Perf.		Eff.		Comp.		Usability						Reliability				Security						Maintenability					Portability			Con	Pre	Nav			
No	Metrics	Cm	Co	Ap	TB	Re	Ca	CE	In	AR	Le	Op	UE	UI	Ac	Ma	Av	FT	Rc	Cn	Ig	NR	At	Au	Md	Ru	An	Mo	Te	Ad	Is	Rp						
37	Use of HTML notation in formatting									x	x																		x	x	x	x	x		x	Binary	Rio and Abreu [2010]
38	Number of divs			x						x					x														x	x	x		x		x	Number	Signore [2005]
39	Presence of tables inside tables								x						x																		x		x	Binary	Rio and Abreu [2010]
40	Average link words											x																					x		x	Real	Ivory [2001b]
41	Link title (with explanatory help)									x			x		x																			x	x	Binary	Olsina and Rossi [2002]
42	Broken links									x	x				x	x		x	x								x	x	x					x	x	Number	Olsina and Rossi [2002]
43	Number of broken links to another sites														x	x		x	x								x	x	x					x	x	Number	Signore [2005]
44	Number of links to another sites					x			x			x																							x	x	Number	Alves and Ponti [2001]
45	Link image count			x										x																					x	Number	Alves and Ponti [2001]
46	Graphics link count								x						x	x											x	x						x	x	Number	Stefani and Xenos [2009]
47	Text link count											x		x																				x	x	Number	Ivory [2001b]
48	Number of internal broken links														x	x		x	x								x	x	x					x	x	Number	Signore [2005]
49	Number of internal links					x						x																							x	x	Number	Alves and Ponti [2001]
50	Number of different broken links															x		x										x							x	x	Number	Olsina and Rossi [2002]
51	Unimplemented link count															x		x																x		x	Number	Olsina and Rossi [2002]
52	Number of orphan pages															x											x	x							x	x	Number	Olsina and Rossi [2002]
53	Quick access page = link count / page count																		x															x			x	Real	Stefani and Xenos [2009]
54	Percent of dead-end web pages															x											x	x							x	x	Real	Olsina et al. [2001]
55	HTML errors per page															x											x	x	x					x	x	Number	Signore [2005]
56	HTML warnings per page															x											x	x	x					x	x	Number	Rio and Abreu [2010]
57	Presence of ALT attribute in image									x	x							x	x								x	x							x		x	Binary	Olsina and Rossi [2002]
58	Image title					x				x	x																x	x							x		x	Scale	Olsina and Rossi [2002]
59	Accessibility issues per page														x																			x	x			Number	Signore [2005]
60	Image number per page			x						x				x														x							x		x	Number	Olsina and Rossi [2002]
61	Average of font size in em (percentage) in CSS							x						x																x	x	x	x	x		x	Number	Rio and Abreu [2010]
62	Average font size in pixel in CSS							x						x																x	x	x	x	x		x	Number	Rio and Abreu [2010]
63	Maximum font size in em in CSS							x						x																x	x	x	x	x		x	Number	Rio and Abreu [2010]
64	Maximum font size in pixel in CSS							x						x																x	x	x	x	x		x	Number	Rio and Abreu [2010]
65	Minimum font size in em in CSS							x						x																x	x	x	x	x		x	Number	Rio and Abreu [2010]
66	Minimum font size in pixel in CSS							x						x																x	x	x	x	x		x	Number	Rio and Abreu [2010]
67	Average heading length									x				x																			x	x		x	Real	Signore [2005]
68	Number of heading in reverse order																																				Number	Signore [2005]

		Characteristics																																	Web features			Auto	Type	Source
		Fun.		Sui.		Perf.		Eff.		Comp.		Usability						Reliability					Security						Maintenability						Portability			Con	Pre	Nav			
No	Metrics	Cm	Co	Ap	TB	Re	Ca	CE	In	AR	Le	Op	UE	UI	Ac	Ma	Av	FT	Rc	Cn	Ig	NR	At	Au	Md	Ru	An	Mo	Te	Ad	Is	Rp							
69	Number of italic text bigger than 20 characters													x	x																				x		x	Number	Rio and Abreu [2010]
70	Number of different text colors in CSS									x				x																					x		x	Number	Rio and Abreu [2010]
71	Number of different text fonts in CSS									x				x																					x		x	Number	Ivory [2001b]
72	Number of sentences per paragraph									x				x																					x		x	Number	Signore [2005]
73	Number of sub-headings per heading									x				x																						x	x	Number	Signore [2005]
74	Number of syllables per word		x	x																														x			x	Number	Signore [2005]
75	Number of words per sentence		x	x																														x			x	Number	Signore [2005]
76	Number of words in metatag description											x																						x			x	Number	Rio and Abreu [2010]
77	Number of words in metatag keywords											x																						x			x	Number	Rio and Abreu [2010]
78	Maximum size of paragraph											x		x																				x	x		x	Number	Signore [2005]
79	Paragraph size											x																						x			x	Number	Signore [2005]
80	Sub-heading length									x																								x			x	Number	Signore [2005]
81	Total number of newlines									x																								x			x	Number	Signore [2005]
82	Total sentences									x																								x			x	Number	Signore [2005]
83	Total syllables									x																								x			x	Number	Signore [2005]
84	Word count					x				x		x																						x	x		x	Number	Ivory [2001b]
85	Number of uppercase sentences									x				x																				x			x	Number	Rio and Abreu [2010]
86	Horizontal scrolling		x	x																																	x	Binary	Olsina and Rossi [2002]
87	Non-frame version					x											x																		x		x	Binary	Olsina and Rossi [2002]
88	Total number of server pages																											x					x			x	Number	Ghosheh et al. [2008]
89	Total number of client pages																											x					x			x	Number	Ghosheh et al. [2008]
90	Total number of form pages																											x	x				x			x	Number	Ghosheh et al. [2008]
91	Total number of form elements																											x	x				x			x	Number	Ghosheh et al. [2008]
92	Total number of client components (style sheet and JavaScript)																											x					x			x	Number	Ghosheh et al. [2008]
93	Total number of link relationships								x																			x					x			x	Number	Ghosheh et al. [2008]
94	Total number of submit relationships								x																			x					x			x	Number	Ghosheh et al. [2008]
95	Total number of build relationships								x																			x					x			x	Number	Ghosheh et al. [2008]
96	Total number of forward relationships								x																			x					x			x	Number	Ghosheh et al. [2008]
97	Total number of include relationships								x																			x					x			x	Number	Ghosheh et al. [2008]
98	Total number of use tag relationships								x																			x					x			x	Number	Ghosheh et al. [2008]
99	Number of relationships over number of web pages								x																			x					x			x	Number	Ghosheh et al. [2008]
100	Number of data exchanged over number of server pages								x																			x					x			x	Real	Ghosheh et al. [2008]
101	Number of include relationships over number of web pages								x																			x					x			x	Real	Ghosheh et al. [2008]

		Characteristics																														Web features			Auto	Type	Source
		Fun.		Sui.		Perf.		Eff.		Comp.	Usability						Reliability				Security					Maintenability					Portability			Con	Pre	Nav			
No	Metrics	Cm	Co	Ap	TB	Re	Ca	CE	In	AR	Le	Op	UE	UI	Ac	Ma	Av	FT	Rc	Cn	Ig	NR	At	Au	Md	Ru	An	Mo	Te	Ad	Is	Rp						
102	Lack of cohesion in methods								x																			x				x				Binary	Chae et al. [2007]
103	Data abstraction coupling								x																			x				x					Chae et al. [2007]
104	Response for classes		x													x												x				x					Chae et al. [2007]
105	Response time	x			x							x					x	x	x		x	x	x	x									x		x	Real	Dominic and Jati [2011]
106	Page rank														x																			x	x	Number	Dominic and Jati [2011]
107	Frequency of update	x										x				x	x	x	x		x	x	x	x									x		x	Real	Olsina and Rossi [2002]
108	Traffic				x	x	x																									x			x	Real	Dominic and Jati [2011]
109	Design optimisation		x													x																x				Binary	Dominic and Jati [2011]
110	Number of items					x																								x	x		x	x	x	Number	Dominic and Jati [2011]
111	Accessibility error														x																		x	x	x	Number	Dominic and Jati [2011]
112	Markup validation		x												x														x				x	x	x	Binary	Dominic and Jati [2011]
113	Number of metatag									x																	x						x		x	Number	Mittal [2017]
114	Minimum meta keyword length											x																					x		x	Number	Mittal [2017]
115	Maximum meta keyword length											x																					x		x	Number	Mittal [2017]
116	Words in ALT images									x	x							x	x								x	x					x	x	x	Number	Mittal [2017]
117	Total number of divisions tag																																x	x	x	Number	Mittal [2017]
118	Number of paragraphs																																x		x	Number	Mittal [2017]
119	Number of headings									x																								x	x	Number	Mittal [2017]
120	Number of headings as link									x																								x	x	x	Number	Mittal [2017]
121	The harmony between using font and the background			x										x																			x			Scale	Kamal et al. [2016]
122	Number of exterior relevant links																																x		x	Number	Kamal et al. [2016]
123	Time of read and use content				x																												x			x	Measure	Kamal et al. [2016]
124	Number of anchor tags									x																										x	Number	Kamal et al. [2016]
125	Number of empty links (links without anchor text)														x																				x	x	Number	Kamal et al. [2016]
126	Number of redirected links																																		x		Number	Kamal et al. [2016]
127	Ratio of links with titles														x													x					x		x	Real	Vaucher et al. [2009]
128	Ratio of links with text														x													x					x		x	Real	Vaucher et al. [2009]
129	Presence of a search engine	x								x																							x		x	x	Binary	Vaucher et al. [2009]
130	Presence of a site map	x								x			x																						x	x	Binary	Vaucher et al. [2009]
131	Indication of location in site												x																						x		Binary	Vaucher et al. [2009]
132	Visited links change color									x	x		x		x																			x		x	Binary	Vaucher et al. [2009]

		Characteristics									Web features			Auto	Type	Source
		Fun.	Sui.		Perf.		Eff.	Comp.			Usability									Reliability					Security						Maintenability							Portability			Con	Pre	Nav			
No	Metrics	Cm	Co	Ap	TB	Re	Ca	CE	In	AR	Le	Op	UE	UI	Ac	Ma	Av	FT	Rc	Cn	Ig	NR	At	Au	Md	Ru	An	Mo	Te	Ad	Is	Rp						
133	Link to home page									x																								x	x	Binary	Vaucher et al. [2009]
134	Support of back button									x			x																					x	x	Binary	Vaucher et al. [2009]
135	Proportion of titles chosen suitably for each icon/title		x								x																					x			Real	Abrahão et al. [2014]
136	Proportion of meaningful messages (error, advise, and warning messages)		x							x																						x			Real	Abrahão et al. [2014]
137	Breadth of the internavigation (BiN)			x						x																								x	x	Number	Abrahão et al. [2014]
138	Depth of the navigation			x						x																								x	x	Number	Abrahão et al. [2014]
139	Number of colors in a page									x				x																			x		x	Number	Sethuraman et al. [2013]
140	Presence of RGB color system									x				x																			x		x	Binary	Sethuraman et al. [2013]
141	The use of red and green colors for framing titles, fonts documentations									x				x																			x		x	Binary	Sethuraman et al. [2013]
142	Underlined text is only for hyperlink												x																				x	x	x	Binary	Sethuraman et al. [2013]
143	Presence of tabbed buttons in each page											x																					x	x	x	Binary	Sethuraman et al. [2013]
144	Presence of about info	x								x																							x			x	Binary	Sethuraman et al. [2013]
145	Not presence of auto refresh option			x									x	x																				x		x	Binary	Sethuraman et al. [2013]
146	Page resolution													x																				x		x	Real	Gautam and Sharma [2014]
147	Safe color is used			x								x		x																				x		x	Binary	Gautam and Sharma [2014]
148	Use color for blind people			x							x		x	x	x		x																x		x	Binary	Gautam and Sharma [2014]
149	Underlined text is used			x										x																			x	x		x	Binary	Gautam and Sharma [2014]
150	CSS attributes	x	x																											x			x	x			Binary	Gautam and Sharma [2014]
151	Frame validity																																x	x			Binary	Gautam and Sharma [2014]
152	Label and caption for link, table and form											x																					x	x		x	Binary	Gautam and Sharma [2014]
153	Description of meta	x										x																					x			x	Binary	Gautam and Sharma [2014]
154	Plug-in support											x																					x		x		Binary	Gautam and Sharma [2014]
155	Attributes of multimedia components																																x				Binary	Gautam and Sharma [2014]
156	One media in one page																													x		x	x	x			Binary	Gautam and Sharma [2014]
157	Using thumbnails	x	x	x								x																					x	x			Binary	Gautam and Sharma [2014]
158	Presence of bulletin board	x																															x			x	Binary	Gautam and Sharma [2014]
159	Presence of information guide	x																															x				Binary	Gautam and Sharma [2014]

		Characteristics																														Web features			Auto	Type	Source
		Fun.		Sui.		Perf.		Eff.		Comp.		Usability						Reliability				Security					Maintenability					Portability			Con	Pre	Nav			
No	Metrics	Cm	Co	Ap	TB	Re	Ca	CE	In	AR	Le	Op	UE	UI	Ac	Ma	Av	FT	Rc	Cn	Ig	NR	At	Au	Md	Ru	An	Mo	Te	Ad	Is	Rp						
160	Presence of customer feedback	x																														x				Binary	Gautam and Sharma [2014]
161	Web terrific											x													x							x				Binary	Gautam and Sharma [2014]
162	Presence of domain name	x																																x	x	Binary	Gautam and Sharma [2014]
163	Presence of information publicity	x																														x				Binary	Gautam and Sharma [2014]
164	Number of links on web page of roadmap tree																								x	x		x						x	x	Number	Kumar et al. [2015]
165	Cyclomatic complexity																								x	x		x						x		Number	Kumar et al. [2015]
166	Average digit of strikeouts per web page										x			x																			x			Real	Kumar et al. [2015]

Abbreviation:

- Fun. Sui. = Functional Suitability
  - Cm = Functional completeness
  - Co = Functional correctness
  - Ap = Functional appropriateness
- Perf. Eff. = Performance efficiency
  - TB = Time behaviour
  - Re = Resource utilization
  - Ca = Capacity
- Comp. = Compatibility
  - CE = Co-existence
  - In = Interoperability
- Usability
  - AR = Appropriateness recognizability
  - Le = Learnability
  - Op = Operability
  - UE = User error protection
  - UI = User interface aethestics
  - Ac = Accessibility
- Reliability
  - Ma = Maturity
  - Av = Availability
  - FT = Fault tolerance
  - Rc = Recoverability

- Security
  - Cn = Confidentiality
  - Ig = Integrity
  - NR = Non-repudiation
  - At = Accountability
  - Au = Authenticity
- Maintainability
  - Md = Modularity
  - Ru = Reusability
  - An = Analysability
  - Mo = Modifiability
  - Te = Testability
- Portability
  - Ad = Adaptability
  - Is = Installability
  - Rp = Replaceability

Appendix B

Table of guidelines

This annex belongs to Chapter 6. It consists of 475 original guidelines from literature. For each row, we list original guideline, modified guideline (if needed), source of guideline, the relation to other guidelines (if they have), the axe of guideline (content, navigation or presentation) and the domain of this guideline.

From 475 original guidelines, we achieve 541 guidelines after refining them.

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
1	Left justified text, text line should not be long	Justify left text	Sun and Zhao [2010]	Same as G70	Presentation	Old people
		Do not use long text line				
2	Ensure fit spacing between lines		Sun and Zhao [2010]	Derive from G88	Presentation	Old people
3	(Font) Slightly larger than size with conventional web pages, the appropriate use of bold, to avoid the use of italics and decorative fonts	Use slightly larger font than size with conventional web pages	Sun and Zhao [2010]		Presentation	Old people
		Use bold font appropriately				
		Avoid using of italics and decorative fonts				
4	Text should have clear large heading	Create clear large heading for text	Sun and Zhao [2010]		Presentation	Old people
5	Provide a text equivalent for images		Sun and Zhao [2010]		Presentation	Old people
6	Support users flexible operations (adjustable font size,background color conversion)	Allow users' flexible operations	Sun and Zhao [2010]		Presentation	Old people
7	Ensure links change color after visit		Sun and Zhao [2010]	Same as G63	Presentation	Old people
8	Colors should be used conservatively	Use colors conservatively	Sun and Zhao [2010]		Presentation	Old people
9	A high contrast between the foreground and background should exist. Background screens should not be pure white or change rapidly in brightness between screens	Create high contrast between the foreground and background	Sun and Zhao [2010]	Same as G39, G189	Presentation	Old people
		Do not use pure white background screens				
		Do not change brightness of background screens rapidly				
10	Don't use colour alone to portray information		Sun and Zhao [2010]		Presentation	Old people
11	Links should be clearly named and no link with the same name should go to a different page	Name links clearly	Sun and Zhao [2010]		Navigation Presentation	Old people
		Do not name link with the same name go to a different page				
12	Present links as lists and clearly separate links	Create links as lists	Sun and Zhao [2010]		Presentation	Old people
		Separate links clearly				

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
13	The main message should be focused on central area of page	Focus main message on central area of page	Sun and Zhao [2010]		Presentation	Old people
14	Provide location of the current page		Sun and Zhao [2010]		Navigation	Old people
15	Navigation should be clear	Create a clear navigation	Sun and Zhao [2010]	Similar to G176	Navigation	Old people
16	Avoid pull down menu		Sun and Zhao [2010]	Similar to G178	Presentation	Old people
17	Do not use a deep hierarchy and group information into meaning categories		Sun and Zhao [2010]		Navigation	Old people
18	Provide only one window open. Pop up and advertisement or overlapping windows should be avoided	Provide only one window open Avoid pop up and advertisement or overlapping windows	Sun and Zhao [2010]		Presentation	Old people
19	Provide sufficient time to read information		Sun and Zhao [2010]		Content	Old people
20	Provide a site-map		Sun and Zhao [2010]		Navigation	Old people
21	Search engine should have to check and correct misspelled function	Create search engine which has check and correct misspelled function	Sun and Zhao [2010]		Content	Old people
22	The use of icons should be simple, and clear	Use simple and clear icons	Sun and Zhao [2010]	Partly same as G180	Content Presentation	Old people
23	Provides an online help guide		Sun and Zhao [2010]		Content	Old people
24	Error messages should be simple and easy to follow	Create simple and "easy to follow" error messages	Sun and Zhao [2010]		Content	Old people
25	Language should be clear and concise	Use clear and concise language	Sun and Zhao [2010]	Similar to G191	Content	Old people
26	Page layout should be avoided irrelevant information	Avoid using irrelevant information in page layout	Sun and Zhao [2010]		Content	Old people
27	Important information should be highlighted	Highlight important information	Sun and Zhao [2010]		Presentation	Old people
28	Page layout, navigation and the use of terminology should be simple, clear, consistent	Make page layout, navigation and terminology be simple, clear, consistent	Sun and Zhao [2010]		Content Navigation	Old people

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
29	Reduce the demand on working memory by supporting recognition rather than recall	Support recognition rather than recall for reducing the demand on working memory	Sun and Zhao [2010]		Content Presentation	Old people
30	Consider page download speed - create 'small' pages	Create page with appropriate size for satisfying page download speed	Sun and Zhao [2010]		Content	Old people
31	Do not require 'double clicks'		Sun and Zhao [2010]		Presentation	Old people
32	All images should be JPGs, GIFs or PNGs. JPGs are used for photos. Graphics should use GIF or PNG formats (logos, cartoons, etc.)	Use JPG, GIF or PNG format for images	Carnegie Mellon University [cited January 2017]		Content	University
		Use JPG format for photos				
		Use GIF or PNG formats for graphics				
33	Images have a resolution of 72dpi (dots per inch) and are in either RGB or indexed color modes	Use resolution of 72 dpi for images	Carnegie Mellon University [cited January 2017]		Content	University
		Create images which are in either RGB or indexed color modes				
34	Navigation should always be on the top and/or left	Place navigation on the top and/or left	Carnegie Mellon University [cited January 2017]		Presentation	University
35	Documents, such as PDFs and MOVs, should open in a new window and be labeled .pdf or .mov in the link	Open documents (such as PDFs and MOVs) in a new window	Carnegie Mellon University [cited January 2017]	It may be contrast to G101 in distracting of user	Presentation	University
		Label .pdf or .mov in the link of documents				
36	Links should be relevant text. Do not link words like "here," "this page," etc.	Use relevant text for link	Carnegie Mellon University [cited January 2017]		Content	University
		Do not use link words as “here”, “this page”...				
37	For body copy, the recommended faces for the web, in order of preference, are Verdana, Arial and Helvetica. The browser should use Verdana first; if it is not available, use Arial and then Helvetica. If none are available, use another sans serif font	Use Verdana first as font	Carnegie Mellon University [cited January 2017]	They prefer sans serif fonts, in constrast of G85	Presentation	University

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
		Use Arial and Helvetica as font if Verdana is not available				
		Use another sans serif font as font if Verdana, Arial and Helvetica are not available				
38	The wordmark should appear in a prominent position on every web page. The best position is the top left corner	Place wordmark in a prominent position on every web page	Carnegie Mellon University [cited January 2017]	Similar to G351	Presentation	University
		The best position of wordmark is top left corner				
39	Page color should be brown, not white	Use brown page color	Lokman et al. [2009]	Partly same as G9, G189	Presentation	
		Do not use white page color				
40	Product display style should be filmstrip, not catalog	Use filmstrip product display style	Lokman et al. [2009]		Presentation	
		Do not use catalog product display style				
41	Header menu background color should be grey, not blue	Use grey header menu background color	Lokman et al. [2009]		Presentation	
		Do not use blue header menu background color				
42	Left menu font color should be white, not mix	Use white left menu font color	Lokman et al. [2009]		Presentation	
		Do not use mix left menu font color				
43	Header background color should be grey, not blue	Use grey header background color	Lokman et al. [2009]		Presentation	
		Do not use blue header background color				
44	Face expression should be mix, not none	Use mix face expression	Lokman et al. [2009]		Presentation	
		Do not use none face expression				
45	Body background color should be dark brown, not white	Use dark brown body background color	Lokman et al. [2009]		Presentation	
		Do not use white body background color				
46	Dominant item should be picture, not text	Use picture as dominant item	Lokman et al. [2009]		Content Presentation	
		Do not use text as dominant item				
47	Main text should not exist	Do not use main text	Lokman et al. [2009]		Presentation	
48	Main background color should be brown, not light blue	Use brown main background color	Lokman et al. [2009]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
		Do not use light blue main background color				
49	Main font style should be italic, not normal	Use italic main font style	Lokman et al. [2009]		Presentation	
		Do not use normal main font style				
50	Main font size should be medium, not large	Use medium main font size	Lokman et al. [2009]		Presentation	
		Do not use large main font size				
51	Right menu link style should be picture, not text	Use picture as right menu link style	Lokman et al. [2009]		Presentation	
		Do not use text as right menu link style				
52	Should not use much flashing or blinking graphics	Do not use much flashing or blinking graphics	AgeLight LCC [2001]		Content Presentation	
53	Should not be excessive pop-up windows and ads banners	Do not use much pop-up windows and ads banners	AgeLight LCC [2001]		Presentation	
54	Avoid distracting background elements, should use a light complementary background color	Avoid distracting background elements	AgeLight LCC [2001]		Presentation	
		Use light complementary background color				
55	Including hyperlinks within longer pages so viewers can "jump" from section to section with a single click	Include hyperlinks within longer pages so viewers can "jump" from section to section with a single click	AgeLight LCC [2001]		Navigation	
56	Leave a wide margin of 1.5 or more inches on the right side of the page		AgeLight LCC [2001]		Presentation	
57	Design for Internet appliances		AgeLight LCC [2001]		Content	
58	Making all graphical links large and static, increase the size of the area around a link, making it selectable	Make all graphical links large and static	AgeLight LCC [2001]		Content Presentation	
		Increase the size of the area around a graphical link				
		Make graphical link selectable				
59	Do not use any coding that will limit a user's ability to adjust or change his or her font, font size or colors		AgeLight LCC [2001]		Content Presentation	
60	When a user enlarges a Web page, images, including logos, banners and buttons, are not enlarged with the rest of the text on a page	Do not enlarge images (including logos, banners and buttons) when enlarging a web page	AgeLight LCC [2001]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
61	Links are consistently underlined to make identifiable and recognizable	Underline links consistently to make identifiable and recognizable	AgeLight LCC [2001]		Presentation	
62	Do not underline text or headlines that are not a link		AgeLight LCC [2001]		Presentation	
63	After being viewed, the link color should change from the traditional blue to purple or red	Change link color from the traditional blue to purple or red after being viewed	AgeLight LCC [2001]	Same as G7	Presentation	
64	Links should be descriptive, but no more than maximum of 10-12 words	Create descriptive links, but no more than maximum of 10-12 words	AgeLight LCC [2001]		Content	
65	Should use short pages for home pages and menu pages	Use short pages for home pages and menu pages	AgeLight LCC [2001]		Content	
66	Should use longer pages for content pages	Use longer pages for content pages	AgeLight LCC [2001]	Contrast to G195	Content	
67	Avoid creating large pages with multiple articles and links. Break topics down into succinct pages instead	Avoid creating large pages with multiple articles and links Break topics down into succinct pages	AgeLight LCC [2001]		Content	
68	Additional pages and articles should be kept smaller then 30,000 bytes in order to achieve a download time of 10 seconds	Keep additional pages and articles smaller than 30,000 bytes in order to achieve a download time of 10 seconds	AgeLight LCC [2001]		Content	
69	Accept redundancy of links both within a site and on navigation bars		AgeLight LCC [2001]		Navigation	
70	Should use left-hand alignment. It offers a high level of readability as compared to justification	Use left-hand alignment	AgeLight LCC [2001]	Same as G1	Presentation	
71	Centered text is best used for titles or very small amounts of copy within a text box	Use centered text for titles or very small amounts of copy within a text box	AgeLight LCC [2001]		Presentation	
72	Should design and apply consistent style sheets throughout site	Design and apply consistent style sheets throughout site	AgeLight LCC [2001]		Presentation	
73	Primary colors include red, blue and yellow that cannot be created by mixing other colors	Do not create primary colors by mixing other colors	AgeLight LCC [2001]		Presentation	
74	Keep colors bright and bold	Use colors bright and bold	AgeLight LCC [2001]		Presentation	
75	Should not use low saturation color (very pale or dark)	Do not use low saturation color	AgeLight LCC [2001]		Presentation	
76	Should not use exceptionally bright, fluorescence or vibrant colors	Do not use exceptionally bright, fluorescence or vibrant colors	AgeLight LCC [2001]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
77	Avoid combinations of blue and yellow or red and green		AgeLight LCC [2001]		Presentation	
78	The safest colors to use are black, white, blue and yellow where as red, green brown, grey and purple can be troublesome	Use safe colors such as black, white, blue and yellow Avoid using colors such as red, green brown, grey and purple	AgeLight LCC [2001]	This is opposite of G39	Presentation	
79	Do not use instructions which refer to objects by color		AgeLight LCC [2001]		Presentation	
80	Use dark type on light or white backgrounds		AgeLight LCC [2001]		Presentation	
81	Keeping to the most basic and common fonts	Keep the most basic and common fonts	AgeLight LCC [2001]		Presentation	
82	Should not use shadow of text	Do not use shadow of text	AgeLight LCC [2001]		Presentation	
83	Use consistent typefaces and fonts throughout your site	Use consistent typefaces and fonts	AgeLight LCC [2001]		Presentation	
84	Twelve to fourteen points are recommended font sizes while headlines and titles are typically two points larger	Use twelve to fourteen point font size Use two points larger of font sizes for headlines and titles	AgeLight LCC [2001]		Presentation	
85	In general, for print applications, serif typefaces are most legible, but on lower resolution and small monitors, this may not always be true	Use serif fonts for print applications Do not use serif fonts on lower resolution and small monitors	AgeLight LCC [2001]	G37 recommends only use sans serif font	Presentation	
86	Keep colors bright and bold. It is usually in the low saturation levels	Keep colors bright and bold	AgeLight LCC [2001]		Presentation	
87	Specific kernings can be made between letters to enhance legibility	Make specific kernings between letters	AgeLight LCC [2001]		Presentation	
88	The leading specified is 2 points larger than the typeface	Create leading fonts	AgeLight LCC [2001]	Same as G2	Presentation	
89	Should use bold or capitalize the first letter of each word instead of all capital letters in a heading	Use bold or capital letter for the first letter of each word in heading Do not use all capital letters in heading	AgeLight LCC [2001]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
90	Test pages as much as possible, from many different perspectives including computer types, browsers and monitor displays and resolutions		AgeLight LCC [2001]		Presentation	
91	Designing for an 800 x 600 resolution will insure the greatest degree of monitor compatibility	Design for an 800 x 600 resolution to insure the greatest degree of monitor compatibility	AgeLight LCC [2001]		Presentation	
92	Offer versions of downloadable documents or videos based on users' connectivity		AgeLight LCC [2001]		Content	
93	Date stamping pages lets site visitors know how current information is and increases their confidence in site	Add date stamps to page to let site's visitors know how current information is and to increase their confidence in site	AgeLight LCC [2001]		Content	
94	Insure every graphic element, logo and photo includes an "ALT tag" with the concise and descriptive description	Ensure an "ALT tag" with the concise and descriptive description for every graphic element, logo and photo	AgeLight LCC [2001]		Content	
95	Archive old articles and features on site, while maintaining the actual page URL		AgeLight LCC [2001]		Content	
96	Simple redirect from old links to a home page is an alternative to a user getting a message as "The page cannot be found"	Redirect from old links to a home page Do not redirect old links to a message as "The page cannot be found"	AgeLight LCC [2001]		Navigation	
97	Including "Boolean" search instructions to improve the user's ability	Provide "Boolean" search instructions	AgeLight LCC [2001]		Content	
98	Adding content and background information of the site (management's team bios, phone numbers, street address...)	Provide content and background information of site	AgeLight LCC [2001]		Content	
99	Trying to link to sites at the highest possible level, such as a home page or top level page in the case "page not found"	Link to sites at the highest possible level	AgeLight LCC [2001]		Navigation	
100	Keep a record of external sites that link to site for checking them frequently to insure they are still working or not		AgeLight LCC [2001]		Content Navigation	
101	Considering to open external links in new browser windows	Consider to open external links in new browser windows	AgeLight LCC [2001]		Presentation	
102	Check how long pages take to download over various connections and on different platforms		AgeLight LCC [2001]		Content	
103	If using tables, provide an alternate text-only version of page	Provide alternate text-only version of page for used table	AgeLight LCC [2001]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
104	If using frames, test to make sure site works well without them	Ensure site works well without frames if using them	AgeLight LCC [2001]		Content	
105	Text versions of sites are essential for severe vision impairments or blindness	Create text versions of sites for severe vision impairments or blindness	AgeLight LCC [2001]		Presentation	
106	Site should have multi language versions	Create multi language versions for international site	Maguire [2011]		Content	International site
107	If not, translating an introductory or 'welcome' page into several of the most important languages that the site is intended for	Translate an introductory or 'welcome' page into several of the most important languages that the site is intended for if site does not have multi language versions	Maguire [2011]		Content	International site
108	Adding translation widget to the website such as from Google Translate or Microsoft Live, combining with the option of selecting human translations	Add translation widget to the website such as from Google Translate or Microsoft Live to combine with the option of selecting human translations	Maguire [2011]		Content	International site
109	Should avoid phrases that are colloquialisms or slang only known to the local country	Avoid phrases that are colloquialisms or slang only known to the local country	Maguire [2011]		Content	International site
110	Getting the spelling right for the correct market	Make the spelling right for the correct market	Maguire [2011]	It concerns to G21	Content	International site
111	Design the website using Unicode	Use Unicode for website	Maguire [2011]		Content	International site
112	The field of form should appropriate for each country	Create field of form appropriate to each country	Maguire [2011]		Content	International site
113	Should use icons, symbols and design features on the sites that will not be confusing to others	Use icons, symbols, design features on the sites that will not be confusing to others	Maguire [2011]		Content Presentation	International site
114	Concentrating on a single usable layout rather than artificially enrich a page for a particular country or culture	Concentrate on a single usable layout rather than artificially enrich a page for a particular country or culture	Maguire [2011]		Content	International site
115	Considering both levels: 'high' and 'low' of cultural context for satisfying both viewpoints	Create both levels 'high' and 'low' of cultural context for satisfying both viewpoints	Maguire [2011]		Content	International site
116	E-commerce sites should use alternative payment mechanisms for countries	Use alternative payment mechanisms for countries for E-commerce sites	Maguire [2011]		Content	International site
117	Should build auto response service that sends an immediate reply to the enquirer informing them that they will receive a full reply within 24 or 48 hours	Create auto response service	Maguire [2011]		Content	International site

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
118	Indication of local time in the home website company	Indicate local time in home site of company	Maguire [2011]		Content	International site
119	The format of time should make it clear which one it is using	Make the format of time clear which one is using	Maguire [2011]		Content	International site
120	Write the date with the month represented in letters e.g. '24 June 2011' to prevent misunderstandings	Display the date with the month represented in letters	Maguire [2011]		Content	International site
121	Offering both measures (metric and imperial system) for ease of reference	Provide both measures (metric and imperial system)	Maguire [2011]		Content	International site
122	Pricing of goods and service in a local currency is the most preferred or alternatively providing a currency calculator on its web site	Use local currency for price of goods and service	Maguire [2011]		Content	International site
123	If there is a specific colour that a region or culture associates with the topic of the website then it could be a good choice of colour for that topic	Choose specific colour that a region or culture associates with the topic of website	Maguire [2011]	It concerns to G8 and generalization of some other guidelines about color as G39, G41, G42, G43	Presentation	International site
124	If there is no specific colour that is suitable for the topic, consider the feeling that website is intending to convey and try to choose a colour that stimulates it for that particular region or culture	Choose a colour that stimulates the topic of website for particular region or culture if there is no specific colour that is suitable for the topic	Maguire [2011]		Presentation	International site
125	Avoid choosing a colour that contradicts the topic area of the website or may be likely to evoke a negative reaction in the audience		Maguire [2011]		Presentation	International site
126	Assume all input is malicious		Microsoft Developer Network [cited January 2017]		Content	
127	Centralize your approach of input validation	Validate inputs	Microsoft Developer Network [cited January 2017]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
128	Do not rely on client-side validation		Microsoft Developer Network [cited January 2017]		Content	
129	Be careful with canonicalization issues		Microsoft Developer Network [cited January 2017]		Content	
130	Constrain, reject, and sanitize your input	Filter input	Microsoft Developer Network [cited January 2017]		Content	
131	Validate for type, length, format, and range	Validate type, length, format, and range	Microsoft Developer Network [cited January 2017]		Content	
132	Separate public and restricted areas		Microsoft Developer Network [cited January 2017]		Content	
133	Use account lockout policies for end-user accounts		Microsoft Developer Network [cited January 2017]		Content	
134	Support password expiration periods		Microsoft Developer Network [cited January 2017]		Content	
135	Be able to disable accounts	Allow disabling accounts	Microsoft Developer Network [cited January 2017]		Content	
136	Do not store passwords in user stores		Microsoft Developer Network [cited January 2017]		Content	
137	Require strong passwords		Microsoft Developer Network [cited January 2017]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
138	Do not send passwords over the wire in plaintext		Microsoft Developer Network [cited January 2017]		Content	
139	Protect authentication cookies		Microsoft Developer Network [cited January 2017]		Content	
140	Use multiple gatekeepers		Microsoft Developer Network [cited January 2017]		Content	
141	Restrict user access to system-level resources		Microsoft Developer Network [cited January 2017]		Content	
142	Consider authorization granularity		Microsoft Developer Network [cited January 2017]		Content	
143	Secure your administration interfaces	Secure administration interfaces	Microsoft Developer Network [cited January 2017]		Content Presentation	
144	Secure your configuration store		Microsoft Developer Network [cited January 2017]		Content	
145	Maintain separate administration privileges		Microsoft Developer Network [cited January 2017]		Content	
146	Use least privileged process and service accounts		Microsoft Developer Network [cited January 2017]		Content	
147	Do not store secrets if you can avoid it		Microsoft Developer Network [cited January 2017]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
148	Do not store secrets in code		Microsoft Developer Network [cited January 2017]		Content	
149	Do not store database connections, passwords, or keys in plaintext		Microsoft Developer Network [cited January 2017]		Content	
150	Avoid storing secrets in the Local Security Authority (LSA)		Microsoft Developer Network [cited January 2017]		Content	
151	Use Data Protection API (DPAPI) for encrypting secrets		Microsoft Developer Network [cited January 2017]		Content	
152	Retrieve sensitive data on demand		Microsoft Developer Network [cited January 2017]		Content	
153	Encrypt the data or secure the communication channel	Encrypt the data Secure the communication channel	Microsoft Developer Network [cited January 2017]		Content	
154	Do not store sensitive data in persistent cookies		Microsoft Developer Network [cited January 2017]		Content	
155	Do not pass sensitive data using the HTTP-GET protocol	Do not pass sensitive data which uses the HTTP-GET protocol	Microsoft Developer Network [cited January 2017]		Content	
156	Use SSL to protect session authentication cookies		Microsoft Developer Network [cited January 2017]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
157	Encrypt the contents of the authentication cookies		Microsoft Developer Network [cited January 2017]		Content	
158	Limit session lifetime		Microsoft Developer Network [cited January 2017]		Content	
159	Protect session state from unauthorized access		Microsoft Developer Network [cited January 2017]		Content	
160	Do not develop your own cryptography	Do not create own cryptography	Microsoft Developer Network [cited January 2017]		Content	
161	Keep unencrypted data close to the algorithm		Microsoft Developer Network [cited January 2017]		Content	
162	Use the correct algorithm and correct key size		Microsoft Developer Network [cited January 2017]		Content	
163	Secure your encryption keys	Secure encryption keys	Microsoft Developer Network [cited January 2017]		Content	
164	Encrypt sensitive cookie state		Microsoft Developer Network [cited January 2017]		Content	
165	Make sure that users do not bypass your checks	Do not allow users to bypass the checks	Microsoft Developer Network [cited January 2017]		Content	
166	Validate all values sent from the client	Validate all values which are sent from the client	Microsoft Developer Network [cited January 2017]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
167	Do not trust HTTP header information		Microsoft Developer Network [cited January 2017]		Content	
168	Do not leak information to the client	Do not provide information to the client	Microsoft Developer Network [cited January 2017]		Content	
169	Log detailed error messages		Microsoft Developer Network [cited January 2017]		Content	
170	Catch exceptions		Microsoft Developer Network [cited January 2017]		Content	
171	Audit and log access across application tiers		Microsoft Developer Network [cited January 2017]		Content	
172	Consider identity flow		Microsoft Developer Network [cited January 2017]		Content	
173	Log key events		Microsoft Developer Network [cited January 2017]		Content	
174	Secure log files		Microsoft Developer Network [cited January 2017]		Content	
175	Back up and analyze log files regularly	Back up log files regularly Analyze log files regularly	Microsoft Developer Network [cited January 2017]		Content	
176	Limit navigational topics		Meloncon et al. [2010]	Similar to G15	Navigation	Children site

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
177	Use literal icons and directional images to point the way through the navigation		Meloncon et al. [2010]		Navigation	Children site
178	Do not include search options, in-text links, or pop-ups—all add an unnecessary layer of complexity	Do not include search options, in-text links, or pop-ups	Meloncon et al. [2010]	Similar to G16	Navigation	Children site
179	Take care to make clickable hotspots obvious through rollover effects (e.g., blinking, changing color)	Consider to make...	Meloncon et al. [2010]		Navigation	Children site
180	Use representational images that children can recognize from their everyday lives		Meloncon et al. [2010]	Partly same as G22	Navigation	Children site
181	Provide multiple options for navigation (e.g., breadcrumbs, prominently-displayed "back" button, browser's "back" button)	Provide multiple options for navigation	Meloncon et al. [2010]		Navigation	Children site
182	Use images from children's everyday lives		Meloncon et al. [2010]		Presentation	Children site
183	Age-appropriate mascots can be helpful, but they should play a role in the interface	Use age-appropriate mascots	Meloncon et al. [2010]		Presentation	Children site
184	Avoid graphics for visual interest alone		Meloncon et al. [2010]		Presentation	Children site
185	Use vivid colors		Meloncon et al. [2010]		Presentation	Children site
186	Avoid excessive use of white		Meloncon et al. [2010]	Partly same as G9, G39	Presentation	Children site
187	Incorporate games that play a role in the site's learning objectives		Meloncon et al. [2010]		Presentation	Children site
188	Surpass minimum WCAG 2.0 guidelines so that all children can participate		Meloncon et al. [2010]		Presentation	Children site
189	Keep sites simple or provide alternatives to complex content	Keep sites simple Provide alternatives to complex content	Meloncon et al. [2010]		Presentation	Children site
190	Use content appropriate for the average reader in the site's target age group		Meloncon et al. [2010]		Content	Children site
191	Use concrete words, active verbs, and concise sentence structure		Meloncon et al. [2010]	Similar to G25	Content	Children site
192	Organize content efficiently and effectively		Meloncon et al. [2010]		Content	Children site
193	Provide clear directions and goals		Meloncon et al. [2010]		Content	Children site

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
194	Limit the length of sentences and paragraphs to increase readability		Meloncon et al. [2010]		Content	Children site
195	Segment page length based on concepts	Segment page length which is based on concepts	Meloncon et al. [2010]	Contrast to G66	Content	Children site
196	Provide useful content		U.S. Dept. of Health and Human Services [2006]		Content	
197	Establish user requirements		U.S. Dept. of Health and Human Services [2006]		Content	
198	Understand and meet user's expectations		U.S. Dept. of Health and Human Services [2006]		Content	
199	Involve users in establishing user requirements		U.S. Dept. of Health and Human Services [2006]		Content	
200	Set and state goals		U.S. Dept. of Health and Human Services [2006]		Content	
201	Focus on performance before preference	Make decisions about content, format, interaction, and navigation before deciding on colors and decorative graphics if user performance is important	U.S. Dept. of Health and Human Services [2006]		Content Navigation Presentation	
202	Consider many user interface issues		U.S. Dept. of Health and Human Services [2006]		Presentation	
203	Be easily found in the top 30	Make site be easily found in the top 30	U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
204	Set usability goals		U.S. Dept. of Health and Human Services [2006]		Content	
205	Use parallel design		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
206	Use personas		U.S. Dept. of Health and Human Services [2006]		Content	
207	Do not display unsolicited windows or graphics		U.S. Dept. of Health and Human Services [2006]		Presentation	
208	Increase web site credibility		U.S. Dept. of Health and Human Services [2006]		Content	
209	Standardize task sequences		U.S. Dept. of Health and Human Services [2006]		Content	
210	Reduce the user's workload		U.S. Dept. of Health and Human Services [2006]		Content	
211	Design for working memory limitations		U.S. Dept. of Health and Human Services [2006]		Content	
212	Minimize page download time		U.S. Dept. of Health and Human Services [2006]		Content	
213	Let users know if a page is programmed to 'time out', and warn users before time expires so they can request additional time	Let users know if a page is programmed to 'time out	U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
		Warn users before time expires so they can request additional time				
214	Display information in a directly usable format		U.S. Dept. of Health and Human Services [2006]		Content	
215	Display information in a directly usable format		U.S. Dept. of Health and Human Services [2006]		Content	
216	Provide feedback when users must wait		U.S. Dept. of Health and Human Services [2006]		Content	
217	Inform users of long download times		U.S. Dept. of Health and Human Services [2006]		Content	
218	Develop pages that will print properly		U.S. Dept. of Health and Human Services [2006]		Content	
219	Do not require users to multitask while reading		U.S. Dept. of Health and Human Services [2006]		Content	
220	Use users' terminology in help documentation		U.S. Dept. of Health and Human Services [2006]		Content	
221	Provide printing options		U.S. Dept. of Health and Human Services [2006]		Content	
222	Provide assistance to users		U.S. Dept. of Health and Human Services [2006]		Content Navigation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
223	Comply with section 508		U.S. Dept. of Health and Human Services [2006]		Content Navigation Presentation	
224	Design forms for users using assistive technologies		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
225	Do not use color alone to convey information		U.S. Dept. of Health and Human Services [2006]		Content	
226	Enable users to skip repetitive navigation links		U.S. Dept. of Health and Human Services [2006]		Navigation	
227	Provide text equivalents for non-text elements		U.S. Dept. of Health and Human Services [2006]		Content	
228	Test plug-ins and applets for accessibility		U.S. Dept. of Health and Human Services [2006]		Content	
229	Ensure that scripts allow accessibility		U.S. Dept. of Health and Human Services [2006]		Content	
230	Provide equivalent pages		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
231	Provide client-side image maps		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
232	Synchronize multimedia elements		U.S. Dept. of Health and Human Services [2006]		Content	
233	Do not require style sheets		U.S. Dept. of Health and Human Services [2006]		Presentation	
234	Provide frame titles		U.S. Dept. of Health and Human Services [2006]		Content	
235	Avoid screen flicker		U.S. Dept. of Health and Human Services [2006]		Presentation	
236	Design for common browsers	Design site for common browsers	U.S. Dept. of Health and Human Services [2006]		Content Presentation Navigation	
237	Account for browser differences		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
238	Design for popular operating systems		U.S. Dept. of Health and Human Services [2006]		Content Presentation Navigation	
239	Design for user's typical connection speed		U.S. Dept. of Health and Human Services [2006]		Content Presentation Navigation	
240	Design for commonly used screen resolutions		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
241	Enable access to the homepage		U.S. Dept. of Health and Human Services [2006]		Navigation	
242	Show all major options on the homepage	Display all major...	U.S. Dept. of Health and Human Services [2006]		Content	
243	Create a positive first impression of your site	Create homepage as the key to conveying the quality of site	U.S. Dept. of Health and Human Services [2006]		Content Presentation	
244	Communicate the web site's value and purpose	Display the purpose and value of the website on the homepage clearly and prominently	U.S. Dept. of Health and Human Services [2006]		Content	
245	Limit prose text on the homepage		U.S. Dept. of Health and Human Services [2006]		Content	
246	Ensure the homepage looks like a homepage		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
247	Limit homepage length		U.S. Dept. of Health and Human Services [2006]		Content	
248	Announce changes to a web site		U.S. Dept. of Health and Human Services [2006]		Content	
249	Attend to homepage panel width		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
250	Avoid cluttered displays		U.S. Dept. of Health and Human Services [2006]		Presentation	
251	Place important items consistently		U.S. Dept. of Health and Human Services [2006]		Presentation	
252	Place important items at top center		U.S. Dept. of Health and Human Services [2006]		Presentation	
253	Structure for easy comparison		U.S. Dept. of Health and Human Services [2006]		Presentation	
254	Establish level of importance		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
255	Optimize display density		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
256	Visually align page elements, either vertically or horizontally	Align page elements visually, either vertically or horizontally	U.S. Dept. of Health and Human Services [2006]		Presentation	
257	Use fluid layouts		U.S. Dept. of Health and Human Services [2006]		Presentation	
258	Avoid scroll stoppers		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
259	Set appropriate page lengths		U.S. Dept. of Health and Human Services [2006]		Content	
260	Use moderate white space		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
261	Choose appropriate line lengths		U.S. Dept. of Health and Human Services [2006]		Content	
262	Use frames when functions must remain accessible		U.S. Dept. of Health and Human Services [2006]		Presentation	
263	Provide navigational options		U.S. Dept. of Health and Human Services [2006]		Navigation	
264	Clearly differentiate navigation elements from one another, but group and place them in a consistent and easy to find place on each page	Differentiate navigation elements from one another clearly Group and place navigation elements in a consistent and easy to find place on each page	U.S. Dept. of Health and Human Services [2006]		Navigation Presentation	
265	Use a clickable 'list of contents' on long pages		U.S. Dept. of Health and Human Services [2006]		Navigation	
266	Provide feedback on users' location		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
267	Place primary navigation menus in the left panel		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
268	Use descriptive tab labels		U.S. Dept. of Health and Human Services [2006]		Content	
269	Present tabs effectively		U.S. Dept. of Health and Human Services [2006]		Presentation	
270	Keep navigation-only pages short		U.S. Dept. of Health and Human Services [2006]		Content	
271	Use appropriate menu types		U.S. Dept. of Health and Human Services [2006]		Presentation	
272	Use site maps		U.S. Dept. of Health and Human Services [2006]		Navigation	
273	Use 'glosses' to assist navigation		U.S. Dept. of Health and Human Services [2006]		Content	
274	Breadcrumb navigation	Use breadcrumb navigation	U.S. Dept. of Health and Human Services [2006]		Navigation	
275	Eliminate horizontal scrolling		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
276	Facilitate rapid scrolling while reading		U.S. Dept. of Health and Human Services [2006]		Presentation	
277	Use scrolling pages for reading comprehension		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
278	Use paging rather than scrolling		U.S. Dept. of Health and Human Services [2006]		Content	
279	Scroll fewer screenfuls	Separate information into shorter pages if users are looking for specific information	U.S. Dept. of Health and Human Services [2006]		Content	
280	Use clear category labels		U.S. Dept. of Health and Human Services [2006]		Content	
281	Provide descriptive page titles		U.S. Dept. of Health and Human Services [2006]		Content	
282	Use descriptive headings liberally		U.S. Dept. of Health and Human Services [2006]		Content	
283	Use unique and descriptive headings		U.S. Dept. of Health and Human Services [2006]		Content	
284	Highlight critical data		U.S. Dept. of Health and Human Services [2006]		Presentation	
285			U.S. Dept. of Health and Human Services [2006]		Content	
286	Use headings in the appropriate HTML order		U.S. Dept. of Health and Human Services [2006]		Presentation	
287	Provide users with good ways to reduce options	Provide good ways to reduce options	U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
288	Use meaningful link labels		U.S. Dept. of Health and Human Services [2006]		Content	
289	Link to related content		U.S. Dept. of Health and Human Services [2006]		Content Navigation	
290	Match link names with their destination pages		U.S. Dept. of Health and Human Services [2006]		Content	
291	Avoid misleading cues to click		U.S. Dept. of Health and Human Services [2006]		Presentation	
292	Repeat important links		U.S. Dept. of Health and Human Services [2006]		Content	
293	Use text for links		U.S. Dept. of Health and Human Services [2006]		Presentation	
294	Designate used links		U.S. Dept. of Health and Human Services [2006]		Presentation	
295	Provide consistent clickability cues		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
296	Ensure that embedded links are descriptive		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
297	Pointing-and-clicking,' rather than mousing over, is preferred when selecting menu items from a cascading menu structure	Use 'pointing-and-clicking' rather than mousing over when selecting menu items from a cascading menu structure	U.S. Dept. of Health and Human Services [2006]		Presentation	
298	Use appropriate text link lengths		U.S. Dept. of Health and Human Services [2006]		Content	
299	Indicate internal vs. external links		U.S. Dept. of Health and Human Services [2006]		Content Navigation	
300	Clarify clickable regions of images		U.S. Dept. of Health and Human Services [2006]		Presentation	
301	Link to supportive information		U.S. Dept. of Health and Human Services [2006]		Content	
302	Use black text on plain, high-contrast backgrounds		U.S. Dept. of Health and Human Services [2006]		Presentation	
303	Format common items consistently	Keep the format of common items consistent from one page to another	U.S. Dept. of Health and Human Services [2006]		Presentation	
304	Use mixed-case for prose text		U.S. Dept. of Health and Human Services [2006]		Presentation	
305	Ensure visual consistency		U.S. Dept. of Health and Human Services [2006]		Presentation	
306	Use bold text sparingly		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
307	Use attention-attracting features when appropriate		U.S. Dept. of Health and Human Services [2006]		Presentation	
308	Use familiar fonts		U.S. Dept. of Health and Human Services [2006]		Presentation	
309	Use at least 12-point font		U.S. Dept. of Health and Human Services [2006]		Presentation	
310	Color-coding and instructions	Make color-coding scheme be quickly and easily understood when using it	U.S. Dept. of Health and Human Services [2006]		Presentation	
311	Emphasize importance		U.S. Dept. of Health and Human Services [2006]		Presentation	
312	Highlighting information	Highlight important information	U.S. Dept. of Health and Human Services [2006]	Same as G27	Presentation	
313	Order elements to maximize user performance		U.S. Dept. of Health and Human Services [2006]		Presentation	
314	Place important items at top of the list		U.S. Dept. of Health and Human Services [2006]		Content	
315	Format lists to ease scanning		U.S. Dept. of Health and Human Services [2006]		Presentation	
316	Display related items in lists		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
317	Introduce each list		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
318	Use static menus		U.S. Dept. of Health and Human Services [2006]		Content	
319	Start numbered items at one	Start the numbering sequence at 'one' rather than 'zero' when items are numbered	U.S. Dept. of Health and Human Services [2006]		Content	
320	Use appropriate list style		U.S. Dept. of Health and Human Services [2006]		Presentation	
321	Capitalize first letter of first word in lists	Use capital letter for first letter of first word in lists	U.S. Dept. of Health and Human Services [2006]		Content	
322	Distinguish required and optional data entry fields		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
323	Label pushbuttons clearly		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
324	Label data entry fields consistently		U.S. Dept. of Health and Human Services [2006]		Content	
325	Do not make user-entered codes case sensitive		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
326	Label data entry fields clearly		U.S. Dept. of Health and Human Services [2006]		Content	
327	Minimize user data entry		U.S. Dept. of Health and Human Services [2006]		Content	
328	Put labels close to data entry fields		U.S. Dept. of Health and Human Services [2006]		Presentation	
329	Allow users to see their entered data		U.S. Dept. of Health and Human Services [2006]		Presentation	
330	Use radio buttons for mutually exclusive selections		U.S. Dept. of Health and Human Services [2006]		Presentation	
331	Use familiar widgets		U.S. Dept. of Health and Human Services [2006]		Presentation	
332	Anticipate typical user errors		U.S. Dept. of Health and Human Services [2006]		Content	
333	Partition long data items		U.S. Dept. of Health and Human Services [2006]		Presentation	
334	Use a single data entry method		U.S. Dept. of Health and Human Services [2006]		Presentation	
335	Prioritize pushbuttons		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
336	Use check boxes to enable multiple selection		U.S. Dept. of Health and Human Services [2006]		Presentation	
337	Label units of measurement		U.S. Dept. of Health and Human Services [2006]		Content	
338	Do not limit viewable list box options		U.S. Dept. of Health and Human Services [2006]		Presentation	
339	Display default values		U.S. Dept. of Health and Human Services [2006]		Content	
340	Place cursor in first data entry field		U.S. Dept. of Health and Human Services [2006]		Presentation	
341	Ensure that double-clicking will not cause problems		U.S. Dept. of Health and Human Services [2006]		Navigation	
342	Use open lists to select one from many	Use open lists for selecting on from many	U.S. Dept. of Health and Human Services [2006]		Presentation	
343	Use data entry fields to speed performance		U.S. Dept. of Health and Human Services [2006]		Presentation	
344	Use a minimum of two radio buttons		U.S. Dept. of Health and Human Services [2006]		Presentation	
345	Provide auto-tabbing functionality		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
346	Minimize use of the shift key		U.S. Dept. of Health and Human Services [2006]		Content	
347	Use simple background images		U.S. Dept. of Health and Human Services [2006]		Presentation	
348	Label clickable images		U.S. Dept. of Health and Human Services [2006]		Content	
349	Ensure that images do not slow downloads		U.S. Dept. of Health and Human Services [2006]		Content	
350	Use video, animation, and audio meaningfully		U.S. Dept. of Health and Human Services [2006]		Content	
351	Include logos		U.S. Dept. of Health and Human Services [2006]	Similar to G38	Content Presentation	
352	Graphics should not look like banner ads	Do not create/use graphics look like banner ads	U.S. Dept. of Health and Human Services [2006]		Presentation	
353	Limit large images above the fold		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
354	Ensure web site images convey intended messages		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
355	Limit the use of images		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
356	Include actual data with data graphics		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
357	Display monitoring information graphically		U.S. Dept. of Health and Human Services [2006]		Presentation	
358	Introduce animation		U.S. Dept. of Health and Human Services [2006]		Content	
359	Emulate real-world objects		U.S. Dept. of Health and Human Services [2006]		Content	
360	Use thumbnail images to preview larger images	Use thumbnail images for previewing larger images	U.S. Dept. of Health and Human Services [2006]		Content	
361	Use images to facilitate learning		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
362	Using photographs of people	Use photographs of people	U.S. Dept. of Health and Human Services [2006]		Content	
363	Make action sequences clear	Structure the content so that the sequence is obvious and consistent when describing an action or task that has a natural order or sequence	U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
364	Avoid jargon		U.S. Dept. of Health and Human Services [2006]		Content	
365	Use familiar words		U.S. Dept. of Health and Human Services [2006]		Content	
366	Define acronyms and abbreviations		U.S. Dept. of Health and Human Services [2006]		Content	
367	Use abbreviations sparingly		U.S. Dept. of Health and Human Services [2006]		Content	
368	Use mixed case with prose		U.S. Dept. of Health and Human Services [2006]	Same as G304	Content Presentation	
369	Limit the number of words and sentences	Limit number of words and sentences	U.S. Dept. of Health and Human Services [2006]		Content	
370	Limit prose text on navigation pages		U.S. Dept. of Health and Human Services [2006]		Content	
371	Use active voice		U.S. Dept. of Health and Human Services [2006]		Content	
372	Write instructions in the affirmative		U.S. Dept. of Health and Human Services [2006]		Content	
373	Make first sentences descriptive		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
374	Organize information clearly		U.S. Dept. of Health and Human Services [2006]		Content	
375	Facilitate scanning		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
376	Ensure that necessary information is displayed	Display necessary information	U.S. Dept. of Health and Human Services [2006]		Content	
377	Group related elements		U.S. Dept. of Health and Human Services [2006]		Content Presentation	
378	Minimize the number of clicks or pages		U.S. Dept. of Health and Human Services [2006]		Navigation	
379	Design quantitative content for quick understanding	Design quantitative content for understanding quickly	U.S. Dept. of Health and Human Services [2006]		Content	
380	Display only necessary information	Use another sans serif font as font if Verdana, Arial and Helvetica are not available	U.S. Dept. of Health and Human Services [2006]	Same as G376	Content	
381	Format information for multiple audiences		U.S. Dept. of Health and Human Services [2006]		Content	
382	Use color for grouping		U.S. Dept. of Health and Human Services [2006]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
383	Ensure usable search results		U.S. Dept. of Health and Human Services [2006]		Content Navigation	
384	Design search engines to search the entire site		U.S. Dept. of Health and Human Services [2006]		Content	
385	Make upper- and lowercase search terms equivalent		U.S. Dept. of Health and Human Services [2006]		Content	
386	Provide a search option on each page		U.S. Dept. of Health and Human Services [2006]		Content	
387	Design search around users' terms		U.S. Dept. of Health and Human Services [2006]		Content	
388	Allow simple searches	Provide simple searches	U.S. Dept. of Health and Human Services [2006]		Content	
389	Notify users when multiple search options exist		U.S. Dept. of Health and Human Services [2006]		Content	
390	Include hints to improve search performance		U.S. Dept. of Health and Human Services [2006]		Content	
391	Provide search templates		U.S. Dept. of Health and Human Services [2006]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
392	Provide captions and text alternatives for images and multimedia		Ministry of Community and Social Services of Ontario [2012]		Content	
393	Use strong contrast between text and background		Ministry of Community and Social Services of Ontario [2012]		Presentation	
394	Create content that can be presented using assistive technologies (like screen readers) without losing meaning		Ministry of Community and Social Services of Ontario [2012]		Content	
395	Use structured content and make it keyboard accessible	Use structured content	Ministry of Community and Social Services of Ontario [2012]		Content	
		Make content keyboard accessible				
396	Avoid CAPTCHAs and give users enough time to read and use content	Avoid CAPTCHAs	Ministry of Community and Social Services of Ontario [2012]		Content	
		Allow users enough time to read and use content				
397	Avoid using time limits when asking users to provide a response or information		Ministry of Community and Social Services of Ontario [2012]		Content	
398	Avoid blinking images		Ministry of Community and Social Services of Ontario [2012]		Content Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
399	Help users navigate and find content		Ministry of Community and Social Services of Ontario [2012]		Navigation	
400	Help users avoid and correct mistakes		Ministry of Community and Social Services of Ontario [2012]		Content	
401	Make the tables accessible		Ministry of Community and Social Services of Ontario [2012]		Content	
402	Video content must have closed captioning	Create closed caption for video content	Ministry of Community and Social Services of Ontario [2012]		Content Presentation	
403	Use headings for different topics and subtopics of the text		Ozok and Salvendy [2004]		Content Presentation	
404	Use same words for same items, rather than using synonyms		Ozok and Salvendy [2004]		Content	
405	Do not compromise structural English rules		Ozok and Salvendy [2004]		Content	
406	Be as explicit as possible; do not leave anything to the user's imagination and interpretation	Create content as explicit as possible Do not leave anything to the user's imagination and interpretation	Ozok and Salvendy [2004]		Content	
407	Avoid ambiguity in your explanations	Avoid ambiguity in explanations	Ozok and Salvendy [2004]		Content	
408	Do not deviate from the main topic of your text		Ozok and Salvendy [2004]		Content	
409	Do not use ill-formed sentences or words for any reason		Ozok and Salvendy [2004]		Content	
410	Make sure none of the statements in the text contradicts each other	Ensure that none of the statements in the text contradicts each other	Ozok and Salvendy [2004]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
411	Know your user; use wording and language that can be understood entirely by your target audience	Use wording and language that can be understood entirely by target audience	Ozok and Salvendy [2004]		Content	
412	Use your words consistently within and across interfaces		Ozok and Salvendy [2004]		Content	
413	Make sure same words have the same meanings within and across interfaces	Keep same words have the same meanings within and across interfaces	Ozok and Salvendy [2004]		Content	
414	Use correct words that accurately describe your item	Use correct words that accurately describe the item	Ozok and Salvendy [2004]		Content	
415	Do not confuse the user with unfamiliar words and sentence structures		Ozok and Salvendy [2004]		Content	
416	Be consistent in terms of character sizes, use of upper/lower case letters, spacing, punctuation, character colors, and wording in your text	Use character sizes, upper/lower case letters, spacing, punctuation, character colors, and wording in text consistently	Ozok and Salvendy [2004]		Content Presentation	
417	Guide your users when they are supposed to perform an action	Guide users when they are supposed to perform an action	Ozok and Salvendy [2004]		Content	
418	Use consistent location of text		Ozok and Salvendy [2004]		Content	
419	Do not use unrecognizable text characters		Ozok and Salvendy [2004]		Content	
420	Have a convenient, easy-to-access layout within and between interfaces	Create a convenient, easy-to-access layout within and between interfaces	Ozok and Salvendy [2004]		Presentation	
421	Use consistent sizes, shapes, and colors for screen elements such as menus, combo boxes, radio buttons, and check boxes		Ozok and Salvendy [2004]		Presentation	
422	Use recognizable screen elements across interfaces		Ozok and Salvendy [2004]		Presentation	
423	Communicate menu structure through numbering	Number every menu item	Leuthold et al. [2008]			
		Announce total number of menu items before the menu				
424	Label all user interface elements		Leuthold et al. [2008]		Presentation	
425	Place buttons after options in forms		Leuthold et al. [2008]			
426	Do not use unnecessary words to create context		Leuthold et al. [2008]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
427	Frame every page with the same elements	Create every page with the same elements	Leuthold et al. [2008]		Presentation	
428	Add navigation menu on all pages, except pages at the end of the page hierarchy		Leuthold et al. [2008]		Navigation	
429	Place generic navigation and continuative links at the bottom of the page		Leuthold et al. [2008]		Navigation	
430	Place search on top of the homepage to facilitate task initiation	Place search on top of the homepage	Leuthold et al. [2008]		Content	
431	Eliminate all visual elements used solely for layout and branding		Leuthold et al. [2008]		Content Presentation	
432	Let people provide answers in a format that they are familiar with from common situations and keep questions in an intuitive sequence	Allow people provide answers in a format that they are familiar with from common situations	Bargas-Avila et al. [2010]		Content	
		Keep question in an intuitive sequence				
433	If the answer is unambiguous, allow answers in any format	Allow answers in any format if the answer is unambiguous	Bargas-Avila et al. [2010]		Content	
434	Keep the form as short and simple as possible and do not ask for unnecessary input	Keep the form as short and simple as possible	Bargas-Avila et al. [2010]		Content	
		Do not require unnecessary input in the form				
435	If possible and reasonable, separate required from optional fields and use color and asterisk to mark required fields	Separate required fields from optional fields	Bargas-Avila et al. [2010]		Content Presentation	
		Use color and asterisk to mark required fields				
436	To enable people to fill in a form as fast as possible, place the labels above the corresponding input fields	Place labels above the corresponding input fields to enable people to fill in a form as fast as possible	Bargas-Avila et al. [2010]		Presentation	
437	Do not separate a form into more than one column and only ask one question per row	Do not separate a form into more than one column	Bargas-Avila et al. [2010]		Content Presentation	
		Create one question per row in form				
438	Match the size of the input fields to the expected length of the answer	Match size of the input fields to expected length of the answer	Bargas-Avila et al. [2010]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
439	Use checkboxes, radio buttons or drop-down menus to restrict the number of options and for entries that can easily be mistyped. Also use them if it is not clear to users in advance what kind of answer is expected from them	Use checkboxes, radio buttons or drop-down menus to restrict the number of options and for entries that can easily be mistyped	Bargas-Avila et al. [2010]		Presentation	
		Use checkboxes, radio buttons or drop-down menus if it is not clear to users in advance what kind of answer is expected from them				
440	Use checkboxes instead of list boxes for multiple selection items	Use checkbox for multiple selection items	Bargas-Avila et al. [2010]		Presentation	
		Do not use list box for multiple selection items				
441	For up to four options, use radio buttons; when more than four options are required, use a drop-down menu to save screen real estate	Use radio buttons for up to four options	Bargas-Avila et al. [2010]		Presentation	
		Use drop-down menu for more than four options				
442	Order options in an intuitive sequence (e.g., weekdays in the sequence Monday, Tuesday, etc.). If no meaningful sequence is possible, order them alphabetically	Order options in an intuitive sequence	Bargas-Avila et al. [2010]		Content	
		Order options alphabetically if no meaningful sequence is possible				
443	For date entries use a drop-down menu when it is crucial to avoid format errors. Use only one input field and place the format requirements with symbols (MM, YYYY) left or inside the text box to achieve faster completion time	Use a drop-down menu for date entries when it is crucial to avoid format error	Bargas-Avila et al. [2010]		Content Presentation	
		Use only one input field for date entries				
		Place format requirements with symbols left or inside the text box to achieve faster completion time				
444	If answers are required in a specific format, state this in advance communicating the imposed rule	State answers in advance communicating the format specification if answers are required in a specific format	Bargas-Avila et al. [2010]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
445	Error messages should be polite and explain to the user in familiar language that a mistake has occurred. Eventually the error message should apologize for the mistake and it should clearly describe what the mistake is and how it can be corrected	Create polite error messages	Bargas-Avila et al. [2010]		Content	
		Create error messages which explain to the user in familiar language that a mistake has occured				
		Create error messages which describe clearly what the mistake is and how it can be corrected				
446	After an error occurred, never clear the already completed fields	Do not clear already completed fields after an error occurred	Bargas-Avila et al. [2010]		Presentation	
447	Always show error messages after the form has been filled and sent. Show them all together embedded in the form	Display error messages after the form has been filled and sent	Bargas-Avila et al. [2010]		Content	
		Show error messages all together embedded in the form				
448	Error messages must be noticeable at a glance, using color, icons and text to highlight the problem area and must be written in a familiar language, explaining what the error is and how it can be corrected	Use color, icons and text to highlight the problem area in error messages	Bargas-Avila et al. [2010]		Content Presentation	
449	Disable the submit button as soon as it has been clicked to avoid multiple submissions		Bargas-Avila et al. [2010]		Presentation	
450	After the form has been sent, show a confirmation site, which expresses thanks for the submission and states what will happen next. Send a similar confirmation by e-mail	Display confirmation site which expresses thanks for the submission and states what will happen next after the form has been sent	Bargas-Avila et al. [2010]		Content	
		Send confirmation email after the form has been sent				
451	Do not provide reset buttons, as they can be clicked by accident. If used anyway, make them visually distinctive from submit buttons and place them left-aligned with the cancel button on the right of the submit button	Do not provide reset buttons, as they can be clicked by accident	Bargas-Avila et al. [2010]		Presentation	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
		Make reset buttons visually distinctive from submit buttons if used them				
		Place reset buttons left-aligned with cancel button on the right of submit button if used them				
452	Due to the high volatility of Internet users, videos should not be too long, i.e., the recommended duration should be between two and ten minutes	Do not create too long videos due to the high volatility of Internet users	Chiuchi et al. [2011]		Content	
453	Determining the appropriate size with three types of connection speed (up to 256kbps, between 256 kbps and 1Mbps, above 1Mbps) of video from two to ten minutes as equation: Average size of video = download per second x display time.	Determine the appropriate size of video with three types of connection speed	Chiuchi et al. [2011]		Content	
454	The preferred use of JPEG and GIF images in an attempt to ensure a better user experience	Use JPEG and GIF images to ensure a better user experience	Chiuchi et al. [2011]		Content	
455	The resolution of image should be set correctly inside the tags, specifying the value of the image that is loaded by the browser to prevent the resizing of images loaded by the browser	Set resolution of image correctly inside the tags	Chiuchi et al. [2011]		Content	
		Specify the value of the image that is loaded by the browser to prevent the resizing of images loaded by the browser				
456	Determining the appropriate size with three types of connection speed (up to 256kbps, between 256 kbps and 1Mbps, above 1Mbps) of a webpage with the waiting time being 10 seconds as equation: Maximum size = download speed x maximum waiting time	Determine the appropriate size of webpage with three types of connection speed	Chiuchi et al. [2011]		Content	
457	Websites should be developed to make available a version where the images are not loaded by the browser whenever the user so wishes	Create a version of website where the images are not loaded by the browser whenever the user so wishes	Chiuchi et al. [2011]		Content	
458	Use of titles for all images of the website	Provide titles for all images of the websites	Chiuchi et al. [2011]		Content	

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
459	The overall readability of the site should remain unchanged. It means that the text-only version and inclusion of titles in the images must be implemented.	Implement text-only version and inclusion of titles in the images	Chiuchi et al. [2011]		Content	
460	Number of pages = (Download time + reading time) / Remaining time of the visit	Create number of pages which follows the formula	Chiuchi et al. [2011]		Content	
461	The need for scrolling or rolling of the website page should be avoided. Therefore, the developer should avoid the use of long pages	Avoid using scrolling or rolling of the website page Avoid the use of long pages	Chiuchi et al. [2011]		Content Presentation	
462	It should use of short texts, with an option to expand them if the reader so desires	Use short texts, with an option to expand them if the reader so desires	Chiuchi et al. [2011]		Content	
463	It should avoid incorporating scripts and flash on items considered essential in the website	Avoid incorporating scripts and flash on items which are considered essential in website	Chiuchi et al. [2011]		Content	
464	It should use the navigational map	Use navigational map	Chiuchi et al. [2011]		Navigation	
465	The use of one of the two resolutions (1024x768 and 1280x800) is recommended to better serve users	Use 1024 or 1280 for resolutions of screen	Chiuchi et al. [2011]		Presentation	
466	Using pages having a width of at least 120 pixels for mobile devices	Use pages which have a width of at least 120 pixels for mobile devices	Chiuchi et al. [2011]		Presentation	
467	It should not to use frames in a website	Do not use frames in a website	Chiuchi et al. [2011]		Presentation	
468	A search engine on the website should be created for users	Provide search engine for the site	Chiuchi et al. [2011]		Content	
469	Control bar: Make the control bar prominent so that the user can easily see, understand, and use it to control the speed at which he or she is going through the program	Make the control bar prominent so that the user can see, understand and use it easily	Xie et al. [2011]		Presentation	Old people
470	Symbols: Use both text and symbol, or text alone, to indicate the function of a clickable element; do not use a symbol alone if its meaning is not self-explanatory to users of various literacy levels	Use both text and symbol, or text alone, to indicate the function of a clickable element Do not use a symbol alone if its meaning is not self-explanatory to users of various literacy levels	Xie et al. [2011]		Presentation	Old people

No	Original guidelines	Modified guidelines	Source	Related to other guidelines	C / N / P	Domain
471	New Windows: Avoid new windows that block the previous window; if this is unavailable, at least design a built-in link (e.g., a clickable button) and place it in a prominent position in the new window to indicate how to close the new window to return to the previous window or navigate to the next step	Avoid new windows that block the previous window Create a built-in link in a prominent position in the new window to indicate how to close the new window to return to the previous window or navigate to the next step	Xie et al. [2011]		Content Presentation	Old people
472	Subtitles: Add a sign to every page (e.g., a button, bar, flag) to indicate whether the subtitles are currently on or off (and how to switch between on and off)	Add a sign to every page to indicate whether the subtitles are currently on or off	Xie et al. [2011]		Presentation	Old people
473	Enlarge photos: Add clear instructions to indicate that a larger version of the photo is available, and how to do so	Add clear instructions to indicate that a larger version of the photo is available, and how to do so	Xie et al. [2011]		Presentation	Old people
474	Site Map: Use an alternative phrase to better convey what this feature means	Use an alternative phrase to better convey what this feature means	Xie et al. [2011]		Navigation	Old people
475	Prior knowledge: Avoid having interactive features that require extensive prior knowledge and skills about computers, medicine, and numeracy		Xie et al. [2011]		Content Presentation	Old people

Appendix C

Source code of tool

C.1 Add guidelines

Here is the source code of function Add guidelines. A new guideline is added in the list of pending guidelines and after verified to accept or not.

```
btnAdd.addActionListener(new ActionListener() {
          public void actionPerformed(ActionEvent arg0) {
               try{
                  Connection conn = null;
                  Statement stmt = null;
                  Statement stmt2 = null;
                  //Register JDBC driver
                  Class.forName(JDBC_DRIVER);
                  //Open a connection
                  conn = DriverManager.getConnection(DB_URL,USER,PASS);
                  //Execute a query
                  stmt = conn.createStatement();
                  stmt2 = conn.createStatement();
                  String sql3;
                  String newGuidelineAdded;
                  String modalVerbPhrase;
                  String adjectivePhrase;
                  String adverbPhrase;
                  String comp1Phrase;
                  String comp2Phrase = "";
                  String verbSelected = "";

                  //Choose verb
                  if ((comboBox_2.getSelectedItem() != null) &&
                      (comboBox_2.getSelectedItem() != "Choose a verb")){
                     verbSelected = comboBox_2.getSelectedItem().toString();
                  }
                  else
                     if ((textField_4.getText() != null)){
```

```
            //use new verb from text field
            verbSelected = textField_4.getText();
            //add new verb to database
            sql3 = "INSERT INTO verb VALUE (\"" + verbSelected +
                "\")";
            stmt.executeUpdate(sql3);
         }

      if ((comboBox_1.getSelectedItem() != null) &&
          (comboBox_3.getSelectedItem() != null)){

         //modal verb
         if (comboBox_1.getSelectedItem().equals("Choose a modal
             verb")){
            if (chckbxNot.isSelected()) modalVerbPhrase = "Do not";
            else modalVerbPhrase = "";
         }
         else
            if (chckbxNot.isSelected()) modalVerbPhrase =
                comboBox_1.getSelectedItem().toString() + " not";
            else modalVerbPhrase =
                comboBox_1.getSelectedItem().toString();
         //adjective
         if (comboBox_3.getSelectedItem().toString().equals("Choose
             an adjective")) adjectivePhrase = "";
         else adjectivePhrase =
             comboBox_3.getSelectedItem().toString() + " ";
         //adverb
         if (comboBox_5.getSelectedItem().toString().equals("Choose
             an adverb")) adverbPhrase = "";
         else adverbPhrase =
             comboBox_5.getSelectedItem().toString() + " ";
         if (comboBox_4.getSelectedItem().toString().equals("Choose
             a complement")) comp1Phrase = "";
         else
            if (textField_3.getText() != null)
               comp1Phrase = comboBox_4.getSelectedItem().toString()
                   + " " + textField_3.getText();
            else  comp1Phrase =
                comboBox_4.getSelectedItem().toString() + " ";

         }
         */
         //Sentence case the verb if have not modal verb
         if (modalVerbPhrase.equals("")){
            char c = verbSelected.charAt(0);
            char cUpper = Character.toUpperCase(c);
            verbSelected = cUpper + verbSelected.substring(1);
            //verbSelected = verbSelected. ;
         }
         //if not, add a space
```

```
                    else modalVerbPhrase += " ";

                    newGuidelineAdded = modalVerbPhrase + verbSelected + " " +
                        adjectivePhrase + textField.getText() + " " +
                        adverbPhrase + comp1Phrase + comp2Phrase;
                    sql3 = "INSERT INTO guideline VALUE (0, \"" +
                        newGuidelineAdded + "\", \"" +
                        comboBox_7.getSelectedItem() + "\", \"" +
                        comboBox_8.getSelectedItem() + "\")";

                    stmt.executeUpdate(sql3);

                    //add id to pending guideline list
                    String sql3b = "SELECT * FROM guideline WHERE content =\""
                        + newGuidelineAdded + "\"";
                    String sql3c = "";
                    ResultSet rs1 = stmt.executeQuery(sql3b);
                    while(rs1.next()){
                          //Retrieve by column name
                          int pending_id = rs1.getInt("id");
                          String pending_id_String = String.valueOf(pending_id);

                          sql3c = "INSERT INTO pending VALUE (" +
                              pending_id_String + ")";
                              stmt2.executeUpdate(sql3c);
                       }
                 }

                    //Clean-up environment
                    stmt.close();
                    stmt2.close();
                    conn.close();
                 }catch(SQLException se){
                    //Handle errors for JDBC
                    se.printStackTrace();
                 }catch(Exception e){
                    //Handle errors for Class.forName
                    e.printStackTrace();
                 }

           }
        });
```

C.2 Verify guidelines

In the function Verify guidelines, a guideline in the pending list is verified whether itself or similar one is already in the list. Here is the source code of function Ver Finding similar

guidelines. We use the Levenshtein distance between two strings to find the similarity.

```
btnFindSimilarGuidelines.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
      try{

            Connection conn = null;
            Statement stmt = null;
            //Register JDBC driver
            Class.forName(JDBC_DRIVER);
            //Open a connection
            conn = DriverManager.getConnection(DB_URL,USER,PASS);
            //Execute a query

          //View all guidelines
          stmt = conn.createStatement();
          String sql;
          int levenshteinmetrics = 1000;   //great number
          int tempLevenshtein = 0;
          double tempCosine = 0;
          double seuilCosine = 0.5;

          sql = "SELECT * FROM guideline";
          ResultSet rs = stmt.executeQuery(sql);
          String inputGuideline = "";
          String outputGuideline = "";

          //inputGuideline = "testing";
          inputGuideline = comboBox_10.getSelectedItem().toString();
          String nextGuideline = "";
          textArea.setText(null);

          while(rs.next()){
            //Retrieve by column name
            nextGuideline = rs.getString("Content");
            //tempLevenshtein = LevenshteinDistance(inputGuideline,
                nextGuideline);
            tempCosine = cosineSimilarity(inputGuideline, nextGuideline);

            //Show results
            if (tempCosine > seuilCosine){
               textArea.append(nextGuideline + " distance: " +
                   String.valueOf(tempCosine) + "\n");
               //levenshteinmetrics = tempLevenshtein;
               //outputGuideline = nextGuideline;
               }
            }

          //Clean-up environment
            rs.close();
            stmt.close();
```

```
            conn.close();

        } catch (SQLException e1) {
           // TODO Auto-generated catch block
           e1.printStackTrace();
        } catch (ClassNotFoundException e1) {
           // TODO Auto-generated catch block
           e1.printStackTrace();
        }finally{
           }//end try


     }
  });
```

C.3 Request guidelines

In this function, we can query the guideline base using criterium, the domain (general, children, etc.), the view or keyword. Here is the source code of this function.

```
  btnRequest.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
              try{
                 Connection conn = null;
                 Statement stmt = null;
                 //Register JDBC driver
                 Class.forName(JDBC_DRIVER);
                 //Open a connection
                 conn = DriverManager.getConnection(DB_URL,USER,PASS);
                 //Execute a query

                    stmt = conn.createStatement();
                    String sql = "";
                    String selectedItemComboBoxView = "";
                    String selectedTextSearchBox = "";
                    Boolean executed = false;

                    selectedItemComboBoxView =
                        comboBox_9.getSelectedItem().toString();

                    selectedTextSearchBox = textField_1.getText();

                    if (selectedItemComboBoxView.equals("Choose a view")){
                      sql = "SELECT * FROM guideline WHERE content LIKE '%" +
                          selectedTextSearchBox + "%' ";
                      executed = true;
                    }
```

```
                if (selectedTextSearchBox.equals("")){
                  sql = "SELECT * FROM guideline WHERE view LIKE '%" +
                      selectedItemComboBoxView + "%' ";
                  executed = true;
                }

                if (!selectedItemComboBoxView.equals("Choose a view") &&
                    !selectedTextSearchBox.equals("")){
                  sql = "SELECT * FROM guideline WHERE content LIKE '%" +
                      selectedTextSearchBox + "%' and view LIKE '%" +
                      selectedItemComboBoxView + "%' ";
                  executed = true;
                }

                ResultSet rs = null;

                if (executed){
                  rs = stmt.executeQuery(sql);
                  //Extract data from result set
                  textArea_1.setText(null);
                  while(rs.next()){
                     //Retrieve by column name
                     String content = rs.getString("content");
                     textArea_1.append(content + "\n");
                     }
                }
                //Clean-up environment
                rs.close();
                stmt.close();
                conn.close();
             }catch(SQLException se){
                //Handle errors for JDBC
                se.printStackTrace();
             }catch(Exception e1){
                //Handle errors for Class.forName
                e1.printStackTrace();
             }//end try
          }
       });
```

Tuan Anh DO

A QUALITY-CENTERED APPROACH FOR WEB APPLICATION ENGINEERING

Résumé :
Avec le développement d'Internet, les applications web sont de plus en plus nombreuses et importantes. De nombreux standards de qualité, des modèles de qualité, des méthodes d'ingénierie web ont été proposés, mais la qualité des applications web n'est pas toujours au niveau souhaité.
Dans cette thèse, nous proposons une approche pour tenter de résoudre ce problème. Elle comporte trois phases itératives: définition, mesure et amélioration de la qualité des applications web. Dans la première phase, nous proposons une définition plus complète et plus riche de la qualité des applications web. La qualité d'une application web n'est pas uniquement perçue comme la qualité d'un logiciel, mais également comme la qualité des informations qu'elle met à disposition. Enfin, elle comprend des éléments de qualité spécifiques à ces applications qui contribuent notamment au succès et à la réputation de l'organisation. Dans la seconde phase, nous construisons une taxonomie de métriques pour mesurer la qualité des applications web. Cette taxonomie est fondée sur le standard ISO25010. Dans la troisième phase, nous avons collecté et adapté les « guidelines » de la littérature pour les mettre à la disposition des concepteurs-développeurs d'applications web. A cet effet, nous avons proposé un méta-modèle de guideline, une grammaire et un outil pour les gérer.

Mots clés :
Application web, qualité des applications web, amélioration continue, métrique de qualité, guideline.

Abstract :
With the development of Internet, web applications are more and more important. Many quality standards, models, web engineering methods were proposed but the quality of many web applications is not yetat the desired level.
In this thesis, we propose an approach contributing to this area. Our approachcontains three iterative phases for, respectively, defining, measuring, and improving quality of web applications. In the first phase, we define a more complete, richer definition of quality of web applications. The latter is not only seen as quality of software, but also as quality of information, and quality of specific web features. In the second phase we build a taxonomy of metrics for measuring quality of web applications. This taxonomy is based on the ISO25010 quality model. In the third phase we collect and adapt guidelines for improving quality of web applications and providing web applications developers with useful advice. Our contribution consists of a guideline meta-model, a grammar, and a tool for managing guidelines.

Keywords :
Web application, web application quality, continuous improvement, quality metrics, guidelines.