HAL
open science

DRARS, A Dynamic Risk-Aware Recommender System

Djallel Bouneffouf

To cite this version:

Djallel Bouneffouf. DRARS, A Dynamic Risk-Aware Recommender System. Computation and Language [cs.CL]. Institut National des Télécommunications, 2013. English. NNT : . tel-01026136

HAL Id: tel-01026136

https://theses.hal.science/tel-01026136v1

Submitted on 20 Jul 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

DOCTORAT EN CO-ACCREDITATION
TELECOM SUDPARIS ET L'UNIVERSITE EVRY VAL D'ESSONNE

Spécialité : Informatique

Ecole doctorale : Sciences et Ingénierie

Présentée par

Djallel Bouneffouf

Pour obtenir le grade de
DOCTEUR DE TELECOM SUDPARIS

DRARS, A Dynamic Risk-Aware Recommender System

Soutenue le 19/12/2013

Devant le jury composé de :

Directeur de thèse

Amel Bouzeghoub	Professeur à Télécom SudParis - SAMOVAR

Rapporteurs

Mohand Boughanem	Professeur à Université Paul Sabatier
Eric Gaussier	Professeur à Université Joseph Fourier (Grenoble I)

Examinateurs

Florence d'Alché-Buc	Professeur à Université d'Evry Val d'Essonne
Sylvie Calabretto	Professeur à INSA LYON, LIRIS
Alda Gançarski	Maître de conférences à Télécom SudParis, SAMOVAR (Encadrant)
Fabrice Jarry	Président & Fondateur de Nomalys

Thèse n° 2013TELE0031

SUMMARY

The vast amount of information generated and maintained everyday by information systems and their users leads to the increasingly important concern of overload information. In this context, traditional recommender systems provide relevant information to the users. Nevertheless, with the recent dissemination of mobile devices (smartphones and tablets), there is a gradual user migration to the use of pervasive computing environments.

The problem with the traditional recommendation approaches is that they do not utilize all available information for producing recommendations. More contextual parameters could be used in the recommendation process to result in more accurate recommendations. Context-Aware Recommender Systems (CARS) combine characteristics from context-aware systems and recommender systems in order to provide personalized recommendations to users in ubiquitous environments.

In this perspective where everything about the user is dynamic, his/her content and his/her environment, two main issues have to be addressed: i) How to consider content dynamicity? and ii) How to avoid disturbing the user in risky situations?. In response to these problems, we have developed a dynamic risk sensitive recommendation system called DRARS (Dynamic Risk-Aware Recommender System), which model the context-aware recommendation as a bandit problem. This system combines a content-based technique and a contextual bandit algorithm.

We have shown that DRARS improves the Upper Confidence Bound (UCB) policy, the currently available best algorithm, by calculating the most optimal exploration value to maintain a trade-off between exploration and exploitation based on the risk

level of the current user's situation. We conducted experiments in an industrial context with real data and real users and we have shown that taking into account the risk level of users' situations significantly increased the performance of the recommender systems.

RÉSUMÉ

.

L'immense quantité d'information générée et gérée au quotidien par les systèmes d'information et leurs utilisateurs conduit inéluctablement à la problématique de surcharge d'information. Dans ce contexte, les systèmes de recommandation traditionnels fournissent des informations pertinentes aux utilisateurs. Néanmoins, avec la propagation récente des dispositifs mobiles (Smartphones et tablettes), nous constatons une migration progressive des utilisateurs vers la manipulation d'environnements pervasifs. Le problème avec les approches traditionnelles de recommandation est qu'elles n'utilisent pas toute l'information disponible pour produire des recommandations. Davantage d'informations contextuelles pourraient être utilisées dans le processus de recommandation pour aboutir à des recommandations plus précises. Les systèmes de recommandations sensibles au contexte (CARS) combinent les caractéristiques des systèmes sensibles au contexte et des systèmes de recommandation afin de fournir des informations personnalisées aux utilisateurs dans des environnements ubiquitaires. Dans cette perspective oú tout ce qui concerne l'utilisateur est dynamique, les contenus qu'il manipule et son environnement, deux questions principales doivent être adressées : i) Comment prendre en compte la dynamicité des contenus de l'utilisateur ? et ii) Comment éviter d'être intrusif en particulier dans des situations critiques ?. En réponse à ces questions, nous avons développé un système de recommandation dynamique et sensible au risque appelé DRARS (Dynamic Risk-Aware Recommender System), qui modélise la recommandation sensible au contexte comme un problème de bandit. Ce système combine une technique de filtrage basée sur le contenu et un algorithme de bandit contextuel. Nous avons montré que DRARS améliore la stratégie de l'algorithme UCB (Upper Confidence Bound), le meilleur algorithme actuellement

disponible, en calculant la valeur d'exploration la plus optimale pour maintenir un compromis entre exploration et exploitation basé sur le niveau de risque de la situation courante de l'utilisateur. Nous avons mené des expériences dans un contexte industriel avec des données réelles et des utilisateurs réels et nous avons montré que la prise en compte du niveau de risque de la situation de l'utilisateur augmentait significativement la performance du système de recommandation.

ACKNOWLEDGEMENTS

I first thank Nomalys for supporting this work and providing me with a rich industrial environment to experiment. More personally, I thank the many people who contributed to make this PhD a great experience for me.

I wish to express my gratitude to my advisors at Telecom SudParis. I would really like to thank Amel Bouzeghoub and Alda Gancarski for their great guidance throughout these years. Their support and advices on research directions were precious to go forward with my work.

I am also grateful to Fabrice Jarry for his kind management. It was a great opportunity for me to benefit from the management expertise he delivers with passion.

At Nomalys, I had the chance to work with various people that all contributed to make work days instructive and enjoyable. I am also thankful to the whole team of the SAMOVAR laboratory, in Telecom SudParis, for its good atmoshpere.

I am particularly thankful to Mohand Boughanem and Eric Gaussier for accepting to review my PhD thesis. I would also like to thank Sylvie Calabretto and Florence d'Alch-Buc who both accepted to be jury examinators.

I am also grateful to Farah Benamara, she has given me the first bases of research when I had my internship in artificial intelligence, six years ago (thank you madam).

I am lucky to have met amazing friends and they have always been a great help.

The permanent support of my family was extremely precious to me. Thank you to my parents, to my brothers and my wife to whom I owe so much.

Contents

SUMMARY **1**

ACKNOWLEDGEMENTS . . . . . **iii**

LIST OF TABLES . . . . . **viii**

LIST OF FIGURES . . . . . **ix**

I INTRODUCTION **1**

1.1 Research issues 3

1.2 Industrial Context 4

1.3 Major Contributions 5

1.4 Thesis Organization 9

II RECOMMENDER SYSTEMS **10**

2.1 Introduction 10

2.2 The User's Profile 13

2.2.1 Defining the User's Profile 13

2.2.2 Modelling the User's Profile 14

2.2.3 Acquiring the User's Profile 17

2.2.4 Discussion 18

2.3 Recommendation Methods 20

2.3.1 Collaborative Filtering 20

2.3.2 Content-Based Filtering 22

2.3.3 Hybrid Approach 23

2.3.4 Machine Learning Approach 24

2.3.5 Discussion 29

2.4 The Dynamicity of the User's Content 29

2.4.1 CBF for User's Content Dynamicity 30

2.4.2 Exploration-Exploitation Trade-Off for Content Dynamicity . 31

2.4.3 Discussion 32

2.5 Context-Aware Recommender Systems (CARS) 33
2.5.1 Context Definition 33
2.5.2 Context Acquisition 34
2.5.3 Context Modelling 34
2.5.4 From Context Awareness to Situation Awareness 45
2.5.5 Recommendation Methods 48
2.5.6 Discussion 56
2.6 Risk Aware Decision 59
2.6.1 Risk Definition 59
2.6.2 Measuring the Risk Value 60
2.6.3 Discussion 62
2.7 Synthesis 63
2.8 Conclusion 64
III MULTI-ARMED-BANDIT PROBLEM 66
3.1 Introduction 66
3.2 Definition of Multi-Armed Bandit Problem 66
3.3 Multi-Armed Bandit Variations 67
3.4 Bandit Algorithms Overview 68
3.4.1 ϵ-Greedy 69
3.4.2 SoftMax 69
3.4.3 Exp3 70
3.4.4 Pursuit 70
3.4.5 Upper Confidence Bound 70
3.4.6 Gittins 71
3.4.7 Discussion 71
3.5 Bandit Algorithm for Recommender Systems 72
3.5.1 EG-greedy 72
3.5.2 LINUCB 73

3.5.3 Discussion 75
3.6 Bandit Algorithm for Risk-Aware Decision 75
3.7 Analysis 76
3.8 Conclusion 76
IV THE DRARS SYSTEM 77
4.1 Introduction 77
4.2 Issues and Motivations 77
4.3 Context, Situation and Profile Modelling 78
4.3.1 The User's Model 79
4.3.2 The User's Profile 79
4.3.3 The User's Context 80
4.3.4 The User's Situation 83
4.4 The Architecture of DRARS 86
4.5 The Functional Description of DRARS 87
4.5.1 Context Acquisition 88
4.5.2 Extracting Information from the Information System 89
4.5.3 Abstracting the Contextual Information 92
4.5.4 Retrieving the Relevant Situation 93
4.5.5 Computing the Risk Level of the Situation 94
4.5.6 Following the dynamicity of the User's Content 98
4.6 Conclusion 102
V IMPLEMENTATION OF DRARS 103
5.1 Host Application of DRARS 103
5.2 Application Scenario 104
5.3 Integrating DRARS in Nomalys 105
5.3.1 Smartphone 105
5.3.2 The Nomalys' Server 106
5.4 Tools Used to Implement DRARS 110

5.4.1 Programming the Server Application 111
5.4.2 Ontologies Construction 111
5.4.3 Database Management 111
5.4.4 Project Management 111
5.5 Conclusion 112
VI EVALUATION OF DRARS 113
6.1 Introduction 113
6.2 Evaluation Framework 113
6.3 Descriptive Analysis 114
6.4 Parametrizing 116
6.4.1 Genetic Algorithm 116
6.4.2 Testing the GA 121
6.5 Off-line Experimental Results 122
6.5.1 Risk Level of the Situations 123
6.5.2 Size of Data 124
6.6 On-line Evaluation of DRARS 125
6.6.1 New Document Exploration 125
6.6.2 R-UCB, VDBE-UCB and EG-UCB Comparison 127
6.7 The Algorithm Complexity 127
6.8 Conclusion 128
VII CONCLUSION AND PERSPECTIVES 129
7.1 Perspectives 130
REFERENCES 132

List of Tables

1 Representing the user's profile 18
2 Acquisition of the user's profile 19
3 The recommendation approaches 29
4 Approaches for following the user's content dynamicity 32
5 Comparing context representation models 56
6 Comparing CARS 57
7 The models of measuring the risk 62
8 Comparing the state of the art works with our requirements 65
9 Comparing the bandit algorithm with our requirements 76
10 The Company relation within the client database 107
11 The Opportunity relation within the client database 107
12 The Contract relation within the client database 107
13 The Action relation within the client database 107
14 Example of diary situations and navigation entries 109
15 Diary situation entries 113
16 Diary situation abstraction 114
17 Diary navigation entries 114
18 Crossover example 120
19 Mutation example 120

List of Figures

1 Case-based Reasoning 25

2 Attribute-value model proposed by [118] 36

3 Object-oriented model proposed by [95] 37

4 A) The XReAl query model; B) The XReAl contextual information document 39

5 A) part of the physical context, B) part of the informational context . 41

6 MUSE ontology 42

7 Dimensions of the situation model [14] 43

8 Ontology model of context proposed by [71] 44

9 Context-aware neural network 55

10 Classification of situations proposed by [122] 61

11 Location ontology 81

12 Time ontology 82

13 Social ontology 83

14 The dimensions aggregated in the situation model 84

15 The architecture of DRARS 86

16 The sequence diagram of the different modules of DRARS 88

17 The transformation of a relational database on a hierarchical database 90

18 Example of XML document after the transformation phase 92

19 Risk modelling 95

20 An example of user's agenda in the user's 104

21 Results list of searching a company using the Nomalys application . . 104

22 The result folder presented as a mind map to the user 105

23 XML document transferred from the mobile to the server 106

24 The system's answer without recommendation 108

25 The system's answer with recommendation 110

26 Risk level of the situations in the 5 clusters 115

27 The risk level of the situations regarding their properties 115

28 Number of populations by generation 121

29 Average CTR for exploration-exploitation algorithms 123

30 Average CTR of exploration-exploitation algorithms in situations with different risk levels 124

31 Average CTR for different data size 125

32 Average number of new documents used in three groups without and with recommendation 126

33 Average number of recommended documents have used multiple times for each navigation session 127

34 Time in ms to recommend 128

Chapter I

INTRODUCTION

The exponential expansion of mobile phones and the number of mobile applications are associated with a proliferation of information whose volume continues to grow. Given this profusion of documents, the users have more and more difficulty to find out relevant information that matches their needs. In this setting, Recommender Systems (RS) have been developed to anticipate the needs of the user by providing recommendations of documents deemed relevant to his interest.

Concretely, RS include all the systems capable of providing recommendations tailored to the users' need, to help them gain access to useful or interest resources in a space of data. In this domain, the user does not need to formulate a query. The query is in fact implied and can be translated as: "Which resources correspond to my interests?" To answer this question, the RS needs to establish a correspondence between the information sought (expressed through the user's profile) and all the documents in the collection.

Modelling the acquisition, the representation and the organization of the user's profile and his context is the most difficult task of a RS whose performance depends mainly on this process. In addition, the profile must be constantly kept up to date so that it not only reflects the interests of the user but also addresses the common problem of the user's content dynamicity.

The acquisition of explicit profiles shows limited performance due to several factors, among which the subjectivity of the user, who initially may not know exactly what he wants, and the dynamicity of the context that may influence the user's profile.

The main strategy to recommend documents to the user is through Content-Based

Filtering (CBF), which identifies similar resources to those assessed by the user according to his content. In particular, only similar resources to resources for which a positive assessment could be obtained may be recommended, which greatly limits the diversity of recommendations [3]. In addition, in most of these systems, the assessments are considered static. However, an appreciation from the user can highly depend on her context. Such a context can be geographic, social, cultural, etc. [3, 93]. For example, a user could love an ice cream in a sunny days, but not by snowy days; another user might love listening to heavy metal music, but not in the office. The context is thus often crucial to the assessment of the user.

Recently, a new generation of RS called Context-Aware Recommender System (CARS) has emerged to take into acount the user's environment, which is highly dynamic in nature. These systems are based on the computational behaviour of the user to model her interests regarding the surrounding environment like location, time and near people. However, with the proliferation of smart phone particularly in the professional field, CARS is nowadays more and more confronted to the risk of upsetting the user. In such a setting, the research problem that this thesis adresses is: How to provide the *right* information, at the *right* time, in the *right* place, in the *right* way to the *right* person? The *right* information can be inferred from interests or derived from previous actions; The *right* time addresses intrusiveness of information delivery. It requires balancing the costs of intrusive interruptions against the loss of context sensitivity of deferred recommendations. The *right* place takes location-based information into account; The *right* way differentiates between multi-model representations; e.g. text, image, etc. espacially for users who may suffer from some disability; The *right* person requires user modeling.

1.1 Research issues

Now, when thinking of the research problem cited above, several research questions arise:

1-User modelling: To make an adapted recommendation to a user, the system needs to collect and analyse the data assessments of this user. The man challenge in this sense is how to construct an adequate structure (model) to stock these informations?

2-Context sensitivity: An important aspect of context-aware systems and CARS in particular, is how the information representing the context is obtained and represented. A particular challenge often consists to infer higher-level goals from low-level observed operations.

3-Management of the cold start and of new content The user's content is the document collection from where the RS has to choose the interesting ones to recommend. These documents may change with the time, and the system has to be continuously adapted to this dynamicity using the user's context information to provide the relevant recommendation. When a new document is introduced into the system, it cannot be considered to perform a content-based recommendation because the user's feedback for this document is not yet available. This problem is known as the "cold start" [102]. RS must cope with this problem in order to follow the dynamicity of the user's content.

4-Non-intrusive recommendation: The system must respect some criteria to not upset the user [61]. If the system gives a random information to the user in some situation, as for example in the work office, the user may have a very bad reaction. However, when the user is at home, this random information can lead to a serendipity, which is explained by the fact that the user is more open to discover a new information at home than at office.

To build a RS performing accurate recommendations, these issues must be carefully

addressed. Indeed, the accuracy of recommendations provided by a RS depends on how the system models the user's profile and context, how the system follows the dynamicity of the user's content and how the system manages the user's situation risk level during the recommendation process.

1.2 Industrial Context

This thesis is a part of the Nomalys project in collaboration with the Nomalys SA, an innovative start-up that is under the responsibility of Mr. Fabrice Jarry.

Nomalys develops a platform for linking the information systems of companies and smartphones for nomad commercial users. These users can access to their information system using their smartphone.

The particularity of the commercial users is their mobility. Their context is always changing (for example, they can be at the office, at the restaurant, etc.). The documents that they also use are in constant change (for example, to prospect a new client, the commercial needs a document related to this client). Due to this dynamic environment, the problem of latency has more impact on this type of systems (that incorporate regularly new documents and with dynamic user's context).

Indeed, documents and users of the application are very numerous and varied (thousands of users and tens of thousands of documents). However, the challenge is to establish recommendation tools able to provide users with relevant information adapted to their needs and contexts.

Academically, the thesis is supervised inside the SAMOVAR research laboratory, a French CNRS (Centre National de Recherche Scientifique) unity located at the TELECOM Sudparis engineer school. SAMOVAR has extensive experience in modelling and building ubiquitous (pervasive) applications.

1.3 Major Contributions

Providing the objectives that we set for this thesis, our contributions are involved in the proposed system named "Dynamic Risk Aware Recommender System"(DRARS).

Modelling the User and the Situation: We have modelled the user with two facets: the profile and the context (Chapter 4) and we propose to define and model the situation as an instanciation of the user context. The user's profile is structured as multidimensional features, and the user's context is modelled with ontologies. As proposed in [14], each situation is linked to a user's interest and stored in a case base.

Following the dynamicity of the user's content: We propose an algorithm, that we called R-UCB, which is a combination of CBF and the Upper Confidence Bound (UCB) algorithm to follow the dynamicity of the user's content (Chapter 4). The UCB algorithm constructs a reward estimate for each document previously seen. The reward is computed as the mean of the observed number of clicks added to an additional term that is inversely related to the number of times the document has been recommended. The document with the highest reward estimate is selected for recommendation. The reward estimates in this way encourages exploration of documents that have been infrequently selected. After selecting the document, the proposed algorithm uses CBF to identify the similar resources to those selected by the UCB algorithm. This approach allows to follow the user's content dynamicity by proposing documents which are sometimes the most probable to be clicked and, other times, documents randomly chosen to improve the knowledge of the system.

Considering the situation risk level and intrusiveness of information delivery: We have considered the situation risk level when managing the exploration-exploitation trade-off in the RS (Chapter 4). This strategy achieves high exploration when the current user's situation is not risky and achieves high exploitation in the inverse case.

We have also aggregated three approaches for computing the risk. The first approach computes the risk using concepts from the application domain, permitting to get the risk directly from the risk of each of those concepts. The second approach computes the risk using the similarity between the current situation and situations stored in the system, assuming that similar situations have the same risk level. The third approach computes the risk using the variance of the reward, assuming that risky situations get very few user's clicks.

Evaluating the proposed approaches: For the validation of the proposed approaches in this thesis (Chapter 6), we evaluate the different models through off-line experiments, recording the user's navigation activities in a first step, and test them in a second step using an iterative process. We have also done on-line experiments, where we evaluate the algorithms regarding the number of users' clicks. The evaluation yields to the conclusion that considering the exploration-exploitation trade-off in the recommendation permits to follow the dynamicity of the user's content. We also conclude that considering the risk level of the situation on the exploration-exploitation strategy significantly increases the performance of the RS.

Industrial Contributions: DRARS is currently integrated in Nomalys Application to make it possible to display real time any type of data (CRM, ERP,) to nomade users (Sales, Purchasing, HR, Marketing,) according to their contextual relevance. Two patent applications are in the process of being finalized after a proof of concept achieved through DRARS.

Published Work: Large portions of contributions made in this work have been published in international conferences:

[25]: Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gancarski. Risk-Aware Recommender Systems. In Springer, editor, ICONIP '13 : The 19th International Conference on Neural Information Processing

[30]: Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gancarski. Contextual Bandits for Context-Based Information Retrieval. In Springer, editor, ICONIP '13 : The 19th International Conference on Neural Information Processing

[29]: Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gancarski. Hybrid-epsilon-greedy for mobile context-aware recommender system. In Springer, editor, PAKDD '12 : The 16th Pacific-Asia Conference Advances in Knowledge Discovery and Data Mining, volume 7301/2012, pages 468 479, Heidelberg ;Dordrecht ;London [etc.], 2012. Collection : Lecture Notes in Computer Science.

[28]: Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gancarski. Following the users interests in mobile context-aware recommender systems : the hybrid-epsilon-greedy algorithm. In IEEE Computer Society, editor, HWISE '12: The Eighth International Workshop on Heterogeneous Wireless Networks, pages 657 662, 2012.

[26]: Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gancarski. A contextual-bandit algorithm for mobile context-aware recommender system. In Springer, editor, ICONIP '12 : The 19th International Conference on Neural Information Processing, volume 7665, pages 324 331, Heidelberg ;Dordrecht ;London, 2012. Collection : Lecture Notes in Computer Science.

[27]: Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gancarski. Exploration/exploitation tradeoff in mobile context-aware recommender systems. In Springer, editor, AI '12 : The Twenty-Fifth Australasian Joint Conferences on Artificial Intelligence, volume 7691, pages 591 601, Heidelberg ;Dordrecht ;London, 2012. Collection : Lecture Notes in Computer Science.

[31]: Djallel Bouneffouf, Amel Bouzeghoub, Alda Lopes Ganarski: Considering the

High Level Critical Situations in Context-Aware Recommender Systems. In CEUR-WS.org, editor, IMMoA12 : 2nd International Workshop on Information Management for Mobile Applications, volume 908, pages 26 32, 2012. In conjunction with VLDB '12 : 38th International Conference on Very Large Databases.

[15]: Djallel Bouneffouf, Applying Machine Learning Techniques to Improve User Acceptance on Ubiquitous Environment. CAiSE (Doctoral Consortium) 2011: 3-14

[21]: Bouneffouf Djallel, Situation-Aware Approach to Improve Context-based Recommender System, arXiv preprint arXiv:1303.0481, 2013

[18]: Bouneffouf Djallel, Improving adaptation of ubiquitous recommander systems by using reinforcement learning and collaborative filtering,arXiv preprint arXiv:1303.2308, 2013

[16]: Bouneffouf Djallel, Evolution of the user's content: An Overview of the state of the art, arXiv preprint arXiv:1305.1787, 2013

[17]: Bouneffouf Djallel, The Impact of Situation Clustering in Contextual-Bandit Algorithm for Context-Aware Recommender Systems, arXiv preprint arXiv:1304.3845, 2013

[23]: Bouneffouf Djallel, Etude des dimensions spéecifiques du contexte dans un syst\eme de filtrage d'informations, arXiv preprint arXiv:1405.6287, 2014

[19]: Bouneffouf Djallel, L'apprentissage automatique, une étape importante dans l'adaptation des systèmes d'information à l'utilisateur, Inforsid, pages 427–428, 2013

[22]: Bouneffouf Djallel, Towards User Profile Modelling in Recommender System, arXiv preprint arXiv:1305.1114, 2013

[20]: Bouneffouf Djallel, Mobile Recommender Systems Methods: An Overview, arXiv preprint arXiv:1305.1745, 2013

[24]: Bouneffouf, Djallel,Recommandation mobile, sensible au contexte de contenus evolutifs: Contextuel-E-Greedy, arXiv preprint arXiv:1402.1986 ,2014

1.4 Thesis Organization

Let us briefly present the organization of this manuscript.

We first review related work in Chapter 2. The review is structured so as to reflect the four main fields of research related to our work: (1) RS and the most common techniques developed in this area. (2) Following the dynamicity of the user's content; (3) Context-aware RS and (4) Risk-aware decisions.

Next, we introduce in Chapter 3 the exploration-exploitation problem and the different algorithms that try to solve it, selecting the most interesting algorithms in this area related to our needs.

Chapter 4 describes the architecture of the proposed framework, from computing context information to recommending documents to the user.

Besides, in Chapter 5 we further discuss the use of our framework to enable recommendation with a realistic scenario. We give an example of running our system in order to summarize the main specifications of this thesis' project. Moreover, we show how the different modules of the developed system can be implemented, explaining the use of the various tools to achieve our goal.

Chapter 6 describes the experimentations done to evaluate our approach. We have first made the parametrization of our algorithm. Then, we have evaluated it in an off-line dataset derived from the journal of the Nomalys company. In addition, we have made an on-line evaluation of the developed RS using the Nomalys users'.

Finally, we conclude our work and contributions in Chapter 7. The discussion includes suggestions for future work related to our RS and its applications.

Chapter II

RECOMMENDER SYSTEMS

2.1 Introduction

The information that mobiles can access becomes very wide nowadays, and the user is faced to a dilemma: there is an unlimited pool of information available to him but he is unable to find the exact information he is looking for. This is why the current research aims to design *Recommender Systems (RS)* able to continually send information that matches the user's interests in order to reduce his navigation time. RS can be defined in several ways. The definition we use in this thesis is the general definition of [38]: "The RS is a tool which provides personalized information to guide the user towards interesting or useful resources within a space of data".

In practice, most RS consist in applications that provide lists of resources to users. Such resources may correspond to different types of data such as films, music, books, news, Web pages (Yahoo, Google), etc..

When the RS emerged in the mid-1990s, they used the notion of ratings as a way to capture user interests for different items. For example, in the case of a RS for restaurants, when Paul assigns a rating of 3 (out of 5) for the restaurant "Woodpecker, i.e., the RS sets Restaurant(Paul, Woodpecker)=3. The recommendation process starts with the specification of the initial set of ratings that is specified by the users. Once these initial ratings are provided, the RS tries to estimate the rating function $F : User \times Item \rightarrow Rating$, for the pairs (user, item) that have not been assessed yet by the users. The *Rating* is generally a real number within a certain range, and $User$ and $Item$ are the domains of users and items respectively. Once the function F is estimated for the whole $User \times Item$ space, the RS can recommend the highest-rated

item for each user.

In this formulation, the recommendation problem is reduced to the problem of estimating ratings for the items that have not been seen by a user. This estimation is usually based on the ratings given by this user to other items, ratings given to this item by other users, and other information as well (e.g., user and item characteristics). This information is generally stored in a structure called the user's profile. RS are defined according to their model for representing the user's profile and the role of this profile in the prediction of relevant items to the user [38].

Before 2007, the majority of the existing recommendation approaches did not take into any contextual information, such as time, location and the company of other people. However, this contextual information may be useful to decide which items to recommend to the user, like for example when recommending a restaurant given the time and location. Motivated by this, context-aware recommender systems (CARS) try to predict user interests by incorporating contextual information into the recommendation process. These interests are also expressed as ratings defined with the rating function $R : User \times Item \times Context \rightarrow Rating$, where $User$ and $Item$ are the domains of users and items respectively, $Rating$ is the domain of ratings, and $Context$ specifies the contextual information associated with the application. To illustrate these concepts, consider the example application for recommending restaurants to users, where users and restaurants are described as relations having the following attributes:

- Restaurant(RestaurantID, Genre, Name, Address, Capacity): the set of all the restaurants that can be recommended;

- User(UserID, Name, Address, Age, Gender, Profession): the people to whom restaurants are recommended.

The contextual information consists of the following relations:

- Open(RestaurantID, Date, TimeOfDay): the date and time periods when the restaurant is open;

- Companion(companionType): represents the type of relation the user may have with a person making him company to go to the restaurant, where attribute companionType has values “alone”, “girlfriend/boyfriend”, “co-workers”, and “others”.

Given these relations, the rating assigned to a restaurant by a person depends on with whom, which day and at what time he goes to the restaurant. For example, the restaurant to recommend to Paul can differ significantly depending on whether he is planning to go on a weekend with his girlfriend or on a workday with his co-workers.

In addition, with the proliferation of smartphones in the professional field, CARS are confronted to two other problems, which are: (1) the dynamicity of the users' content: professional users acquire and maintain a large amount of content undergoes frequent updates; (2) the risk of upsetting the user: for example, when where the professional users are in meeting, with a client or with a provider.

In such a setting, different requirements have to be taken in consideration when developing RS: (1) considering the dynamicity of the user's content, (2) considering the dynamicity of the user's context and (3) considering the risk to upset the user.

In this chapter, we describe the different works that try to answer these different requirements.

The rest of this chapter is structured as follows. Section 2.2 describes the spine of the RS which is the user's profile. Then, Section 2.3 introduces the main techniques that have been developed to implement RS. The following sections study the above mentioned requirements to take into account in RS, namely: dynamicity of the user's content (Section 2.4), the notions of context and situation in RS (Section 2.5) and risk-aware decisions in recommendation (Section 2.6). We close this discussion focusing on connections among these research areas (Section 2.7).

2.2 The User's Profile

The notion of profile appeared in the 1970s decade, mainly due to the need to create custom applications that could be adapted to the user.

In this section we treat the different aspects of the user's profile, defining it, its features and its indicators of interest, and then we describe the different approaches of modelling and acquiring the user's interests.

2.2.1 Defining the User's Profile

RS belong to a more general framework of systems called Personalized Access Systems. These systems integrate the user as an information structure, in the process of selecting relevant information to him [103].

In a RS, the user's profile is a kind of query about his interests, describing features that can be shared with a group of individuals. Comparing these features to incoming documents, the system can select those likely to be of interest [8].

2.2.1.1 The User's Profile Features

The features that characterize the user's profile are his acquired knowledge in different subjects (background), his objectives (goals) and his interests [36].

Background: The background concerns all the information related to the user's past experience. This includes his profession, experience in fields related to his work, and how the user is familiar with the working environment of the system.

Objectives: The objectives are the user's needs when he searches for information.

Interests: The interests are the documents that the user has consulted or notified to the system directly or indirectly through its feedback by indicators of interests.

2.2.1.2 Indicators of Interests

The implicit detection of interests is through observable behaviours collected by the system when the user interacts with his environment. In this context, several behaviours can be considered, such as:

- Document annotation;
- Click of the mouse on a document;
- Eye-tracking.
- Scrolling a document using the mouse or keyboard;
- Navigating to reach a document;
- Time spent on a document;

These behaviours during user/system interactions are indicators of the relevance of a document and provide useful evidence to predict interests indirectly expressed by the user.

The structures that store the user's profile features are modelled as described in what follows.

2.2.2 Modelling the User's Profile

User modelling is a research field that concerns the improvement of man-machine interaction by predicting the interests of users [103].

Modelling the user's profile consists of designing a structure for storing all the information which characterizes the user and describes his interests, his background and his objectives [100].

There are several ways to model or represent the user's profile. We describe here the different existing techniques.

2.2.2.1 Vector Representation

Vector representation is based on the classic vector space model of Salton [100], where the profile is represented as an m-dimensional vector, where each dimension corresponds to a distinct term and m is the total number of terms that exist in the user's profile.

The vector representation has been the first model of the user's profile exploited. The weighting of terms is usually based on a diagram of the TF/IDF format commonly used in IR [100].

The weight associated to each term represents the degree of importance in the user's profile. Different RS use such representation, like an on line newspaper in [90] or the recommendation of web services in [42].

In addition to its simplicity of implementation, the use of several vectors to represent the profile permits to take into account the different interests; but the default of this representation is in the lack of semantics (no connexion between terms).

2.2.2.2 Graph Representation

The graph representation is based on an associative interconnection of the nodes, where each node corresponds to a concept that represents the user's interest.

For example, the system proposed by [76] uses the concepts existing in WordNet to group similar terms. They called their model "Synset Similarity Model" because it extends the classical vector space model by using WordNet concepts, called synsets, to index documents rather than keywords and by adopting a similarity function able to deal with synsets.

A similar approach has been used in [104]. Initially, each semantic network contains a collection of nodes in which each node represents a concept. The nodes contain a single vector of weighted terms. When a new user information is collected, the profile is enriched by updating the weighted terms vector in the corresponding nodes.

The graph representation has the advantage of connecting information [115, 76], but the problem is the absence of hierarchical relation among the concepts of the network, reflecting the semantic generalization / specialization between concepts (e. g., Statistics is one specialization of Mathematics).

2.2.2.3 Ontologies Representation

Ontologies may be used to represent the semantic relations among the informational units that make the user's profile [86]. This representation allows to overcome the limitations of the graph representation by presenting the user's profile in the form of a hierarchy of concepts.

Using ontologies, each concept in a hierarchy represents the knowledge of an area of the user's interests. The relationship (generalization / specification) between the elements of the hierarchy reflects in a more realistic way the interest of the user.

The representation of the profile based on ontologies creates some problems related to the heterogeneity and the diversity of the user's interests. In fact, users may have different perceptions of the same concept, which leads to inaccurate representations [58]. For instance, the concept Java may be interpreted as a programming language, but also as an island.

2.2.2.4 Multidimensional Representation

The user's profile can contain several types of information such as demographics, interests, purposes, history and other information [92].

In [68], the authors represent the contents of the user's profile by a structured model with predefined categories considered as dimensions: personal data, data source, data delivery, behavioural data and security data. Each of these dimensions has his own representation (vector, graph or ontology). This model has been proposed in the development of a digital library service.

In the same context, authors in [66] propose a set of open dimensions that can contain

most of the information that characterizes the user. The authors propose different dimensions, like personal data, the domain ontology, the expected quality, customization, security, preferences and miscellaneous information.

The multidimensional representation has the advantage of providing a better interpretation of the semantics of the user's profile. However, the problem in this approach is in the difficulty of interpreting the role, and also the importance, of each dimension.

2.2.3 Acquiring the User's Profile

In the recommendation process, a crucial step is the construction of the user's profile that truly reflects his interests. However, this step is not an easy task because the user may not be sure of his interests [11], on one hand, and often does not want or even can not make efforts for its creation, on the other hand.

In this context, several approaches have been proposed for the acquisition of the user's profile.

2.2.3.1 Explicit Approach

The explicit approach promotes the description of the user's profile through a set of keywords explicitly provided by the user [8]. The method requires much from the user because, if he is not familiar with the system and the vocabulary of incoming documents, it becomes difficult for him to provide the proper keywords that describe his interests.

2.2.3.2 Implicit Approach

As it is not reasonable to ask the user a set of keywords describing his preferences, the idea is to observe the user's behaviour through his interactions with the system to learn his profile. This behaviour gives implicit indicators about the user's interest, as referred in Section 2.2.1.2.

Most systems construct the user's profile by learning from consulted documents. This

profile is generally based on the vector model and different indicators of interests are used, such as the movements and mouse clicks, for example. The terms' weight adjustment is computed using machine learning techniques such as neural networks, genetic algorithms and others [4].

2.2.3.3 Hybrid Approach

An hybrid approach permit to combine both precedent techniques. At the first step, the user must provide a set of keywords describing preferences to initialize the profile; then, at each arrival of a new document, the system uses the user's profile to select the documents potentially fitting his interests, and displays them to the user. The user indicates the system one relevant document and one irrelevant document to him through indicators of interests. This information is used to adjust the description of the user's profile in order to reflect his new preferences [8].

2.2.4 Discussion

We present now a synthesis of the different approaches to represent and acquire the user's profile discussed above.

Table 1: Representing the user's profile

Representation technique	**Advantage**	**Disadvantage**
Vector representation	Takes into account the diversity of interests and their dynamicity through time and is easy to implement	Lack of semantics
Graph representation	Semantic relationships between interests	Absence of hierarchical relations between the network's concepts
Ontology representation	Semantic and hierarchical relations between the ontology's concepts	Heterogeneity and diversity of the user's interests
Multidimensional representation	Better interpretation of the semantics of the user's profile	Difficulty in interpreting the role of each dimension

Table 1 summarizes the advantages and disadvantages of different approaches on user's profile modelling. From the table, we note that the graph representation solves the shortcomings of the vector representation by establishing relationships between interests in the user's profile. Moreover, the ontology representation allows to expand the limits of graph representation by including a hierarchy between interests. However, representing interests as concepts of an ontology may generate some problems related to the heterogeneity and the diversity of the user's interests. In fact, users may have different perceptions of the same concept. Finally, the multidimensional representation allows a better interpretation of the semantics of the user's profile, having, however, the difficulty in defining the role of each dimension.

Table 2: Acquisition of the user's profile

Representation technique	**Advantage**	**Disadvantage**
Explicit approach	The system is sure for the user's interests	Hard for the user to provide the appropriate keywords, the profile is static and the system is user-dependent
Implicit approach	The profile is not restricted to the keywords already entered by the user	The problem of cold start (no recommendation when the user profile is empty)
Hybrid approach	The system is sure for the user's interests, it allows the system to change the profile	Hard for the user to provide the appropriate keywords and the system is user-dependent

Table 2 summarizes the advantages and disadvantages of different user's profile acquisitions approaches. One observation is that the hybrid approach is the most appropriate method for the user's interest acquisition, in the sense that it avoids the user dependence by using implicit approach, and solves the cold start problem using explicit approach.

From this analysis, we conclude the most appropriate approaches to model and acquire the user's profile are respectively a multidimensional approach and a hybrid

approaches.

The representation and acquisition of the user's profile are part of the recommendation process. In what follows, we study the method used for the selection of documents to be recommended, also part of the recommendation process.

2.3 Recommendation Methods

It is possible to classify RS in different ways. The most common classification is based on three recommendation approaches: Collaborative Filtering, Content-based Filtering and Hybrid-based Filtering [3, 101, 64, 13].

2.3.1 Collaborative Filtering

Collaborative Filtering (CF) exploits the user's feedback on resources (documents) presented to him. The user's assessments are generally represented by some kind of notation, like from one to five starts. They are either assigned explicitly by users or implicitly from the indicators of interest (Section 2.2).

The first RS has been designated as CF system [35]. This system allows users the access to their emails. Depending on the appreciations of other users about the emails they receive, the system makes recommendations. The authors have called this approach "Collaborative Filtering" because users could work together to set undesired emails.

More specifically, CF uses a matrix which rows correspond to the users and the columns to resources. Each cell in the matrix correspond to a note provided by a user for a specific resource. The goal is to predict notes to resources for which the users have not yet provided a rating, and then recommend the best resources w. r. t. the predicted notes. CF is generally classified into two approaches: the memory-based approach and the model-based approach [35].

The memory-based approach treats the whole matrix for finding similarities between users. This approach has the advantage of being both simple to implement and

efficient, and is dynamically adapted when new notes are entered into the matrix. However, its complexity is such that its use is only possible in a space data relatively small.

The base structure for these methods is the user-item ratings matrix $R = User \times Item$. Let $r_{u,i}$ be the rating assigned by user u to the item i. To find missing ratings and recommend items with the highest ratings, an aggregation function which uses ratings given by other users is generally used:

$$r_{u,i} = \overline{r_u} + K \sum_{u' \in U} sim(u, u') \times (r_{u',i} \overline{r'_u}) \tag{1}$$

In this aggregated rating, K is an constant and each contribution is weighted using a similarity metric between users. To compute similarity between users, most popular approaches use the set of items co-rated by two users u and u', and noted $I_{u,u'}$. The most common similarity metric in CF methods is the users Pearson correlation [105]:

$$sim(u, u') = \frac{\sum_{i \in I_{u,u'}} (r_{u,i} - \overline{r_u})(r_{u',i} - \overline{r_{u'}})}{\sqrt{\sum_{i \in I_{u,u'}} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum_{i \in I_{u,u'}} (r_{u',i} - \overline{r_{u'}})^2}} \tag{2}$$

In the model-based approach, a probabilistic model is built from the users assessments (for example, Bayesian classification), and it is applied to evaluate conditional probabilities, if the rating prediction is based on the expected value:

$$r_{u,i} = E(r_{u,i}) = \sum_{r=0}^{n} P(r_{u,i} = r | r_{u,i'}, i' \in I) \tag{3}$$

This model runs faster than the memory-based model, but the construction of the model takes time [35].

CF gives an interesting recommendation only if the overlap between users' history is high (similarity on user's profiles) and the users' content is static [73]. In addition, it suffers from the "cold start" problem since the system can provide relevant recommendations only if the user provides enough rating for a sufficient number of resources (if there is no rating for a new resource, this resource can not be recommended).

2.3.2 Content-Based Filtering

Content-Based Filtering (CBF) analyses the resources to determine which ones are likely to be interesting for a given user. This domain is highly similar to IR. Indeed, the same techniques are used, the difference being essentially in the absence of explicit requests made by the user. Therefore, many of the general concepts of recommendation based on the content come from IR.

CBF recommendation is based on notes assigned by users to multiple documents and the similarity between documents according to certain criteria. For example, in order to recommend websites not previously seen to a user, the system searches for similarities with respect to certain characteristics (type of website, content, etc.), with websites that have been previously assessed a high score by the same user. From the similarity computation, only websites with a high degree of similarity will be recommended [9].

Generally, the content of a document is described by keywords; and for this, the system must know the importance of each word in a document associating it with a weight. Weights can be computed in several ways, being the most known method the so called TF/IDF. This method tries to find the importance of a word in a text using its "term frequency" (TF) and "inverse document frequency" (IDF) values. In IR, terms are the resulting word roots after a pre-processing step including steeming and eliminating stop-words. Let the content of a document d be defined by $d = \{w_{k_1,d}, ..., w_{k_n,d}\}$, where w_{1d},..., w_{Kd} are computed using in Eq. 4.

$$w_{k,d} = TF(k,d) \times IDF(k) = \frac{f_{k,d}}{max_{k'} f_{k',d}} \times log(\frac{N}{n_k}) \tag{4}$$

In Eq. 4, N is the number of documents in the collection, n_k is the number of documents where the term k appears and $f_{k,d}$ is the number of times term k appears in document d.

Given two vectors $\overrightarrow{w_u}$ and $\overrightarrow{w_i}$ representing respectively the user's interests and the

document, the utility function f can then be defined using a distance metric between these two vectors. The cosine similarity measure is generally used:

$$f(u,i) = cos(\overrightarrow{w_u}, \overrightarrow{w_i}) = \frac{\overrightarrow{w_u} \cdot \overrightarrow{w_i}}{||\overrightarrow{w_u}|| \cdot ||\overrightarrow{w_i}||} = \frac{\sum_k w_{k,u} w_{k,i}}{\sqrt{\sum_{k,u} w_{k,u}^2}\sqrt{\sum_{k,i} w_{k,i}^2}} \tag{5}$$

CBF identifies new documents which match an existing user's profile. However, the recommended documents are always similar to the documents previously selected by the user [88]. The main limitation of content-based recommendation is that it requires a sufficient number of attributes that describe the resources. That is why it is appropriate to recommend text resources or when textual descriptions of resources have been entered manually.

2.3.3 Hybrid Approach

The hybrid approach combines the collaborative and the content-based filtering methods to reduce their limitations. Among the several ways to combine these two methods, we distinguish the following ones.

2.3.3.1 Extending CBF with properties of CF

Several hybrid recommendation systems extend CBF with properties of CF, as [99]. In this work, authors compute the similarity between two users' profiles based on all the documents each user has assessed, rather than only on assessed documents users have in common. The advantages of this method is to decrease the cold start problem if the users have not assessed the same documents.

2.3.3.2 Extending CF with properties of CBF

Existing techniques to extend CF with properties of CBF are used to make clustering on a group of profiles based on the content. For example, in [123], authors use clustering indexing to create a set of user's profiles.

2.3.3.3 Unifying CBF and CF

An example of unification of CBF and CF is the work described in [124], where authors propose to use characteristics of both methods in a single classifier, using Bayesian regression to classify the interesting and the non-interesting documents.
Studies in [99, 8] have shown that the hybrid approach unifying CBF and CF has given better results compared to pure recommendation approaches, like CBF or CF. The limitation of the hybrid approach is that a new user of such a system must provide relevance assessments for a minimum number of resources before the system can provide relevant recommendations (cold start problem).
Another limitation of the hybrid approach is the inability to recommend interesting documents when the frequent changes of the user's content is high.

2.3.4 Machine Learning Approach

Machine learning methods can be used to learn models and predict documents and are generally used to improve the precedent techniques. We describe here mainly techniques used in RS.

2.3.4.1 Naive Bayes

Naive Bayes classifier is a common probabilistic approach which can be used to associate a document to a certain category. A simple and binary example of such categories is to classify documents as relevant or irrelevant for the user [85]. Let C denote a category, and $k_1, ..., k_n$ the keywords of the document to classify. The problem is to evaluate the probability $P(C|k_1, ..., k_n)$. Under the keywords independence assumption, this probability is proportional to:

$$P(C|k_1, ..., k_n) \propto P(C) \prod_{i=1}^{n} P(k_i|C) \tag{6}$$

All individual probabilities $P(k_i|C)$ in the above equation can be evaluated with learning on a training set. The method consists in computing the probability for each

category C and assigning the document to the one with the highest value. It has been shown that the naive Bayes classifier gives good accuracy results [85].

This model gives great result, but the construction of the model takes time [35].

2.3.4.2 Case-based Reasoning (CBR)

Case-based Reasoning (CBR), a well-known artificial intelligence technique, has already proven its effectiveness in many domains. The fundamental concept in CBR is that similar problems will have similar solutions. CBR is a method of solving a new problem by analysing the solutions to previous, similar problems. It can be applied to complicated and unstructured problems relatively easily.

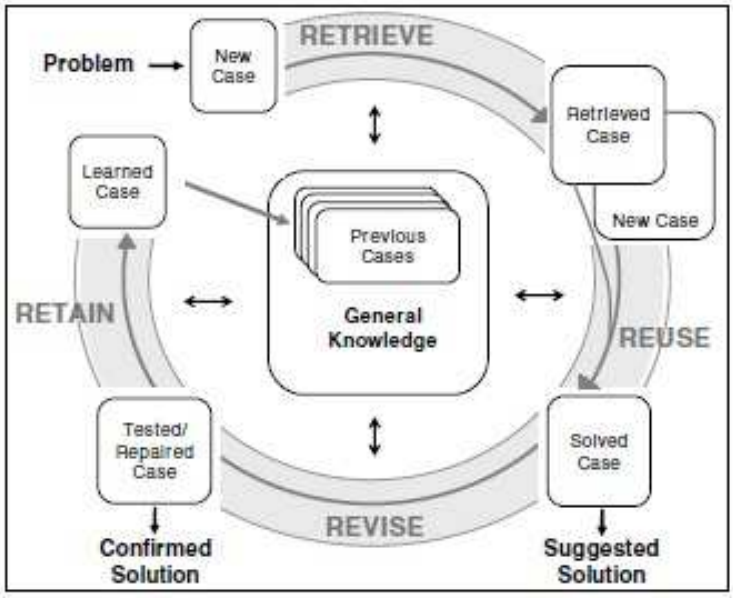

Figure 1: Case-based Reasoning

As shown in Figure 1, CBR is typically described as a cyclical process comprising the four REs : (1) REtrieve the most similar case or cases, (2) REuse the information and knowledge in that case to attempt to solve the problem, (3) REvise the proposed solution if necessary, and (4) REtain the parts of this experience likely to be useful for future problem solving.

In [78], a case is modelled by objective attributes describing the product (content model) and subjective attributes describing implicit or explicit interests of the user in this product (evaluation model). Formally, the case c is defined as $c \in X \times E$. The content model (X), in this system, is represented by a vector space $X = \prod_{i=1}^{n} X_i$ where, for instance, x_1 is the restaurant code (integer); x_2 is the restaurant name (string); x_3 is the restaurant address (string); x_4 is the cuisine type (string); x_5 is the approximate price (real); x_6 is the capacity (integer) and x_7 is the air-conditioning (boolean). The evaluation model (E) is also an heterogeneous vector space $E = \prod_{i=1}^{m} E_i$ where some $e_i \in E_i$ describe explicit interests attributes like a general evaluation of the product provided by the user or a quality price ratio. Some other e_i describe implicit evaluation attributes like the rate of time spent by the user to read product information. There is also a special attribute, called drift attribute, that measures how recently the user expressed his interest in the product. When this drift attribute becomes very small the system tends to reduce the importance of the information contained in the case associated to that product, and eventually can discard the case.

The recommendation process starts with the user providing some preferences about a new restaurant he/she is looking for and with a new restaurant r (source case). The goal of the system is to evaluate if this new restaurant r could be interesting for the user. With these preferences and input product the system searches similar restaurants in the case base (**retrieval phase**) to find restaurants that could be used to compute the interest prediction of the new restaurant r. In the **reuse** phase the system basically extract from the retrieved cases $(c_i, i = 1, ..., k)$ the interest attributes, or in our terminology the evaluation model. In the **revise** phase the system assumes that the user's interest in the new restaurant r is similar to his/her interest in the retrieved restaurants, hence, for each retrieved case c_i it extracts the interest attributes (evaluation model) and compute a global interest value $V(i)$ for

each retrieved case. $V(i)$ is a weighted average sum of the interest attributes multiplied by the drift attribute. Then a global interest confidence value $I(r)$ for the product r is computed as a weighted average of the interest values of the retrieved cases:

$$I(r) = \frac{\sum_{i=1}^{k} V(i) Sim(r, c_i)}{\sum_{i=1}^{k} V(i) Sim(r, c_i)} \tag{7}$$

If the interest confidence value of the new restaurant is greater than a certain value (a confidence threshold), then the new restaurant is recommended to the user. Otherwise, the CBR cycle terminates with no recommendation and the system just provides a negative advice to the user about the queried restaurant. The review phase is implemented by asking the user for the correct evaluation of the restaurant and after that a new case (the product and the evaluation) is retained in the case base. Also implicit evaluation indicators are retained as derived from the analysis of the user/system interaction.

The case base reasoning approach give interesting recommendation in dynamic environment but its learning is slow.

2.3.4.3 Artificial Neural Network (ANN)

To make recommendation in [44] a ANN and specially a multilayer feed-forward architecture is adopted, with various numbers of hidden layers and distributions of units among layers. The connection from unit m to unit n is characterized by a real-number weight w_{mn} with initial value positioned at random in the range [-1, 1]. When a pattern μ is impressed on the input interface, the activities of the input units propagate through the entire network. Each unit in a hidden layer or in the output layer receives a stimulus, where the am are the activities of the units in the immediately preceding layer. The activity of generic unit m in the hidden or output layers is in general a nonlinear function of its stimulus, $\alpha_m = g(u_m)$. In [44] the unit activation functions $g(u)$ are selected between the logistic (sigmoid), hyperbolic

tangent and linear forms. The system response may be decoded from the activities of the units of the output layer while the dynamics is particularly simple: the states of all units within a given layer are updated successively, proceeding from input to output. Several training algorithms exist that seek to minimize the cost function with respect to the network weights. For the cost function the authors make the traditional choice of the sum of squared errors calculated over the learning set, or more specifically

$$E = \sum_{\mu} E^{(\mu)} = \frac{1}{2} \sum_{\mu,i} (t_i^{(\mu)} - o_i^{(\mu)})^2 \tag{8}$$

where $t_i^{(\mu)}$ and $o_i^{(\mu)}$ denote, respectively, the target and actual activities of unit i of the output layer for input pattern (or example) μ. The most familiar training algorithm is standard back-propagation denoted SB, according to which the weight update rule to be implemented upon presentation of pattern μ is

$$\Delta\omega^(\mu)_{mn} = -\eta \frac{\eth E^(\mu)}{\eth W_{mn} + \alpha \Delta\omega_{mn}^{(\mu-1)}} \tag{9}$$

where η is the learning rate, α is the momentum parameter, and $\mu - 1$ is the pattern impressed on the input interface one training step earlier. The second term on the right-hand side, called the momentum term, serves to damp out the wild oscillations in weight space that might otherwise occur during the gradient-descent minimization process that underlies the back-propagation algorithm. In this work the ANN is trained with a modified version of the SB algorithm.. In this algorithm, denoted MB, the weight update prescription corresponding to Eq. 9 reads

$$\Delta\omega^(\mu)_{mn} = -\eta \frac{\eth E^(\mu)}{\eth w_{mn} + \alpha S_{mn}^{\mu-1}} \tag{10}$$

the momentum term being modified through the quantity

$$S_{mn}^{(\mu-1)} = \frac{S_{mn}^{(\mu-2)} e + \Delta w_{mn}^{(\mu-1)}}{e+1} \tag{11}$$

In the latter expression, e is the number of the current epoch, with $e = 0, 1, 2, 3, ...$ The replacement $w_{mn}^{(\mu-1)}$ of by $S_{mn}^{(\mu-1)}$ in the update rule for the generic weight allows

earlier patterns of the current epoch to have more influence on the training than is the case for standard back-propagation. The advantage of the ANN is the rapidity of learning, but it have a problem in dynamic environment.

2.3.5 Discussion

We present in Table 3 a synthesis of the recommendation approaches, and their advantages / disadvantages.

Table 3: The recommendation approaches

Recommendation approach	**Advantage**	**Disadvantage**
CF [35, 44]	Gives an interesting recommendation when the overlap between users' history is high and the users' content is static	Inability to recommend new documents
CBF [9, 78, 85]	Is able to recommend new documents	Can not make recommendation when a new user arrives
Hybrid Approach [123, 99, 124]	Overcome the limitations of both approaches (can make recommendation when a new user arrives and can recommend new documents)	Problem with the frequent changes of the user's content

From Table 3, we can observe that the inability of CF to recommend new documents is reduced by combining it with the CBF technique [72]. However, the user's content in real applications undergoes frequent changes. These issues make CBF and CF approaches difficult to apply [47].

Generally the precedent approaches improved by supervised learning as ANN, CBR or Naive bayes approaches, but they still have the problem to follow the dynamicity of the user content.

The dynamicity of the user's contents is the subject of the next section.

2.4 The Dynamicity of the User's Content

As stated in Section 2.3, the dynamicity of the user's content still remains a problem for an accurate recommendation. This section aims to explain this problematic point in outlining the proposals that have been made in research with their advantages and disadvantages.

The dynamicity of the user's content is reflected in the difficulty of generating recommendations for the user when old documents may expire and new documents may emerge continuously. This dynamicity lets the user's profile in a permanent cold start, also called latency [102].

The cold start problem occurs, in fact, when a new document is integrated into the system and the relation between the user's interests and the document is not yet available. Therefore, the new document will not be involved in the recommendations. The problem of latency has more impact on systems that incorporate new documents regularly, such as systems recommending news articles [72, 38].

To overcome this latency problem, two approaches are proposed, the first one uses CBF and the second one uses an exploration-exploitation strategy. These two approaches are described in what follows.

2.4.1 CBF for User's Content Dynamicity

CBF tries to solve cold start by techniques based on content similarities. When a new document is introduced, the technique based on the content evaluates the similarity of the new document with the available documents in the user's profiles to make recommendation. However, the use of this technique based on the content leads to a lack of diversity in recommendations, which hinders the performance of the RS.

A new filtering technique (based on content) exploiting ontologies has also been suggested as a solution to the problem of latency. The system uses location ontologies, in order to classify and categorize a new arrived documents and generate the users'

profiles. This technique has been especially used by [123] to recommend restaurants and by [86] to recommend scientific papers. However, the limitation of this technique is the need to pre-construct the domain knowledge ontologies.

In [123], the authors tackle the user's contents dynamicity by proposing a system with a dynamic user's interest, where the user's interests at different moments should have different weights. To catch a user's short-term interests without losing the long-term interests. There are four major phases in the proposed approach: (1) In the first phase the system groups the new documents with similar contents. (2) In the second phase the system calculates the weight of each document based on the previously given user's preferences in each document, as well as the corresponding timestamps. Then, the interest vector of each user is constructed to represent the user's current short-term interests. (3) After that, the system identifies which users are the neighbours of the active user based on the similarity of their interest vectors. (4) Finally, the system estimates the interests for the active user, to take into account in the document selection, based also on the interests of his/her neighbours.

The limitation of this approach is in the deterioration of the quality of recommendations if the user's social group is heterogeneous.

2.4.2 Exploration-Exploitation Trade-Off for Content Dynamicity

Few works found in the literature [72, 73] solve the problem of content dynamicity by addressing it as a need for balancing exploration and exploitation studied in the multi-armed bandit problem.

A bandit algorithm B exploits its past experience to select documents (arms) that appear more frequently. Besides, these seemingly optimal documents may in fact be suboptimal, because of the imprecision in B's knowledge. In order to avoid this undesired situation, B has to explore documents by choosing seemingly suboptimal documents so as to gather more information about them. Exploration can decrease short-term

user's satisfaction when some suboptimal documents may be chosen. However, obtaining information about the documents' average rewards can refine B's estimate of the documents' rewards and in turn increases the long-term user's satisfaction.

Clearly, neither a purely exploring nor a purely exploiting algorithm works well, and then a good trade-off is needed.

The exploration-exploitation approach is used in [72] for ads recommendation and in [73] for news recommendation.

2.4.3 Discussion

To discuss the two approaches for following the user's content dynamicity, we present, in Table 4, their advantages and disadvantages.

Table 4: Approaches for following the user's content dynamicity

Approach	**Advantage**	**Disadvantage**
CBF [38, 86]	Recommendations similar to the user's history	The lack of diversity in recommendations.
Exploration-Exploitation Trade-off [73, 72]	Recommending diversified information to the user	The risk to upset the user with randomly choose documents

From Table 4, we can observe that the techniques based on CBF have the advantage of exploiting to user's history in the recommendation. However, these techniques have a lack of diversity of content in recommendations.

In the other case, exploration-exploitation trade-off approaches have allowed recommending varied and diverse documents to the user, but the risk is to upset the user with exploratory recommendation.

We can conclude that a trade-off has to be done among the exploration-exploitation approach and the CBF approach.

Moreover, in order to be well accepted by the user, exploratory recommendations should be managed regarding the user's context, where some situations are not adequate for exploration (for example, when the user is in professional meeting).

In what follows, we describe the impact of the user's context in the recommendation process.

2.5 Context-Aware Recommender Systems (CARS)

Before the first CARS workshop on 2009, the majority of RS focus on recommending the most relevant documents to users and did not take into consideration any contextual information, such as time, location and nearby people. However, when recommending a personalized content, it is not sufficient to consider only user's profiles and documents. It is also important to recommend documents adequate to the user's situation. Therefore, a good recommendation depends on how well the RS has incorporated the relevant contextual information into the recommendation process. Recently, some RS have taken the context into account, being called Context-Aware Recommendation System (CARS). However, for a long time many works deal with the context in other areas, like IR, mobile-learning and advertising since context become inescapable.

In this section, we define the general notion of context, how to get context information, how the context can be modelled and how the context is used to recommend pertinent information to the user. We finish with a discussion comparing the presented approaches.

2.5.1 Context Definition

The notion of context appeared in several disciplines, like computer science, linguistics, philosophy, psychology, etc., and every discipline gives its own definition, often different from the others, which is more specific than the generic definition given by [114] "conditions or circumstances that have an effect on something". Therefore, there are several definitions of context across varied disciplines.

Since our concern is RS, and since the general concept of context is very wide, we focus on ubiquitous and mobile context-aware computing, a field directly related to

RS having the context as the core concept.

In context-aware computing. Authors in [2] have considered the context as a key component to increase human-machine interactions [2], and they have given the subsequent definition of context that is now ordinarily accepted: "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." [2].

Using the above definitions of the context and situation, in what follows we present the different techniques used to obtain contextual information, and then we describe existing context models.

2.5.2 Context Acquisition

According to Grudin [59], the context acquisition is the process through which contextual information is captured.

The context can be obtained by different methods, depending on the contextual information that the system needs. In the literature, there are three types of contextual information and correspondent acquisition method [59]:

- Context detected: this type of information is acquired through physical sensors or software sensors, such as temperature, atmospheric pressure or noise;

- Context derived: such contextual information is calculated during execution, such as time and date;

- Context explicitly provided: this is the case when the user explicitly gives to the system the necessary information, like for example when the user feeds his agenda with his planned meetings.

The structure that stores the contextual information can be modelled as described in what follows.

2.5.3 Context Modelling

Context models formalize the representation of the context as a structure (ontology, class of vectors of terms, set of concepts, etc.) or a set of specific and different information structures. We present now the most interesting models found in the literature. A deep and complete survey and analysis framework for context models has been published in [10].

2.5.3.1 Attribute-Value Models

The most simple context models are based on attribute-value pairs to represent context, where attributes capture various characteristics of contextual elements.
An attribute-value model is used in [118] for representing the context of each user's item. First, they build a user profile. The basic data source of the technology is a user-items matrix, $A(m, n)$. It stores the ratings which are given by m users for n items. m denotes the users information $U(u_1, u_2, ..., u_m)$, and n denotes the items information $I(i_1, i_2, ..., i_n)$. If a user u rates an item i, it generate a rating Ru, i, which is between 0 and 5. Where 5 is the best rating. So their context is modelled as follows: $C = (C_1, C_2, ..., C_n)$, where C presents one type of contexts, such as Time and i consists of many different variables. For example, in the type of Time, there are several values (such as morning, noon, afternoon, and evening). And a user may have different interest for the same item in different variables of one type of context. So they give the definition of $C_i = (C_{i1}, C_{i2}, ..., C_{ik})$.

To introduce context on their model they consider a $User \times Item \times Context \rightarrow Rating$ model. Figure 2 shows their specific three-dimension rating model. In the Figure 2, there is $R_{us_1,i_1,C_{11}}$ which means the user us_1 gives a rating whose value is 3 for the item i_1 in the variable c_{11} of context C_1. So there are many models which store the ratings given by all users for every item in each context variable.

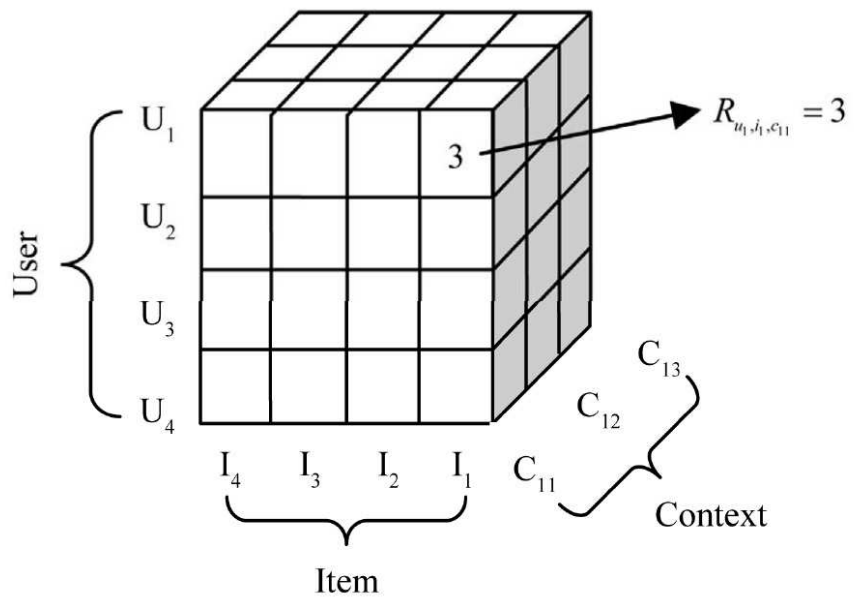

Figure 2: Attribute-value model proposed by [118]

Attribute-value models are frequently used because they are particularly easy to manage. However, they lack of semantic, since no relation between attributes and/or values are given.

2.5.3.2 Object-Oriented Models

The main benefits of object modelling come from being commonly understood by application developers to know the encapsulation between object and class [10].
In [95], the context is described by a number of elements, which characterize the various aspects of the entity's situation. This entity can be a user, an object, a place, among others. In the proposed model, the context is represented by the class *Description_of_Context* of the UML class diagram shown in Figure 3. This class represents the contextual factors which are subclasses of an abstract class called *element_of_Context*.

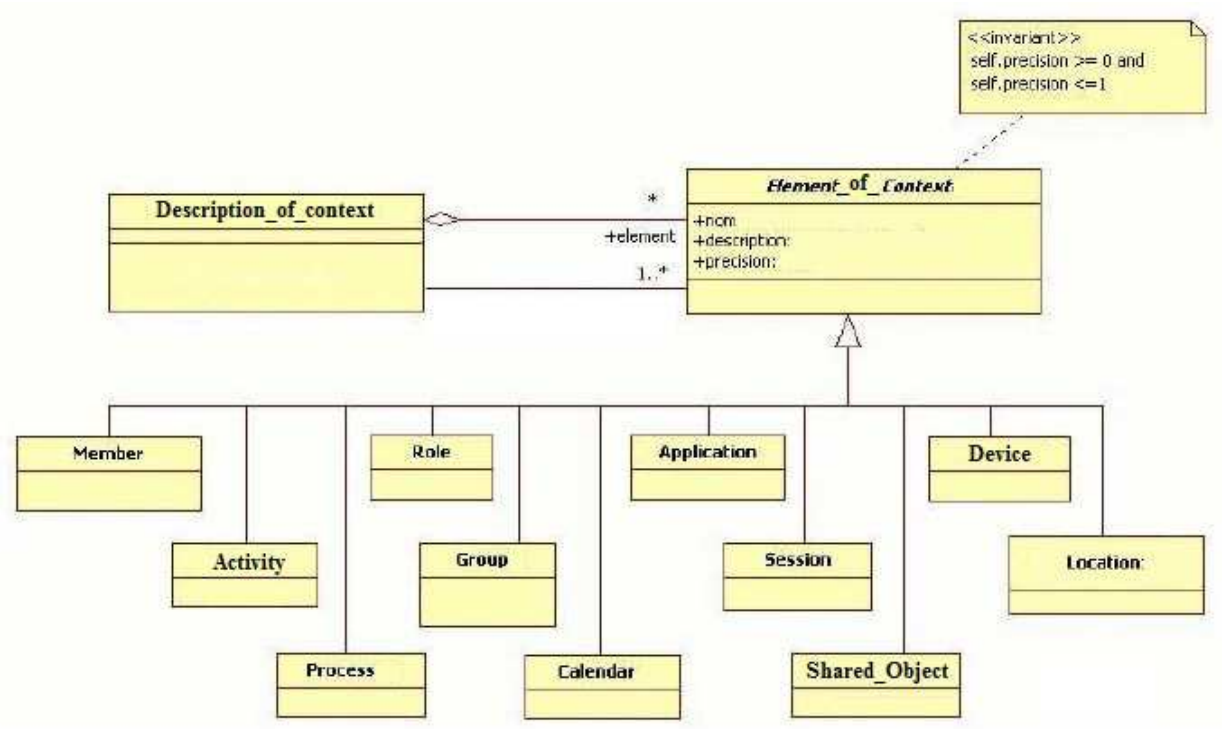

Figure 3: Object-oriented model proposed by [95]

2.5.3.3 Markup Models

Markup scheme models use a markup-text representation to model the context data, and they generally use the Extensible Markup Language (XML). The aim of this approach is to encourage a tree representation and enable more reasoning [10].

For example, in [79], the user's context is modelled using XML [49]. In this model, tags are used to describe the context of users. In addition, spatial tags are used to decline the user's preferences on content based on predefined localization zones. When a mobile client makes a queries to get a document, his current contextual information is considered for executing the queries. Authors propose an XML-based model called XREAL that formalizes the contextual information as well as the queries, which are represented as recursive relational algebra trees.

The XReAL models the mobile user using an identification attribute, called MCID, and a sequence of elements referring to contextual information: physical context, environmental context, informational context, personal context, social context, application context, and system context. Figure 4.A shows the XML schema of a mobile client at an abstract level.

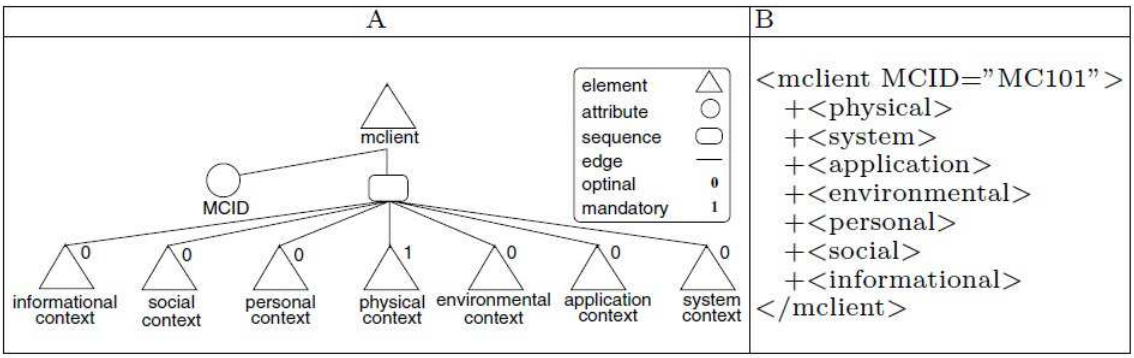

Figure 4: A) The XReAl query model; B) The XReAl contextual information document

Any mobile client is assigned a MCID number, to be recognized by the system. Physical context provides information related to location and time. The location is a position, elevation, and direction. The position could be represented using a geographical coordinates and/or relative coordinates, such as a street, area and city. The time represents time zone, which could be inferred from the location information. The time zone determines the absolute time, day, week, month, quarter, and year. Physical context might help to infer information at a generic level related to environmental context, such as weather and light. Other methods are needed to determine an accurate environmental information.

Informational context formalizes information of interest to the mobile client, such as currency rates, stoke quotes and sports scores. Personal context specifies information

such as health, mood, biographical information, habit and activities. Social context formalizes information concerning group activity and social relationships. Application context models information, such as email received and websites visited. The system context represents information related to systems used by the client and specifications of her mobile, such as processor, and memory.

The user of a mobile client might provide personal and social information to be recorded as contextual information related to her mobile client. It is assumed that the minimum level of information is the information of physical context. So, the physical context element is a mandatory element. However, the other elements are optional. Furthermore, it is assumed that there is a repository of contextual information related to the environment, in which mobile clients are moving, such as parking spots or food shops.

Examples. Figure 4.B shows the contextual information document specified using XReAl. The contextual information document is assigned MC101 as an ID. The XML language is very flexible in representing variety of contextual information. Figure 5 depicts part of the physical and informational contexts. Figure 5.A shows a representation for information of the relative position, and Figure 5.B illustrates a representation for business information as a part of informational context.

A	B
<relative_position> <country>Germany</country> <city>Bruchsal</city> <area>south</city> <street>Durlacher<street> <postal_code>76646</postal_code> </relative_position>	<quote> +<value> +<newspaper> +<section> +<date> +<description> </quote>

Figure 5: A) part of the physical context, B) part of the informational context

Another work using the markup model is the "context model layer" described in [89]. This model is composed of three layers. The first layer (conceptual layer) aims at representing real world context entities by concept, such as the name of a person, the type of a sensor, or the characteristics of a device. The second layer (logical layer) aims at enhancing the resultant conceptual context model with special characteristics of context information that can be classified into static and dynamic, or profiled, inferred and sensed. The final layer (physical layer) allows to describe context data in flat textual XML schemas.

2.5.3.4 Semantic Models

Semantic models are generally based on ontologies because they are an interesting tool to share knowledge, reusing knowledge and they have the possibility of logical inference [10]. Ontologies describe context data semantically independently of programming languages. They make it easy to automatically deduce further implicit high-level context data or situations (such as user's activities) from low-level context data (such as location, temperature or noise). In addition, they offer a support for interoperability and heterogeneity since ontologies can be shared. Among the ontologies developed for context modelling we can cite: CONON (CONtext ONtology) is one of the first approaches that used OWL for modelling context in pervasive computing environments [112]. SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications) is a set of ontologies expressed in OWL and designed to model and support pervasive computing applications [45]. CoDaMoS defines an ontology around four concepts used to model Users (preferences, profile and current activity), the Environment (time, location information and environmental conditions such as lighting), Platforms (a hardware and software description of a device) and Services (software which provides a service to a user) [96]. In addition, the ontology identifies context data about the user such as profiles, roles, moods, tasks and I/O devices.

MUSE is a multi-ontology representation in which each ontology corresponds to an

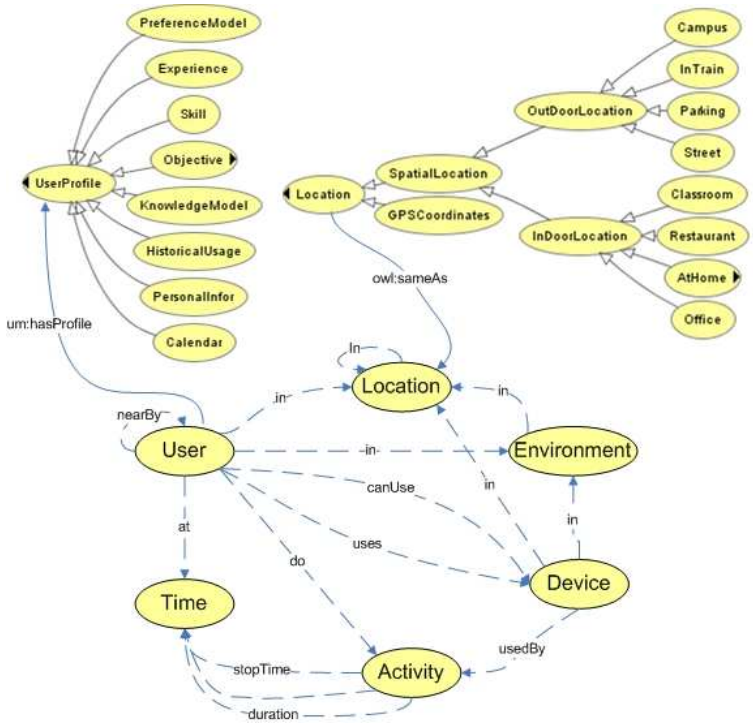

Figure 6: MUSE ontology

element or a context dimension (User, Activity, Environment, Device, Location and Time) (see Figure 6) [32]. This approach proposes a bridge over these dimensions with semantic relations (e.g., in, use, nearBy, do) to express facts like: Where is the user? In which environment is he/she? Which activity is he/she doing? Which device is he/she using? Some of these ontologies are based on existing standards like CC/PP for Device ontology and W3C for Time ontology.

A recent work in context aware information retrieval field uses an ontology-based context model [14]. They developed an ontology for modelling the interests of the mobile user and exploited spatial and temporal ontologies for modelling the semantics of the spatio-temporal situations of mobile users. The situation model is represented by an aggregation of the following four dimensions (Figure 7):

- Location type: refers to a location class name, such as beach or school.

- Season: refers to one season of the year.

- Day of the week: refers either to workday, weekend or holiday.

- Time of the day: refers to the time zone of a day, such as morning or night. More specifically, a situation S is represented as a vector whose features X are the values assigned to each dimension:

$$S = (X_l, X_u, X_v, X_w) \tag{12}$$

Where X_l (resp. X_u, X_v, X_w) is the value of the location type (resp. season, day of the week and time of the day) dimension.

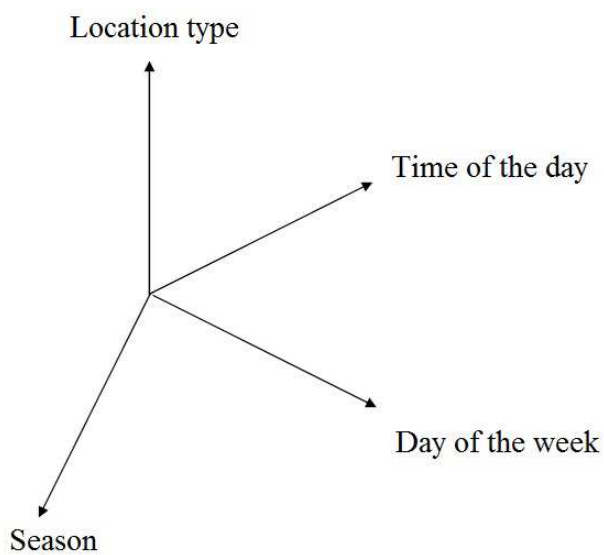

Figure 7: Dimensions of the situation model [14]

In [71], the context model has four dimensions, as shown in Figure 8: spatial (location and points of interest), social (e.g., personal information and activity being performed), temporal (date and time) and computational (mobile device). These dimensions have been explored in the acquisition of knowledge about users and photos for context-aware recommendation of photos. The location attribute is extracted from the user's mobile device (GPS). Other attributes such as place description (e.g., shopping, beach, etc.) are derived from freely available web services such as WikiMapia.

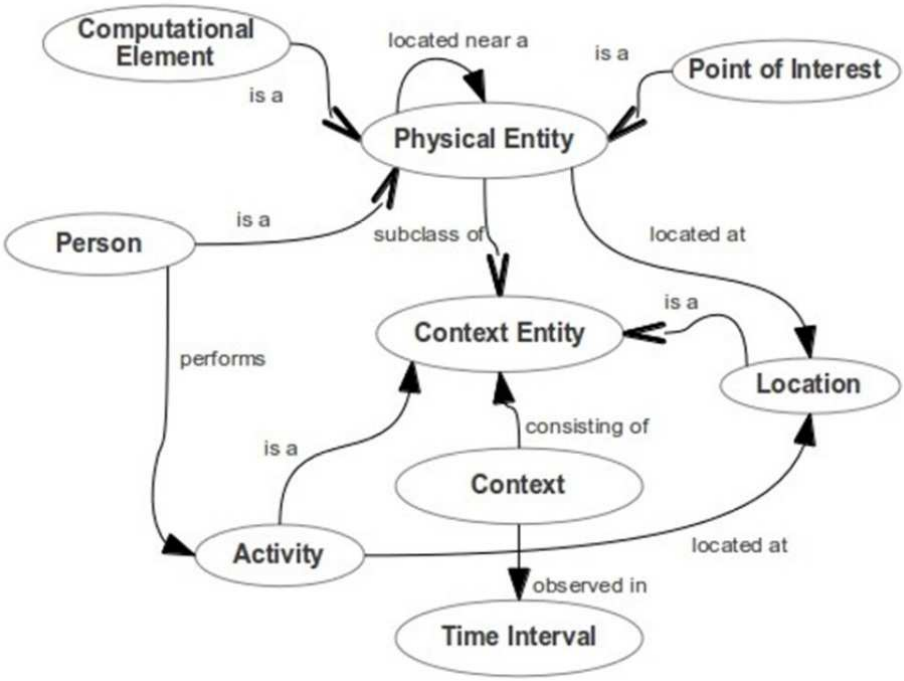

Figure 8: Ontology model of context proposed by [71]

Date and time considered are those of use in the system. The activity needs to be explicitly informed by the user and can be chosen among a set of presented options or reported manually. Examples of activities are: sports, festivals, and landscapes.

2.5.4 From Context Awareness to Situation Awareness

In spite of many benefits from adopting context awareness, some researchers state that context awareness often becomes insufficient if the application requirements need to consider the relationship between various context data and history of users and applications actions over a period of time [119]. In order to palliate this weakness, the concept of situation has been introduced. Situation aware applications [83] describe a new class of context-aware applications that are capable of recognising a user's situation and adapt themselves to a current, and maybe future, user's situation. Situation awareness focuses on the modelling of a user's environment to help him/her

to be "*aware of his/her current situation*". The same piece of information may have different meanings and usages for different people in the same environment. Therefore, the system must know the context of the current user in order to provide him/her with appropriate information, or to perform certain tasks. Feng *et al* [53] provided a clearer distinction between these two concepts: Context awareness allows systems to dynamically adapt to changes in a user's task domain, by updating relevant information and service provision, whereas situation awareness focuses on information about the state of the environment in which these tasks are carried out. Situation awareness implies context awareness, but not *vice versa*.

2.5.4.1 Definitions of the Term Situation

A plethora of definitions of the term situation exist in the literature. McCarthy introduced in 1963 the theory of situation calculus [81, 82]. He defined the concept of situation as "*the complete state of the universe at an instant of time*". Reiter formalised this notion using action theory and proposed a new definition of situation: "*a finite sequence of actions. It's not a state, it's not a snapshot, it's a history*" [97]. Endsley introduced the theory of situation awareness [50]. He defined situation awareness informally and intuitively as "*knowing what's going on*" and, more formally, as "*the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future*".

More recently, in pervasive computing domain, Yau *et al.* defined the situation as follows: "*The situation of an application software system is an expression on previous device-action record over a period of time and/or the variation of a set of contexts relevant to the application software on the device over a period of time. Situation is used to trigger further device actions*" [119, 120]. This definition introduces the concept of context and shows the relation between context and situation. However, this definition is not complete since there is no difference between context, and situation

and the meaning of situation is not clear. Accordingly, the situation seems to be composed of various information as well as context and it has more expressive power than context since it contains previous device-action record over a period of time. Situation does not trigger actions, but on a given situation, the actions are triggered by recognising a set of context information.

The authors in [121] define a situation as "*an external semantic interpretation of sensor data*". Interpretation means that situations assign meanings to sensor data. The adjective "*external*" means that the interpretation is done at the application level rather than at sensors level. The term "*semantic*" means that the interpretation assigns meaning to different types of sensors' data. In this latter definition, temporal aspects are not considered. However, a situation is not an activity as it involves rich temporal aspects (e.g., time-of-day, duration, frequency, sequence). [32] proposes the following definition: "*A situation is a set of semantic relations between concepts (in one context dimension or between several context dimensions) which are valid and stable during an interval of time*", where the term "*dimension*" here, means a context type such as location, time, activity, etc.

Another point of view is proposed in [87] where situation awareness is compared to problem recognition. At a certain point of time, a system which has abilities of situation awareness as well as context awareness understands its context, recognises a problem, and makes plans to resolve it. A situation is then considered as a problem: "*a problematic and developing state of a computational element characterised by its context*". This definition is limited since it focuses on problem recognition of a computational element.

The definition proposed by [32] in a previous work of our team seems more appropriate for our goal: It makes a distinction between context and situation; meanings are assigned to context data; a set of sensor data may be considered as a context dimension; and temporal aspects are involved. Therefore, this definition is the one

we have chosen for this thesis.

2.5.4.2 Key features of situations

A situation mostly depends on the following three parameters: available sensors, system environment (domain knowledge) and application's requirements (interesting states to observe). In other words, the same raw data can be interpreted in different situations according to application's requirements. For example, meeting and room not free are two possible interpretations depending on application's requirements: user-centred or location-centred. A situation is then an interesting abstracted state where certain actions can be taken or certain recommendation can be proposed when this situation occurs. One key issue in situation awareness is situation identification. In the literature, a huge number of techniques have been used to identify situations. Most of them can be classified into two categories, namely, data-driven or knowledge-driven approaches depending on the type of sensor and the level of situation identification.

In what follow we describe how the context is used to recommend pertinent information for user.

2.5.5 Recommendation Methods

Context-Aware Recommender Systems (CARS) argue that contextual information does matter in RS and that it is important to take this information into account when providing recommendations. In this setting, we describe the main techniques for exploiting context in recommendation.

2.5.5.1 Contextual Content-Based Filtering

In [71], the authors present a CARS for contextual photos named MMedia2U. It allows for the recommendation of photos even those that have never been evaluated by users. The recommendation is performed only from the context in which the photos

were created. This allows users, without a history of use, to receive recommendations based on their current context.

The user chooses the desired "activity/interest"; then, the system captures the current user's context (location and date/time), and sends all this information to the server. This, in turn, returns a list of recommended images. If the user selects a photo, he can see its position and the distance between him and the place where the photo was captured. Concretely, a similarity measure is used in the system in order to retrieve those photos created in contexts more similar to the user's current context. The algorithm developed uses the user's context as indicative of his preferences and the context of items (i.e., photos) as a representation of its features. In their system, the context of items is the context in which the photos were created.

To make recommendations, a similarity is calculated between the context of the user U and the context of an item I using the following equation:

$$Similarity(U, I) = \sum_{c \in Context} w_c * sim_c(U, I) \tag{13}$$

In Eq. 13, the similarity is calculated without the need of training data. In this case, c is an attribute belonging to the dimensions of the context model (e.g., location); w_c is the weight of attribute c (e.g., location has a weight of 50%) and sim_c is the similarity function for attribute c. Those pictures that have the highest value of similarity are the ones recommended to user U. The function sim_c is particular to each type of context and application domain. Each context model dimension has a method to calculate its similarity. The location similarity, for instance, is calculated by measuring the distance between the place where the picture was taken and the current user's location. The similarity for activity is calculated by comparing the activity that the user is doing and the activity in which the image was generated. Eq. 14 is used with numeric context attributes. The similarity is defined by how close

two values are.

$$Similarity(U, I) = 1 - \frac{V_c(U) - V_c(I)}{max(c) - min(c)} \tag{14}$$

In Eq. 14, $V_c(U)$ and $V_c(I)$ represent, respectively, the values of context c for the user and the item; $max(c)$ and $min(c)$ represent, respectively, the maximum and minimum values for the compared attribute of context c (e.g., for $c = hour$ of the day, $min(c) = 0$, and $max(c) = 12$). The similarity between dates can compare the various attributes related to the moment the photo was taken and the moment the user is found. Some attributes compared were: the hour of the day, day of the week and month of the year. The date attributes were compared individually to analyse the influence of each one (e.g., the similarity between hour of the day has a greater influence on the choice of the user than the similarity of the day of the week). The similarity between the months of the year can be calculated by Eq. 14, adapting it to cyclic values (e.g., the distance between January and December is 1, instead of 11).

2.5.5.2 Contextual Collaborative Filtering

Authors in [118] propose a contextual memory-based CF, which is based on the traditional memory-based CF calculating the similarity. This approach calculates the similarity between the active user's current context and the other contexts associated to items previously seen, finds out the ratings already given by the active user about the items under the contexts which are similar with user's current context, and then predicts which items the active user will prefer in the current context.

Concretely, the system tries to find out which variables of the other context are the same with the variables of current context at first. Then, they calculate the similarity between the two variables using Eq. 15.

$$sim_t(x, y, i) = \frac{\sum_{u=1}^{m}(R_{u,i,x} - \overline{R_i})(R_{u,i,y} - \overline{R_i})}{\sqrt{\sum_{u=1}^{m}(R_{u,i,x} - \overline{R_i})^2}\sqrt{\sum_{u=1}^{m}(R_{u,i,y} - \overline{R_i})^2}} \tag{15}$$

In Eq. 15, t denotes one type of context, $R_{u,i,x}$ means that user u gave a rating for item i on the variable x in the context t, and $\overline{R_i}$ represents the mean of the ratings

for the item i. They use Eq.15 to calculate the similarity between the active user's current context and the other contexts in which some variables are the same with ones of current context.

They take the $sim_t(C, s, i)$ as the similarity where C means the current context, and s denotes the variables of the current context which are the same as the ones in the other contexts.

Before selecting the nearest neighbors of the active item, the system tries to know the ratings which are rated by the active user for the items in the current context at first by using Eq.15, and they define the ratings as follows:

$$R_{u,i,C} = \frac{\sum_{s \in C} R_{u,i,x}(\sum_{t=1}^{n} sim_t(C, s, i))}{\sum_{t=1}^{n} |sim_t(C, s, i)|} \tag{16}$$

Then, they use the ratings combined with Eq. 15 to calculate the similarity between item i and item j.

To generate the prediction, and after getting the nearest neighbors, the system utilizes Eq. 16 to predict the active user's ratings $P_{a,i,C}$ for item i in context C.

2.5.5.3 Contextual Hybrid Filtering

In [5], authors describe a CARS for service recommendation and they demonstrate that combining CF and CBF for discovery and selection of service gives high performance as well as it overcomes the problem of new consumer, and new service entrance. The practical results show that this hybrid RS has more performance and better quality of recommendation in comparison with CF methods.

Concretely, the hybrid method begins the recommendation process with selecting one of the available recommender systems regarding selection criteria. When the appropriate recommender system is selected, the other recommender systems will not play any role in the recommendation process. The presented recommender system consists of two parts: Collaborative Filtering Recommender System and Content Recommender System. When the consumer profile enters the recommender system,

at first, the neighbors of the consumers' mentioned service are found according to the below stages.

Calculating similarity: The similarity between the services is calculated by adjusting cosine similarity formula, which is one of the most used methods:

$$ServiceSim(i,j) = \frac{\sum_{c \subset RB_{i,j}} (r_{ci} - \overline{r_c})(r_{cj} - \overline{r_c})}{\sqrt{\sum_{c \subset RB_{i,j}} (r_{ci} - \overline{r_c})^2} \sqrt{\sum_{c \subset RB_{i,j}} (r_{cj} - \overline{r_c})^2}} \tag{17}$$

In Eq. 17, $RB_{i,j}$ denotes the set of consumers who have rated both service i and service j , r_{ci} denotes the consumer's rate for service i, r_c determines the average of the consumer's rates.

Selecting similar neighbors:

$$Similarity(i) = Sk | Sk \in L(i), ServiceSim(Sk, Si) > 0, Sk \neq Si \tag{18}$$

Sk and Si are service k and service i, respectively. $L(i)$ is a collection of services such that their similarity rate with service i is calculated. In other words, the services having a positive similarity rate with service i are considered as neighbors of i.

As a result, the recommender system makes its prediction by means of CF method and according to the below formula:

$$Pred(c,i) = \frac{\sum_{j \in ratedservices} ServiceSim(i,j).r_{ci}}{ServiceSim(i,j)} \tag{19}$$

In Eq. 19 r_{ci} is the consumers rate for service i.

2.5.5.4 Machine Learning Approach

Different machine learning techniques has been studied for recommending items in CARS. We describe in what follows some of them.

Case-Based Reasoning. In order to build the CBR model for CARS, authors in [70] define the similarity function that is used to find k previous cases similar to the new case. The similarity score between a new case N and a previous case C is

calculated using the Eq. 20.

$$Similarity(N, C) = \frac{\sum_{i=1}^{n} f(N_i, C_i) \times W_i}{\sum_{i=1}^{n} W_i} \tag{20}$$

Where N_i is the i_{th} feature value of the new case, C_i is the i_{th} feature value of the old case, n is the number of features, $f(N_i, C_i)$ is the distance function between N_i and C_i calculated by Eq. 21, and W_i is the weight of the i_{th} feature. The value of the similarity score is between 0 and 1. The more the two cases N and C are similar, the more the similarity score becomes close to 1.

$$f(N_i, C_i) = \begin{cases} 1 - d & if \quad 0 \leq d \leq 1 \\ 0 & if \quad d > 1 \end{cases} \tag{21}$$

$$d = \frac{|N_i - C_i|}{Max - Min} \tag{22}$$

In Eq.22, Max gives the maximum value among the ith feature values for all cases in the case base and the Min gives the minimum value among the ith feature values for all cases in case base.

The most distinguished advantage of CBR is that it can learn continuously by just adding new cases to the case base.

Naive Bayes Algorithm. In [107] the naive bayes algorithm is used for off-line-processing that produces user-mode learning model from profile data, past context data, feedback logs, and item data. The User-modes are defined as user interests in selecting information such as location of users' favorite contents, user preference, and user's favorite method for retrieving items.

User-mode learning model is the set of intensities of user-modes for all profile/context combinations. The intensity of the user preferences is based on the probability of category j_i of user-mode i in each profile/context combination. It is given by:

$$P(M_{ij_i}|C_{1k_1}, C_{2k_2}, ..., C_{Hk_H}) = \frac{P(M_{ij_i}, C_{1k_1}, C_{2k_2}, ..., C_{Hk_H})}{\sum_{j_i} P(M_{ij_i}|C_{1k_1}, C_{2k_2}, ..., C_{Hk_H})} \tag{23}$$

In Eq. 23, M_{ij_i} represents the user-mode j_i of user-mode category i ($i = 1, 2, ..., I, j_i = 1, 2.., J_i$); C_{hk_h}: represents the Profile/context k_h of user profile/context category h ($h = 1, 2..., H$ and $k_h = 1, 2..., K_h$).

The Naive Bayes algorithm is used to calculate Eq. 23, under the Naive Bayes assumption between profile/context sets, where:

$$P(M_{1j_i}|C_{ik_1}, C_{2k_2}, ..., C_{Hk_H}) = P(M_{ij_i}) \prod_{h=1}^{H} P(C_{hk_h}|M_{ij_i}) \tag{24}$$

In Eq. 24, $P(C_{hk_h}|M_{ij_i})$ is calculated using frequency $Freq(X)$, which is defined as the amount of user feedback in context during the experiment period, as indicated in Eq.25.

$$P(C_{hk_h}|M_{ij_i}) = \frac{Freq(M_{ij_i}, C_{hk_h})}{Freq(M_{ij_i})} \tag{25}$$

Artificial Neural Network (ANN). Using ANN, the aim of the work in [48] is to pinpoint which member of a household gave a specific rating to a movie. The solution authors propose relies on the identification of three simple classifiers $c1$, $c2$, $c3$ that are then combined through a machine learning approach. The problem is formulated as follows. Given the tuple $t :< h, s, r, ts >$, representing the rating r, given within the household h to the movie s, and with timestamp ts, the goal is to learn a function g that identifies the user u who performed that rating.

With the assumption that the users belonging to a given household are known, the authors define the following measures, which take into account the parameters of the tuple separately.

Analysis of the distribution of values given by a user (classifier *c1*): The authors aggregate the frequency of user ratings dividing them into five classes:

1. 0-40
2. 41-60
3. 61-80
4. 81-90

5. 91-100

The aim of such analysis is to model each user based on the values of his ratings. Each user is thus represented by one distribution normalized by an index of Z-score normalization that is compared with the given rating.

Analysis of the distribution of times when the rating was given by a user (classifier $c2$): Similarly to the previous case, the input data is processed with the following four groups:

1. morning (7am to 12pm);
2. afternoon (1pm to 5pm);
3. evening (6pm to 10pm);
4. night (11pm to 6am).

This model represents the user habit of giving his ratings at certain times. The values they obtain are then compared with the timestamp ts that appears in the tuples.

Analysis of movies to determine if two users have seen the same movie (classifier $c3$): The authors analyze the users who have seen a movie. Each of those users is compared with the set of Candidate Users using the following Jaccard distance:

$$\frac{\cap(Su, Sv)}{\cup(Su, Sv)} \tag{26}$$

In Eq. 26, where Su and Sv are the set of movies rated by the users u and v, respectively. Subsequently, the three formulations are combined through a stacking algorithm, where a meta-learner based on neural networks derives the best combination of output of the base learners. Details of the architecture of the neural network are shown in Figure 9.

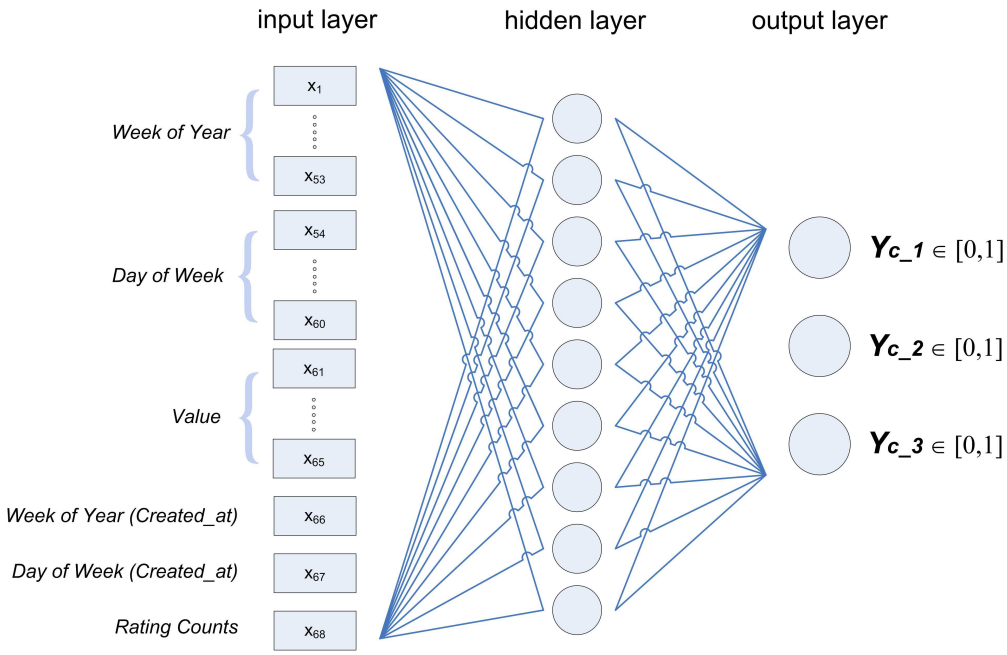

Figure 9: Context-aware neural network

The neural network is trained using the following input parameters (68 features):

1. the distribution of the number of users who rated a movie per week (53 features);

2. the distribution of the number of users who rated a movie per day of the week (7 features);

3. the distribution of ratings given to a movie, divided into five groups (5 features);

4. the submission date of the rating, identified by week of the year (from 1 to 53) and by day of the week (from 1 to 7) (2 features);

5. the number of ratings given to a movie (1 feature).

The output layer of the network consists of three nodes which express a membership value to the three simple classifiers mentioned above.

Finally, the predicted user is chosen according to the following algorithm:

1. Given the training data (t_1, c_1), ..., (t_N, c_N) and the tuple t

2. For i = 1 to 3:

(a) $TS_i \longleftarrow$ a selection of N random examples from the training set

(b) $f_i \longleftarrow$ the result of training base learning algorithm on TS_i

(c) draw the output y_{c1} , y_{c2} , y_{c3} from the neural network

3. Output from the combined classifier: $g(t) = majority(y_{c1} \cdot f_1(t), y_{c2} \cdot f_2(t), y_{c3} \cdot f_3(t))$, where t_i is a tuple and c_i is a 3-bit code word vector, which represents the classification of c_1, c_2, and c_3 for tuple t_i. For example, the vector [0, 1, 1] describes the situation in which the tuple is classified correctly only by the classifiers c_2 and c_3.

The neural network, thus, provides a probability value (i.e., between 0 and 1), which is used as a weight in the majority voting discribed in [69] that is defined by the function $g(t)$.

2.5.6 Discussion

We present now a synthesis concerning the context representation and CARS approaches.

A huge number of works related to different research domains have proposed models to represent context. We have chosen in Table 5 a few of them to illustrate the major existing models.

Table 5: Comparing context representation models

Context model	**Advantage**	**Disadvantage**
Attribute-Value [118]	It is particularly easy to manage and easy to implement	Lack of semantics
Object-Oriented [95]	Correlate relationships between class and object, easy to implement	Difficult to update
Markup [79, 89]	Hierarchical structure	Not flexible
Semantic [14, 71]	Better interpretation of the context semantic	Limited reasoning performance in real-world applications

According to the survey of context modelling [10], ontological models have clear

advantages regarding support for interoperability and heterogeneity as well as reasoning capability. However, limited reasoning performance further reduces the scalability of these approaches in real-world applications.

We present in Table 6 a synthesis of CARS systems, comparing them according to how they acquire the context (Context Acquisition), how they model the context (Context model), and how they recommend information according to the context (Recommendation approach).

Table 6: Comparing CARS

CARS	**Recommendation approach**	**Context model**	**Context Acquisition**
[118]	CF	Attribute-Value	Context detected and derived
[52]	CBF	Markup	Context detected and derived
[5]	Hybrid	Attribute-Value	Context detected
[71]	CBF	Semantic	Context detected and explicitly provided
[70]	CBF	Attribute-Value	Context detected and explicitly provided
[107]	CBF	Attribute-Value	Context detected and explicitly provided
[48]	CBF	Attribute-Value	Context detected and explicitly provided

From Table 6, we can observe that the majority of existing CARS uses the recommendation techniques that exist in the classic RS, like CF, CBF and Hybrid recommendation, where the context is just integrated as a parameter on their equation,

which means that those works did not create a specific recommendation technique for CARS.

However, integrating the context as a parameter may reduce the possibilities to exploit several granularity levels of context data. For example, when the user's profile has an interest associated to GPS coordinates, it is difficult to recommend this interest in the future, since it is improbable to find again the user at exactly the same place in the sense of same GPS coordinates. However, if we can fall over automatically from the physical location of the user to its semantic representation (e.g. place, city, type of location, ...), we can exploit a larger panel of location types and group places of the same type.

We can also observe that the most of the existing CARS uses the Attribute-Value context model based on an algorithm of exact matching on these attributes. This can be explained by the simplicity of its implementation. The main critics of this approach is its limited capabilities in supporting reasoning on context, on higher context abstractions.

We see that major existing works use a method for context acquisition, which is actually explained by the high technological development of the mobile devices.

Finally, from the evaluation done in [52], where the authors have compared machine learning algorithms in CARS, namely Case-based Reasoning, Neural Network, Naive Bayesian and Decision Tree, the CBR gives the best results and it has been explained by the dynamicity of this algorithm.

We have described, in this section, how existing CARS manage the user's context in order to make an accurate recommendation.

Another criterion that must be considered is to avoid intrusive recommendations according to user situation. In what follows, we study the risk of upsetting the user and the different existing approaches to compute it.

2.6 Risk Aware Decision

The majority of existing approaches to RS focus on recommending the most relevant documents to the users using the contextual information and do not take into account the risk of disturbing the user in specific situations. However, in many applications, such as recommending a personalized content, it is also important to incorporate the risk of upsetting the user into the recommendation process in order not to recommend documents to users in certain circumstances, for instance, during a professional meeting, early morning, late-night. Therefore, the performance of the RS depends on the degree to which it has incorporated the risk into the recommendation process.
In this section, we define the notion of risk as well as how it can be calculated.

2.6.1 Risk Definition

The norm ISO 31000 (2009) defines the risk as the "effect of uncertainty on objectives". In this definition, uncertainties include events which may happen or not.
Many definitions of risk exist in common usage, however in this thesis we consider the definitions in domains close to RS. Therefore, we take the following two definitions from the reinforcement learning and the financial domain, respectively:

Reinforcement learning. "The risk is the reward criteria which takes not only into account the expected reward, but also some additional statistics of the total reward such as its variance, its value at risk, etc" [108].

Finance. "The risk is the probability that the investment return will be different than expected, which leads to the possibility of losing some or all of the original investment" [37].

Using both definitions of the risk, in what follows we describe the different techniques that exist to measure it.

2.6.2 Measuring the Risk Value

Some existing works take into account the risk in different application areas, like reinforcement learning and robotics. In these works the risk is measured with two types of uncertainty: parametric and inherent. Some works also use an hybrid of both types.

2.6.2.1 The Variance of the Cost

The variance of the cost approach is related to the imperfect knowledge of the problem parameters. For instance, in the context of Markovien Decision Process (MDPs) and addressing inherent uncertainty, Howard and Matheson [80] have proposed to use an exponential utility function, where the parameter of the exponent controls the risk sensitivity.

Another approach considers the percentile performance criterion [1], in which the average environment's reward and its variance lead to decide the best action to select using the following Eq:

$$z = argmax_z(E[z] - k * var[z]) \tag{27}$$

In Eq. 27 z is a random variable for profit and k is a constant referred to as risk aversion factor, and it indicates excessive deviation from the expected values.

The main advantage of variance of cost approach is the ability for computing the risk without the need of an expert, however this leads to a cold start at the beginning of the process.

2.6.2.2 The Expected Environment Cost

The expected environment cost approach is related to the stochastic nature of the system. For example, Geibel and Wysotzki [55] have considered an MDPs model where some states have been error states. They define the risk as the probability of entering such a state when each policy is followed. Then, they try to find good

policies with a risk smaller than some threshold predefined by the user. This problem is formalized as a constrained MDPs with a risk function based on a cumulative return. The authors present a reinforcement learning algorithm that aims at finding good deterministic policies.

Another work presented in [122] classifies such situations or states as nominal or adverse. Nominal states are not dangerous; in the inverse case, adverse situations are dangerous and can be specified or unexpected (see Figure 10).

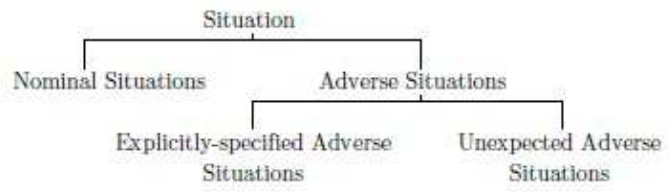

Figure 10: Classification of situations proposed by [122]

The authors of [60] have shown that step-wise exploration by a pre-specified backup policy is safe. The step-wise exploration proposed is based on the assumption about the world that neighboured states of safe states are safer than others.

The expected environment cost approach allows to avoid the cold start using an expert to predefine the dangerous state, however this approach is unable to detect new dangerous states.

2.6.2.3 Hybrid Approach

The hybrid approach is a combination of both the expected environment cost and the variance of the cost [41].

In [41], the authors develop a policy gradient algorithm for criteria that involve both the expected cost and the variance of the cost. The authors prove the convergence of these algorithms to local minima and demonstrate their applicability in a portfolio planning problem.

2.6.3 Discussion

We present now a synthesis of the approaches to compute the risk.

These approaches are grouped in Table 7 according to the models of measuring the risk and their advantages /disadvantages. From Table 7, we can observe that the hybrid

Table 7: The models of measuring the risk

Techniques	**Advantage**	**Disadvantage**
The variance of the cost [80, 1]	No user dependent	Cold start problem
The expected environment cost [60, 122, 55]	No cold start problem	These strategies often depend on manually tuning
Hybrid [41]	Overcome the limitations of the precedent approaches	Lack of semantics

approaches have been developed by combining the two latest techniques; so that, the inability of the "expected environment cost approach" to detect new dangerous states (cold start) is reduced by "the variance of the cost approach" [41]. However, hybrid approaches do not consider neither the similarity between states nor a semantics description of the states, where describing states using concepts and computing the similarity between them using ontologies can contribute on improving the detection of similar danger states.

We describe, in this section, the impact of the risk in the decision process in uncertain environments of the real world, and how different techniques compute this risk.

In what follows, we try to analyse the different criteria that we have studied to get an accurate RS.

2.7 Synthesis

Table 8 classifies the existing works mentioned along this chapter, according to our requirements sketched in the introduction (Chapter 2), which are:

Follows the Dynamicity of User's Content (FDUC): This requirement is

reflected in the difficulty of generating recommendations when old documents expire and new documents emerge continuously. Two approaches have tackled this problem. The techniques based on CBF ([38, 86] have the advantage of exploiting the user history in the recommendation, but they have a lack of diversity of content in recommendations. Exploration-exploitation trade-off approaches ([73, 72]) have allowed recommending varied documents to the user, but the risk is to upset the user with exploratory recommendation.

Context-Aware (CA): This requirement guarantees that the context is considered during the recommendation process. Indeed, a suitable recommendation depends on how well the context is modelled. Recently, CARS use different approaches to model the context. Attribute-Value model used in [54, 40, 118, 5] is particularly easy to manage and easy to implement but it lacks of semantic. The Object-Oriented model used in [12, 95] allows a correlate relationships between class and object, easy to implement but difficult to update. Markup model considered in [79, 89] have a hierarchical structure but is not flexible. Semantic model used in [14, 71] have a better interpretation of the semantic of the context but they do not consider the heterogeneity and the diversity of the context.

Risk-Aware (RA): This requirement expresses the ability to be aware of the risk level of the user's situation during the recommendation process in order to not recommend documents in risky situations for example. Three techniques are proposed to compute the risk. The "variance of the cost" approaches ([80, 1]) have the advantage to be not user dependent but they still have the deal with the cold start problem. The "expected environment cost" approaches [60, 122, 55] have the disadvantage to depend on manually tuning techniques. The hybrid approaches ([41]) have been developed by combining the two latest techniques so that, the inability of the "expected environment cost approach" to detect new dangerous states (cold start) is reduced by "the variance of the cost approach" [41], but they suffer from the lack of

semantic.

Recommender Systems (RS): This latest requirement is simply to check that the studied works are RS or not.

2.8 Conclusion

We explore, in this chapter, the various requirements for an accurate RS.

We observe that the most appropriate representations of the user's profile and context are respectively the multidimensional representation, and the ontology representation.

We also observe that the risk is never considered by existing RS, and that the most interesting approach to follow the dynamicity of the user's content is through a trade-off among the exploration-exploration and CBF.

In the next chapter, we discuss the main approach used, together with CBF to follow the dynamicity of the user's content: the exploration/exploitation trade-off (as stated in Section 2.4).

Table 8: Comparing the state of the art works with our requirements

References	RS	CA	FDUC	RA
[43]	+			
[44]	+			
[76]	+			
[86]	+			
[104]	+			
[92]	+			
[68]	+			
[101]	+			
[64]	+			
[13]	+			
[35]	+			
[9]	+			
[99]	+			
[91]	+			
[124]	+			
[123]	+			
[85]	+			
[95]		+		
[79]		+		
[89]		+		
[14]		+		
[71]	+	+		
[5]	+	+		
[52]	+	+		
[118]	+	+		
[70]	+	+		
[107]	+	+		
[48]	+	+		
[11]	+		+	
[75]	+		+	
[4]	+		+	
[72]	+		+	
[73]	+		+	
[80]				+
[1]				+
[55]				+
[122]				+
[60]				+
[41]				+

Chapter III

MULTI-ARMED-BANDIT PROBLEM

3.1 Introduction

According to Section 2.4, the main solution to prevent the problem of the dynamicity of the user's content is to consider the exploration-exploitation trade-off in the user's preferences learning. This trade-off is managed through the bandit problem. The present chapter is dedicated to explain this problem.

The chapter is organised as follows. Section 3.3 outlines the typical problems of multi-armed bandit. Then, Section 3.4, describes in more detail strategies that can be used for the multi-armed bandit problem. Section 3.5 describes how the multi-armed bandit algorithm that can be used in RS and risk aware decision. We conclude with an analysis which offers an adequate solution to the requirements in terms of recommendation (Section 3.7).

3.2 Definition of Multi-Armed Bandit Problem

The Multi-Armed Bandit (MAB) or k-armed bandit problem is a sequential decision making process studied in the fields of statistics [56], machine learning [7], among others.

The problem is originally described by [98] as a process, where the system must select one of several arms or resources. For each selected arm, a reward is received. The objective is to find a selection strategy that maximises the cumulative reward. In the scope of RS, an arm is a resource or a document from the collection and a reward corresponds to the satisfaction of the user's interest w. r. t. the document. As explained in Section 2.2, this satisfaction can be explicitly expressed or implicitly

inferred (e.g., from the number of clicks on the document). The objective of the recommendation is to maximise the cumulative satisfaction of the user by recommending him interesting documents. RS based on MAB problem commonly assumes to have no prior knowledge about the reward of each document and thus should explore rewards from different documents in an effective strategy. The best strategies are those that incorporate the need to balance exploration (pulling different documents to identify the best one) and exploitation (pulling the expected best document to maximise the reward).

MAB problems have also applications in areas as diverse as clinical drug trials [116], web advertising [73] and many other decision-making problems [106].

3.3 Multi-Armed Bandit Variations

In the MAB problem, the system pulls resource i at time t and receives a reward $r_i(t)$ from that resource. The objective is to find a strategy that maximizes the sum of the received rewards after time T.

This problem has been studied both in finite time T [7, 111] and also for infinite time T [67].

Furthermore, many different problems have been derived from the classical MAB problem, as, for example, the following ones.

The stochastic multi-armed bandit problem considers that each resource i has reward $r_i(t)$ at time t generated from a probability distribution R_i. The system does not have prior knowledge of these distributions and the distributions are fixed over time [106].

The adversarial multi-armed bandit problem has been studied in [7], where the rewards of each resource have been set a priori by an adversary. The rewards generated by the adversary are bounded in [0, 1].

The objective of the adversarial framework is to identify the best resource to repeatedly play for all iterations, rather than finding the best resource for each individual iteration. This is a restrictive assumption in realistic scenarios in which the optimal resource to select can change between iterations.

The one-armed bandit problem is a special case of the MAB problem. The system must choose between a resource with unknown expected reward and a resource with known expected reward. The problem, in this form, has been first studied in [46, 57] for sequential clinical trials, where a treatment has to be chosen between a drug with known probability of success and a new drug with unknown probability of success.

The MAB problem with covariates first introduced in [116], considers the scenario where the system observes context information (like number of click, time spent) prior to each resource. In this problem, the expected reward of each resource is a function of this context information represented in the form of a covariance.

It has been argued in [116] that such context information exists in many applications and incorporating this into bandit problems is a more realistic representation of real-world problems. For example, covariate information such as age, sex and weight could influence the probability of success of a food recommendation [74].

All of these variants to the MAB problem need to be solved by trying to make the best strategy in exploration-exploitation. In what follows we describe these strategies (Bandit algorithms) and we try to find out the most interesting algorithm adapted to our needs.

3.4 Bandit Algorithms Overview

In this section we describe the different bandit algorithms that try to solve the MAB problem.

3.4.1 ϵ-Greedy

ϵ-greedy is the most widely used algorithm to solve the MAB problem and has been first described by Watkins [7, 51, 88, 73, 72]. The ϵ-greedy approach consists in choosing a random resource with ϵ-frequency, and otherwise choosing the resource with the highest estimated mean, the estimation being based on the rewards observed. $\epsilon \in [0, 1]$ is predefined by the user.

The simplest variant of ϵ-greedy is the ϵ-beginning algorithm. This strategy consists in doing the exploration all at once at the beginning. For a given number $I \in N$ of iterations, the resources are randomly selected during the $\epsilon * I$ first iterations. During the remaining $(1 - \epsilon) * I$ iterations, the resource of highest estimated mean is selected.

Another variant of the ϵ-greedy is the ϵ-decreasing. The ϵ-decreasing consists in using a decreasing ϵ_i, where i is the index of the current iteration. The value of the decreasing ϵ_i is given by $\epsilon_i = \epsilon_0/i$ where $\epsilon_0 > 0$. The choice of ϵ_0 is given by the user.

3.4.2 SoftMax

The SoftMax algorithm consists of a random resource selection according to a probability distribution. The resource k is chosen with probability $p_k = e^{u_k/t} / \sum_{i=1}^{n} e^{u_i/t}$ where u_i is the mean of the rewards brought by resource i and $t \in R+$ is a parameter named temperature. The choice of the value t is given by the user.

SoftMax has been proposed in [77]. More generally, all methods that choose resources according to a probability distribution, are called probability matching methods.

The SoftMax algorithm has the same variant algorithms as the ϵ-greedy, like the decreasing-SoftMax, where the temperature decreases, or the beginning-SoftMax, where the temperature is very high at the first iterations and becomes zero in the remaining iterations.

3.4.3 Exp3

A variant of the SoftMax algorithm, the Exp3 "exponential weight algorithm for exploration and exploitation", is introduced in [6].

The probability of choosing resource k at the iteration of index t is defined by

$$p_k = \frac{(1-y) * (w_k(t))}{(\sum_{j=1}^{K} w_j(t))} + \frac{y}{K} \tag{28}$$

In Eq. 28, $w_j(t+1) = w_j(t) * e^{y(r_i(t)/p_j(t)K)}$ if the resource j has been selected at time t with $r_j(t)$ being the observed reward, $w_j(t+1) = w_j(t)$ otherwise.

The choice of the value $y \in [0,1]$ is done by the user. The main idea is to divide the current reward $r_j(t)$ by the probability $p_j(t)$ that the resource has been chosen.

3.4.4 Pursuit

Pursuit algorithms [94] start with uniform probabilities assigned to each resource d_i, $p_i(0) = 1/N$. Where N is the number of resources. At each iteration t, the probabilities are recomputed as follows:

$$p_i(t+1) = \begin{cases} p_i(t) + lr(1 - p_i(t)) & if \quad q > \epsilon \\ p_i(t) + lr(0 - p_i(t)) & otherwise \end{cases} \tag{29}$$

In Eq. 29, $lr \in [0,1]$ is a learning rate.

3.4.5 Upper Confidence Bound

The Upper Confident Bound (UCB) assigns to each resource an optimistic reward estimate with a certain confidence interval and to greedily choose the resource with the highest optimistic mean [7]. Unobserved resources will have an over-valued reward mean that will lead to further exploration of those resources.

Initially, each resource is selected once. Afterwards, at iteration t, the algorithm greedily picks the resource $d(t)$ as follows:

$$d(t) = argmax_{i=1....N}(u_i + \sqrt{\frac{log(t)}{n_i}}) \tag{30}$$

In Eq. 30, n_i is the number of times that the resource i is selected, u_i is its current average reward and N the number of resources.

Many variants of UCB have been proposed in the generalized model of MDPs. 30 different algorithms are discussed in [84]. But, in the simpler situation, all these variants are equivalent to UCB.

3.4.6 Gittins

The authors in [56] have shown that the optimal resource could be selected using the Gittins indices by considering the future reward distributions of each resource independently.

Specifically, one Gittins index is assigned to each resource, corresponding to the expected reward of staying on a resource for an optimal length of time.

The Gittins rule is then to pull the resource with the highest index value V given by Eq. 31.

$$V(n,m) = max\{p/(1\gamma), (n/n+m)*[1+\gamma*V(n+1,m)]+(m/n+m)*\gamma*V(n,m+1)\} \tag{31}$$

In Eq. 31, n (resp. m) denotes the number of times a positive reward of 1 has been observed in previous iterations after pulling resource (resp. reward of 0), where γ is a discount factor and p is the probability of choosing the resource.

This method significantly reduces the complexity of the computation and the optimality of Gittins indices has been proved in [63, 113, 62, 110].

3.4.7 Discussion

From the empirical evaluation done by [111], in the case where the resources' reward distributions are normally distributed, simple algorithms with no theoretical guarantees, such as ϵ-greedy, significantly outperform more complicated algorithms, such as Exp3 or UCB. But the ranking of the algorithms change significantly when switching to real-world data. Pricing methods such as UCB outperform simple algorithms in

the case of the networking data examined.

Bandit algorithms are used in different domains, as we see in Section 1, and, nowadays, the tend is to apply them in RS to follow the dynamicity of the user's content. In what follows, we discuss the different approaches that have been done in this sense.

3.5 Bandit Algorithm for Recommender Systems

In this section, we describe the algorithms that consider the exploration-exploitation trade-off in RS.

Compared to the standard MAB problem with a fixed set of possible actions, in RS, the old documents may expire and new documents may frequently emerge.

In this setting, the algorithm needs to explore continuously new documents which may not be desirable to perform the exploration all at once at the beginning, as the begining strategy in [51], or to decrease monotonically the effort on exploration, as the decreasing strategy in [88].

Few research works are dedicated to study the MAB in RS, where they consider the user's behaviour as the context of the bandit problem.

3.5.1 EG-greedy

In [73], authors extend the ϵ-greedy strategy by updating the exploration value ϵ dynamically. In each iteration, they run a sampling procedure to select a new ϵ from a finite set of candidates.

Probabilities associated to the candidates are uniformly initialized and updated with the Exponentiated Gradient (EG) [65]. This updating rule increases the probability of a candidate ϵ if it leads to a user's click. Compared to both ϵ-beginning and ϵ-decreasing strategy, this technique improves the results [73]. First they assume that they have a finite number of candidate values for ϵ, denoted by (ϵ_1, ..., ϵ_T), and they try to learn the optimal ϵ from this set. To this end, they introduce $p = (p_1, ..., p_T)$, where p_i stands for the probability of using ϵ_i in the ϵ-greedy algorithm.

These probabilities are initialized to be $\frac{1}{T}$ at the beginning and then iteratively updated through iterations. They use a set of weights $w = (w_1, ..., w_T)$ to keep track of the performance of each ϵ_i and update them using the EG algorithm. The idea is to increase w_i if the algorithm receives a click from using ϵ_i. Finally, the algorithm calculates p by normalizing w with smoothing. Algorithm 1 shows the ϵ-greedy algorithm with the EG update.

Algorithm 1 EG-greedy

Input: $(\epsilon_1, ..., \epsilon_T)$: candidate values for ϵ

β, τ and k: parameters for EG

N: number of iterations

$p_k \Leftarrow \frac{1}{T}$ and $\mathrm{w}_k \Leftarrow 1$, $k = 1, ..., T$

for i=1 **to** N **do**

Sample d from discrete $(p_1; ...; p_T)$

Run the ϵ-greedy with ϵ_d

Receive a click feedback c_i from the user

$w_k \Leftarrow w_k \exp(\frac{\tau[c_i I(k = d) + \beta]}{p_k})$, $k = 1, ..., T$

$p_k \Leftarrow (1 - k)(w_k/(\sum_{j=1}^{T} w_j) + k/T)$, $k = 1, ..., T$

end for

In Algorithm 1, $I[z]$ is the indicator function and τ and β are smoothing factors in weights updating. k is a regularization factor to handle singular w_i.

3.5.2 LINUCB

In [72], authors model the recommendation as a bandit problem with covariance. Assuming that the expected pay-off of a document is linear in its features (user's clicks, time spent), the authors propose an approach in which a learning algorithm selects sequentially documents to serve users based on features information about the users and the documents.

To maximize the total number of user's clicks, this work proposes the LINUCB algorithm. The pseudo-code of LINUCB is sketched in Algorithm 2.

Algorithm 2 LINUCB

Input: $\alpha \in R+$, A_t

for t=1 **to** T **do**

Observe features of all documents $a \in A_t : x_{t,a} \in R^d$

for all $a \in A_t$ **do**

if a is new **then**

$\mathbf{A}_a \leftarrow I_d$ (d-dimensional identity matrix)

$b_a \leftarrow 0_{d*1}$ (d-dimensional zero vector)

end if

$\Theta_a \leftarrow \mathbf{A}_a^{-1} * b_a$

$p_{t,a} \leftarrow \Theta_a^\top x_{t,a} + \alpha\sqrt{x_{t,a}^\top \mathbf{A}_a^{-1} x_{t,a}}$

end for

Choose document $a_t = argmax_{a \in A_t} p_{t,a}$ with ties broken arbitrarily, and observe a real-valued payoff r_t

$\mathbf{A}_{a_t} \leftarrow \mathbf{A}_{a_t} + x_{t,a_t} x_{t,a_t}^\top$

$b_{a_t} \leftarrow b_{a_t} + r_t x_{t,a_t}$

end for

In Algorithm 2, A_t is the set of documents at iteration t, where $x_{a,t}$ is the feature vector of document a with d-dimension, Θ_a is the unknown coefficient vector of the feature $x_{a,t}$, α is a constant and $\mathbf{A}_a = D_a^\top D_a + I_d$.

D_a is a design matrix of dimension $m \times d$ at trial t, whose rows correspond to m training inputs (e.g., m contexts that are observed previously for document a), and $b_a \in R^m$ is the corresponding response vector (e.g., the corresponding m click/no-click user feedback). Applying ridge regression to the training data (D_a, c_a) gives an estimate of the coefficients: $\Theta_a = (D_a^\top D_a + I_d)^{-1} D_a^\top c_a$, where I_d is the $d \times d$ identity

matrix and c_a are independent conditioned by corresponding rows in D_a.

3.5.3 Discussion

The authors in [73, 72] describe a smart way to solve the MAB problem in RS. However, none of them considers neither the context of the user nor the risk of the user's situation during the recommendation.

3.6 Bandit Algorithm for Risk-Aware Decision

In this section we describe the algorithm that tries to manage the risk of the situations in the exploration-exploitation trade-off.

A recent work [109] has treated the risk problem in the context of Markovien Decision Process (MDPs), where the authors propose an algorithm named Value-Difference Based Exploration (VDBE), to extend ϵ-greedy by introducing a state-dependent exploration probability, rather than a hand-tuning of the parameter ϵ.

On one hand, the system makes more exploration in situations when the knowledge about the environment is uncertain (low information about the environment), which can be recognized by large changes in the error value function.

On the other hand, the exploration rate is reduced as the system's knowledge becomes certain about the environment, which can be recognized by very small changes in the error value function.

The following equation adapts such desired behaviour:

$$\epsilon_{t+1}(s) = \gamma * f(s_t, a_t, \sigma) + (1 - \gamma) * \epsilon_t(s) \tag{32}$$

In Eq. 32, $f(s, a, \sigma) = 1 - e^{-|Q_t(s,a) - Q_{t-1}(s,a)|/\sigma}$, where $Q_t(s, a)$ describes the quality of action a in state s at time t, $\gamma \in [0, 1]$ is a parameter determining the effect of the selected action on the exploration rate, σ is a positive constant called inverse sensitivity, In this work, authors have shown that low inverse sensitivities cause full exploration even at small value changes. On an other side, high inverse sensitivities

cause a high level of exploration only at large value changes.

At the beginning of the learning process, the exploration rate is initialized by $\epsilon_{t=0}(s) = 1$ for all states s.

3.7 Analysis

To analyse the different bandit algorithm techniques, we compare them regarding the predefined requirements (Section 2.1), namely to be adequate to follow the dynamicity the user's content (FDUC), to be Context-Aware (CA), and to be Risk-Aware (RA). We also verify if the algorithm is used in RS or not (RS).

Table 9: Comparing the bandit algorithm with our requirements

Algorithms	FDUC	RS	CA	RA
ϵ-greedy	+			
Sofmax	+			
EXP3	+			
Pursuit	+			
UCB	+			
Gittins	+			
EG-greedy	+	+		
LINUCB	+	+		
VDBE	+			+

From Table 9, we observe that only two algorithms are applied on the RS domain. Moreover, none of the works using the mentioned algorithms considers both risk and context in their exploration.

3.8 Conclusion

In this chapter, we outline the different algorithms that can be used for the MAB problem. We also describe the bandit algorithms that can be used in the RS system and the risk aware decision. We conclude with an analysis that allows us to detect the lack of the existing algorithms, which is the non consideration of the context and the risk in the exploration-exploitation trade-off, particularly in the scope of RS.

Chapter IV

THE DRARS SYSTEM

4.1 Introduction

We present, in this chapter, this thesis' proposal: the "Dynamic Risk Aware Recommender System", named DRARS. This proposal aims to develop a RS that tackles the main challenges raised in this area which are:

1) How to model the user's context and his profile?

2) How to follow the dynamicity of the user's content?

3) How to consider the risk in the RS?

The objective of DRARS is to take into account these tasks and the relationship between them in the recommendation process. We show throughout this chapter how we have modelled the implied concepts and designed DRARS in order to achieve this goal.

This chapter is organized as follows. Section 4.2 presents our issues and motivation for the proposed approaches when building DRARS. In Section 4.3, we model the user's context and profile. Section 4.4 presents an overview of the functional architecture of typical RS. Section 4.5 focuses on the functional architecture of our system, describing its main modules. The last section concludes the chapter.

4.2 Issues and Motivations

The motivations for the various elements of different approaches composing our contribution are based in the conclusions obtained from the state of the art, which we briefly recall in the following.

Modelling the User's Profile and Context

The most appropriate approach to represent the user's profile is by a multidimensional model (Section 2.2) and the best approach to represent the context is through ontologies (Section 2.5). Thus we propose a multidimensional profile and a semantic model to represent the spatio-temporal-social context of the user's navigation activity.

Following the dynamicity of the User's Content

To follow the dynamicity of the user's content, it is adequate to combine the CBF and the exploration-exploitation techniques, as some existing works do (Section 2.4). We propose thus such an hybrid approach for the recommendation process in DRARS (detailed in Section 4.5). Note that the best approach to recommend documents is through an hybrid approach combining CBF and CF (Section 2.3). However, we only use CBF because in our context the information used to recommend is specific to each user (no overlap between users' profiles).

Risk-Awareness

As stated in Chapter 2, no work has addressed the risk of upsetting the user in the recommendation process. In DRARS, concerning this setting, we consider the risk level of the situation when managing the exploration-exploitation trade-off, in-order to help the RS to be adapted with the user's dynamic environment. This strategy achieves high exploration when the current user's situation is not risky and achieves high exploitation in the inverse case.

4.3 Context, Situation and Profile Modelling

Interesting approaches have been proposed in several research fields like Pervasive systems [32, 33, 34] or Context-Based Information Retrieval [14]. We were inspired by all these works to propose user, context and situation models that feet with our

requirements.

4.3.1 The User's Model

We define the user's model as a tuple $UR = (Ctx, Prf)$, where Ctx represents the user's context and Prf represents the user's profile. In what follows, we describe these components.

4.3.2 The User's Profile

From Chapter 2, we observe that the most interesting approach to model the user's profile is through a multidimensional model, this is why we construct a user's profile that contains different dimensions, namely the user's interest, the user's agenda and his personal information.

Formally, the profile $Prf \subseteq Pd \times Ag \times UI$, where Pd is the user's personal data, Ag the user's agenda and UI the user's interests.

Personal Data

Personal data is given by the user in the beginning (before DRARS recommendations). It contains for example the user's profession, his name, his address and his identifier. Formally, the personal data $Pd = \{Pd_1, ..., Pd_n\}$, where n is the number of features.

The Agenda

Before each meeting, the user has to fill-up the agenda with information concerning the person(s) to meet, the time and the location. The agenda is a set of quadruplets $Ag = \{(L^i, T^i, Ir^i, Act^i)\}$, where $L^i \in L$ is the user's location, $T^i \in T$ is the time value, $Ir^i \in Ir$ is the user's interlocutor and $Act^i \in Act$ the user's action.

The User's Interest

The interest of a user is a complex notion. Let us take an example. Paul prefers French food when he goes to the restaurant in France, as usual. When he travels, he

prefers local food.

The condition (when he travels or when he is in France) is related to the situation of the individual.

Paul's interest concerning food in each situation is what we refer to as interest. As we can see from this example, interests are contextual, and might depend on many factors that range from one's own location to a friend's situation.

In a RS, interests are built after navigation activities done by the user; they contain the set of documents of the navigation in the current situation.

A navigation activity expresses the following sequence of events: the user opens the system and navigates to get the desired information; the user expresses his interests on the visited documents. We assume that a document is relevant if there are some observable user's behaviours, like clicking on a document.

The User's Interests (UI) are deduced during the user's navigation activities. They are defined as UI^k=$\{(d^j, cl^j, ts^j, rec^j)\}$ and UI^k where $d^j \in D$ is a document from the collection, $cl^j \in N$ the correspondent click value, $ts^j \in R^+$ the time spent on d^j and $rec^j \in N$ the number of times that d^j is recommended.

4.3.3 The User's Context

The user's context is an adaptation of a previous work in our team [32] and has the following multi-ontology representation: Ctx=($\cup O_{\delta \in \Delta}$), where each ontology O_δ corresponds to a context dimension δ, being Δ the set of all context dimensions. An Ontology O is a quintuple O=*(C, P, R, Ax, I)*, where *C, P, R, Ax* and I are respectively the sets of Concepts (conceptual entities of the domain), Properties of a concept, Relations (relationships between concepts and/or properties), Axioms and Instances. We focus on three dimensions Δ=$\{Location, Time, Social\}$ since they cover all the needed information for our application domain.

4.3.3.1 Location Modelling

There are different ways to characterize a location. As returned by location sensor systems (GPS), the location is a position in systems based on geographic coordinates, or may also be defined by an address.

Simple automated place labelling systems already exist (Google maps, Yahoo local, among others) and consist of merging data such as postal addresses with maps.

In our user's context model, we construct a spatial ontology to represent and reason on geographic information.

To define the location aspects characterizing the situation of the user, we propose to abstract location in some specific regions that we believe have an effect on the user's behaviour. We thus chose to represent the location through region, department and city. Indeed, we have found that the activity of the user can be altered by variation of the location according to each of these dimensions.

Figure 11 shows a sample of the constructed ontology.

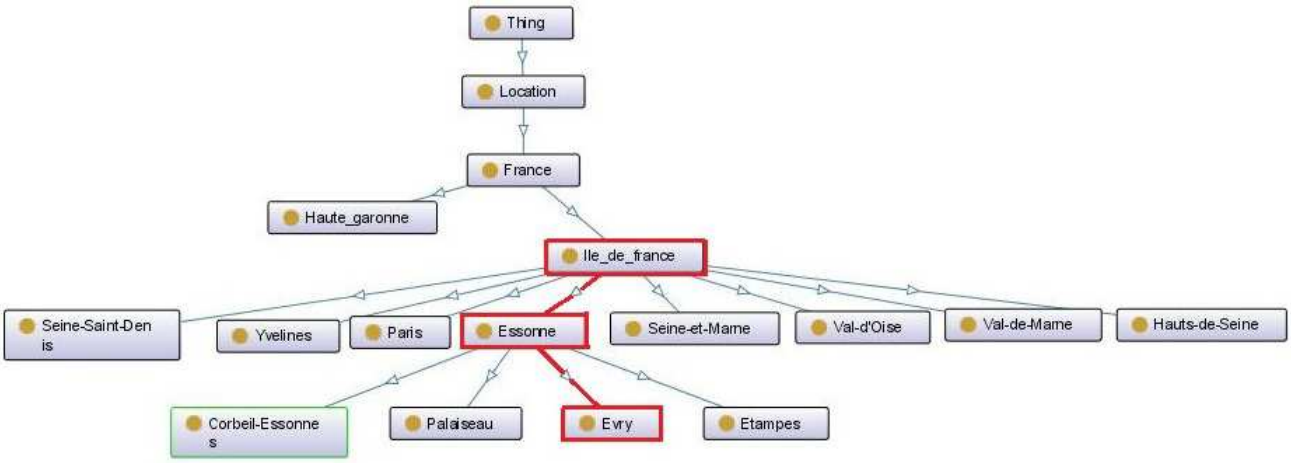

Figure 11: Location ontology

Using the ontology of Figure 11, for example, from the location "38.86, 2.34" returned by a GPS, we get the value Evry, which corresponds to the path (**region:** Ile de france, **department:** Essonne, **city:** Evry) as it is shown in dark grey in Figure 11.

4.3.3.2 Time Modelling

To make a good representation of temporal information and its manipulation, the trend is towards semantic approaches with temporal ontologies. To define the temporal aspects of the user's situation, we propose to abstract time in specific periods through three layers: day, week and month.

We consider the months of the year (e. g., April (A), September (S), among other). The days of the week can be separated in work days (Wd) and holidays (H). A day is related to the month in the current situation of the user. For example, in our notation, "work days (A)" means a workday in April.

Concerning the time of the day, we use the five main periods, namely morning, noon, afternoon, evening and night. These periods are related to a day of the week. For example "afternoon (AWd)" means afternoon of a workday in April. Figure 12 gives a screen-shot of the ontology we propose to represent the time.

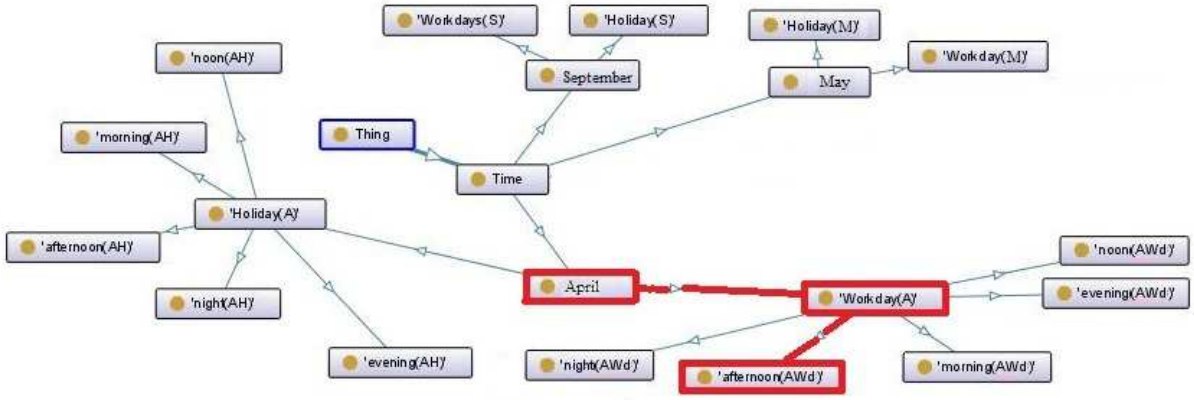

Figure 12: Time ontology

Using the ontology from Figure 12, for example, from the time value "Mon April 3 15:10:00 2011", we get the value "afternoon (AWd)", which allows to get the path (**Month:** April, **Day of the week:** work days (A), **Time of day:** afternoon (AWd)) as it shown in dark grey in Figure 12.

4.3.3.3 Social Modelling

The social dimension of context is in our case the user's interlocutors extracted from the user's agenda, and it is modelled according to our application host which is addressed to a commercial trader. The most important information about the user on commercial trading is the set of companies which he prospects. This is why we have constructed ontologies for all companies, which gives the classification hierarchy about the interlocutor's companies (e. g. a finance client, a bank, etc).

Figure 4.3.3.3 gives a screen-shot of an ontology representing a company. For example, suppose the interlocutor "Paul Gerard" works at "Arval". The information about this company leads, in the ontology, to the path (**Subsidiary name:** Arval, **Name of the company:** BNP, **Type of finance-company:** Banks, **Type of company:** Finance-company) as it is shown in dark grey in Figure 4.3.3.3.

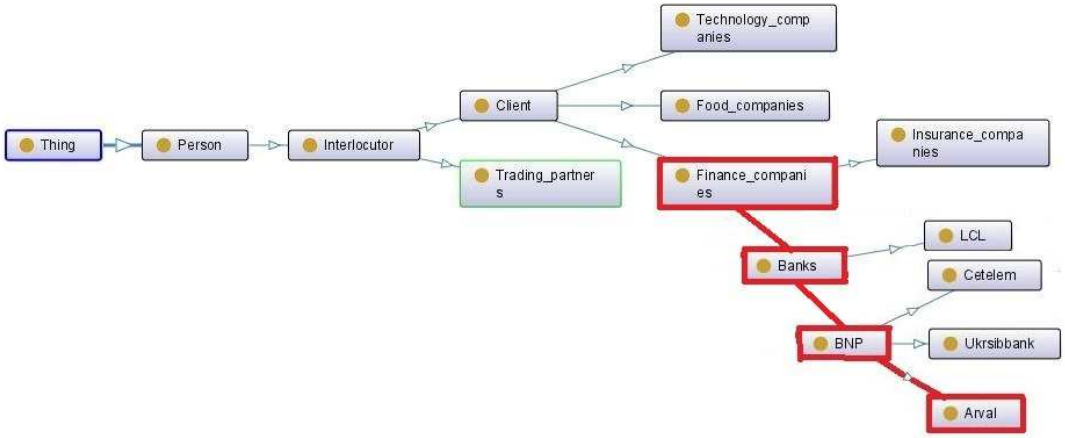

Figure 13: Social ontology

4.3.4 The User's Situation

We adopt the definition proposed by our team [32]: Situations are external semantic interpretations of low-level context, permitting a higher-level specification of human behaviour. In other words, it is a projection on the multidimensional context space. We choose to use the vector model for aggregating location, time and social as user's

situation, where the user's situation is represented by a point in the space defined by those three dimensions, as shown in Figure 14.

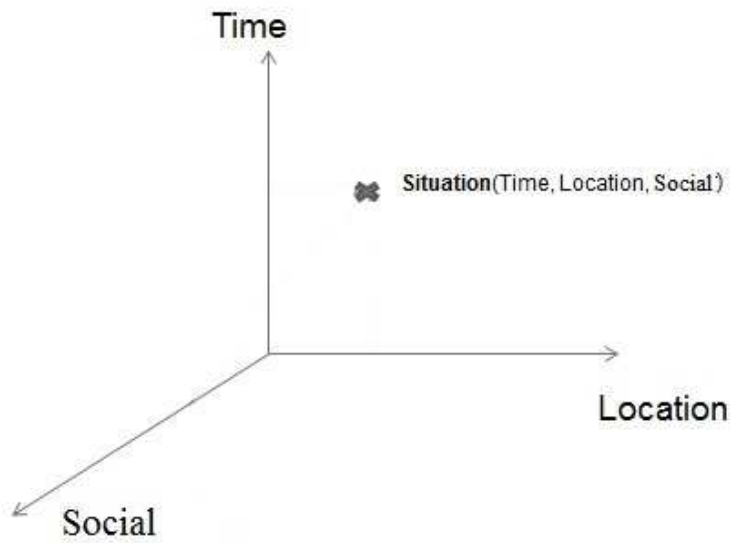

Figure 14: The dimensions aggregated in the situation model

More formally, a situation S is an instantiation of the user's context:
$S = (O_{\delta_1}.c_1, O_{\delta_2}.c_2, ..., O_{\delta_n}.c_n)$ where $O_{\delta_1}, O_{\delta_2}, ..., O_{\delta_n}$ are the ontologies modelling the context (Section 4.3.3), $c_1, c_2, ..., c_n \in C$ and n is the number of context dimensions. According to our need, we consider a situation as a triple $S = (O_{Location}.c_i, O_{Time}.c_j, O_{Social}.c_k)$ where $c_i, c_j, c_k \in C$.

As an example, suppose that the following data are sensed from the user's mobile phone: the GPS shows the latitude and longitude of a point "48.89, 2.23"; the local time is "$Oct_3/12 : 10/2012$" and the agenda states that the user has a meeting with "Mr. Smith". The corresponding situation is: $S = (\text{“}48.89, 2.23\text{”}, \text{“}Oct_3_12 : 10_2012\text{”}, \text{“}Mr.Smith\text{”})$.

Among the set of captured situations, some of them are characterized as Critical Situations.

Risk Modelling

We can notice from Chapter 2 that no work in RS has studied the risk associated to the user's situation.

Since we focus on RS, and since the general concept of risk is very wide, we define the risk in RS, which is not yet done.

Definition:

"The risk in recommender systems is the possibility to disturb or to upset the user which leads to a bad answer of the user".

From the precedent definition of the risk, we have proposed to consider in our system Critical Situations (CS) which is a set of situations where the user needs the best information that can be recommended by the system, because he can not be disturbed. This is the case, for instance, of a professional meeting. In such a situation, the system must exclusively perform exploitation rather than exploration-oriented learning. In other cases where the risk of the situation is less important (like for example when the user is using his information system at home, or he is on vacation with friends), the system can make some exploration by recommending information without taking into account his interest.

To consider the risk level of the situation in RS, we go further in the definition of situation by adding it a risk level R, as well as one to each concept:

$S[R]$=$(O_{\delta_1}.c_1[cv_1], O_{\delta_2}.c_2[cv_2], ..., O_{\delta_n}.c_n[cv_n])$ where CV=$\{cv_1, cv_2, ..., cv_n\}$ is the set of risk levels assigned to concepts and it is discussed in Section 4.5.6.3, $cv_i \in [0, 1]$. $R \in [0, 1]$ is the risk level of situation S, and the set of situations with $R > th_R$ are considered as critical situations (CS). th_R is the risk threshold described in Section 4.5.

In the following, we describe the different modules of the system DRARS.

4.4 The Architecture of DRARS

The general architecture of DRARS is illustrated in Figure 15. This architecture is based on modelling the user through his context and his profile.

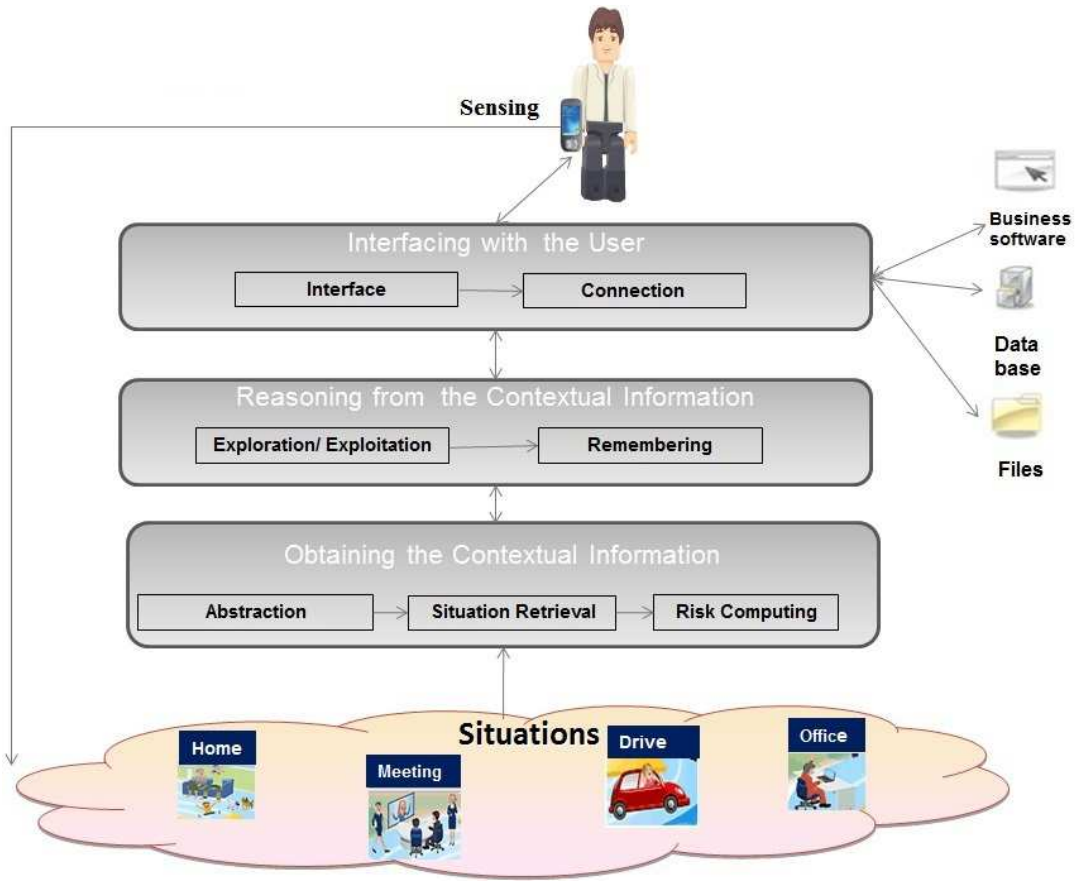

Figure 15: The architecture of DRARS

In this architecture, we identify the following main components:

1- Sensing: permits DRARS to extract the user's context; it is performed inside the mobile phone.

2- Obtaining the Contextual Information: this module permits to extract information from the context such as the risk of the situation.

3- Reasoning from the Contextual Information: this component of the system uses

the context to select the most interesting information to recommend and also update the memory of the system with a new experience.

4- Interfacing with the User: permits DRARS to interact with the phone and with data (data is heterogeneous and can be a database, a file, or a software).

In what follows, we describe the different components of these modules.

4.5 The Functional Description of DRARS

In this section, we describe the functional architecture of the proposed RS. The sensing module is in the mobile phone and the other modules are in the server. The interface is the platform which permits to interact with the phone and persistent data, and manages the different process described in the following sections.

Figure 16 gives the sequence diagram of the different modules of DRARS.

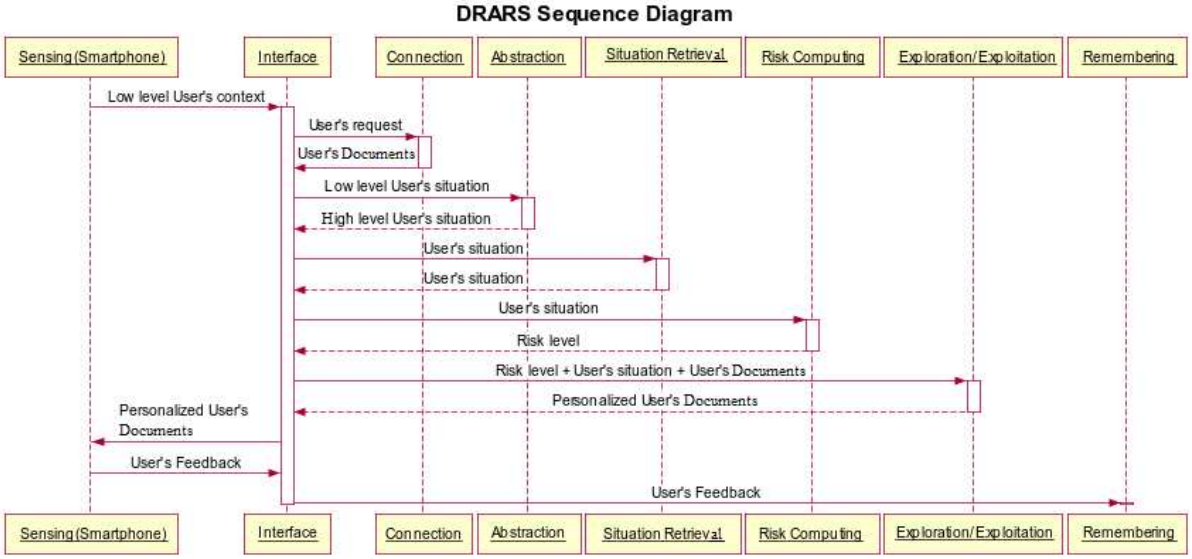

Figure 16: The sequence diagram of the different modules of DRARS

4.5.1 Context Acquisition

The Sensing module allows to obtain the user's request, together with the corresponding contextual information (time and GPS). This information is stored in an XML document, still inside the user's mobile phone, and sent to the interface module on the server.

Our system uses two approaches to get the contextual information: context detection and context explicitly provided.

4.5.1.1 Context Detection

The information given by context detection is acquired through two physical sensors: GPS and clocks. A GPS permits to get a very low semantic information, being limited to the geographical coordinates longitude and latitude. A clock permits to get time information in hours/minutes/seconds, also with few semantic information.

4.5.1.2 Context Explicitly Provided

The context is explicitly provided when the user gives explicitly to the system the context information. In our work, this is the case of the agenda, used by the user to select explicitly with whom he is at rendezvous.

4.5.2 Extracting Information from the Information System

The Connection module (connector) of DRARS extracts global information from the information system. This task is done before the recommendation process. The connector is executed in the server and performs the following tasks:

- Connecting to the client database and extracting the data.
- Transforming relational data into hierarchical data.
- Transferring the information to the interface using XML.

4.5.2.1 Connecting to the client database and extracting the data

To extract the information that the user wants to visualise in his mobile phone, the Connector makes a query corresponding to this information on a Web Service exposed by the user's company (This company permits Nomalys application to request its data).

4.5.2.2 Transforming relational data into hierarchical data

To display data from an information system on a mind map, the connector has to generate a hierarchical database (single relation) from a relational database (N relations). Figure 17 gives an example of transforming a relational database on a hierarchical database.

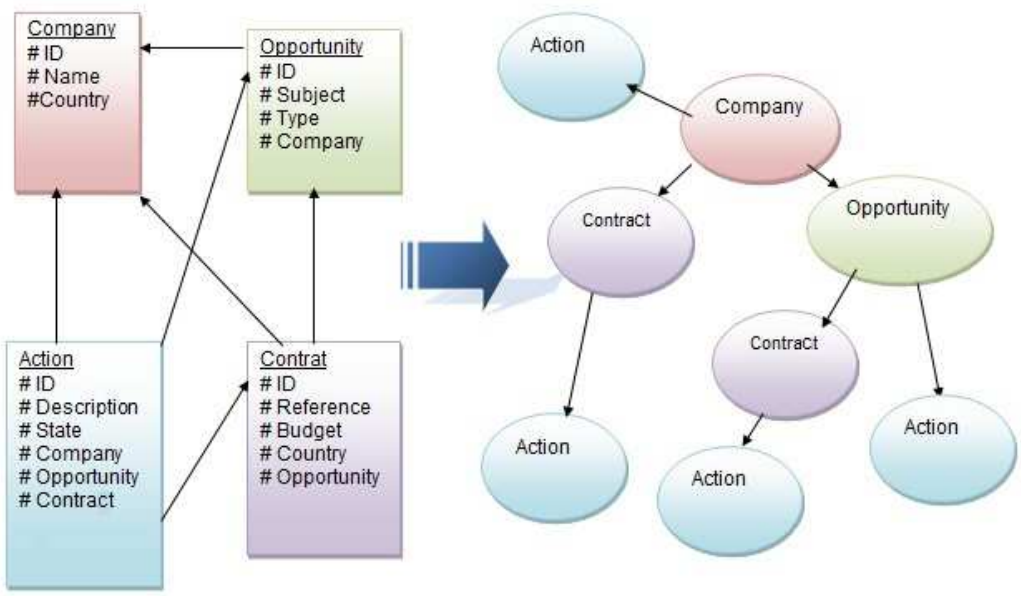

Figure 17: The transformation of a relational database on a hierarchical database

Generally, the Nomalys client database has a schema based on a set of relations, like for example: Company, Action, Opportunity, etc.

These relations are stored in the form of tables in a relational database. Each of these relations has a primary key that uniquely defines each of its tuples (records) and some relations have one or more foreign keys that point to other relations.

Relations that have no foreign key are "root relations". The other relations are called "child relations".

For instance, as indicated in Figure 17, in the client database there are four relations: Company, Opportunity, Contract and Action. Company is a root relation and the other ones are child relations.

The connection module connects to the client database, and, for each relation, it retrieves all records with a foreign key to the root relation "Company".

In a relational database, child relations are related through foreign keys to parent relations, but, in a hierarchical database, parent relations are linked to child relations. In this example, the opportunities are not duplicated because they have only one foreign key ("Company"). However, the Action relation is replicated three times because it has foreign keys simultaneously to the Company, Opportunity and Contract relations.

4.5.2.3 Transferring the information to the interface using XML

The data extracted from the client database is converted into XML (Extensible Markup Language). An XML document is structured in a hierarchical manner: it has always a root node within which nodes are nested one inside the other. This makes XML particularly suitable to represent hierarchical databases.

Figure 18 shows an example of data after processing it in hierarchical data and conversion to XML. We can see that the parent Company relation points to child relations "Opportunity", "Contract" and "Actions".

```
<Entity Company>
<Data Company Number = '0 '>
<Children>
<Entity = 'Opportunity' ID = '0, 1'/>
<Entity = 'Contract' ID = '0, 2' />
< Entity= 'Action' ID = '0, 1, 2, 3, 4'/>
</ Children>

<ID value = '0 '/>
<Value name = 'Nomalys' />
<Country Value = 'France' />
</ Data Corporation>
</ Entity Company>
```

Figure 18: Example of XML document after the transformation phase

After the XML transformation, the system needs to select the data to recommend to the user. To this end, the system has to proceed different actions, starting by abstracting the context.

4.5.3 Abstracting the Contextual Information

The information provided by the sensors needs to be interpreted. The DRARS abstraction module is used to accomplish this task by analysing and transforming raw data into high level formats that are easier for use.

In fact, sensors typically provide technical data that is not suitable for direct use by the application.

In our case, data obtained through GPS (latitude, longitude) are abstracted as a name place (more semantic concept) as described in the Section 4.3.3.

Once the abstraction gives the context with a semantic representation, the DRARS system must extract from its knowledge the most similar situation to the current one. We present in what follows the module of DRARS that tackles this task.

4.5.4 Retrieving the Relevant Situation

To manipulate the user's situations and correspondent interests in DRARS, we have structured both of them in a case-base structure $Cas = \{(S^i, UI^i)\}$, where $S^i \in PS$ and $UI^i \in UI$. Let S^t be the current user's situation, and PS the set of past situations. The system compares S^t with the situations in PS in order to choose the most similar one, S^p:

$$S^p = argmax_{S^i \in PS} sim(S^t, S^i) \tag{33}$$

In Eq. 33, the semantic similarity metric is computed by:

$$sim(S^t, S^i) = \sum_{\delta \in \Delta} \alpha_\delta sim_\delta(c^t_\delta, c^i_\delta) \tag{34}$$

In Eq. 34, sim_δ is the similarity metric related to dimension δ between two concepts c^t_δ and c^i_δ; α_δ is the weight associated to dimension δ and it is set out by using an arithmetic mean as follows:

$$\alpha_\delta = \frac{1}{t-1}(\sum_{k=1}^{t-1} y^k_\delta) \tag{35}$$

In Eq. 35, $y^k_\delta = sim_\delta(c^K_\delta, c^p_\delta)$ at trial $k \in \{1, ..., t-1\}$ from the $t-1$ previous recommendations, where $c^p_\delta \in S^p$. The idea here is to augment the importance of a dimension with the previously corresponding computed similarity values, reflecting the impact of the dimension when computing the most similar situation in Eq.34.

The similarity between two concepts of a dimension δ in an ontological semantics depends on how closely c^t_δ and c^i_δ are related in the corresponding ontology. To compute sim_δ, we have used a path-based measure done in[117] that takes into account the

depth of the concepts in the hierarchy:

$$sim(c1, c2) = 2 * M_3/(M_1 + M_2 + 2 * M_3) \tag{36}$$

In the Eq. 36, M_1 and M_2 are the number of $is-a$ links from $c1$ and $c2$, respectively, to their least common subsumer (LCS), and M_3 is the number of $is-a$ links from the LCS to the root of the ontology.

4.5.5 Computing the Risk Level of the Situation

In a contextual environment, the exploration-exploitation trade-off is directly related to the risk level of the situation, this is why computing the risk level of the situation is indeed indispensable.

As we observe from the state of the art (Section 2.6), the best approach to compute the risk is from a hybrid approach that combines both the variance of the cost and the expected environment cost. We also need to add semantics to compute the risk.

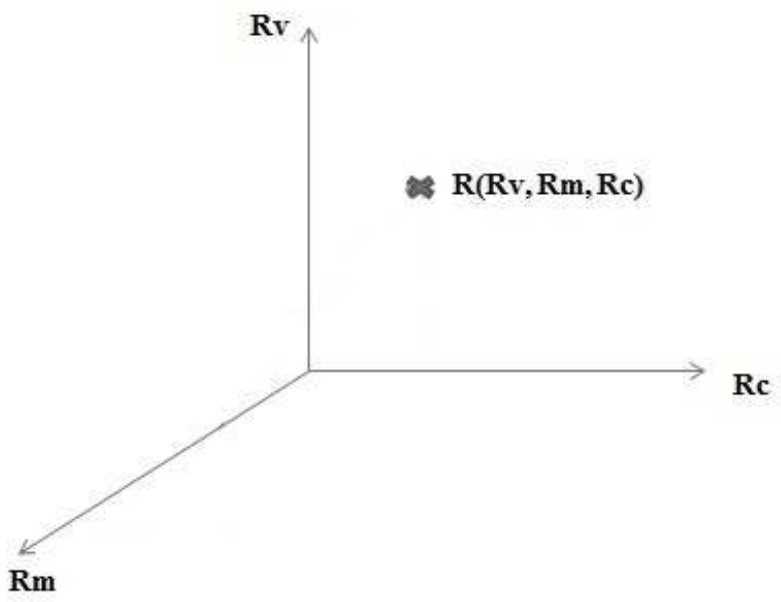

Figure 19: Risk modelling

As it is shown in Figure 19, we have aggregated three approaches for computing the risk. The first one is computing the risk R_c using concepts. This approach permits to get the risk of the situation directly from the risk of each of its concepts. The second approach is computing the risk R_m using the semantic similarity between the current situation and situations stocked in the system. R_m comes from the assumption that similar situations have the same risk level. The third approach is computing the risk R_v using the variance of the reward. In this case, we assume that risky situations get very low number of user's clicks.

In what follows, we describe the three approaches and their aggregation.

4.5.5.1 Risk Computed using the Variance of the Reward

To compute the risk of the situation using the variance of the reward, we suppose that the distribution of the click through rate (CTR) of the situations follows a normal distribution. From this assumption, and according to confidence interval theory [39], we compute the risk using Eq. 37. Here, the idea is that, more the CTR of situations is low (low number of user's clicks) more the situation is risky.

$$R_v(S^p) = \begin{cases} 1 - \frac{ctr(S^p) - Var}{1 - Var} & \text{if} \quad ctr(S^p) > Var \\ 1 & \text{Otherwise} \end{cases} \tag{37}$$

In Eq. 37, the risk threshold Var is computed as follows :

$$Var = E(ctr(S)) - \alpha * \sigma(ctr(S)) \tag{38}$$

In Eq. 38, σ is the variance of $ctr(S)$ and α is constant fixed to 2 according to Gauss theory [39]. The $ctr(S)$ is computed as follows :

$$ctr(S) = \frac{click(S)}{rec(S)} \tag{39}$$

In Eq. 45, $click(S)$ gives the number of times that the user clicks in documents recommended in S and $rec(S)$ gives the number of times that the system has made recommendation in the situation S.

4.5.5.2 Risk Computed using Concepts

Computing the risk using concepts gives a weighted mean of the risk level of the situation concepts:

$$R_c(S^t) = \sum_{\delta \in \Delta} \mu_\delta cv_\delta^t \qquad \text{if} \qquad CV \neq \emptyset \tag{40}$$

In Eq. 40, cv_δ^t is the risk level of dimension δ in S^t and μ_δ is the weight associated to

dimension δ, set out by using an arithmetic mean as follows:

$$\mu_\delta = \frac{1}{|CS|}(\sum_{S^i \in CS} cv_\delta^i) \tag{41}$$

The idea in Eq. 41 is to make the mean of all the risk levels associated to concepts related to the dimension δ in CS.

4.5.5.3 Risk Computed using Semantic Similarity between the Current Situation and Past Situations

The risk may also be computed using the semantic similarity between the current situation and CS stocked in the system. This permits to give the risk of the situation from the assumption that a situation is risky if it is similar to a pre-defined CS. The risk $R_m(S^t)$ obtained this way is computed using Eq. 42

$$R_m(S^t) = \begin{cases} 1 - B + sim(S^t, S^m) & if\ sim(S^t, S^m) < B \\ 1 & otherwise \end{cases} \tag{42}$$

In Eq. 42, the risk is extracted from the degree of similarity between the current situation S_t and the centroid critical situation S^m (Eq. 43). B is the similarity threshold (discussed in Section 6.4). From Eq. 42, we see that the situation risk level $R_m(S^t)$ increases when the similarity between S^t and S^m increases. The critical situation centroid is selected from CS as follows:

$$S^m = argmax_{S^f \in CS} \frac{1}{|CS|} \sum_{S^e \in CS} sim(S^f, S^e) \tag{43}$$

4.5.5.4 Risk Computed Using the Different Risk Approaches

The risk complete level $R(S^t)$ of the current situation is computed by aggregating the R^c, R^v and R^m as follows:

$$R(S^t) = \sum_{j \in J} \lambda_j R_j(S^t) \tag{44}$$

In Eq. 44, R_j is the risk metric related to dimension $j \in J$, where $J = \{m, c, v\}$; λ_j is the weight associated to dimension j and it is set out using a genetic algorithm described in Chapter 6.

4.5.6 Following the dynamicity of the User's Content

After retrieving the set D^p of documents clicked in situation S^p, the system observes rewards of each document $d \in D^p$ in previous trials in order to choose the one with the greatest reward r. This reward is precisely the CTR of a document, and it is computed as follows:

$$ctr(d^j) = \frac{cl^j}{rec^j} \tag{45}$$

In Eq. 45, cl^j and rec^j are integers. To consider the document that are clicked but not yet recommended, $rec^j \in [1, +\infty]$.

As it is described in the state of the art (Section 2.4) the best approach to follow the dynamicity of the user's content is by combining CBF and a bandit algorithm. In this sense, we have used a combination of CBF and the UCB algorithm. We have called this algorithm R-UCB because it is a derivation from the UCB algorithm.

We notice that we have decided to use the UCB algorithm, because it gives the best result in the off-line evaluation described in [73] (Section 3.4).

In what follows, we describe the UCB algorithm used in the DRARS system and how we have adapted it to make it risk-aware.

4.5.6.1 The ϵ-UCB Algorithm

To combine CBF and the UCB algorithm we propose the ϵ-UCB algorithm, which is sketched in Alg. 3. For a given user's situation, the algorithm recommends a predefined number of documents, specified by parameter N. Specifically, in trial t, this algorithm computes the index $ctr(d) + \sqrt{\frac{log(t)}{n_d}}$ (the UCB algorithm Eq. 30) of each document d, being D^p the set of documents clicked in situation S^p, $ctr(d)$ is the mean reward obtained by document d, $\sqrt{\frac{log(t)}{n_d}}$ is its corresponding confidence interval, so that n_d is the number of times that document d was recommended. With the probability 1-ϵ, this algorithm selects the document that achieves a highest upper confidence bound : $d_t = argmax_d ctr(d) + \sqrt{\frac{log(t)}{n_d}}$; and with the probability ϵ, it

uniformly chooses any other documents. The idea here is to introduce the random exploration in UCB algorithm as it is done in [73].

Algorithm 3 ϵ-UCB

Input: $\epsilon, D^p, D^t, RD = \emptyset, N$

Output: RD

for i=1 **to** N **do**

$q = Random(0, 1)$

$$d_i = \begin{cases} argmax_{d \in (D^p - D^i)} ctr(d) + \sqrt{\frac{log(t)}{n_d}} & if\ q > \epsilon \\ Random(D^p - D^i) & otherwise \end{cases}$$

$d_t = CBF(d_i, D^t - RD)$

$D^i = D^i \cup d_i$

$RD = RD \cup d_t$

end for

In Alg. 3, D^i is the set of documents selected by the UCB algorithm, D^t is the set of documents received from the connector, D^p is the set of documents included in the user's interests UI^p corresponding the most similar situation (S^p) to the current one (S^t); RD is the set of documents to recommend; *ctr()* gives the CTR of a document; $Random()$ is the function returning a random element from a given set; q is a random value uniformly distributed over [0, 1] which defines the exploration-exploitation trade-off; ϵ is the probability of recommending a random exploratory document.

The CBF algorithm (Alg . 4) computes the similarity between each document d from D^t (except already recommended documents RD) and the best document d_i, and returns the most similar one. The degree of similarity between d and d^i is determined by using the Jaccard similarity measure, as indicated in Eq. 46:

$$sim(d_i, d) = \frac{|term_{d_i} \cap term_d|}{|term_{d_i} \cup term_d|} \tag{46}$$

In Eq. 46, $term_{d_i}$ gives the set of terms in document d_i. Note that we have used a

Jaccard similarity measure rather than Cosine similarity based weighting of keyword or term, due to low number of terms that exist in each document.

Algorithm 4 CBF

Input: D^t, d_i

Output: d_t

$d_t = argmax_{d \in (D^t)} sim(d_i, d)$

4.5.6.2 The R-UCB Algorithm

To improve the adaptation of the ϵ-UCB algorithm (Alg. 3) to the risk level of the situations, the R-UCB algorithm computes the probability of exploration ϵ, by using the situation risk level $R(S^t)$, as indicated in Eq. 47. A strict exploitation (ϵ=0) leads to a non optimal documents selection strategy, this is why R is multiplied by $(1 - \epsilon_{min})$, where ϵ_{min} is the minimum exploration allowed in CS and ϵ_{max} is the maximum exploration allowed in all situations (these metrics are discussed in Chapter 6).

$$\epsilon = \epsilon_{max} - R(S^t) * (\epsilon_{max} - \epsilon_{min}) \tag{47}$$

Depending on the risk level of the current situation S^t, two scenarios are possible:

(1) If $R(S^t) < th_R$, S^t is not critical; the ϵ-UCB algorithm is used with $\epsilon > \epsilon_{min}$.

(2) If $R(S^t) \geq th_R$, S^t is critical; the ϵ-UCB algorithm is used with ϵ=ϵ_{min} (high exploitation).

We still consider an ϵ_{min} random exploration indispensable to avoid that document selection in CS become less optimal.

The risk threshold th_R is computed as follows:

$$th_R = \sum_{th_j} \lambda_j th_j \tag{48}$$

In Eq. 48, th_j is the risk threshold metric related to dimension $j \in \{m, v\}$; λ_j is the weight associated to dimension j and it is set out using a process described in Chapter 6.

Algorithm 5 The R-UCB algorithm

Input: $S^t, D^t, D^p, RD = \emptyset, B, N, \epsilon_{min}, , \epsilon_{max}, R(S^t)$

Output: RD

$\epsilon = \epsilon_{max} - R(S^t) * (\epsilon_{max} - \epsilon_{min})$ $//R(S^t)$ is computed as described in Eq. 44

if $R < th_R$ **then**

RD=ϵ-UCB(ϵ, D^p, D^t, RD, N)

else

if $R \geq th_R$ **then**

RD=ϵ-UCB(ϵ_{min}, D^p, D^t, RD, N)

end if

end if

To summarize the algorithm R-UCB, the system makes a low exploration when the current user's situation is critical; otherwise, the system performs high exploration. In this case, the degree of exploration decreases when the risk level of the situation increases.

4.5.6.3 Remembering

After receiving the user's reward, the remembering module improves the DRARS document-selection strategy with the new observation: in situation S^t, document d obtains a reward $r(d)$. Depending on the similarity between the current situation S^t and its most similar situation S^p, two scenarios are possible:

-If sim(S^t, S^p) $\neq 1$: the current situation does not exist in the case base; the system adds to the case base the new case composed of the current situation S^t and the current user's interests UI^t.

-If sim(S^t, S^p) $= 1$: the situation exists in the case base; the system updates the case having premise situation S^p with the current user's interests UI^t.

After this process, the system propagates the risk to the concepts of the ontology

using Eq. 49 and propagates the risk in CS using Eq. 50 :

$$\forall cv \in S^t \;\; cv = \frac{1}{|CV_{cv}|}(\sum_{S^i \in CV_{cv}} cv_i^{\eta}) \tag{49}$$

The idea in Eq. 49 is to make the mean of all the risk levels associated to concepts cv related to situations S^i in the user's situation history for the dimension η. In Eq. 49, CV_{cv} gives the set of situations where cv has been computed.

$$R(S^t) = \frac{1}{T}(\sum_{k=1}^{k=T} R(S_k^t)) \tag{50}$$

The idea in Eq. 50 is to make the mean of all the risk levels associated to the situation S^t in the user's situation historic. In Eq. 50, $k \in [0, T]$ gives the number of times that the risk of S^t is computed.

4.6 Conclusion

We have presented in this chapter the system developed under this thesis' project: "Dynamic Risk Aware Recommender System", named DRARS. This project aims to develop a RS that tackles the main challenges raised in this area. We have combined a multi-dimension user's profile and a semantic model of spatio-temporal-social context to manage the user's interest. We have used a combination of CBF and bandit algorithm to follow the dynamicity of the user's contents. We also consider the risk in recommendation by computing the risk level of the situations.

In the following, we describe how the different modules composing the DRARS architecture are implemented in the host application.

Chapter V

IMPLEMENTATION OF DRARS

In this chapter, we give an example of running the DRARS system that summarizes the main specifications of this thesis' project. In addition, we show how the different modules of the system are implemented in the Nomalys application, presenting the various tools used to achieve our goal.

5.1 Host Application of DRARS

Our DRARS system has been implemented to be integrated in the Nomalys application. Nomalys is an application for smartphone and tablets that is currently present in the three mobile operating systems that share most of the market, namely iOS, Android and BlackBerry OS. The application allows people who are often in travel to see the data in their information systems in real time and anywhere. The information is displayed in the form of a mind map (see Figure 22), which goes very well with the constraints of mobile devices, including smartphone and tablets with screens smaller than the office screen. Another particularity of Nomalys application is to connect to multiple types of information systems.

The system developed in this thesis is integrated in the Nomalys application to provide the most relevant information to the user according to his context (considering location, time, and social dimensions), allowing to enrich and improve his information. The application is delivered as Software as a Service (SaaS), which requires no major installation for use and is rather like the client/server model, where the client is the user's smartphone, and the server is the secure server that contains Nomalys.

In what follows, we give a simple scenario of using the DRARS system in the Nomalys application.

5.2 Application Scenario

To illustrate how DRARS is integrated in Nomalys, in this section we describe a simple example of the execution of our system in the Nomalys application.

Suppose company "X" provides to all its traders the Nomalys application. Paul is a sales representative of the company. Regarding Paul's agenda (Figure 20), he has a meeting with an interlocutor of the Nespresso company in Paris on 09/04/2013.

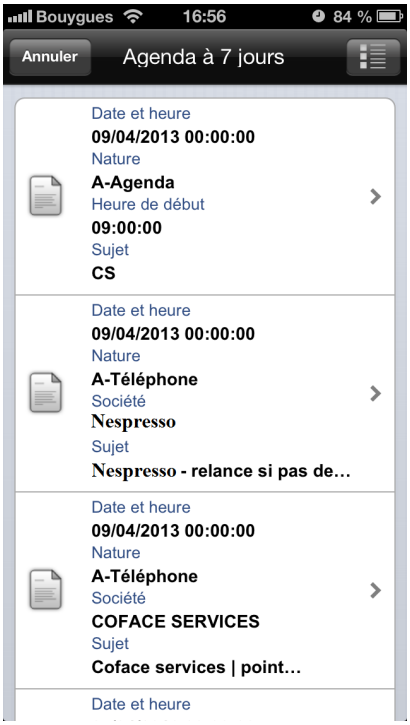

Figure 20: An example of user's agenda in the user's

Figure 21: Results list of searching a company using the Nomalys application

When Paul arrives at his meeting, he uses his smartphone to connect to his company's database and gets some information to show to the client. He writes his request example "Nespresso" in the search area. From this search, he gets the resulting list of folders (Figure 21) and, when he clicks on one of those folds, he gets a mind map (Figure 22).

Figure 22: The result folder presented as a mind map to the user

From Figure 22, we see that the folder has 5 entities (relations). The goal of DRARS is to decide which entity to open or to know it content (e. g. the entity "Reclamations" in Figure 22) and which entity to close (e. g. the entity "Actions" in Figure 22), depending on the user's context.

5.3 Integrating DRARS in Nomalys

The DRARS architecture is divided into two parts (smartphone and the Nomalys server) which are described in what follows.

5.3.1 Smartphone

The smartphone contains the Sensing module of the recommender system and the interface of the Nomalys application, where the user interacts. As stated in Section 4.4 the Sensing module gets the user's request, the time and the GPS coordinates. All that information is stored in an XML document in the user's smartphone and sent

to the Interface module Section 4.4 in the server. The XML document is structured as shown in Figure 23, where the Context element contains the context's different dimensions (time, location, interlocutor) and the Request element contains the identifier of the requested entity, for example in the Figure 23 the requested entity has the identifier "10234 ".

```
<Entity User='0'>
<Context>
<Time = 'Opportunity' ID = '0, 1'/>
<Location longitude='48N' latitude='20E' />
<Interlocutor = 'Nespresso'/>
</ Context>

<Request>
<ID Document value = '10234 '/>
<ID Database value = '1234' />
</ Request> </ Entity User>
```

Figure 23: XML document transferred from the mobile to the server

5.3.2 The Nomalys' Server

When the server receives the XML document containing the user's request, the interface module activates the Connector in order to extract the data from the relational database and transform it into hierarchical data (XML). Then, if there is no recommendation, the interface transfers directly the information to a mobile user.

In what follows we give an example without recommendation and after we include the recommendation.

5.3.2.1 The System's Answer without Recommendation

To illustrate how the Nomalys answers a user's request without recommendation, let a client database contain four entities (relations), Company, Opportunity, Contract and Action, shown in tables 10 to 13.

Table 10: The Company relation within the client database

ID	Name	Country
0	Nomalys	France
1	Selligent	Belgium
2	MS CRM	USA

Table 11: The Opportunity relation within the client database

ID	Subject	Type	Company
0	Opportunity find by Jack	Commercial	0
1	Opportunity find by provider	Industrial	0
2	Opportunity find by provider	Commercial	0

Table 12: The Contract relation within the client database

ID	Reference	Budget	Company	Opportunity
0	Ref.32.065	35k	0	0
1	Ref.55.321	112k	0	1
2	Ref.59.812	9k	0	0

Table 13: The Action relation within the client database

ID	Description	Statut	Company	Opportunity	Contrat
0	Call client	Waiting	0	0	0
1	test application	done	0	1	1
2	Contact a accounting	done	0	1	1

Suppose the user wants to display the folder Nespresso, which means all tuples contained in relations that are linked to the company "Nespresso" through a foreign key.

To this end, the Connection module connects to the database, and, for each customer relation, the Connector retrieves all its tuples with a foreign key to the Nespresso company. The extracted data is converted to XML and sent to the interface and then to the mobile user. Figure 24 shows what the user can see from his mobile application without the recommendation process.

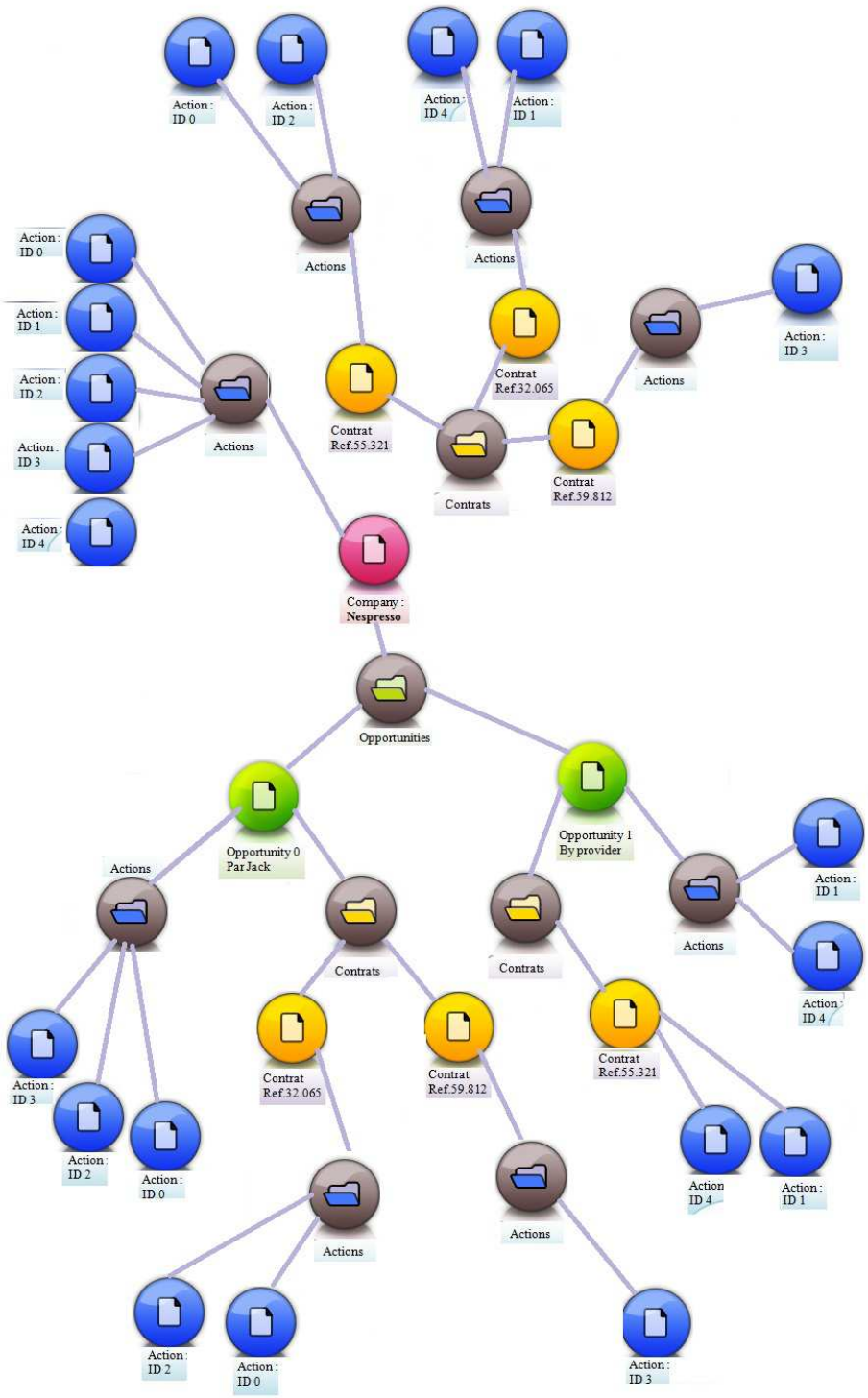

Figure 24: The system's answer without recommendation

We observe from Figure 24 that all entities are open and shown to the user.

5.3.2.2 The System's Answer with Recommendation

When the interface receives the information from the Connection module, it executes the recommendation process rather than transferring directly the results.

As an example, let Table 14 describe a set *Cas* of cases (S_i, UI_i), *i=1, 2, 3,* existing in the case base.

Table 14: Example of diary situations and navigation entries

IDS	User	Time	Location	Client	Relations and Rewards
1	Paul	noon (AW)	Evry	Carte Noire	Opportunities=1, Contracts=4, Actions=1
2	Paul	afternoon (AH)	Roubaix	Quick	Opportunities=0, Contracts=2, Actions=1
3	Paul	morning (AH)	Toulouse	McDonald's	Opportunities=1, Contracts=1, Actions=1

The Situation Retrieval module captures the current situation S (Paris, afternoon(AW), Nespresso), as outlined in Section 4.5, and starts to compute the similarity between S and each of the situations in the case base {*S1, S2, S3*} by applying Eq. 36 and selects the closest situation which is *S3*. The algorithm also computes the risk level of the situation S using the Eq. 44 and gets for example the value 0,95. This value gives a very high probability to open the Contract relation which contains the highest number of rewards (*CTR*).

The interface transfers the document to the user's smartphone. Figure 25 plots the result that the user observes in his smartphone.

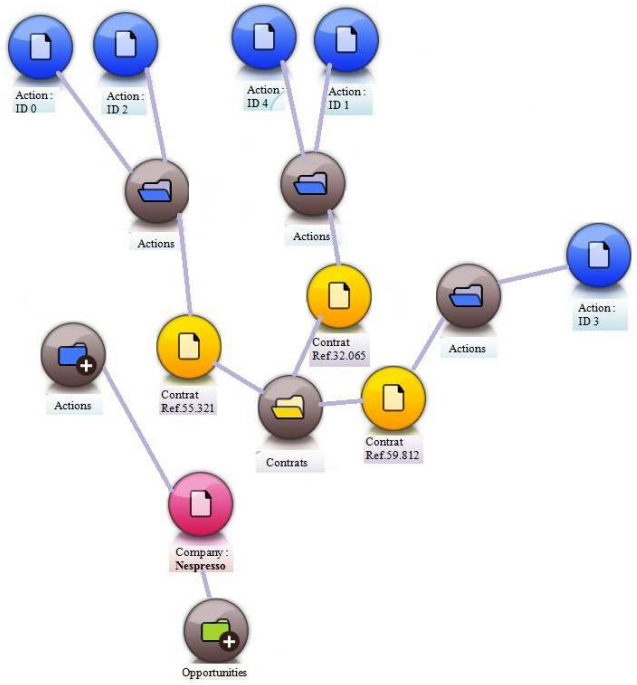

Figure 25: The system's answer with recommendation

We observe, from Figure 25, that the entity Action and Opportunities are closed and the entity Contracts is open, as we expect.

We have discussed in this section how DRARS is executed inside Nomalys, illustrating it with a simple scenario. In what follows we give a small description of the different tools used in this setting.

5.4 Tools Used to Implement DRARS

In this section, we describe the tools used during the different implementation phases of DRARS.

5.4.1 Programming the Server Application

In order to program the server application, we have used C sharp, which is an object oriented programming language, created by Microsoft.

5.4.2 Ontologies Construction

We have used Protégé to build our ontologies. Protégé is a free, open-source platform that provides tools to construct ontologies. Its interface allows the creation, visualization, and manipulation of ontologies in various representation formats. Further, Protégé can be extended by a plug-in architecture and a Java-based Application Programming Interface (API) for building knowledge-based tools and applications.

5.4.3 Database Management

Used to store the Nomaly's data, Microsoft SQL Server is a database management system (abbreviated DBMS or RDBMS for "relational database management system") developed and marketed by Microsoft Corporation.
It was originally co-developed by Sybase and Microsoft. The original version was released on Unix platforms and OS/ 2. Since then, Microsoft has brought this DBMS on Windows and it is now only supported by this system.

5.4.4 Project Management

We have used a web-based project management named Redmine which is open source. It is used to monitor projects during their implementation. We have also used Subversion, which is a revision control system, to store information during the thesis' development. Subversion allows for archiving all intermediate versions, as well as the differences between the versions.

5.5 Conclusion

In this chapter, we give an example of running our system to explain its main specifications and show how the different modules are implemented in the Nomalys application. We also introduce the various tools used to achieve our goal.

The next chapter is dedicated to the evaluation of our system.

Chapter VI

EVALUATION OF DRARS

6.1 Introduction

In order to evaluate empirically the performance of our approach, in the absence of a standard evaluation framework, we propose an evaluation framework based on a diary set of study entries.

The main objectives of the experimental evaluation are:

(1) Find the optimal parameters of the DRARS system;

(2) Evaluate the performance of the proposed R-UCB algorithm using an off-line and an on-line evaluation.

In the following, we describe our experimental datasets and then present and discuss the obtained results.

6.2 Evaluation Framework

We have conducted a diary study with the collaboration of Nomalys. This company provides a history application, which records the time, the current location, the social and navigation information of its users during their application use. The diary study lasted 2 months and has generated 356 738 diary situation entries. Table 15 illustrates three examples of such entries where each situation is identified by IDS.

Table 15: Diary situation entries

IDS	Users	Time	Place	Client
1	Paul	11/05/2012	75060 Paris Cedex 02	NATIXIS
2	Fabrice	15/05/2012	2 rue Kellermann - 59100 Roubaix - France	MGET
3	Paul	19/05/2012	90 Boulevard Pasteur, 75015 Paris	AMUNDI

Each diary situation entry represents the capture of contextual time, location and social information. For each entry, the captured data are replaced with more abstracted information using time, spatial and social ontologies (Section 3). Table 16 illustrates the result of such transformations corresponding to entries in Table 15.

Table 16: Diary situation abstraction

IDS	Users	Time	Place	Client
1	Paul	Workday (AW)	Paris	Finance client
2	Fabrice	Workday (A)	Roubaix	Social client
3	John	Holiday (A)	Paris	Telecom client

From the diary study, we have obtained a total of 5 518 566 entries concerning the user's navigation (number of clicks and time reading an entry), expressed with an average of 15.47 entries per situation. Table 17 illustrates examples of such diary navigation entries, where an entry is identified by IdDoc.

Table 17: Diary navigation entries

IdDoc	IDS	Click	Time
1	1	2	2'
2	1	4	3'
3	2	1	5'

6.3 Descriptive Analysis

To analyse the different risk level of situations of our dataset, we have computed the risk of each situation in the dataset using Eq. 41. Then, we group the situations depending on their levels of risk in different intervals: [1%, 20%],]20%, 40%],]40%, 60%],]60%, 80%],]80%, 100%], where 1% corresponds to the less risky situations and 100% corresponds to the highest risk level situations. We plotted the situation distribution in the 5 intervals as a pie chart in Figure 26.

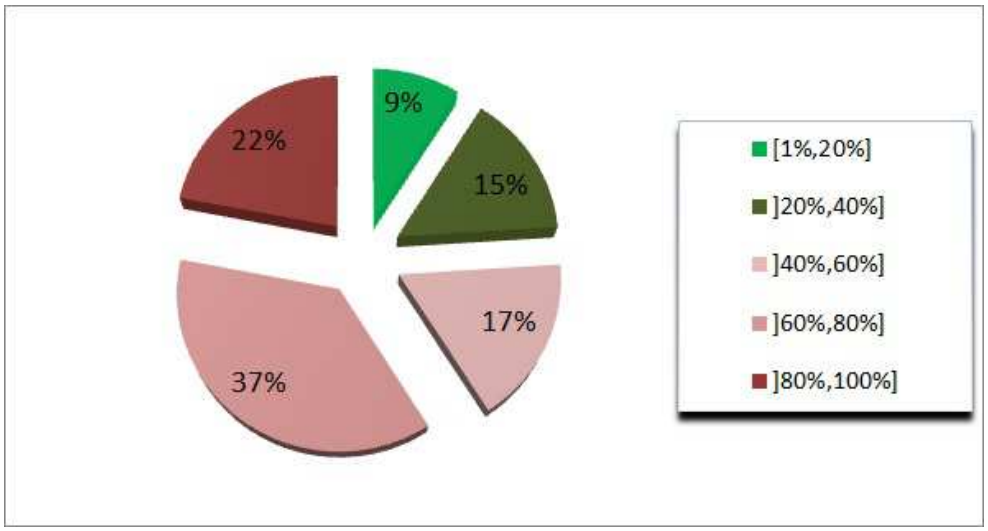

Figure 26: Risk level of the situations in the 5 clusters

In Figure 26, the largest cluster takes 37% of the situations, while the smallest cluster contains 9% of the situations. We can say from the figure that our application domain is risky, with more than the half of the situations in the interval]60%,100%]. We have further studied the situation composition of the 5 intervals with respect to localization, time, social context, age, gender categories, high time spent and high number of user's clicks, and presented the results in Figure 27 as a heatmap graph. Each square's gray level indicates the rate of a feature on the corresponding cluster, from black (high content) to white (low content).

From the figure, we have found that the situations with high risk level]80%,100%] are mostly office situations under working day; the situations with high risk level are mostly transportation situations under working day; the situations with moderate risk level]60%,80%] are mainly restoration situations under working day; the situations with low-moderate risk level]40%,60%] are predominantly home situations under working day; the situations with low risk level]20%,40%] are mostly situations under holidays.

We have also observed that female have less situations with high risk level than male,

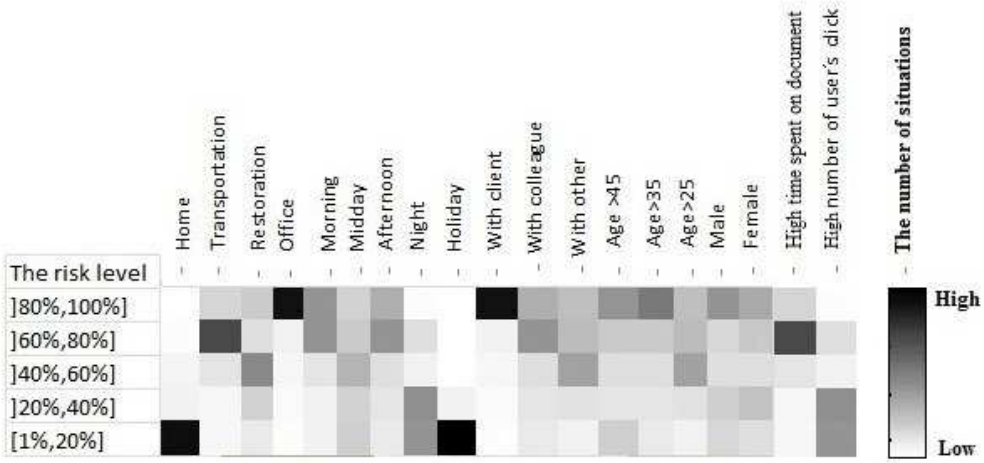

Figure 27: The risk level of the situations regarding their properties

and users with age between 20 and 35 have few situations with high risk level. This observation can lead to program more exploration for young people (35>age>20) than people with age between 35 and 45, as well as for female than male.

6.4 Parametrizing

In this section, we present the algorithm we use to find the optimal parameters of the DRARS system, which are the th_R threshold, ϵ_{min} and ϵ_{max} (Section 4.5.6.2). Computing the threshold is very important because misclassifying a non-CS as CS can be tolerated, but the opposite may be catastrophic. Computing ϵ_{min} and ϵ_{max} is also important because giving a very high ϵ_{min} can lead to a very high exploration, which can be inadequate in CS.

We have used a genetic algorithm to tune these three learning parameters of the DRARS system.

6.4.1 Genetic Algorithm

A Genetic Algorithm (GA) begins with a set of candidate solutions called chromosomes, composed of genes. A set of chromosomes is called a population. In our experiment a gene represents a learning parameter, in our case the th_R threshold, ϵ_{min} and ϵ_{max}. Moreover, each chromosome $P_i = (th_{R_i}, \epsilon_{mini}, \epsilon_{maxi})$ in the GA represents a tuple of learning parameters' values, $P_i \in P$, being P a set of candidate solutions.

From the existing populations of chromosomes, a new population is created with the hope of getting a better one. The chromosomes that are selected for reproduction have to be the most adapted to their environment. This process is repeated until satisfying some stop condition, like for example if the new population is not better than the old population. The global process of the GA of our system is described in Alg. 6.

Algorithm 6 Genetic Algorithm

Input:

$th_R : (th_{R_1}, th_{R_2}, ..., th_{R_M})$

$\epsilon_{min} : (\epsilon_{min_1}, \epsilon_{min_2}, ..., \epsilon_{min_M})$

$\epsilon_{max} : (\epsilon_{max_1}, \epsilon_{max_2}, ..., \epsilon_{max_M})$

N: Number of iterations

M: Number of individual

Output: P': survivor populations

P=$\emptyset$

for $i = 1$ **to** M **do**

$P_i = random(B, \epsilon_{min}, \epsilon_{max})$ // Base population $P : (P_1, P_2, ..., P_M)$ randomly generated $P' = P \cup \{P_i\}$

end for

c = Fitness (P)

for $t = 1$ **to** N **do**

P' = Selection(P', c)

P' = Crossover(P')

P' = Mutation(P')

c = Fitness(P')

end for

As shown in Alg. 6, the GA of our system is composed of four main algorithms, namely Fitness, Selection, Crossover and Mutation. Selection, Crossover and Mutation change the chromosomes in the existing populations in order to create new ones. From the new created populations, Fitness returns the number of clicks from the user, which are used by the Selection algorithm to retrieve the chromosomes that have the highest number of clicks. These process is repeated until reaching the predefined number N of iterations.

6.4.1.1 Fitness

The Fitness algorithm (Alg. 7) returns the number of clicks by testing the chromosomes in the environment. The number of clicks gives the rate of adaptation of chromosomes to their environment. Concretely, the Fitness algorithm runs in the DRARS system with a population P and gets user's clicks that correspond to this population.

Algorithm 7 Fitness

Input:

$P : (P_1, P_2, ..., P_M)$

$c = \emptyset$

Output: c : click feedback

for i = 1 to M **do**

  Run R-UCB (Alg. 5) with P_i

  Receive a click feedback c_i from the user

  $c = c \cup c_i$

end for

6.4.1.2 Selection

The Selection algorithm is based on the principle of survival of the fittest. It creates a new generation of chromosomes from the previous generation. Chromosomes with better fitness values increase in number while chromosomes with less values decrease in number (chromosomes with better fitness values means the chromosome that leads to user's clicks on the Fitness algorithm). The Selection algorithm is sketched in Alg. 8.

Algorithm 8 Selection

Input:

c: click feedback

N: number of chromosomes

$P = (P_1, P_2, ..., P_M)$: initial chromosomes

$P' = \emptyset$: survivor populations

Output: P'

for $i = 1$ **to** N **do**

$f = argmax_j(c_j)$ // j is the index of the chromosome in P

$P' = P' \bigcup P_f$

end for

In Alg. 8, M is the number of the initial chromosomes, c_i is an array that gives the number of user's clicks when the algorithm uses chromosome P_i in the Fitness algorithm.

6.4.1.3 Crossover

The Crossover algorithm randomly exchanges the genes of two chromosomes to create two progenies or offsprings. The crossover operator mimics biological recombination between two organisms. We recall that genes, in our case, are the information of the DRARS's learning parameters $(th_{R_i}, \epsilon_{min_i}, \epsilon_{max_i})$.

As an example, consider parents (Parent 1 and Parent 2) and a crossover point at position 2 sketched on Table 18.

In this example, we have two parents (Parent1, Parent2) that combine their genes to give two offsprings (Offspring1, Offspring2). Offspring1 inherits values in position 1 and 2 from Parent1 and the value in position 3 from Parent2. Similarly, Offspring2 inherits values in position 3 from Parent2 and the rest from Parent1.

Alg. 9 sketches the different steps of the Crossover algorithm.

Table 18: Crossover example

Individual	Value at position 1	Value at position 2	Value at position 3
Parent1	0,15	0,25	0,13
Parent2	0,2	0,5	0,25
Offspring1	0,15	0,25	0,25
Offspring2	0,2	0,5	0,13

Algorithm 9 Crossover

Input:

P : Set of Populations

Cr : number of chromosomes selected for crossover

$P' = \emptyset$: Offspring chromosomes

Output: P'

for $t = 1$ **to** Cr **do**

P_i = Random(P); P_j = Random(P) // randomly select two chromosomes

P_e = Random(P_i, k) // randomly select $k \in [0, |P_i|]$ genes from P_i

$P_e = P_e \bigcup$ Random(P_j, s) // randomly select s genes from P_j, and $k+s \in [0, |P_j|]$

$P' = P' \bigcup P_e$ // insert P_e on the new population P'

end for

6.4.1.4 Mutation

Mutation simply randomly changes the genes' information. Mutation helps in avoiding the possibility of mistaking a local optimum for a global optimum. It can occur for any genes with usually very small probability. For example, consider chromosome P=(0,2;0,5;0,25) in Table 19 with mutation point at position 3.

Table 19: Mutation example

Individual	Value at position 1	Value at position 2	Value at position 3
Before mutation	0,2	0,5	0,25
After mutation	0,2	0,5	0,15

The value 0,25 at position 3 flips to 0,15 after mutation. Alg. 10 presents the Mutation algorithm.

Algorithm 10 Mutation

Input: P'

mut : the number of mutations

Output: P''

for $t = 1$ **to** mut **do**

P'_i = Random(P') // randomly select 1 chromosome

k = Random(N) // randomly select position $k \in [0, |P'_i|]$

$P'_i[k]$ = Random(N) // $N \in [0, 1]$

$P'' = P'' \bigcup P'_i$

end for

Return P''

6.4.2 Testing the GA

To get the optimal chromosome, we have done an off-line test in our experiments. We have firstly collected, from the Nomalys' historic, a collection Cas of 100000 cases $Cas^i = (S^i, D^i)$ that contains a couple of situation S^i and its correspondent displayed documents $D^i, i \in 1, ..., 100000$. The testing step consists of running the algorithm by confronting it at each iteration to a case randomly selected from Cas.

The size of the chromosomes population is constant throughout a GA run; for our experiments we have used a population of 120 initial chromosomes. The initial population is completely random.

Figure 28 shows how the number of the fittest chromosomes change in each generation (new population). We observe that population fitness tends to stabilize around the 15th generation. For this reason, we have decided to stop the GA after 15 generations. At this point, we have gathered the resulting population and we have observed that $B \in [0.7, 0.82]$, $\epsilon_{min} \in [0.05, 0.13]$, $\epsilon_{max} \in [0.47, 0.56]$. Consequently, we select as

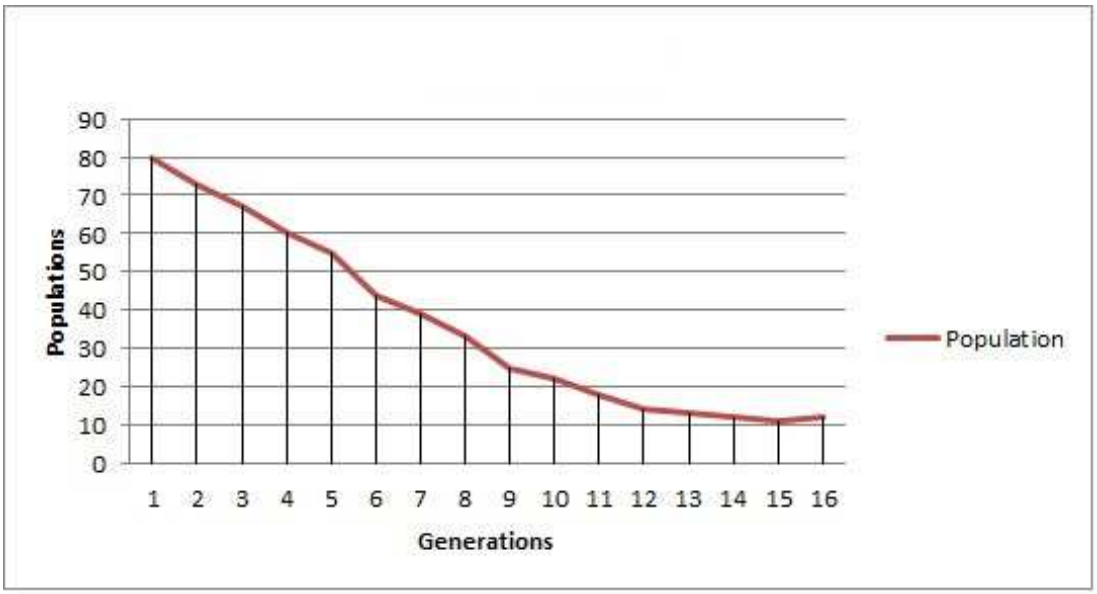

Figure 28: Number of populations by generation

optimal value a random individual from the final population for testing our DRARS in off-line and on-line process.

6.5 Off-line Experimental Results

In this section, we evaluate the main component of our system, which is the R-UCB algorithm. To test it, in our experiments, the testing step is similar to the one described in Section 6.4.2. We have collected, from the Nomalys' historic, a collection Cas of 100000 cases Cas^i which are different than used to parametrize the GA. The testing step consists of running the algorithm by confronting it at each iteration to a case randomly selected from Cas. For each iteration i the algorithm need to select or recommend 10 documents $d \in D^i$, note that the algorithm is only confronted to the case Cas where $|D| > 20$ and $D \in Cas$. We compute the average CTR (click feedback) every 1000 iterations and we have run the simulation until the number of iterations reaches 10000, which is the number of iterations where all the tested algorithms have converged. Note that, due to our goal on evaluating the RS in a periods

of time or in iterative process, we have used the average CTR rather than traditional Recall used in IR that do not allowed this kind of evaluation.

In the first experiment, in addition to a pure exploitation baseline, we have compared the R-UCB algorithm to the algorithms described in the related work (Chapter 3): ϵ-UCB and beginning-UCB, decreasing-UCB, VDBE-UCB , EG-UCB, which correspond to ϵ-UCB using respectively decreasing exploration, beginning exploration, VDBE exploration and EG exploration. In Figure 29, the horizontal axis represents the number of iterations and the vertical axis is the performance metric.

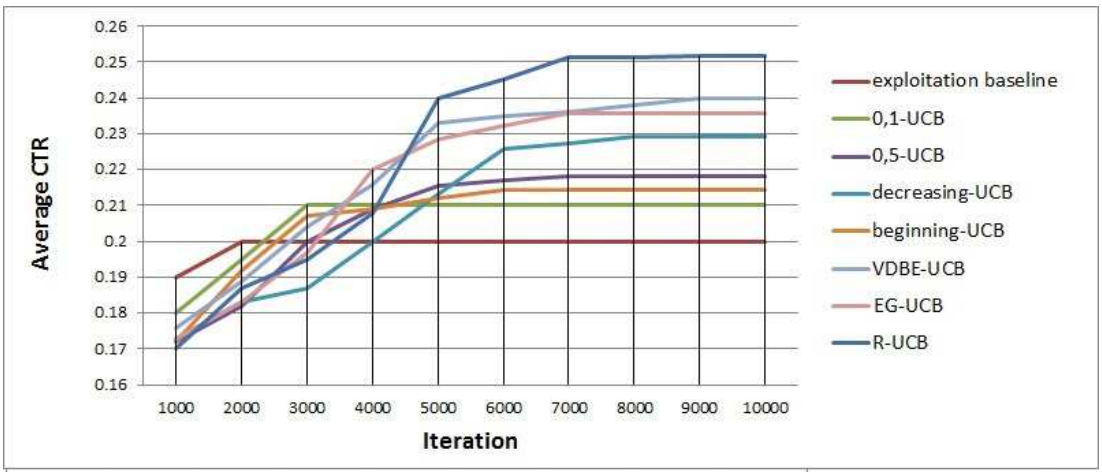

Figure 29: Average CTR for exploration-exploitation algorithms

We have parametrized the different algorithms as follows: ϵ-UCB was tested with two parameter values: 0.5 and 0.1; decreasing-UCB and EG-UCB use the same set ϵ_i = 1- 0.01 * i, i = 1,...,100; decreasing-UCB starts using the highest value and reduces it by 0.01 every 100 iterations, until it reaches the smallest value. We have several observations regarding the different exploration-exploitation algorithms. For the decreasing-UCB algorithm, the converged average CTR increases as the ϵ decreases (exploitation augments). For the 0.1-UCB and 0.5-UCB, neither a small exploration of 10% for 0.1-UCB nor a big exploration of 50% for 0.5-UCB give good results. This

confirms that a static exploration is not interesting in this dynamic environment. While the EG-UCB algorithm converges to a higher average CTR, its overall performance is not as good as the VDBE-UCB algorithm that considers the uncertainty of its knowledge for each situation.

The R-UCB and VDBE-UCB algorithms effectively have the best convergence rate, increasing the average CTR by a factor of 1.5 over the baseline for VDBE-UCB and 2 for R-UCB. This improvement comes from a dynamic exploration-exploitation trade-off, controlled by considering the situations.

Finally, as we expect, the R-UCB outperforms VDBE-UCB, which is explained by the good estimation of the risk based on our semantic approach. The R-UCB algorithm takes full advantage of exploration when the situations are not dangerous (non-CS), giving opportunities to establish good results when the situations are critical (CS).

6.5.1 Risk Level of the Situations

To compare the algorithms in situations with different risk levels, we run the tested algorithms in the groups of situations with different risk level described in Section 6.3. To better visualize the comparison results, Fig. 30 shows algorithms' average CTR graphs with the previous referred risk levels. Our first observation is that the R-UCB algorithm outperforms all other exploration-exploitation algorithms, at every levels. We notice that, in high risk situations, low exploration (0.1-UCB) is better than high exploration (0.5-UCB). The gap between the R-UCB results and the other algorithms increases with the risk of the situations. This improvement comes from the safety exploration made by the R-UCB. During a CS, R-UCB makes a safety exploration by selecting the documents with the highest CTR from the non-CS set which are similar to the current one.

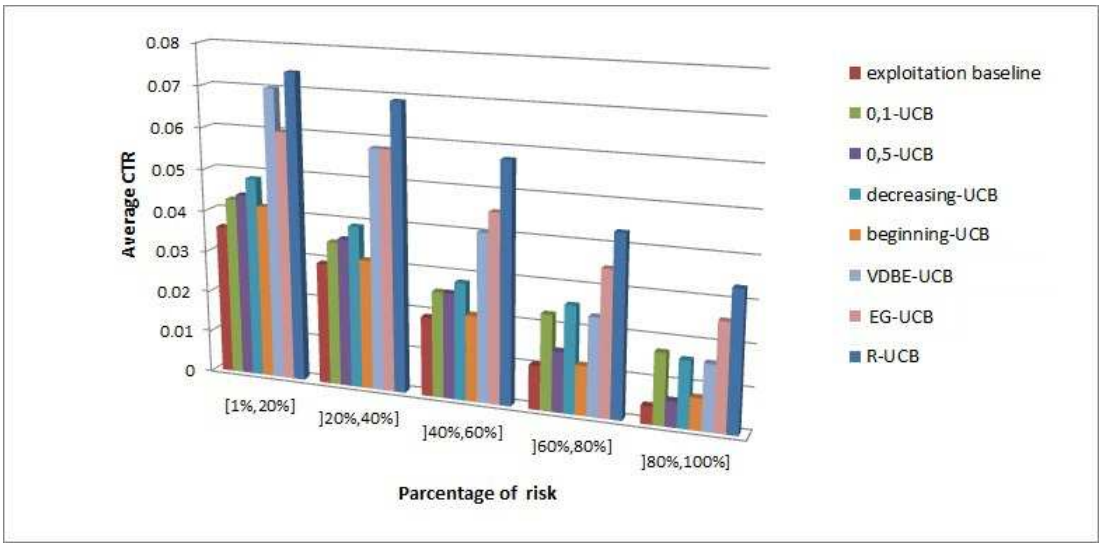

Figure 30: Average CTR of exploration-exploitation algorithms in situations with different risk levels

6.5.2 Size of Data

To compare the algorithms when the case base is sparse in our experiments, we reduce the case base size of 50%, 30%, 20%, 10%, 5%, and 1%, respectively. To better visualize the comparison results, Fig. 31 shows algorithms' average CTR graphs with the previous referred data sparseness levels. Our first observation is that all algorithms are useful at every level. We notice that decreasing data size does not significantly improve the performance of 0.5-UCB and 0.1-UCB. Except for exploitation baseline, beginning-UCB seems to have a rather poor performance. Its results are worse than any other strategy independently of the chosen parameters. The reason lies in the fact that this algorithm makes the exploration only at the beginning. We can also observe that R-UCB outperform the other existing algorithms. This performance comes from the use of the CBF that helps R-UCB in the recommendation process by selecting more attractive documents to recommend.

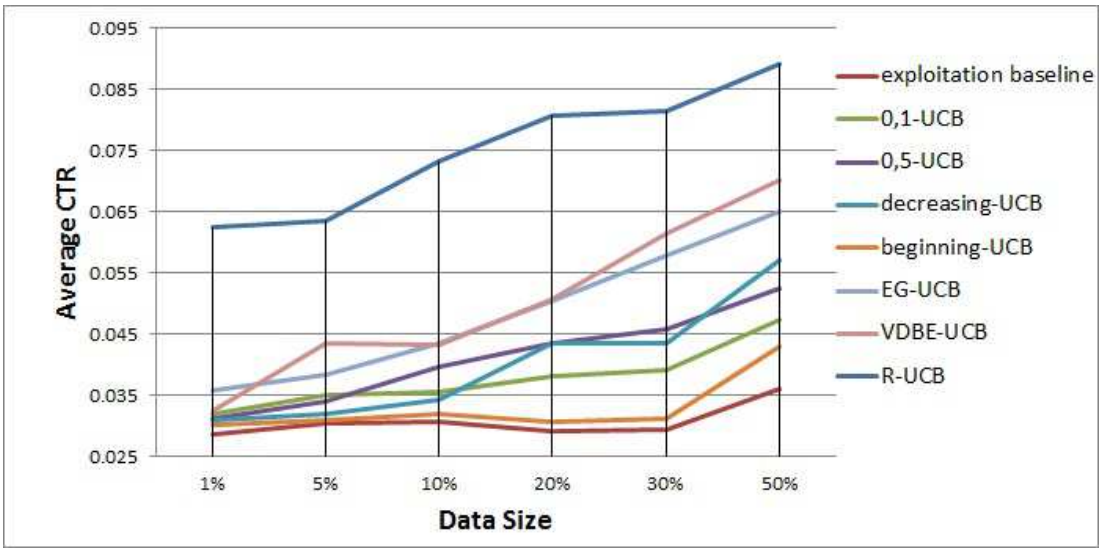

Figure 31: Average CTR for different data size

6.6 On-line Evaluation of DRARS

Based on the results from our off-line evaluations, we have selected the most promising techniques, R-UCB, VDBE-UCB and EG-UCB, and tested them with real users in their real working environments.

We use the 3000 users of Nomalys application in full time. To qualify, participants are required to use Nomalys for more than 1 hour/week.

To this end, we have randomly split the users in three groups. During the first week of the study, the system records the documents used by each participant without recommendation. During the second week of the study, we have equipped the first group with the DRARS system running the R-UCB algorithm, the second group running the VDBE-UCB, and the last group running EG-UCB algorithm.

6.6.1 New Document Exploration

With a large number of users, we can not easily follow the average CTR of each user. For this reason, we use a metric to see how the usage of the RS impacted the

user's usage of documents that they had not previously seen (new documents). An increase in the usage of such documents would indicate that the system was recommending documents that were useful. By comparing the number of new documents with and without recommendation in the 1st, 2nd and 3rd groups, we got the impact that recommendations had on the use of new documents. Figure 32 illustrates this comparison by week and by group.

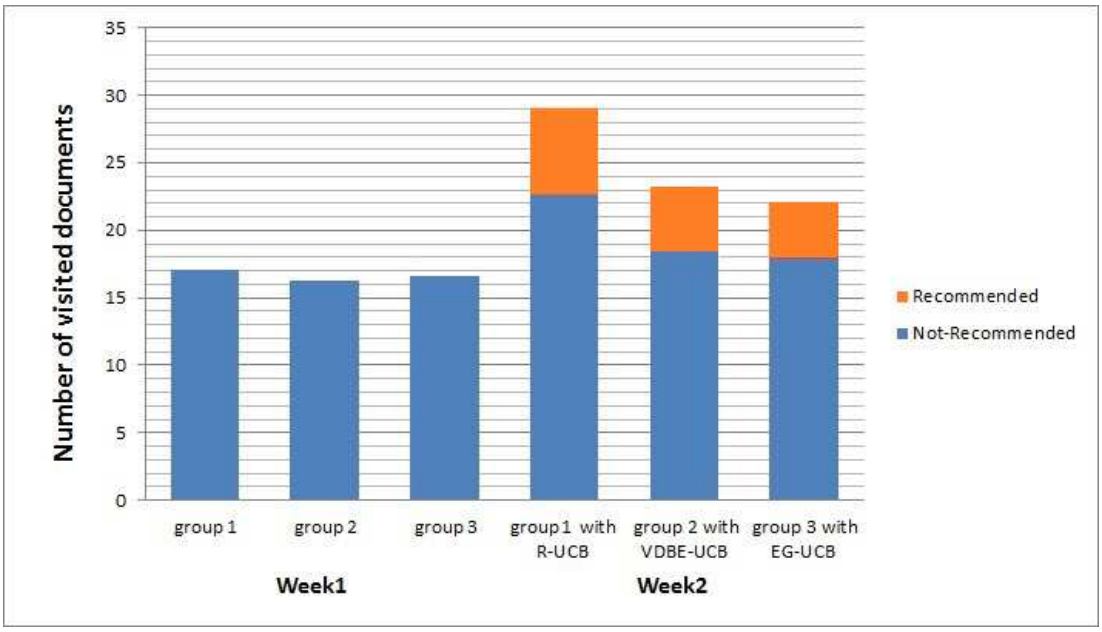

Figure 32: Average number of new documents used in three groups without and with recommendation

Figure 32 shows a main effect for the week on the number of visited new documents. The average number of new documents used in the first week has been 17.12, 16.31 and 16.63 for groups 1, 2 and 3 respectively, and 29, 23.21 and 22.12 for groups 1, 2 and 3 respectively on the second week. Moreover, group 1 has significantly more new documents used than groups 2 and 3 in the second week.

Based on the results illustrated in Fig. 32, we can estimate the proportion of new visited documents used in the second week due to the introduction of the recommender system, and the proportion of new documents that would have been used by chance (without recommendation).

In the second week, with groups 1, 2 and 3 respectively, an average of 6.3, 4.7 and 4.3 documents actually appeared in the recommender list before being used (i.e. new recommended documents), which represents a very good improvement w. r. t. the first week without recommendation.

Another interesting finding is that, excluding the average of 16.68 new documents without recommendation (first week) from the number of new documents with recommendation (second week), for groups 1, 2 and 3, and also excluding the average number of new recommended documents, we get an average number of 6.01, 1.82 and 1.13 new documents in the second week with groups 1, 2 and 3, respectively. We believe that the majority of these extra documents were discovered through the use of DRARS.

The improvement on the number of visited new documents, on one hand, corresponds most of all to recommended documents; on the other hand, an important part is discovered during recommendations, which shows an unintentional benefit of the system which promotes document discovery.

6.6.2 R-UCB, VDBE-UCB and EG-UCB Comparison

To compare the R-UCB, VDBE-UCB and EG-UCB algorithms, we look at the number of recommended documents that have been used multiple times (i. e. more than twice) in each session and the time spent in each document. Figure 33 shows the results. We observe, from the figure, that the time spent in each document does not significantly change in the three groups, which means that the exploration-exploitation trade-off does not impact the user's time spent. However, it has shown

that group 1 has used more new documents than groups 2 and 3, which confirms that our exploration-exploitation trade-off have allowed more exploration of documents from the users than the other strategies.

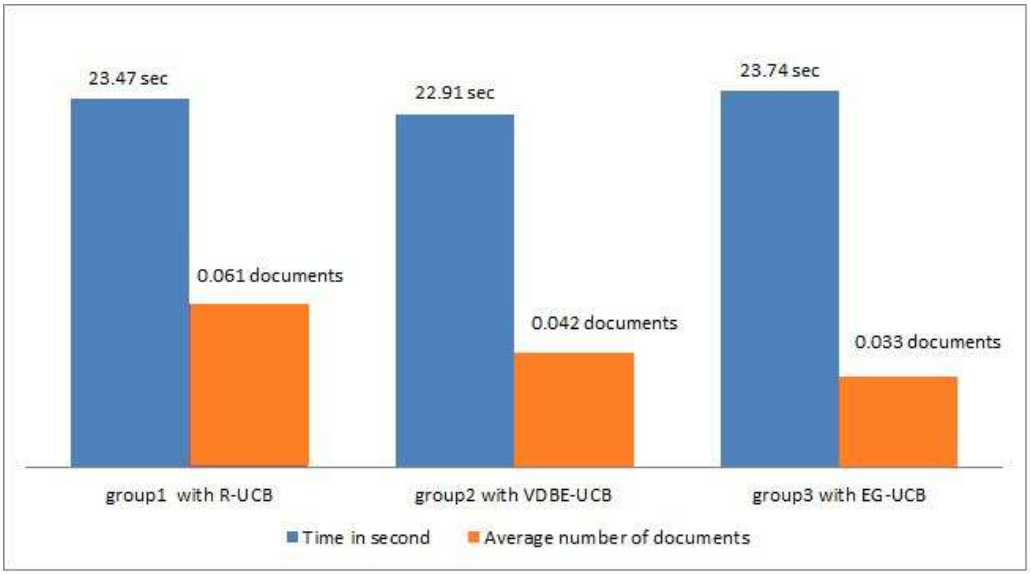

Figure 33: Average number of recommended documents have used multiple times for each navigation session

6.7 The Algorithm Complexity

Our own algorithm is expected to be at most O(S^2C^2), where S is the number of situations to be matched, and C is the number of concepts of the ontologies. The particularity of the input data used by our system would suggest C to be rather small and static. Therefore, S should dominate the complexity O(S^2).

We have implemented the generalization module of the system in $C\sharp$ programming language. We have used a personal computer with two-core CPU, 2.80GHz each, 6GB RAM and 64-bit operating system. The test run on randomly generated documents recommendation confirms the estimated computational complexity and shows that, in this environment, 25000 situations are processed to make recommendation in less

than 1,2s.

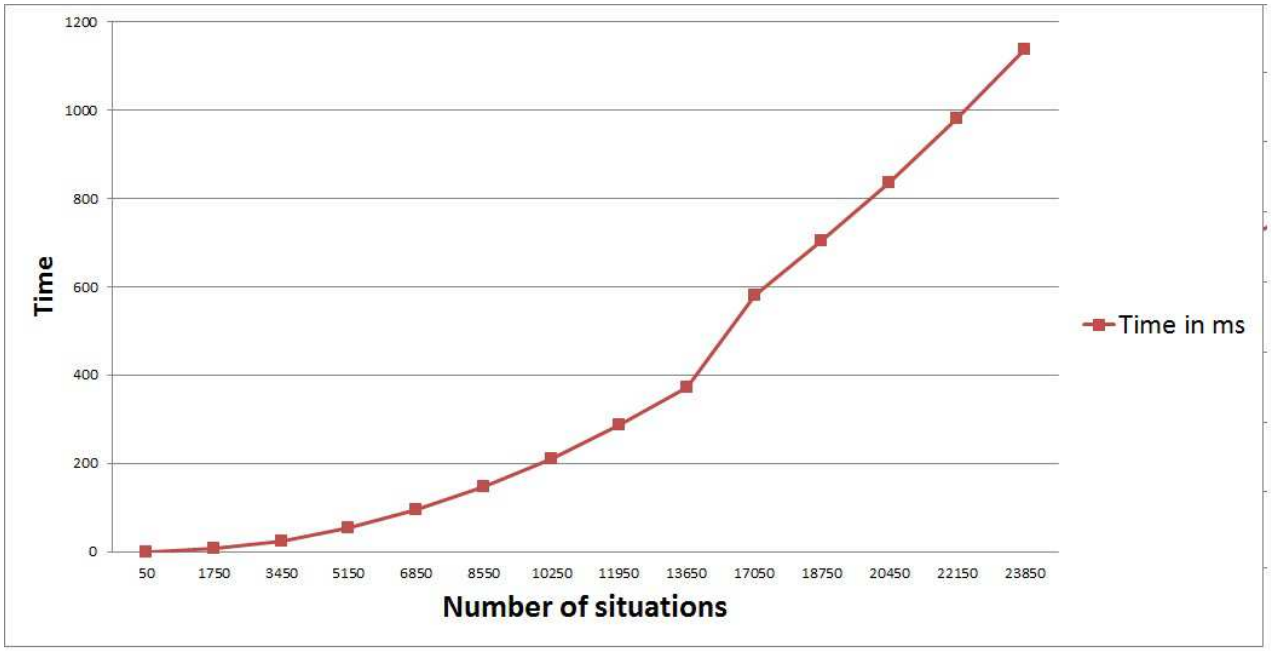

Figure 34: Time in ms to recommend

6.8 Conclusion

In this chapter, we have validated our work with a series of both off-line and on-line studies which offer promising results. More precisely, this study yields to the conclusion that considering the risk level of the situation on the exploration-exploitation strategy significantly increases the performance of the recommender system.

Chapter VII

CONCLUSION AND PERSPECTIVES

As the amount of data maintained by mobile information systems keep increasing, and finding the pertinent information to deliver to the users becomes a heavy problem, we have presented throughout this manuscript, our approach to design a Dynamic Risk Aware Recommender System. This system aims at proposing more relevant information to users in risky and dynamic environments.

To this end we have set out throughout this manuscript the following steps.

(1) We have explored various requirements for an accurate CARS, namely considering the dynamicity of the user's content, considering the dynamicity of the user's context and the situation risk level. We observe that the most appropriate representations of the user's profile and context are respectively the multidimensional representation, and the ontology representation. We also observe that the risk is never considered in existing RS, and that the most interesting approach to follow the dynamicity of the user's content is through a trade-off among bandit algorithm and CBF technique.

(2) We have then explored the multi-armed Bandit problem as a solution for the content dynamicity requirement. We describe the bandit algorithms that can be used in recommender systems and in the risk aware decision. We conclude that existing algorithms, do not consider of the context and the risk in the exploration-exploitation trade-off.

(3) We introduce our proposal named DRARS, for "Dynamic Risk Aware Recommender System". This system aims at tackling the main requirement defined in this thesis. We have combined a multi-dimension user's profile and a semantic model of spatio-temporal-social context to manage the user's profile, context and situation.

We have used a combination of CBF and bandit algorithm to follow the dynamicity of the user's contents. We also consider the risk on recommendation by computing the risk level of the user situations.

(4) We sketch a running example of our system that summarizes the main specifications of our proposal and shows how the different modules are implemented in the Nomalys application. We have also explained the installation of the various tools used to achieve our goal.

(5) We have validated our proposal with a series of both off-line and on-line studies. These studies yield to the conclusion that considering the risk level of the situation on the exploration-exploitation strategy significantly increases the performance of the recommender system.

Finally, we have studied the various flaws of our system. This helps us to provide several research perspectives for enhancement and future work.

7.1 Perspectives

Experimental evaluations of our contributions have shown their effectiveness in several aspects, and open perspectives in improving our system. More specifically, our perspectives focus on the following points:

(1) Enhancing the user's context model by introducing other contextual dimensions. One line of research in this direction is to include the social context defined by the user's friends or social group and try to include the interest of the user's group when computing user's interest. This allows to refine the relevance of recommendations.

(2) In our model, we have related interests for each situation, but some users' interests may be related to several situations.

In this sense, our perspective is to generalize the user's interests in more than one situation by creating situations clusters.

(3) As part of our research perspectives in the long term, we envisage to study

the dynamicity of content in IR, where we plan to apply the bandit algorithm as it is done in the RS. We also plan to extend the notion of risky situation to IR.

(4) Regarding the risk aware recommendation, we plan to study the

notion of safety exploration or how we can make exploratory recommendation without upsetting the user. One possible direction to solve this problem is to study the history of the user in risky situations and construct a specific profile related to that kind of situations.

(5) We have used only user's clicks as a feedback of the user's interest. We plan to consider different features, like the time spent on a document. In this sense, we have to associate a weight for each feature.

(6) With the emergence of distributed infrastructures, e.g., peer-to-peer Semantic Web, Grid, etc., recommender systems may become decentralized and recommendations will be based upon opinions from most trusted peers rather than most similar ones. Likewise, for social filtering we cannot rely upon conventional collaborative filtering methods only, owing to the neighborhood computation scheme's poor scalability. Some more natural and, most important scalable neighborhood selection process becomes indispensable, e.g., based on trust networks.

.

REFERENCES

[1] A. Filar, J., Krass, D., and W. Ross, K., "Percentile performance criteria for limiting average Markov decision processes," in *Advanced Courses*, 1995.

[2] Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P., "Towards a better understanding of context and context-awareness," in *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, HUC '99, (London, UK), pp. 304–307, Springer-Verlag, 1999.

[3] Adomavicius, G. and Tuzhilin, A., "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, pp. 734–749, June 2005.

[4] Anderson, C. R., *A machine learning approach to web personalization*. PhD thesis, 2002.

[5] Atefeh Jajvand, Mir Ali Seyyedi, A. S., "A Hybrid Recommender System for Service Discovery," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 2, pp. 1344–1347, 2013.

[6] Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E., "Gambling in a rigged casino: The adversarial multi-armed bandit problem," in *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, FOCS '95, (Washington, DC, USA), pp. 322–, IEEE Computer Society, 1995.

[7] Auer, P., Cesa-Bianchi, N., and Fischer, P., "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, pp. 235–256, May 2002.

[8] Baeza-Yates, R. A. and Ribeiro-Neto, B., *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing, 1999.

[9] Balabanovic, M. and Shoham, Y., "Content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, pp. 66–72, 1997.

[10] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., and Riboni, D., "A survey of context modelling and reasoning techniques," *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 161 – 180, 2010. ¡ce:title¿Context Modelling, Reasoning and Management¡/ce:title¿.

[11] Bila, N., Cao, J., Dinoff, R., Ho, T. K., Hull, R., Kumar, B., and Santos, P., "Mobile user profile acquisition through network observables and explicit user queries," in *Proceedings of the The Ninth International Conference on Mobile Data Management*, MDM '08, (Washington, DC, USA), pp. 98–107, IEEE Computer Society, 2008.

[12] Bila, N., Cao, J., Dinoff, R., Ho, T. K., Kumar, B., Lieuwen, D., and Santos, P., "Intuitive network applications: Learning user context and behavior," vol. 13, (New York, NY, USA), pp. 31–47, John Wiley & Sons, Inc., Aug. 2008.

[13] Bogers, T. and van den Bosch, A., "Collaborative and Content-based Filtering for Item Recommendation on Social Bookmarking Websites," Oct. 2009.

[14] Bouidghaghen, O., Tamine, L., and Boughanem, M., "Context-aware user's interests for personalizing mobile search," in *Mobile Data Management (1)*, pp. 129–134, 2011.

[15] Bouneffouf, D., "Applying machine learning techniques to improve user acceptance on ubiquitous environement," *arXiv preprint arXiv:1301.4351*, 2013.

[16] Bouneffouf, D., "Evolution of the user's content: An overview of the state of the art," *arXiv preprint arXiv:1305.1787*, 2013.

[17] Bouneffouf, D., "The impact of situation clustering in contextual-bandit algorithm for context-aware recommender systems," *arXiv preprint arXiv:1304.3845*, 2013.

[18] Bouneffouf, D., "Improving adaptation of ubiquitous recommander systems by using reinforcement learning and collaborative filtering," *arXiv preprint arXiv:1303.2308*, 2013.

[19] Bouneffouf, D., "'l'apprentissage automatique', une étape importante dans l'adaptation des systèmes d'information à l'utilisateur," *Inforsid*, pp. 427–428, 2013.

[20] Bouneffouf, D., "Mobile recommender systems methods: An overview," *arXiv preprint arXiv:1305.1745*, 2013.

[21] Bouneffouf, D., "Situation-aware approach to improve context-based recommender system," *arXiv preprint arXiv:1303.0481*, 2013.

[22] Bouneffouf, D., "Towards user profile modelling in recommender system," *arXiv preprint arXiv:1305.1114*, 2013.

[23] Bouneffouf, D., "\'etude des dimensions sp\'ecifiques du contexte dans un syst\eme de filtrage d'informations," *arXiv preprint arXiv:1405.6287*, 2014.

[24] Bouneffouf, D., "Recommandation mobile, sensible au contexte de contenus\'evolutifs: Contextuel-e-greedy," *arXiv preprint arXiv:1402.1986*, 2014.

[25] Bouneffouf, D., Bouzeghoub, A., and Ganarski, A. L., "Risk-aware recommender systems," in *Neural Information Processing*, pp. 57–65, Springer, 2013.

[26] Bouneffouf, D., Bouzeghoub, A., and Gançarski, A. L., "A contextual-bandit algorithm for mobile context-aware recommender system," in *Neural Information Processing*, pp. 324–331, Springer, 2012.

[27] Bouneffouf, D., Bouzeghoub, A., and Gançarski, A. L., "Exploration/exploitation trade-off in mobile context-aware recommender systems," in *AI 2012: Advances in Artificial Intelligence*, pp. 591–601, Springer, 2012.

[28] Bouneffouf, D., Bouzeghoub, A., and Gancarski, A. L., "Following the user's interests in mobile context-aware recommender systems: The hybrid-e-greedy algorithm," in *Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on*, pp. 657–662, IEEE, 2012.

[29] Bouneffouf, D., Bouzeghoub, A., and Gançarski, A. L., "Hybrid-ε-greedy for mobile context-aware recommender system," in *Advances in Knowledge Discovery and Data Mining*, pp. 468–479, Springer, 2012.

[30] Bouneffouf, D., Bouzeghoub, A., and Gançarski, A. L., "Contextual bandits for context-based information retrieval," in *Neural Information Processing*, pp. 35–42, Springer, 2013.

[31] Bouneffouf, D., Bouzeghoub, A., Gançarski, A. L., and others, "Considering the high level critical situations in context-aware recommender systems," *IMMoA'12: 2nd International Workshop on Information Management for Mobile Applications*, vol. 908, pp. 26–32, 2012.

[32] Bouzeghoub, A., Do, K., and Lecocq, C., "A Situation-Based Delivery of Learning Resources in Pervasive Learning," in *Second European Conference on Technology Enhanced Learning (EC-TEL 07). Creating New Learning Experiences on a Global Scale*, (Crete, Greece), pp. 450–456, 2007.

[33] Bouzeghoub, A., Do, K., and Wives, L., "Situation-aware adaptive recommendation to assist mobile users in a campus environment," *Advanced Information Networking and Applications, International Conference on*, vol. 0, pp. 503–509, 2009.

[34] Bouzeghoub, A., Taconet, C., Jarraya, A., Do, N., and Conan, D., "Complementarity of Process-oriented and Ontology-based Context Managers to Identify Situations," in *Proc. 5th International Conference on Digital Information Management*, (Thunder Bay, Canada), June 2010.

[35] Breese, J. S., Heckerman, D., and Kadie, C., "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, UAI'98, (San Francisco, CA, USA), pp. 43–52, Morgan Kaufmann Publishers Inc., 1998.

[36] Brusilovsky, P., "Adaptive hypermedia," *User Modeling and User-Adapted Interaction*, vol. 11, pp. 87–110, Mar. 2001.

[37] BuisinessDictionary.com, 2013.

[38] Burke, R., "Hybrid recommender systems: Survey and experiments," *User Modeling and User-Adapted Interaction*, vol. 12, pp. 331–370, Nov. 2002.

[39] Casella, G. and Berger, R., *Statistical inference*. Duxbury Press Belmont, Califfornia, 1990.

[40] Castelli, G., Rosi, A., Mamei, M., and Zambonelli, F., "A simple model and infrastructure for context-aware browsing of the world," *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications*, pp. 229–238, 2007.

[41] Castro, D. D., Tamar, A., and Mannor, S., "Policy gradients with variance related risk criteria," in *Proceedings of the International Conference on Machine Learning*, 2012.

[42] Chan, N. N., Gaaloul, W., and Tata, S., "A web service recommender system using vector space model and latent semantic indexing," in *Proceedings of the 2011 IEEE International Conference on Advanced Information Networking and Applications*, AINA '11, (Washington, DC, USA), pp. 602–609, IEEE Computer Society, 2011.

[43] Chan, N. N., Gaaloul, W., and Tata, S., "A web service recommender system using vector space model and latent semantic indexing," in *Proceedings of the 2011 IEEE International Conference on Advanced Information Networking and Applications*, AINA '11, (Washington, DC, USA), pp. 602–609, IEEE Computer Society, 2011.

[44] Charalampos Vassilou, D. S. and Martakos, D., "A recommender system framework combining neural networks and collaborative filtering," in *Proceedings of the 5th WSEAS International Conference on Instrumentation, Measurement, Circuits and Systems*, pp. 285–290, Springer-Verlag, 2006.

[45] Chen, H., Perich, F., Finin, T., and Joshi, A., "Soupa: Standard ontology for ubiquitous and pervasive applications.," in *First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous 2004).*, (Boston, EU), Agosto 2004.

[46] Chernoff, H., "Sequential models for clinical trials," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Le Cam, L. M. and Neyman, J., eds.), vol. 4, pp. 805–812, University of California Press, 1967.

[47] Chu, W. and Park, S.-T., "Personalized recommendation on dynamic content using predictive bilinear models," in *Proceedings of the 18th international conference on World wide web*, WWW '09, (USA), pp. 691–700, ACM, 2009.

[48] Claudio Biancalana, Fabio Gasparetti, A. M. A. M. and Sansonetti, G., "Context-aware movie recommendation based on signal processing and machine learning," in *Context-aware recommender systems workshop*, (Chicago, Illinois, USA.), Agosto 2011.

[49] Connolly, D., Khare, R., and Rifkin, A., "The Evolution of Web Documents: The Ascent of XML," *World Wide Web Journal*, vol. 2, no. 4, pp. 119–128, 1997.

[50] Endsley, M., "Toward a theory of situation awareness in dynamic systems," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 37, pp. 32–64(33), March 1995.

[51] Even-dar, E., Mannor, S., and Mansour, Y., "Pac bounds for multi-armed bandit and markov decision processes," in *In Fifteenth Annual Conference on Computational Learning Theory*, COLT '02, pp. 255–270, 2002.

[52] Fbio Santos da Silva, L. G. P. A. and Bressan, G., "Personaltvware: An infrastructure to support the context-aware recommendation for personalized digital tv," *International Journal of Computer Theory and Engineering*, vol. 4, no. 2, pp. 131–136, 2012.

[53] Feng, Y.-H., Teng, T.-H., and Tan, A.-H., "Modelling situation awareness for context-aware decision support," *Expert Syst. Appl.*, vol. 36, pp. 455–463, Jan. 2009.

[54] Ge, Y., Xiong, H., Tuzhilin, A., and Xiao, K., "An Energy-Efficient Mobile Recommender System," 2010.

[55] Geibel, P. and Wysotzki, F., "Risk-sensitive reinforcement learning applied to control under constraints," *Journal. Artificial. Intelligence. Research.*, vol. 24, pp. 81–108, July 2005.

[56] Gittins, J., *Multi-armed bandit allocation indices.* Wiley-Interscience series in systems and optimization, Wiley, 1989.

[57] Glazebrook, K. D., "On the evaluation of suboptimal strategies for families of alternative bandit processes," *Journal of Applied Probability*, vol. 19, pp. 716–22, 1982.

[58] Godoy, D. and Amandi, A., "User profiling in personal information agents: a survey," *Knowledge Engineering Review*, vol. 20, no. 4, pp. 329–361, 2005.

[59] Grudin, J., "Partitioning digital worlds: focal and peripheral awareness in multiple monitor use," in *ACM Conference on Human Factors in Computing Systems*, pp. 458–465, 2001.

[60] Hans, A., Schneegass, D., Schäfer, A. M., and Udluft, S., "Safe exploration for reinforcement learning," in *European Symposium on Artificial Neural Networks*, pp. 143–148, 2008.

[61] Hu, R. and Pu, P., "Acceptance issues of personality-based recommender systems," in *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, (New York, NY, USA), pp. 221–224, ACM, 2009.

[62] Ishikida, T. and Varaiya, P., "Multi-armed bandit problem revisited," *The Journal of Optimization Theory and Applications*, vol. 83, pp. 113–154, Oct. 1994.

[63] Kaspi, H. and Mandelbaum, A., "Multi armed bandits in discrete and continuous time," *The Annals of Applied Probability*, vol. 8, no. 4, 1998.

[64] Kim, J.-H., Kang, U.-G., and Lee, J.-H., "Content-based filtering for music recommendation based on ubiquitous computing," in *Intelligent Information Processing III* (Shi, Z., Shimohara, K., and Feng, D., eds.), vol. 228 of *IFIP International Federation for Information Processing*, pp. 463–472, Springer US, 2007.

[65] Kivinen, J. and Warmuth, M. K., "Exponentiated gradient versus gradient descent for linear predictors," *Information and Computation*, vol. 132, 1995.

[66] Kostadinov, D., Bouzeghoub, M., and Lopes, S., "Query rewriting based on user's profile knowledge," in *Bases de Données Avancées*, BDA '07, 2007.

[67] Lai, T. L. and Robbins, H., "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[68] Lakiotaki, K., Matsatsinis, N. F., and Tsoukias, A., "Multicriteria user modeling in recommender systems," *IEEE Intelligent Systems*, vol. 26, no. 2, pp. 64–76, 2011.

[69] Lam, L. and Suen, S. Y., "Application of majority voting to pattern recognition: An analysis of its behavior and performance," *Trans. Sys. Man Cyber. Part A*, vol. 27, pp. 553–568, Sept. 1997.

[70] Lee, J. S. and Lee, J. C., "Context awareness by case-based reasoning in a music recommendation system," in *UCS*, (Springer-Verlag Berlin Heidelber), Agosto 2007.

[71] Lemos, F., Carmo, R., Viana, W., and Andrade, R., "Towards a context-aware photo recommender system," in *Context-Aware Recommender System Workshops*, 2012.

[72] Li, L., Chu, W., Langford, J., and Schapire, R. E., "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World Wide Web*, WWW '10, (USA), pp. 661–670, ACM, 2010.

[73] Li, W., Wang, X., Zhang, R., and Cui, Y., "Exploitation and exploration in a performance based contextual advertising system," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, (USA), pp. 27–36, ACM, 2010.

[74] Liang, Y., Wang, X., and Yi, Y., "One-armed bandit process with a covariate," *Annals of the Institute of Statistical Mathematics*, vol. 65, no. 5, pp. 993–1006, 2013.

[75] Lopez-Lopez, L. M., Castro-Schez, J. J., Vallejo-Fernandez, D., and Albusac, J., "A recommender system based on a machine learning algorithm for b2c portals," in *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '09, (Washington, DC, USA), pp. 524–531, IEEE Computer Society, 2009.

[76] Lops, P., Degemmis, M., and Semeraro, G., "Improving social filtering techniques through wordnet-based user profiles," in *Proceedings of the 11th international conference on User Modeling*, UM '07, (Berlin, Heidelberg), pp. 268–277, Springer-Verlag, 2007.

[77] Luce, R. D., *Individual choice behavior: A theoretical analysis*. Wiley, 2011.

[78] M. Montaner, B. L. and de la Rosa, J. L., "Improving case representation and case base maintenance in recommender systems," in *In S. Craw and A. Preece, editors, Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning*, pp. 234–248, Springer-Verlag, 2002.

[79] Mansour, E. and Hopfner, H., "Towards an XML-Based Query and Contextual Information Model in Context-Aware Mobile Information Systems," in *Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, MDM '09, (Washington, DC, USA), pp. 574–579, IEEE Computer Society, 2009.

[80] Marcus, S. I., Fernández-Gaucherand, E., Hernández-Hernandez, D., Coraluppi, S., and Fard, P., "Risk sensitive markov decision processes," *Progress in Systems and Control Theory*, vol. 22, pp. 263–280, 1997.

[81] McCarthy, J., "Situations, actions and causal laws," tech. rep., Stanford University, 1963. Reprinted in Semantic Information Processing (M. Minsky ed.), MIT Press, Cambridge, Mass., 1968, pages 410-417.

[82] Mccarthy, J. and Hayes, P. J., "Some philosophical problems from the standpoint of artificial intelligence," *Machine Intelligence*, vol. 4, pp. 463–502, 1969.

[83] Meissen, U., Pfennigschmidt, S., Voisard, A., and Wahnfried, T., "Context- and situation-awareness in information logistics," in *Proceedings of the 2004 international conference on Current Trends in Database Technology*, EDBT'04, (Berlin, Heidelberg), pp. 335–344, Springer-Verlag, 2004.

[84] Meuleau, N. and Bourgine, P., "Exploration of multi-state environments: Local measures and back-propagation of uncertainty," *Machine Learning*, vol. 35, pp. 117–154, May 1999.

[85] Michael Pazzani, Daniel Billsus, S. M. and Wnek, J., "Learning and revising user profiles: The identification of interesting web sites," in *Machine Learning*, (Boston, EU), Agosto 1997.

[86] Middleton, S. E., Shadbolt, N. R., and De Roure, D. C., "Ontological user profiling in recommender systems," *ACM Transactions on Information Systems*, vol. 22, pp. 54–88, Jan. 2004.

[87] Minsoo, K., "A formal definition of situation towards situation-aware computing," in *Visioning and Engineering the Knowledge Society. A Web Science Perspective* (Lytras, M., Damiani, E., Carroll, J., Tennyson, R., Avison, D., Naeve, A., Dale, A., Lefrere, P., Tan, F., Sipior, J., and Vossen, G., eds.), vol. 5736 of *Lecture Notes in Computer Science*, pp. 553–563, Springer Berlin Heidelberg, 2009.

[88] Mladenic, D., "Text-learning and related intelligent agents: A survey," *IEEE Intelligent Systems*, vol. 14, no. 4, pp. 44–54, 1999.

[89] Mostefaoui, S. K., "A context model based on UML and XML schema representations," in *Proceedings of the 2008 IEEE/ACS International Conference on Computer Systems and Applications*, AICCSA '08, (Washington, DC, USA), pp. 810–814, IEEE Computer Society, 2008.

[90] Mukhopadhyay, D., Dutta, R., Kundu, A., and Dattagupta, R., "A product recommendation system using vector space model and association rule," in *Proceedings of the 2008 International Conference on Information Technology*, ICIT '08, (Washington, DC, USA), pp. 279–282, IEEE Computer Society, 2008.

[91] Nicholas, I. S. C. and Nicholas, C. K., "Combining content and collaboration in text filtering," in *In Proceedings of the IJCAI99 Workshop on Machine Learning for Information Filtering*, pp. 86–91, 1999.

[92] Niederée, C., Stewart, A., Mehta, B., and Hemmje, M., "A Multi-Dimensional, Unified User Model for Cross-System Personalization," in *Workshop on Environment For Personlized Access Workshop 2004*, 2004.

[93] Palmisano, C., Tuzhilin, A., and Gorgoglione, M., "Using context to improve predictive modeling of customers in personalization applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 11, pp. 1535–1549, 2008.

[94] Papadimitriou, G. I., "A new approach to the design of reinforcement schemes for learning automata: Stochastic estimator learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 6, pp. 649–654, Aug. 1994.

[95] Pinheiro, M., *Adaptation contextuelle et personnalisée de l'information de conscience de groupe au sein des systèmes d'information coopératifs*. PhD thesis, Joseph-Fourier University, 2006.

[96] Preuveneers, D., den Bergh, J., Wagelaar, D., Georges, A., Rigole, P., Clerckx, T., Berbers, Y., Coninx, K., and de Bosschere, K., "Towards an extensible context ontology for ambient intelligence," in *In: Proceedings of the Second European Symposium on Ambient Intelligence*, pp. 148–159, Springer-Verlag, 2004.

[97] Reiter, R., *Knowlege in action: logical foundations for specifying and implementing dynamical systems*. Cambridge, MA, USA: MIT Press, 2001.

[98] Robbins, H., "Some Aspects of the Sequential Design of Experiments," in *Bulletin of the American Mathematical Society*, vol. 58, pp. 527–535, 1952.

[99] Rojsattarat, E. and Soonthornphisaj, N., "Hybrid recommendation: Combining content-based prediction and collaborative filtering," *Intelligent Data Engineering and Automated Learning*, vol. 2690, pp. 337–344, Dec. 2003.

[100] Salton, G., Wong, A., and Yang, C. S., "A vector space model for automatic indexing," *Communication. ACM*, vol. 18, pp. 613–620, Nov. 1975.

[101] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J., "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, WWW '01, (New York, NY, USA), pp. 285–295, ACM, 2001.

[102] Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M., "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '02, (New York, NY, USA), pp. 253–260, ACM, 2002.

[103] Shani, G., Rokach, L., Meisels, A., Naamani, L., Piratla, N. M., and Ben-Shimon, D., "Establishing user profiles in the mediascout recommender system," in *CIDM'07*, pp. 470–476, 2007.

[104] Stefani, A. and Strappavara, C., "Personalizing Access to Web Sites: The SiteIF Project," *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98*, June 1998.

[105] Su, X. and Khoshgoftaar, T. M., "A survey of collaborative filtering techniques," in *Advances in Artificial Intelligence*, (Boston, EU), Agosto 2009.

[106] Sutton, R. S. and Barto, A. G., *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, 1998.

[107] Takashi Shiraki, Chihiro Ito, T. O., "Large scale evaluation of multi-mode recommender system using predicted contexts with mobile phone users," in *Context-aware recommender systems workshop*, (Chicago, Illinois, USA.), Agosto 2011.

[108] Tamar, A., Castro, D. D., and Mannor, S., "Policy evaluation with variance related risk criteria in markov decision processes," *CoRR*, vol. abs/1301.0104, 2013.

[109] Tokic, M., "Adaptive epsilon-greedy exploration in reinforcement learning based on value differences," in *Proceedings of the 33rd annual German conference on Advances in Artificial Intelligence*, KI'10, (Berlin, Heidelberg), pp. 203–210, Springer-Verlag, 2010.

[110] Tsitsiklis, J. N., "A Short Proof For The Gittins Index Theorem," *The Annals Of Applied Probability*, vol. 4, no. 1, pp. 194–199, 1994.

[111] Vermorel, J. and Mohri, M., "Multi-armed bandit algorithms and empirical evaluation," in *Proceedings of the 16th European conference on Machine Learning*, ECML'05, (Berlin, Heidelberg), pp. 437–448, Springer-Verlag, 2005.

[112] Wang, X., Zhang, D., Gu, T., and Pung, H., "Ontology based context modeling and reasoning using owl," *Pervasive Computing and Communications Workshops, IEEE International Conference on*, vol. 0, p. 18, 2004.

[113] Weber, R., "On the Gittins Index for Multiarmed Bandits," *The Annals of Applied Probability*, vol. 2, no. 4, pp. 1024–1033, 1992.

[114] Webster, N. and MacKechnie, J., *Webster's New Twentieth Century Dictionary of the English Language, Unabridged: Based Upon the Broad Foundations Laid Down by Noah Webster*. Collins World, 1975.

[115] Wibowo, A., Handojo, A., and Halim, A., "Application of topic based vector space model with wordnet," in *Uncertainty Reasoning and Knowledge Engineering (URKE), 2011 International Conference on*, vol. 1, pp. 133–136, 2011.

[116] Woodroofe, M., "A one-armed bandit problem with a concomitant variable.," *Journals - American Statistical Association*, vol. 74, pp. 799–806, 1979.

[117] Wu, Z. and Palmer, M., "Verbs semantics and lexical selection," in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, (Stroudsburg, PA, USA), pp. 133–138, Association for Computational Linguistics, 1994.

[118] Xueqing Tan, P. P., "A contextual item-based collaborative filtering technology," *Intelligent Information Management*, vol. 4, no. 1, pp. 85–88, 2012.

[119] Yau, S., Huang, D., Gong, H., and Seth, S., "Development and runtime support for situation-aware application software in ubiquitous computing environments," in *In COMPSAC*, pp. 452–457, IEEE, 2004.

[120] Yau, S., Wang, Y., Huang, D., and In, H., "Situationaware contract specification language for middleware for ubiquitous computing," in *In Proceedings of the 9th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS 2003*, pp. 93–99, IEEE Computer Society Press, 2003.

[121] Ye, J., Dobson, S., and McKeever, S., "Situation identification techniques in pervasive computing: A review," *Pervasive and Mobile Computing*, vol. 8, pp. 36–66, Feb. 2012.

[122] Yu, X., Li, Y., Wang, X., and Zhao, K., "An autonomous robust fault tolerant control system," in *Information Acquisition, 2006 IEEE International Conference on*, pp. 1191–1196, 2006.

[123] Zhang, D., "Collaborative filtering recommendation algorithm based on user interest evolution," vol. 129 of *Advances in Intelligent and Soft Computing*, pp. 279–283, Springer Berlin Heidelberg, 2012.

[124] Zhang, Y. and Koren, J., "Efficient bayesian hierarchical user modeling for recommendation system," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, (New York, NY, USA), pp. 47–54, ACM, 2007.