MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABDELHAMID IBN BADIS - MOSTAGANEM

Faculté des Sciences Exactes et d'Informatique

Département de Mathématiques et informatique

Filière : Informatique

MEMOIRE DE FIN D'ÉTUDES

Pour l'Obtention du Diplôme de Master en Informatique

Option : **Ingénierie des Systèmes d'Information**

Présenté par :

Cynthia Michelle Ramaliarisoa

THEME :

Raisonnement à partir des cas pour l'aide à la décision

Soutenu le : 02/07/2023

Devant le jury composé de :

Mr C. Abdallah Bensalloua	MCA	Université de Mostaganem	Président
Mr F. Henni	MAA	Université de Mostaganem	Examinateur
Mr B. Atmani	PR	Université de Mostaganem	Encadreur
Mr A. Mansoul	MCA	Université de Skikda	Co-Encadreur

Année Universitaire 2022-2023

Dédicaces

Je tiens à dédier ce travail à mes chers parents, Mr Rahajarison RAFIANDRANA et Mme Sidonie RAMALIARISOA qui ont été d'un immense soutien tout au long de mon parcours académique. Leur amour inconditionnel a été pour moi source de motivation pour bien travailler.

Je dédie également ce travail à ma tante, Mme Micheline RAMAVONIRINA qui m'a aussi soutenue depuis le début.

Remerciements

Premièrement, je remercie le bon Dieu qui ne m’a jamais laissée tomber et m’a fortifiée durant toutes ces années d’étude en Algérie.

Je tiens aussi à remercier Mr Atmani de m’avoir encadré tout au long de ce travail, de sa bienveillance envers moi, mais également Mr Mansoul pour son aide précieuse qui a permis de bien mener ce travail.

Je tiens également à exprimer ma profonde reconnaissance envers les membres du jury, Mr C. Abdallah Bensalloua et Mr Henni, pour avoir accepté d'évaluer mon projet

Je souhaite également remercier Mme Abid, Mr Henni et tous les enseignants qui ont contribués à ma formation et qui n’ont pas hésités à me tendre leur main.

Et enfin, mes remerciements vont aussi à ceux et celles qui ont contribués de près ou de loin à la réalisation de ce mémoire de fin d’étude. Votre soutien a été essentiel et je vous en suis extrêmement reconnaissante.

خلاصة :

ستدلال القائم ع ا الة (CBR)، المعروف أيضًا باسم ستدلال عن طر ق القياس ،و عملية معرفية ستخلص ف ا المرء است تاجات أو قرارات بناءً ع حالات سابقة مماثلة. إنه ينطوي ع إيجاد أوجه ال شابه ب ن الموقف ا ا أو المش لة ا الية والمواقف السابقة المعروفة سابقًا.

يجعل CBR من الممكن مقارنة جميع ا الات ا زنة مع ال وضع ا ا للت بؤ بحل جديد والذي، بدوره، س تم إضافته كحل ل الة ا الية. عتمد ذا النوع من التفك ع حل المشكلات من خلال إيجاد حا لات مماثلة قاعدة ا الة وتكييف حلول ا مع ا الة قيد النظر. ينقسم البحث عن حالات مماثلة إ مرحلت ن: التصفية و ختيار. ناك طرق مختلفة لتحديد ا صائص ذات الصلة. يمكننا استخدام: جميع ا صائص ،عض ا صائص ا ددة الما، ا صائص ك تمي ًا، أيضًا مقياس ال شابه متضمن العملية.

ال دف من ذا المشروع و دمج طر قة تمي ية ساب ال شابه من أجل إثراء عملية صنع القرار لدينا. ذه طروحة ، سوف نركز حصر ًا ع تنفيذ التفك القائم ع ا الة (CBR).

ال لمات الدالة:

دعم القرار، دعم القرار الط، نظام دعم القرار، ستدلال القائم ع ا الة (CBR)، ا الات، ال شابه، أمراض القلب التاجية، jColibri.

Résumé

Le raisonnement à partir des cas (RàPC), également connu sous le nom de raisonnement par analogie, est un processus cognitif dans lequel on tire des conclusions ou des décisions sur la base de cas similaires antérieurs. Cela implique de trouver des similitudes entre une situation actuelle ou un problème et des situations passées déjà connues.

Le RàPC permet de comparer tous les cas mémorisés avec la situation actuelle pour prédire une nouvelle solution qui, à son tour, s'ajoutera comme solution au cas courant. Ce type de raisonnement est basé sur la résolution des problèmes en retrouvant des cas similaires dans la base de cas et en adaptant leurs solutions au cas considéré. La recherche de cas similaire se décompose en deux phases : le filtrage et la sélection. Il existe différentes façons de déterminer quelles sont les caractéristiques pertinentes. Nous pouvons utiliser : toutes les caractéristiques, certaines caractéristiques déterminantes dans le passé, les caractéristiques les plus discriminantes, etc., aussi une mesure de similarité est mise à contribution dans le processus.

L'objectif de ce projet est l'intégration d'une méthode discriminante pour le calcul de similarité afin d'enrichir notre processus d'aide à la décision. Dans ce mémoire, nous nous concentrerons exclusivement sur la mise en œuvre du Raisonnement à Partir des Cas (RàPC) pour la classification des maladies coronariennes.

Mots-clés:

Aide à la décision, Aide à la décision Médicale (ADM), Système d'aide à la décision (SAD), Raisonnement à partir des cas (RàPC), Cas, Similarité, Maladies coronariennes (CAD), JColibri.

Abstract

Case-Based Reasoning (CBR), also known as reasoning by analogy, is a cognitive process in which one draws conclusions or decisions based on previous similar cases. It involves finding similarities between a current situation or problem and previously known past situations.

The CBR makes it possible to compare all the stored cases with the current situation to predict a new solution which, in turn, will be added as a solution to the current case. This type of reasoning is based on solving problems by finding similar cases in the case base and adapting their solutions to the case considered. The search for similar cases is divided into two phases : filtering and selection. There are different ways to determine which characteristics are relevant. We can use: all the characteristics, certain defining characteristics in the past, the most discriminating characteristics, also a measure of similarity is involved in the process.

The objective of this project is the integration of a discriminating method for calculating similarity in order to enrich our decision-making process. In this manuscript, we will focus exclusively on the implementation of Case-Based Reasoning (CBR).

Keywords:

Decision support, Medical decision support, Decision Support System, Case-Based Reasoning(CBR), cases, similarity, coronary Heart Disease, jColibri.

Liste des figures

Liste des tableaux

Liste des abréviations

Abréviation	Expression Complète	Page
RàPC	Raisonnement à Partir des Cas	5
CAD	Coronary Artery Disease	5
IA	Intelligence Artificielle	8
SADM	Système d'Aide à la Décision Médicale	9
SAD	Système d'Aide à la Décision	10
SIH	Système d'Information Hospitalier	10
k-PPV	K Plus Proches Voisins	11
RNA	Réseau de Neurones Artificiels	14
SMA	Système Multi-Agents	29

Table des matières

Introduction Générale

Les maladies coronariennes (CAD) constituent un problème de santé majeur dans le monde entier, nécessitant des stratégies de prise de décision médicale efficaces et précises. L'intégration de méthodes avancées dans le processus d'aide à la décision peut jouer un rôle crucial pour améliorer la pertinence et la fiabilité des recommandations cliniques. Dans ce projet, nous proposons d'enrichir le processus de remémoration des cas (maladies coronariennes), guidé par l'outil jColibri qui implémente le Raisonnement à partir des Cas (RàPC), en intégrant la règle de Bayes, une méthode discriminante pour le calcul de similarité.

L'objectif principal de cette étude est d'explorer comment l'intégration de la règle de Bayes peut améliorer le calcul de similarité dans le processus de remémoration des maladies coronariennes. La règle de Bayes est une méthode statistique puissante qui permet d'estimer la probabilité d'un événement en utilisant des connaissances a priori. En l'intégrant dans l'outil jColibri, nous cherchons à améliorer la précision des remémorations en tenant compte de l'information contextuelle et des probabilités conditionnelles.

Pour atteindre cet objectif, une revue approfondie de la littérature est réalisée pour examiner l'utilisation de la règle de Bayes dans le domaine médical, ainsi que les techniques existantes de calcul de similarité. Nous évaluerons ensuite l'applicabilité de cette méthode discriminante dans le contexte du RàPC implémenté par l'outil jColibri.

En intégrant la règle de Bayes dans le processus de remémoration des maladies coronariennes implémenté par l'outil jColibri, nous visons à améliorer la précision et la pertinence des recommandations cliniques. Cette approche permettra d'identifier de manière

plus précise les cas similaires et d'optimiser les plans de prévention et de traitement pour les patients atteints de maladies coronariennes. Nous prévoyons également que l'intégration de cette méthode discriminante contribuera à réduire les erreurs de diagnostic et à améliorer les résultats cliniques.

Notre contribution dans ce projet se concentre spécifiquement sur la prédiction des maladies coronariennes par le biais du RàPC en utilisant l'outil jColibri. L'objectif principal de cette étude est d'explorer comment le RàPC, mis en œuvre à travers l'outil jColibri, peut être utilisé pour améliorer la précision et la fiabilité de la prédiction des maladies coronariennes.

En se basant sur une approche centrée sur les cas, le RàPC permet d'exploiter les connaissances et les expériences passées pour établir des relations entre les cas similaires et en déduire des recommandations appropriées. L'utilisation de l'outil jColibri facilite cette démarche en fournissant une plateforme robuste pour la gestion des cas, l'extraction des caractéristiques pertinentes, le calcul de similarité et la prédiction des maladies coronariennes par la méthode des k plus proches voisins.

Notre mémoire est structuré en trois chapitres qui couvrent différents aspects de la prise de décision médicale et de l'application du RàPC en utilisant l'outil jColibri.

Dans le premier chapitre « Aide à la décision médicale », nous introduisons le concept d'aide à la décision médicale et son importance dans le domaine de la santé. Nous explorons les principes fondamentaux de l'aide à la décision et les méthodes couramment utilisées.

Le deuxième chapitre se concentre sur le RàPC en tant qu'approche de prise de décision. Nous présentons en détail les principes et les concepts clés du RàPC, ainsi que les étapes du processus de raisonnement à partir des cas. Nous discutons des avantages et des limites du RàPC, en mettant en évidence son application dans la prédiction des maladies coronariennes.

Le troisième chapitre porte sur la conception et l'implémentation du processus RàPC complet en utilisant l'outil jColibri. Nous commençons par présenter jColibri et ses fonctionnalités clés. Ensuite, nous détaillons la conception du processus RàPC spécifique aux maladies coronariennes. Nous décrivons ensuite l'implémentation pratique du processus RàPC en utilisant jColibri, en configurant les paramètres et les fonctionnalités nécessaires. Enfin, nous évaluons les performances du processus RàPC avec jColibri, en analysant les résultats obtenus et en discutant des implications et des perspectives d'amélioration.

Enfin, nous terminons ce mémoire par une conclusion et quelques perspectives.

Chapitre 1
Aide à la décision médicale

1.1 Introduction

Depuis plus d'une décennie, l'intelligence artificielle (IA) vit une accélération dans son développement et son adoption. En médecine, elle intervient dans la recherche fondamentale et clinique, la pratique hospitalière, les examens médicaux, les soins ou encore la logistique. Ce qui contribue à l'affinement des diagnostics et des pronostics, à une médecine encore plus personnalisée et ciblée, à des avancées dans les technologies d'observations et d'analyses ou encore dans les outils d'interventions chirurgicales et autres robots d'assistance. Dans ce chapitre, nous allons parler de l'aide à la décision médicale, qui fait intervenir des notions issues de l'IA utilisées dans le domaine médical.

1.2 Aide à la décision

La décision est un concept très riche, où on peut trouver plusieurs éléments ontologiques tels que : l'objet et l'organe de la décision, le type de prise de décision (routine, créatif, etc.), la portée de la décision (stratégique, tactique, opérationnelle, etc.) et le contrôle des éléments de la décision. [1]

Cependant le concept aide à la décision est l'activité de celui qui, prenant appui sur des modèles clairement explicités mais non nécessairement clairement formalisés, aide à obtenir des éléments de réponse aux questions que se pose un intervenant dans un processus de décision, éléments concourants à éclairer la décision et normalement à prescrire, ou simplement à favoriser, un comportement de nature à accroître la cohérence entre l'évolution d'un processus d'une part, les objectifs et le système de valeurs au service desquels cet

intervenant se trouve placé d'autre part. Elle ne consiste que partiellement en une recherche de la vérité, mais est plus souvent utilisée comme une aide à la réflexion et à la communication destinée au décideur, elle l'aide à construire et à faire partager ses convictions [2]. L'activité d'aide à la décision s'articule autour d'un processus de décision qui est un ensemble d'activités déclenché par un stimulant, et en aboutissant à un engagement spécifique à l'action, il est présenté selon les quatre artéfacts cognitifs suivants : [3]

- Représentation du problème
- Formulation du problème
- Modèle d'évaluation
- Recommandation finale

Donc, La prise de décision a toujours été un problème pour les décideurs. Au moment où la prise de décision est particulièrement critique, il est important d'éliminer les différents facteurs tels que le stress, le bruit, la maladie ou la fatigue afin d'obtenir à la fin une décision précise et vraie. Même dans les décisions prises dans les cas appropriés, différents facteurs environnementaux peuvent entraîner des résultats imprévus par la suite.

1.3 Aide à la décision médicale

Dans cette section, nous allons voir la notion de décision médicale, puis parler des systèmes d'aide à la décision médicale (SADM) ainsi que les différents types de SADM qui existent. …

1.3.1 Notion décision médicale

En médecine, les professionnels de la santé utilisent plus d'un million d'éléments d'information pour une prise en charge de leurs patients et près d'un tiers de leur temps est passé à enregistrer et synthétiser de l'information. Par conséquent, la prise de décision en médecine n'est pas une tâche aisée et nécessite la prise en compte des paramètres cliniques du

patient, du contexte de la maladie et de leurs connaissances médicales pour évoquer et/ou confirmer un diagnostic ou une conduite à tenir. En outre, leur sixième sens ne doit jamais être remplacé par une machine et donc les systèmes d'aide à la décision dans le domaine médical peuvent néanmoins fournir une alerte, suggestion ou une recommandation pour aider les professionnels de la santé à devenir plus efficaces pour une meilleure prise en charge des patients. [4]

1.3.2 Système d'aide à la décision médicale (SADM)

De manière générale, les systèmes d'aide à la décision ou SAD peuvent être utilisés dans le domaine médical, où il est impératif de faire fonctionner ces systèmes pour garantir des solutions efficaces pour des problèmes médicaux. De même, le concept de SAD est largement utilisé dans les applications médicales et largement étudié par le public scientifique. Nous pouvons dire que les SADM sont des outils informatiques basés sur des applications logicielles qui peuvent aider les professionnels de la santé à prendre les meilleures décisions. Une utilisation courante des SADM consiste à analyser les données des patients passées, actuelles et nouvelles et à identifier ou suggérer des lacunes, des erreurs, des problèmes de sécurité ou des améliorations du parcours de soins pour l'utilisateur.

1.3.3 Types de systèmes d'aide à la décision médicale

Les SADM ont été développés et beaucoup d'entre eux ont été utilisés en tant que systèmes autonomes ou faisant partie des Systèmes d'Information Hospitalier ou SIH locaux. Ainsi et depuis l'émergence des SADM et les forts intérêts qu'ils apportent au domaine médical, plusieurs systèmes ou prototypes ont été développés et proposés dans la littérature selon les besoins des professionnels de la santé. [5]

1.3.3.1 Selon la représentation des connaissances

Selon différents travaux, deux types de SADM ont été proposés, des systèmes ou approches basés sur les connaissances et des systèmes ou approches basés sur les données. [6]

1.3.3.1.1 SADM basé sur les connaissances

La base de connaissances du système d'aide à la décision contient des informations sur les maladies, leurs signes et leurs symptômes. Le moteur d'inférence établit une correspondance entre les signes, les symptômes et les maladies du patient ; et peut suggérer des diagnostics à prendre en compte par les médecins. Ces systèmes ne génèrent généralement pas qu'un seul diagnostic, mais génèrent généralement un ensemble de diagnostics basés sur les informations disponibles. Étant donné que le clinicien en sait souvent plus sur le patient que ce qui peut être mis dans l'ordinateur, le médecin pourra éliminer certains des choix.

1.3.3.1.2 SADM basé sur les données

Contrairement aux SADM basés sur les connaissances, certains de ces systèmes basés sur les données utilisent une forme d'intelligence artificielle appelée apprentissage machine qui permet à l'ordinateur d'apprendre des expériences passées et / ou de reconnaitre des modèles dans les données médicales. La recherche sur l'efficacité du SADM provient en grande partie de quelques institutions où ces systèmes ont été développés, bien qu'au cours des dernières années, à mesure que les systèmes commerciaux se sont répandus, il existe une littérature croissante sur leur efficacité dans divers contextes. Naïve bayes, les arbres de décisions et les K-plus proches voisins (ou k-PPV) sont parmi les types d'approches utilisés dans la littérature des SADM fondés sur les données en médecine. [7]

Les k-plus proches voisins, est une méthode d'apprentissage à base d'instances. Elle ne comporte pas de phase d'apprentissage en tant que telle. Les instances faisant partie de l'ensemble d'apprentissage sont seulement enregistrées. Lorsqu'une nouvelle instance à

classer arrive, elle est comparée aux instances d'apprentissage à l'aide d'une mesure de similarité. Ses k plus proches voisins sont alors considérés : nous observons leur catégorie et celle qui revient le plus parmi les voisins est affectée à l'instance à classer. C'est là une version de base de l'algorithme que l'on peut affiner. Cette technique a été utilisée dans diverses applications en médecine.

1.3.3.2 Selon le mode d’intervention

1.3.3.2.1 Assistance documentaires

Les SADM d'assistances documentaires sont des systèmes à base de données dont leurs objectifs sont d'accroître les connaissances des professionnels de la santé en donnant l'accès à la documentation médicale comme les systèmes de recherche d'information médicale. Cependant, cette aide à la décision reste au stade classique du stockage et retrait de l'information. [1]

1.3.3.2.2 Consultants

Les SADM consultant sont des systèmes qui utilisent les techniques d'IA dont le but est de fournir des avis ou conseils sur des diagnostics ou des traitements basés sur les données des patients. Les systèmes experts font partie de cette catégorie. Ces SADM raisonnent sur des situations médicales définies préalablement et fournissent aux professionnels de la santé des conclusions argumentées selon les méthodes de raisonnement employées. [1]

1.3.3.3 Selon le mode de raisonnement

Le raisonnement est une véritable puissance et une tâche importante effectuée par le moteur d'inférence des SADM, qui combine les connaissances médicales avec des données spécifiques au patient pour générer des décisions pertinentes. Les méthodologies de raisonnement fournissent des outils et des techniques puissants pour manipuler les connaissances pour faire une inférence et une décision ainsi que pour résoudre le problème. [1]

1.3.3.3.1 Raisonnement à base de règles

Ce mode de raisonnement repose sur des règles de types SI-ALORS, le raisonnement à base de règles également connu sous le nom des systèmes experts a été largement utilisé dans le domaine médical dans ses différents aspects. Un SADM basé sur des règles a été mis en place pour prédire le Covid-19 et le système expert a pu facilement pré-cribler les patients sans l'intervention d'aucun deuxième individu. [8]

1.3.3.3.2 Raisonnement à base de cas

Le paradigme du raisonnement à base de cas ou le RàPC a une base d'expériences passées, appelée base de cas, et tente de résoudre de nouveaux problèmes en récupérant des solutions similaires dans cette base et en les adaptant à de nouveaux problèmes. Le RàPC, dans une certaine mesure, imite le comportement humain dans des activités comme la gestion et le diagnostic dans lesquelles les connaissances et l'expérience antérieures sont le moteur de la recherche de solutions pour de nouveaux problèmes quasi équivalents. En médecine, le RàPC est un mode de raisonnement utilisé car souvent, les connaissances médicales sont difficiles à modéliser avec les méthodes de représentation formelles. Les conditions essentielles pour le succès de ce mode de raisonnement dans les SADM sont l'application des bonnes mesures de similitude et les mécanismes efficaces de rechercher des cas similaires.

1.3.3.3.3 Raisonnement à base d’ontologie

Ce mode de raisonnement utilise l'ontologie pour la représentation des connaissances car cette structure est mieux adaptée pour encapsuler les concepts et les relations entre les termes associés à la médecine, c'est-à-dire qu'elle est capable de recueillir les connaissances médicales d'une manière formelle, ce qui permet de les partager et de les réutiliser chaque fois que cela est nécessaire. [9]

1.3.3.3.4 Raisonnement à base de modèles

Ce mode de raisonnement est utilisé par exemple pour diagnostiquer un artefact en observant son comportement manifesté et en le comparant au comportement prédit du modèle de l'artefact. [10]

1.3.3.3.5 Raisonnement Bayésien

Ce mode de raisonnement est basé sur des probabilités conditionnelles et prédit la probabilité postérieure d'un évènement spécifique compte tenu de l'occurrence d'un autre évènement. [11]

1.3.3.3.6 Réseaux de neurones

Les réseaux de neurones artificiels ou RNA ont été utilisés dans le diagnostic de diverses affections telles que le cancer colorectal, la maladie du pancréas, le diabète précoce et le cancer du côlon. Le RNA a également été utilisé dans le domaine de la cardiologie et de la pédiatrie. Des RNA ont été utilisés dans la recherche pour les bios marqueurs et dans le traitement d'image médicale en utilisant l'apprentissage profond. [7]

1.3.3.3.7 Réseaux sémantiques

Ce mode de raisonnement dans les SADM repose sur l'utilisation des réseaux sémantiques qui sont une représentation graphique déclarative dans des modèles de nœuds et d'arcs interconnectés. [12]

1.4 Aide à la décision pour les maladies coronariennes (CAD)

L'aide à la décision pour les maladies coronariennes (CAD) est un domaine essentiel dans la pratique médicale, car il s'agit d'une pathologie cardiovasculaire fréquente et potentiellement grave. Les systèmes d'aide à la décision médicale dans le contexte des maladies coronariennes visent à fournir aux professionnels de la santé des informations pertinentes et des recommandations pour optimiser les décisions cliniques liées à la prévention, au diagnostic, au traitement et à la gestion des patients atteints de CAD.

Ces systèmes utilisent différentes approches pour soutenir les médecins dans leur prise de décision. Avant de voir quelques exemples d'aides à la décision pour les maladies coronariennes, nous allons définir ce qu'est une maladie coronarienne.

1.4.1 Définition maladie coronarienne

La maladie coronarienne est une maladie qui touche les artères ayant pour fonction d'alimenter le cœur en sang (artères coronaires). Elle est souvent causée par l'athérosclérose, une accumulation de plaques à l'intérieur de la paroi des artères. Cette accumulation rétrécit peu à peu l'intérieur des artères et ralentit le flot de sang. [13]

1.4.2 Évaluation du risque cardiovasculaire

Les outils d'aide à la décision permettent d'évaluer le risque cardiovasculaire global d'un patient en prenant en compte divers facteurs tels que l'âge, le sexe, les antécédents familiaux, les facteurs de risque (hypertension, diabète, tabagisme, etc.) et les résultats d'examens médicaux. Ces évaluations aident à identifier les patients à haut risque et à prendre des mesures préventives appropriées.

1.4.3 Diagnostic du CAD

Les systèmes d'aide à la décision peuvent aider les médecins à diagnostiquer précisément les maladies coronariennes en analysant les symptômes, les antécédents médicaux, les résultats d'examens (électrocardiogramme, tests d'effort, imagerie cardiaque, etc.) et les biomarqueurs spécifiques. Ils peuvent également aider à stratifier les patients en fonction de la sévérité de la maladie et des risques associés.

1.4.4 Choix du traitement

Les systèmes d'aide à la décision fournissent des recommandations sur les options thérapeutiques disponibles pour les patients atteints de CAD, en tenant compte des caractéristiques individuelles du patient, des comorbidités et des résultats d'examens. Ils

peuvent aider à déterminer les médicaments appropriés, les interventions chirurgicales (angioplastie, pontage coronarien) ou d'autres approches de traitement.

1.4.5 Suivi et gestion des patients

Les outils d'aide à la décision aident à surveiller les patients atteints de CAD, en fournissant des rappels pour les visites de suivi, les examens médicaux réguliers et les ajustements de traitement. Ils peuvent également aider à évaluer la réponse au traitement, à détecter les complications potentielles et à recommander des modifications de l'approche thérapeutique si nécessaire.

L'aide à la décision pour les maladies coronariennes vise à intégrer les connaissances médicales actuelles, les directives cliniques et les données probantes dans un format facilement accessible et utilisable par les professionnels de la santé. Ces outils visent à améliorer la prise de décision, la qualité des soins, l'efficacité des traitements et les résultats cliniques pour les patients atteints de CAD.

1.5 État de l’art sur l’utilisation des SADM

Dans cette section, nous allons voir quelques SADM qui ont été élaborés pour aider les personnels de la santé.

1.5.1 SADM pour la détection des erreurs médicamenteuses et les évènements indésirables médicamenteux [14]

Dans ce travail, après avoir intégré dans les dossiers médicaux électroniques des alertes automatisées, les SAD ont démontré leur efficacité pour détecter les erreurs de médication et les évènements indésirables médicamenteux. En tenant connaissance que les patients gravement malades courent un risque accru à ces deux évènements et à des résultats négatifs liés à ces évènements, tirer parti du SAD permet l’amélioration des résultats pour ces

patients. Le SAD ici consiste à faire avancer les signaux d'alerte en utilisant des analyses de trajectoire pour prédire les évènements cliniques, au lieu d'attendre que ces évènements se produisent.

1.5.2 SADM pour la gestion de la ventilation mécanique [15]

La ventilation mécanique est un traitement efficace en unité de soins intensifs mais elle peut avoir des effets indésirables importants. C'est pourquoi, des SAD sont conçus pour rendre la gestion de la ventilation mécanique plus sûre, plus cohérente et plus protectrice des poumons tout en mettant en œuvre des protocoles explicites. Les outils SAD donne des résultats comparables aux cas dans lesquels la ventilation a été initiée par des experts. Ils sont particulièrement utiles dans les situations d'urgence ou lorsque des cliniciens moins expérimentés doivent initier un traitement par ventilateur.

1.5.3 SADM pour l'amélioration du respect des directives transfusionnelles [16]

La capacité de transfuser en toute sécurité des produits sanguins sauve sans aucun doute des vies. Cependant, malgré les avancées scientifiques en médecine transfusionnelle, des facteurs de risque importants demeurent inhérents aux transfusions de produits sanguins. Le SAD ciblant la transfusion de produits sanguins a réussi à améliorer le respect des lignes directrices fondées sur des données probantes. Des preuves solides de l'utilisation de SAD pour améliorer la pertinence de l'utilisation des transfusions de globules rouges et de plasma ont été démontrées.

1.6 Conclusion

Nous venons de voir que l'intelligence artificielle a un rôle à jouer dans le monde qui est, technologiquement parlant, avancé. Plusieurs notions issues de l'IA entrent en jeu dans l'univers des entreprises et surtout dans celui de la médecine dû au fait que les données augmentent exponentiellement et que la prise de décision est importante pour les personnels

qui travaillent dans la médecine. Nous avons vu ce que les systèmes d'aide à la décision ont leur place dans le domaine médical et qu'il en existe plusieurs types, adapté chacun pour un type de maladie. Dans le prochain chapitre, nous allons plus nous focaliser dans les systèmes d'aide à la décision qui utilisent le raisonnement à partir des cas et nous allons entrer plus en profondeur dans ce type de raisonnement.

Chapitre 2
Raisonnement à partir des cas(RàPC)

2.1 Introduction

Dans le premier chapitre, nous avons parlé des systèmes d'aide à la décision médicale. Nous avons évoqué certaines méthodes issues de l'intelligence artificielle utilisées dans les SADM. Une de ces méthodes est le raisonnement à partir des cas ou le RàPC. Les méthodologies de raisonnement fournissent des outils et des techniques puissants pour manipuler les connaissances dans le but d'aboutir vers une décision afin de résoudre des problèmes. Dans ce chapitre, nous allons aborder les différents aspects du raisonnement à partir des cas en commençant tout d'abord par la définition du RàPC, puis de la notion de cas et expériences pour entamer ensuite quelques RàPC utilisés dans SADM auparavant et enfin parler du processus de RàPC lui-même.

2.2 Définition du RàPC

Le RàPC, est un paradigme de l'intelligence artificielle et des sciences cognitives qui modélise le processus de raisonnement comme étant principalement basé sur la mémoire. [17]

Ceux qui utilisent ce raisonnement résolvent de nouveaux problèmes en récupérant des « cas », une notion que nous allons décrire plus tard dans notre travail, stockés qui décrivent un ensemble de résolution de problèmes antérieurs similaires et en adaptant les solutions de ces problèmes aux nouveaux problèmes rencontrés. [17]

2.3 Cas

Comme nous avons cité en haut, le RàPC fait intervenir la notion de cas. Nous pouvons même dire que le RàPC est essentiellement basé sur les cas, c'est pourquoi nous allons voir en détail dans ce qui suit ce que c'est le cas et les éléments qui le constitue.

2.3.1 Définition de cas

Un cas est une expérience représentée par une connaissance. Cette expérience constitue une leçon permettant au système de RàPC de résoudre des problèmes de différentes natures. Selon le domaine d'application et les objectifs à atteindre, les informations contenues dans le cas varient. Nous pouvons définir un cas comme étant la description informatique d'un épisode de résolution de problème. [18]

Un cas contient généralement deux parties : la description du problème, la solution au problème, et auxquelles peut s'ajoute une troisième partie : le résultat de la solution. [19]

2.3.1.1 Description du problème

Une description du problème contient toutes les informations descriptives nécessaires à la résolution du problème, les objectifs à atteindre. C'est le problème qui déclenche le processus du RàPC.

2.3.1.2 Solution au problème

La partie solution décrit comment le problème a été résolu et elle regroupe la description de la solution apportée par le raisonnement, sa justification, son évaluation ainsi que les étapes qui ont mené à cette solution. Une description de la solution permet de réutiliser une solution précédente sans repartir à zéro quand un cas similaire arrive.

La manière de représenter une solution varie. La solution peut être simplement une solution au sens strict tout comme elle peut contenir des commentaires, illustrations, actions ou des conseils sur l'utilisation de la solution.

2.3.1.3 Résultat de la solution

La description de la solution contient les explications de ce qui a été réalisé, si c'était un échec ou un succès, la stratégie de répartition en cas d'échec et comment éviter l'échec.

Le RàPC prend en considération l'ensemble des expériences réussies et ratées : celles qui sont réussies mettent en œuvre des solutions réussies et aboutissent à des conseils comme refaire l'expérience. Tandis que celles qui sont ratées mettent en œuvre des solutions échouées et conseillent d'éviter cette expérience. Ce qui nous donne deux ensembles de cas : C + (positifs) et C – (négatifs).

Nous avons deux types de cas : cas source et cas cible. Dans le cas source, les parties problème et solution sont renseignées tandis que dans le cas cible, seule la partie problème est renseignée.

2.3.2 Représentation des cas

Les humains partagent leurs connaissances de diverses manières. Il existe trois couches pour représenter les connaissances, comme nous le voyons dans la figure ci-dessous.

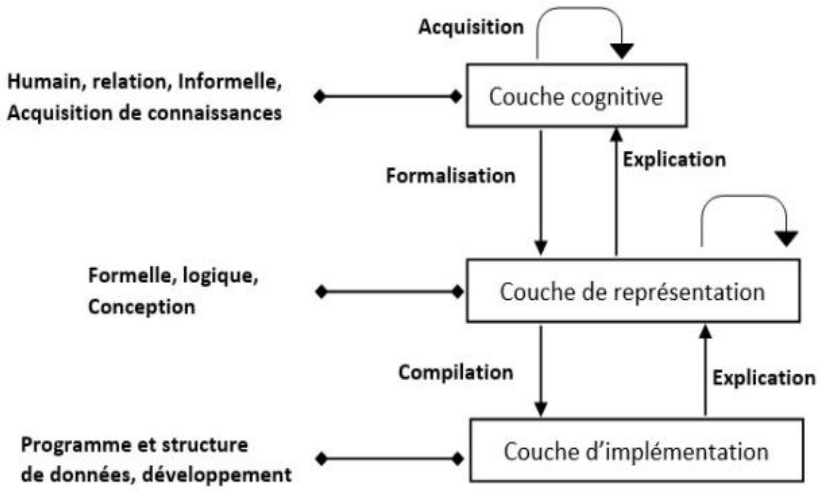

Figure 1 - Couches de représentation [1]

Pour manipuler donc ces cas, c'est-à-dire des problèmes et des solutions, dans une implémentation, la précision est nécessaire. Ainsi, le moyen le plus simple que les chercheurs ont trouvé pour représenter un cas est d'utiliser des paires descripteur-valeur.

2.3.3 Base de cas

Les cas sont stockés dans ce que nous appelons base de cas. La base de cas est une mémoire contenant un ensemble de cas utilisés dans le cycle du RàPC, dans le but de trouver une solution à un problème. C'est donc une source de données, en général finie. De manière simple, la base de cas contient tous les cas sources qui vont nous inspirer pour résoudre un nouveau problème dans notre RàPC. C'est le centre d'un système de RàPC.

Nous pouvons distinguer trois types d'organisation possibles de la base de cas :

- Organisation plate : les cas sont organisés de manière linéaire comme vecteur, tableau, etc. Elle est adaptée à un petit nombre de cas et où les cas n'ont pas de relation entre eux.

- Organisation structurée : ayant comme structure les hiérarchies et les réseaux.

- Organisation non structurée : les cas dans les textes et les images.

Pour créer une base de cas, trois points doivent être considérés : [18]

- La structure et la représentation des cas ;

- Le modèle de la mémoire utilisée pour organiser la base de cas ;

- La sélection des indices qui sont utilisés pour identifier chaque cas afin de faciliter l'organisation et la recherche du cas le plus approprié au problème posé. Ces indices peuvent être indexés de manière manuelle ou automatique.

2.4 Processus du RàPC

Le processus du RàPC est une méthodologie cyclique composée de cinq phases qui sont l'élaboration du problème, la remémoration (ou la recherche de cas similaires), l'adaptation, la révision du cas sélectionné et la mémorisation (ou l'apprentissage) comme nous pouvons voir sur la figure ci-dessous. [20]

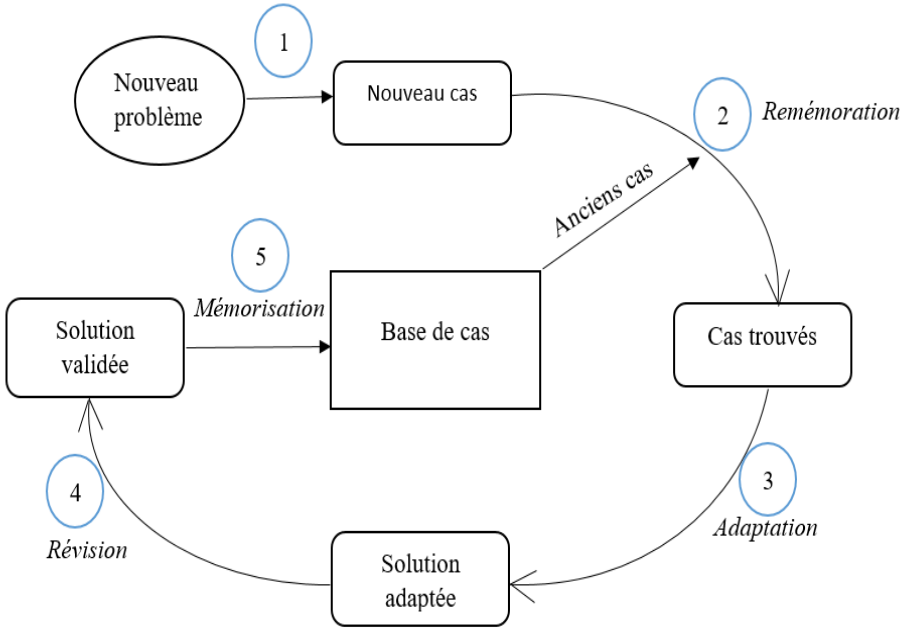

Figure 2 - Cycle du RàPC

2.4.1 Élaboration du problème

Elle se fait régulièrement au début du cycle du RàPC. C'est une tâche qui part de la nécessité d'obtenir un nouveau problème de la part d'un utilisateur. Les utilisateurs doivent saisir le nouveau problème comme les cas présents dans la base des cas, mais ce n'est pas souvent possible donc nous ne pouvons pas décrire les problèmes immédiatement comme des cas.

L'élaboration de problème, ce que nous appelons problème de génération de requête peut se faire de plusieurs manières, ayant comme but d'acquérir le moins d'informations possible mais suffisantes pour le résoudre ce problème de requête. Il y a deux manières de procéder :

- Utiliser une formulation spécifique, éventuellement standardisée du problème.
- Dialoguer avec l'utilisateur.

2.4.2 Remémoration

La remémoration ou encore appelée la recherche de cas similaires a pour but de déterminer les cas qui sont les plus similaires au nouveau problème. La partie problème du cas cible est alors utilisé à titre indicatif et la base de cas est l'espace de recherche. La recherche commence lorsque le nouveau problème est facilement disponible et se termine quand un cas est remémoré c'est-à-dire trouvé, devenant disponible pour la tâche suivante du processus.

En RàPC, la recherche est fortement liée à la représentation des cas, de leur indexation et à la mesure de similarité. Plusieurs techniques, méthodes et mesures de similarités ont été utilisées pour la remémoration des cas dans un système de RàPC. Nous allons les voir plus tard dans notre travail.

2.4.3 Adaptation

Elle est également appelée la réutilisation du cas trouvé. Elle consiste à proposer une solution pour résoudre le nouveau problème en réutilisant les solutions du ou des cas remémorés. Elle est assez simple lorsque le nouveau problème est identique au problème de cas remémorés. Mais nécessite une adaptation lorsqu'elle diffère. Elle peut se faire manuellement ou automatiquement à l'aide d'algorithmes, de méthodes, de règles, etc.

Voici quelques types d'adaptation automatique : [18]

- L'adaptation générative : dans le cas où nous disposons de toutes les connaissances pour résoudre le problème à partir de zéro. Le cas alors retrouvé retrace le raisonnement ayant mené à la solution ;
- L'adaptation transformationnelle : dans le cas où nous ne disposons pas toutes les connaissances pour résoudre le problème à partir de zéro ;
- L'adaptation compositionnelle : c'est le fait de composer les différentes solutions proposées par deux ou plusieurs cas similaires remémorés ;
- L'adaptation hiérarchique : les cas sont organisés en plusieurs niveaux dans la hiérarchie de généralisation.

2.4.4 Révision

La révision ou la validation vise à évaluer l'applicabilité de la solution proposée. Les évaluations peuvent être effectuées dans le monde réel ou dans une simulation. La simulation est plus simple et moins coûteuse mais peut négliger des aspects pratiquement importants.

2.4.5 Mémorisation

La mémorisation ou l'apprentissage consiste à mettre à jour la base de cas en ajoutant le nouveau cas (cas cible) pour la résolution de problèmes futurs. Le succès tout comme l'échec dans la résolution de ce cas cible peut être ajouté dans la base de cas pour l'enrichir, permettant ainsi d'augmenter l'expérience du système.

2.5 Distance, mesure de similarité et recherche de cas similaires

Comme nous avons vu dans le cycle du RàPC, plus particulièrement dans la phase de remémoration, nous avons besoin de rechercher des cas similaires à notre nouveau cas dans la base de cas. Pour cela, dans cette partie, nous allons voir quelques méthodes et algorithmes qui nous permettront cette recherche de cas similaires.

2.5.1 Distances

Les approches à base de distance sont les plus utilisées dans les applications de RàPC pour le calcul de similarité entre les descripteurs des cas. Nous allons voir quelques distances utilisées selon le type de données. Supposant que nous avons deux cas représentés par deux vecteurs $X = x_1, x_2,\ldots, x_k$ et $Y = y_1, y_2,\ldots, y_k$ où k est le nombre de descripteurs.

- Données quantitatives

Distances	Formules
Euclidienne	$\sqrt{\sum_{i=1}^{k}(\lvert X_i\text{-}Y_i\rvert)^2}$
Manhattan	$\sum_{i=1}^{k}(\lvert X_i\text{-}Y_i\rvert)$

Tableau 1 - Distances pour les données quantitatives [1]

- Données qualitatives

Les distances utilisées peuvent être des distances qui donnent la même importance pour tous les attributs comme la distance de Hamming et le Simple Matching Coefficient (SMC) ou alors des distances qui donnent une importance pour certains attributs par rapport aux autres comme la SMC pondérée, le coefficient de Jaccard, etc.

2.5.2 Mesure de similarité

La recherche des cas similaires au problème à résoudre est basée sur le concept des mesures de similarité. Les mesures de similarité cherchent des correspondances entre les

descripteurs problèmes des cas dans la base de cas et ceux du nouveau cas à l'aide d'un algorithme de recherche.

La mesure de similarité est une somme pondérée de calcules locaux de similarité pour chacun des descripteurs possibles des cas. Les mesures de similarité peuvent être locales, c'est-à-dire calculées entre les descripteurs de deux cas, et généralement basées sur la notion de distance ; ou globales et sont calculées au niveau des cas en agrégeant les similarités locales.

2.5.3 Recherche de cas similaires

Un des algorithmes utilisés pour la recherche de cas similaires est l'algorithme des k plus proches voisins (k-PPV). C'est un algorithme d'apprentissage automatique supervisé qui attribue une catégorie à un élément en fonction de la classe majoritaire de ses k plus proches voisins dans l'échantillon d'entraînement. [21]

Pour mieux comprendre cet algorithme, nous allons voir la figure ci-dessous qui est une représentation graphique d'une table de 16 données, chaque donnée ayant chacune 2 caractéristiques (force et courage) de valeurs entières et comme résultat nous avons deux classes possibles : Chevalier et Fantassin. Et nous allons introduire une nouvelle donnée appelée cible avec les caractéristiques (12 ; 12,5) et nous allons chercher ses 7 plus proches voisins, c'est-à-dire k=7.

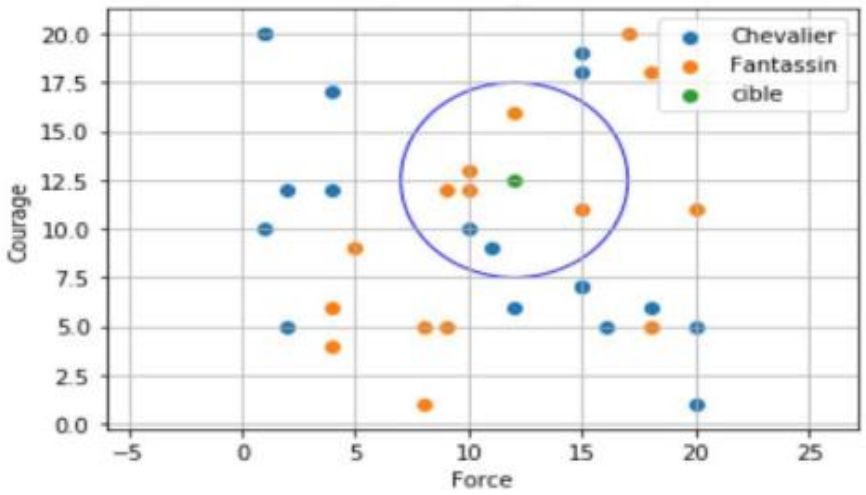

Figure 3- Représentation graphique pour illustrer k-PPV [21]

Nous pouvons déduire de la figure que notre cible est de la classe Fantassin vu que parmi les 7 plus proches voisins, le nombre de Fantassins est égal à 5 or que celui des Chevaliers est seulement 2.

2.6 État de l'art sur l'aide à la décision médicale par RàPC

La médecine a eu recours au RàPC en raison de l'immense quantité de données accumulées au cours des années, dans de grandes bases de données. Sans l'usage de puissants outils d'analyse, l'homme n'aurait pas pu à lui seul exploiter et tirer profit de la masse considérable de données. Ainsi, l'intégration du RàPC dans le domaine médical fut un grand apport aux processus de traitement des données médicales. En effet, le RàPC offre aux praticiens un outil d'exploitation des situations médicales passées pour aider à résoudre des situations médicales souvent délicates à résoudre.

Cependant malgré le succès du RàPC, il présenta quelques limites qui ont poussé à investir d'autres axes afin de les surmonter et la notion d'intégration du RàPC avec d'autres techniques a vu le jour, ainsi les méthodes de fouille de données ont été largement utilisées dans de nombreuses études en médecine, en particulier, les arbres de décision et les réseaux

de neurones. D'autres approches d'intégration, pour améliorer le processus de recherche de cas similaires et solutionner un problème courant, ont été proposées, nous passerons en revus quelques-unes :

- Dans leur travail intitulé « Risk prediction in surgery using case-based reasoning and agent-based modelization » (Prédiction du risque en chirurgie par raisonnement à base de cas et modélisation par agents, en français), Bruno Pérez et ses compagnons ont combiné le RàPC avec le système multi-agent ou SMA pour faire émerger des situations à riques a priori imprévisibles. L'objectif de leur système était de définir des seuils d'alerte dans un contexte non déterministe où surviennent des évènements imprévisibles que nous n'avons pas souhaités. Le RàPC dans ce travail a permis une gestion analytique des données. [22]

- Comme nous le savons tous, l'année 2020 a été marquée par l'apparition d'une pandémie mondiale, qui n'est autre que le Covid-19. Oyelade, O. et Ezugwu, A. dans un cadre de RàPC pour la détection et le diagnostic précoces du Covid-19, ont mené une étude qui vise à tirer parti de la riche base de données de cas de Covid-19 afin de résoudre de nouveaux cas de Covid-19 et aider les médecins à facilement diagnostiquer des cas suspects sans effectuer des tests de laboratoire. Dans cette étude, ils ont utilisé un modèle de RàPC amélioré qui s'appuie sur une nouvelle sélection de caractéristiques et sur le modèle mathématique basé sur la sémantique pour le calcul de similarité des cas. Les résultats ont révélé que l'approche du RàPC amélioré a réussi à classer les cas suspects dans leur catégorie avec une précision de 94,54 %. [23]

- Le docteur Timothy M Rawson dans son travail a évalué un algorithme de raisonnement à partir des cas dans le monde réel par rapport aux décisions des prescripteurs. Les recommandations de prescription faites par le RàPC ont été alors comparées aux décisions prises par les médecins en pratique clinique. Comme résultat, sur 224 patients, les recommandations faites par le RàPC étaient appropriées

chez 202 sur les 224 contre 186 sur 224 dans la pratique. Ce qui nous montre que l'algorithme a fourni des recommandations d'antibiotiques appropriées par rapport aux choix effectués dans les pratiques des cliniques actuelles. [24]

- En 2020, Gu, D., Su, K., et Zhao, H. ont proposé un SAD auxiliaire qui combine le RàPC avec l'apprentissage d'ensemble pour améliorer la précision de la prédiction de la récidive du cancer du sein. Le système proposé a fourni une interprétation basée sur des cas de sa prédiction, ce qui aide les médecins à évaluer la fiabilité des prévisions du système et à prendre leurs décisions en fonction de ce qui est proposé par le système. Comme résultat, ils ont trouvé que le système fournit des prédictions raisonnablement précises et il a été bien accueilli par les oncologues. [25]

- En 2019, Abdelhak Mansoul, Bagdad Atmani, Mohamed Benamina et Sofia Benbelkacem ont proposé une approche qui combine le RàPC avec les règles d'association pour la classification dans la gynécologie et obstétrique. Les règles d'association étaient utilisées afin de réduire la solution de l'espace de recherche de cas similaires, cet espace est ensuite utilisé par le RàPC pour calculer une solution au nouveau problème auquel nous faisons face. [26]

- En 2022, Hatoon S AlSagri, Mourad Ykhlef, Mirvat Al-Qutt et al. ont proposé un système de recommandation de remèdes contre la dépression qui utilise le RàPC avec une forêt aléatoire. L'objectif du système était premièrement de créer un module d'adaptation automatisé, piloté par les données et qui peut évoluer sans intervention humaine ; et ensuite de fixer des poids dans la mesure de similarité des cas dans l'importance des caractéristiques, extraite du système d'identification de la dépression. La précision de récupération du RàPC a atteint 82% et la précision de l'adaptation automatique a atteint 88%. [27]

- En 2023, Henni et al. ont proposé un système d'aide à la décision pour la prédiction de la maladie coronarienne. Dans leur travail, ils ont combiné le RàPC avec un

modèle de forêt aléatoire (RFM). Leur approche consiste à optimiser le modèle de forêt aléatoire grâce à un réglage des hyper paramètres, pour réduire le nombre d'attributs. Ils ont alors intégré le résultat de cette approche dans le RàPC, plus précisément dans l'étape de remémoration du cycle de RàPC, pour améliorer son temps d'exécution. Comme résultat, ils ont constaté que le RFM surpasse les modèles les plus récents publiés pour le diagnostic du CAD. [28]

2.7 Conclusion

En guise de conclusion, le raisonnement à partir des cas ou RàPC est un processus cyclique qui aide à résoudre un nouveau problème en utilisant les anciens problèmes contenus dans la base de cas. Il permet de comparer les cas mémorisés avec le nouveau cas afin de trouver une nouvelle solution qui s'ajoutera ensuite à l'expérience. Dans le processus du RàPC, nous avons la phase de remémoration qui fait intervenir le calcul de similarité et la recherche de cas similaires. Ces deux étapes font appel à des index statiques et dynamiques. Nous avons également vu que le RàPC est utilisé dans plusieurs branches dans le domaine de la médecine et aide notamment les médecins et les cliniciens dans leur rôle qui est de soigner les personnes malades.

Chapitre 3
Approche proposée : Conception et implémentation du RàPC

3.1 Introduction

Précédemment, nous avons vu les différents éléments qui composent le RàPC, les différentes étapes de son cycle ainsi que quelques travaux dans le passé qui ont utilisés le RàPC avec d'autres techniques pour la mise en place d'un système d'aide à la décision médicale. Dans ce chapitre, le travail consiste à présenter l'approche que nous proposons pour notre système d'aide à la décision. En premier lieu, nous allons voir en détail l'architecture proposée pour bien mener notre travail. Ensuite, nous allons passer à l'implémentation de notre système basé sur le RàPC, pour la prédiction de la maladie cardiovasculaire, dans la plateforme jColibri.

3.2 Architecture de l'approche proposée

L'approche proposée a été inspiré de l'approche proposée par Henni et al. [28]. Elle se divise en deux parties : l'analyse discriminante et le RàPC. La figure suivante nous montre comment nous allons procéder pour la mise en place de notre système d'aide à la décision pour la prédiction des maladies coronariennes.

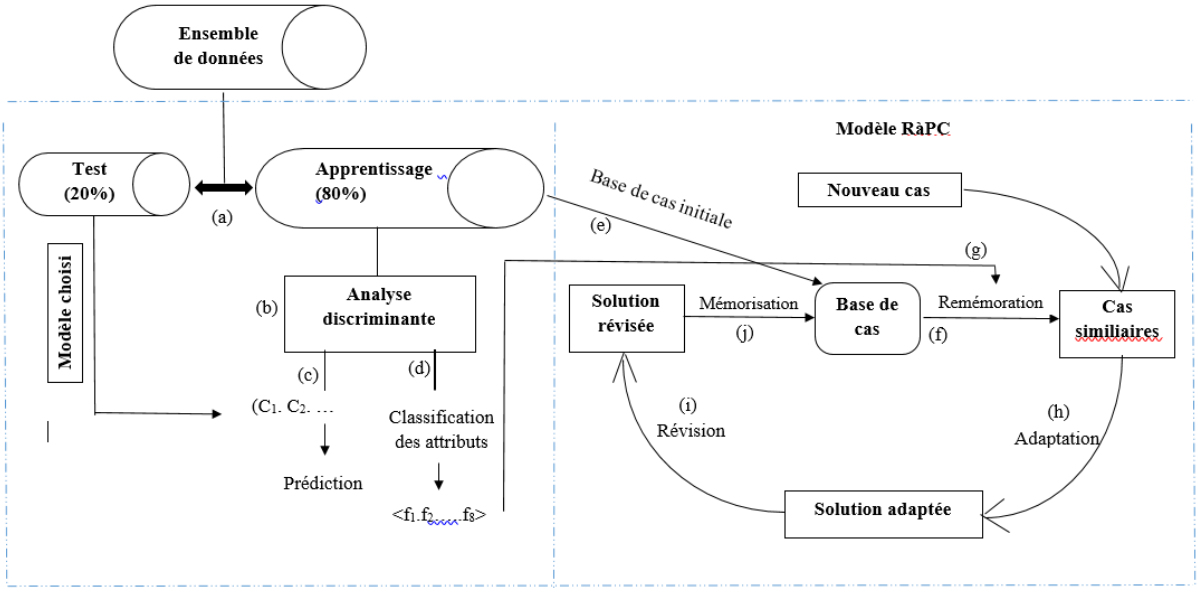

Figure 4 - Approche proposée

a) Cette étape consiste à diviser les données en deux parties : les données d'apprentissage et les données de test.

b) Cette étape consiste à effectuer une analyse discriminante, afin d'obtenir le meilleur modèle en utilisant la règles de Bayes.

c) Une fois que le meilleur modèle est obtenu, il servira à la classification des nouveaux cas.

d) En utilisant les probabilités conditionnelles une classification par ordre de pertinence des attributs est réalisée. Le résultat de cette étape sera alors utilisé dans le cycle du RàPC, plus précisément dans l'étape de remémoration.

Après ces différentes étapes de l'analyse discriminante, nous passons maintenant au système RàPC, partie qui nous concerne directement.

e) Dans cette étape, la base de cas de notre système de RàPC est constituée des données d'apprentissage que nous avons obtenues dans l'étape (a).

f) C'est la phase de remémoration où sont définies les différentes mesures de similarité, locale ainsi que global pour le calcul de similarité.

g) L'étape de remémoration du RàPC est appliquée sur les données de test de l'étape (a) pour évaluer les performances du modèle. La recherche des cas similaires au nouveau cas ne se fait plus en utilisant tous les attributs des données mais se fait seulement à travers les attributs les plus pertinents obtenus dans l'étape (d).

h) La phase d'adaptation du RàPC.

i) La phase de révision du RàPC.

j) La phase d'apprentissage du RàPC.

La suite de notre travail consistera uniquement à l'implémentation du système de RàPC.

3.3 Implémentation du RàPC pour la prédiction de la maladie coronarienne

L'objectif de cette partie est la mise en place d'un système d'aide à la décision médicale en utilisant le RàPC. Dans cette partie, nous allons voir en premier lieu une introduction à la plateforme jColibri et le développement d'une application RàPC sur jColibri. Ensuite, les étapes de la création de l'application RàPC pour la maladie coronarienne sous jColibri et enfin la présentation de l'application elle-même.

3.3.1 Plateforme jColibri et développement d'une application RàPC

Dans cette section, nous parlerons principalement de la plateforme jColibri et nous allons voir comment développer une application RàPC sur cette plateforme.

3.3.1.1 Présentation de la plateforme jColibri

jColibri est une plateforme orientée objet développée en Java pour l'implémentation de système basé sur le RàPC. Il comprend des mécanismes pour les différentes étapes du RàPC, c'est-à-dire pour la remémoration, l'adaptation, la révision et la mémorisation. Il a été mis en place par le groupe GAIA, qui signifie Group for Artificial Intelligence Applications, de l'université Compultence de Madrid.

La plateforme jColibri a deux versions majeures : [29]

- jColibri version 1 : qui a été la première version. Il inclut une interface graphique utilisateur complète qui guide les utilisateurs dans l'implémentation d'un système RàPC. Il est plus recommandé aux non-développeurs qui veulent créer leur système sans avoir à programmer.

- jColibri version 2 : jColibri2 est une nouvelle implémentation qui met en place une architecture nouvelle et claire qui est divisée en deux couches : une couche en boîte blanche déjà existante et une autre en version boîte noire, encore à développer, pour les designers. La version en boîte blanche est destinée aux développeurs Java qui veulent inclure dans leurs applications RàPC les fonctionnalités de jColibri.

3.3.1.2 Développement d'une application RàPC sous jColibri

Par le fait que jColibri2 contient les fichiers requis pour importer le projet framework dans Eclipse, c'est-à-dire .classpath et .project, l'utilisation de l'IDE Eclipse est recommandé pour développer une application RàPC.

Une application RàPC sous jColibri doit implémenter l'interface StandardCBRApplication contenu dans le package jcolibri.cbrapplications, dont le contenu est illustré dans la figure 5. Dans cette interface, nous trouvons trois étapes :

- Precycle : qui initialise l'application RàPC et qui n'est exécuté qu'une seule fois. C'est dans cette étape que l'application charge la base de cas et exécute des algorithmes coûteux.
- Cycle : qui exécute le cycle du RàPC. Il peut être exécuter plusieurs fois.
- PostCycle : qui est un code de post-exécution et de maintenance.

```java
public interface StandardCBRApplication
{
    /**
     * Configures the application: case base, connectors, etc.
     * @throws ExecutionException
     */
    public void configure() throws ExecutionException;

    /**
     * Runs the precyle where typically cases are read and organized into a case base.
     * @return The created case base with the cases in the storage.
     * @throws ExecutionException
     */
    public CBRCaseBase preCycle() throws ExecutionException;

    /**
     * Executes a CBR cycle with the given query.
     * @throws ExecutionException
     */
    public void cycle(CBRQuery query) throws ExecutionException;

    /**
     * Runs the code to shutdown the application. Typically it closes the connector.
     * @throws ExecutionException
     */
    public void postCycle() throws ExecutionException;
}
```

Figure 5 - Interface StandardCBRApplication

3.3.2 Implémentation de l'application RàPC pour la maladie coronarienne

Notre classe principale est nommée HeartDisease.java. Sa méthode main est illustrée par la figure 6 suivante :

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    HeartDisease heart = getInstance();
    heart.showMainFrame();
    try
    {
        heart.configure();
        heart.preCycle();

        QueryDialog qf = new QueryDialog(main);


        boolean cont = true;
        while(cont)
        {
            qf.setVisible(true);
            CBRQuery query = qf.getQuery();

            heart.cycle(query);
            int ans = javax.swing.JOptionPane.showConfirmDialog(null, "CBR cycle finished, query again?", "Cycle finished", javax.swing.JOptionPane.YES_NO_OPTION);
            cont = (ans == javax.swing.JOptionPane.YES_OPTION);
        }
        heart.postCycle();
    }catch(Exception e)
    {
        org.apache.commons.logging.LogFactory.getLog(HeartDisease.class).error(e);
        javax.swing.JOptionPane.showMessageDialog(null, e.getMessage());
    }
    System.exit(0);

}
```

Figure 6 - Méthode main de HeartDisease

3.3.2.1 Représentation de la base de cas de HeartDisease

Comme nous l'avons évoqué dans le chapitre 2, un système de RàPC a besoin d'une base de cas pour fonctionner. Dans cette section, nous allons détailler la base de cas que nous allons utiliser pour notre application et comment la charger sur jColibri.

3.3.2.1.1 Présentation de la base de données utilisée

La base de données que nous allons utiliser est la base de données Heart Disease de Cleveland dans l'UCI Machine Learning Repository [30], contenant 14 attributs dont l'un est l'attribut cible, appelé Maladie. Nous trouverons dans le tableau qui suit les informations sur chacun de ces attributs.

Attributs	**Descriptions**	**Types**	**Valeurs**
Age	Age en année	Entier	[29 - 77]
Sex	Sexe	Discret	2 = masculin 1 = féminin
ChestPain	Type de douleur thoracique	Discret	1 = angine de poitrine atypique 2 = angine atypique 3 = douleur non angorale 4 = asymptomatique
Bp	Pression artérielle au repos (en mm Hg à l'admission à l'hôpital)	Entier	[94 - 200]
Chol	Cholestérol sérique en mg/dl	Entier	[126 – 564]
Fbs	Glycémie à jeun (>120 mg/dl)	Discret	1 = faux 2 = vrai
Ekg	Résultats de l'électrocardiographie au repos	Discret	1 = normal 2 = présentant une anomalie de l'onde ST-T 3 = affiche une hypertrophie ventriculaire gauche probable ou définitive selon les critères d'Estes.
MaxHr	Fréquence cardiaque maximale atteinte	Entier	[71 – 202]

Angina	Angine induite par l'exercice	Discret	1 = Non 2 = Oui
Depression	Dépression ST induite par l'exercice par rapport au repos	Réel	[0.00 – 62.00]
Slope	Pente du segment ST d'exercice maximal	Discret	1 = montée en flèche 2 = plat 3 = descente
NbVessels	Nombre de grands vaisseaux (0,3) colorés par la fluoroscopie	Discret	{1, 2, 3, 4}
Thal	Type de défaut	Discret	1 = normal 2 = défaut fixe 3 = défaut réversible
Maladie	Présence ou absence de la maladie cardiovasculaire	Booléen	Presence Absence

Tableau 2 - Description des attributs

3.3.2.1.2 Base de cas sur jColibri

Pour stocker les données qui vont constituer notre base de cas, nous utilisons HSQLDB ou HyperSQL, un système de gestion de base de données (SGBD) écrit en Java. [31]

jColibri2, à travers Hibernate qui est un middleware utilisé pour le mappage objet-relationnel et pour effectuer une persistance efficace des objets dans les bases de données ou les fichiers XML [32], permet l'utilisation de n'importe quel SGBD. jColibri2 représente les cas en utilisant Java Beans. Java Bean est n'importe quelle classe qui a des méthodes getter et setter pour chaque attribut public [29], dont la modification et la manipulation sont effectuées

automatiquement à travers une technologie Java totalement transparente pour les développeurs, appelée « introspection ». Ainsi, en utilisant Java Beans dans jColibri2, les développeurs peuvent définir leurs cas comme une classe normale Java. En plus, Hibernate utilise Java Beans pour stocker les informations dans une base de données.

La seule restriction des Java Beans qui composent un cas c'est qu'ils doivent définir un identifiant Id qui va les identifier dans la table heartDisease du SGBD. Cette restriction est défini dans l'interface CaseComponent du package jcolibri.cbrcore.

Plusieurs classes et interfaces sont définies dans le package jcolibri.cbrcore pour développer une application RàPC :

- L'interface CaseComponent (vu en haut) : définit un composant d'un cas. Les cas sont composés d'instances de cette interface.
- La classe Attribut : identifie un attribut d'une CaseComponent.
- La classe CBRCase : représente toute structure de cas de jColibri. Elle est composée de plusieurs CaseComponents.
- L'interface CBRCaseBase : définit les méthodes que n'importe quelle base de cas doit mettre en œuvre pour être utilisée de manière transparente par jColibri2.
- La classe CBRQuery : représente une requête RàPC en la définissant comme une description du problème/cas, sans la solution.
- L'interface Connector : déclare les méthodes nécessaires pour accéder aux cas stockés dans une base de données.

3.3.2.2 Cas et requête de HeartDisease sous jColibri2

Dans jColibri2, les cas et les requêtes sont composés de CaseComponents. La classe CBRQuery contient la description du problème et nous savons qu'un cas, défini dans la classe CBRCase, est divisé en quatre composants : description du problème, solution du problème, résultat de la solution et justification de la solution. Ainsi, CBRCase va hériter CBRQuery.

La figure 7 ci-dessous nous montre comment sont définis les différents composants de la classe CBRCase.

```
public class CBRCase extends CBRQuery  {
    CaseComponent solution;
    CaseComponent justificationOfSolution;
    CaseComponent result;

     //Returns the justificationOfSolution.
    public CaseComponent getJustificationOfSolution() {
        return justificationOfSolution;
    }

     //Sets the Justification of Solution component.
     //@param justificationOfSolution The justificationOfSolution to set.
    public void setJustificationOfSolution(CaseComponent justificationOfSolution) {
        this.justificationOfSolution = justificationOfSolution;
    }

    //Returns the result.
    public CaseComponent getResult() {
        return result;
    }

    //Sets the Result component
    public void setResult(CaseComponent result) {
        this.result = result;
    }

     //Returns the solution.
    public CaseComponent getSolution() {
        return solution;
    }

     //Sets the solution component
    public void setSolution(CaseComponent solution) {
        this.solution = solution;
    }

    public String toString()
    {
        return super.toString()+"[Solution: "+solution+"][Sol.Just.: "+justificationOfSolution+"][Result: "+result+"]";
    }
}
```


Figure 7 - Classe CBRCase

Pour la description du cas, nous avons la classe HeartDiseaseDescription, contenu dans le package jcolibri.examples.heartDisease, dont le code est le suivant (sans les getters et les setters, qui doivent obligatoirement être présents ; ils le sont sous eclipse).

```java
public class HeartDiseaseDescription implements jcolibri.cbrcore.CaseComponent {

    String  CaseId;
    Double Age;
    Integer Sex;
    Integer ChestPain;
    Double Bp;
    Double Chol;
    Integer Fbs;
    Integer Ekg;
    Double MaxHr;
    Integer Angina;
    Double Depression;
    Integer Slope;
    Integer NbVessels;
    Integer Thal;

    public String toString()
    {
        return "("+CaseId+";"+Age+";"+Sex+";"+ChestPain+";"+Bp+";"+Chol+";"+Fbs+""
                + ";"+Ekg+";"+MaxHr+";"+Angina+";"+Depression+";"+Slope+";"+NbVessels+";"+Thal+")";
    }

    @Override
    public Attribute getIdAttribute() {
        // TODO Auto-generated method stub
        return new Attribute("CaseId", this.getClass());
    }

}
```

Figure 8 - Classe HeartDiseaseDescription

La partie solution est quant à elle implémentée sous la classe HeartDiseaseSolution ci-dessous.

```java
public class HeartDiseaseSolution implements jcolibri.cbrcore.CaseComponent {
    String id;
    String Maladie;

    public String toString()
    {
        return "("+id+";"+Maladie+")";
    }

    @Override
    public Attribute getIdAttribute() {
        // TODO Auto-generated method stub
        return new Attribute("id", this.getClass());
    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getMaladie() {
        return Maladie;
    }

    public void setMaladie(String maladie) {
        this.Maladie = maladie;
    }


}
```

Figure 9 - Classe HeartDiseaseSolution

3.3.2.3 Mécanisme de chargement des cas et de la base de cas sur jColibri2

Les systèmes RàPC doivent avoir accès aux cas d'une manière efficace, ce qui est un problème qui devient de plus en plus difficile au fur et à mesure où la taille de la base de cas augmente. jColibri2 divise alors le problème de gestion de la base de cas en deux points qui ont tout de même un lien : le mécanisme de persistance et l'organisation en mémoire.

3.3.2.3.1 Connecteurs de jColibir2

Le mécanisme de persistance que nous avons évoqué en haut est construit autour des connecteurs. Les connecteurs sont des objets qui savent comment accéder et récupérer des cas de la base de données et les retournent au système RàPC d'une manière uniforme. L'utilisation des connecteurs donne à jColibri la flexibilité envers le stockage physique, ce qui permet à ses utilisateurs de choisir celui qui leur convient le mieux.

jColibri2 inclut trois connecteurs dans son package connectors : DataBaseConnector, PlainTextConnector, OntologyConnector mais dans notre travail, nous allons seulement utiliser DataBaseConnector. Ce dernier gère la persistance des cas dans les bases de données et il utilise les librairies de Hibernate.

jColibri2 définit une interface Connector qui inclut des méthodes pour lire la base de cas dans la mémoire et la met à jour dans un média persistant. Tout connecteur doit implémenter les méthodes définies dans cette interface, telles que : initFromXMLfile, close qui , storeCases, deleteCases , retrieveAllcases, retrieveSomeCases.

3.3.2.3.2 Organisation des cas en mémoire

C'est la manière d'organiser les cas une fois qu'ils sont chargés dans la mémoire. L'organisation de la base de cas (linéaire, arbre, etc.) peut avoir une grande influence sur le processus du RàPC. La base de cas implémente une interface commune, appélée CBRCaseBase, qui permet aux méthodes de RàPC d'accéder aux cas. (Figure 10)

```java
public interface CBRCaseBase {

    public void init(Connector connector) throws jcolibri.exception.InitializingException;


    public void close();



    /**
     * Returns all the cases available on this case base
     *
     * @return all the cases available on this case base
     */
    public Collection<CBRCase> getCases();


    /**
     * Returns some cases depending on the filter
     * @param filter a case base filter
     * @return a collection of cases
     */
    public Collection<CBRCase> getCases(CaseBaseFilter filter);


    /**
     * Adds a collection of new CBRCase objects to the current case base
     *
     * @param cases
     *            to be added
     */
    public void learnCases(Collection<CBRCase> cases);

    /**
     * Removes a collection of new CBRCase objects to the current case base
     *
     * @param cases
     *            to be removed
     */
    public void forgetCases(Collection<CBRCase> cases);
}
```

Figure 10 - Interface CBRCaseBase

jColibri2 offre trois possibilités d'organiser la base de cas mais dans notre projet, nous allons utiliser jcolibri.casebase.LinealCaseBase, qui enregistre les cas sous forme de liste.

3.3.2.4. Chargement de la base de cas de la maladie coronarienne

Maintenant que nous avons vu les mécanismes de chargement de cas pour une application RàPC sur jColilbri2, nous allons présenter les méthodes configure (figure 11) et precycle (figure 12) de notre classe HeartDisease.

```
@Override
public void configure() throws ExecutionException {
    // TODO Auto-generated method stub
    try {
        // Create a data base connector
        _connector = new DataBaseConnector();
        // Init the ddbb connector with the config file
        _connector.initFromXMLfile(jcolibri.util.FileIO
                .findFile("jcolibri/examples/heartDisease/databaseconfig.xml"));
        // Create a Lineal case base for in-memory organization
        _caseBase = new LinealCaseBase();

        // Create the dialogs
        similarityDialog = new SimilarityDialog(main);
        resultDialog     = new ResultDialog(main);
        autoAdaptDialog  = new AutoAdaptationDialog(main);
        revisionDialog   = new RevisionDialog(main);
        retainDialog     = new RetainDialog(main);

    } catch (Exception e) {
        throw new ExecutionException(e);
    }
}
```

Figure 11 - Méthode configure de la classe HeartDisease

```
@Override
public CBRCaseBase preCycle() throws ExecutionException {
    // TODO Auto-generated method stub
    // Load cases from connector into the case base
            _caseBase.init(_connector);
            // Print the cases
            java.util.Collection<CBRCase> cases = _caseBase.getCases();
            for(CBRCase c: cases)
                System.out.println(c);
            return _caseBase;

}
```

Figure 12 - Méthode preCycle de la classe HeartDIsease

Pour la configuration du connecteur, nous avons le fichier « databaseconfig.xml » (Figure 13). Il définit :

- HibernateConfigFile : qui localise le fichier de configuration de Hibernate ;
- DescriptionMappingFile : qui localise le fichier de mappage de la description des cas ;
- DescriptionClassName : qui représente la classe de la description des cas;
- SolutionMappingFile : qui localise le fichier de mappage de la solution ;
- SolutionClassName : qui représente la classe de la solution des cas.

```
<DataBaseConfiguration>
    <HibernateConfigFile>jcolibri/examples/heartDisease/hibernate.cfg.xml</HibernateConfigFile>
    <DescriptionMappingFile>jcolibri/examples/heartDisease/HeartDescription.hbm.xml</DescriptionMappingFile>
    <DescriptionClassName>jcolibri.examples.heartDisease.HeartDiseaseDescription</DescriptionClassName>
    <SolutionMappingFile>jcolibri/examples/heartDisease/HeartSolution.hbm.xml</SolutionMappingFile>
    <SolutionClassName>jcolibri.examples.heartDisease.HeartDiseaselSolution</SolutionClassName>

</DataBaseConfiguration>
```

Figure 13 - Fichier databaseconfig.xml

Le fichier de configuration de Hibernate nous dit comment accéder à notre SGBD, qui dans notre cas est le HSQLDB. Ce fichier dans notre application est hibernate.cfg.xml. localisé dans le package jcolibri.examples.heartDisease.

Les fichiers de mappage définissent comment mapper un Java Bean avec une table d'une base de données. Dans notre cas, nous avons les classe HeartDiseaseDescription et HeartDiseaseSolution qui contiennent respectivement la description et la solution de notre cas. Nous allons mapper les deux classes dans la même table, appelée heartDisease sur HSQLDB.

3.3.2.5 Remémoration des cas

Maintenant que nous avons les méthodes configure() et precycle() de notre application HeartDisease, nous allons lancer le cycle de notre RàPC et cela commence par la remémoration. Comme nous avons évoqué dans le deuxième chapitre, l'étape de la remémoration consiste à trouver les cas similaires à notre nouveau cas.

jColibri2 nous propose dans la classe NNScoringMethod qui se trouve dans le package jcolibri.method.retrieve.NNretrieval pour la remémoration. Cette classe implémente une méthode qui utilise le principe des k-ppv en comparant les attributs deux à deux en utilisant des fonctions de similarité globale pour comparer les attributs composés et des fonctions de similarité locales pour les attributs simples.

Dans notre application, le composant de cas HeartDiseaseDescription de nos cas est un attribut composé de plusieurs attributs simples (Age, Sex, etc.). Nous allons donc appliquer la fonction de similarité globale pour la description de cas, la fonction moyenne (average) ; et des fonctions de similarité locale comme Equal, Interval, etc. pour chaque attribut simple. NNScoringMethod va ensuite calculer la similarité locale de chaque attribut et puis la similarité globale qui est la moyenne de chaque similarité locale.

La configuration de ces fonctions de similarité se trouve dans la classe NNConfig du package jcolibri.method.retrieve.NNretrieval. Nous y trouvons :

- La fonction de similarité globale pour la description ;
- Les fonctions de similarité locale pour chaque attribut composé, à l'exception de la description ;
- Les fonctions de similarité locale pour chaque attribut simple ;
- Et le poids de chaque attribut.

Nous allons maintenant voir en détail les fonctions de similarité locale et globale que nous avons utilisé pour notre application.

3.3.2.5.1 Calcul de similarité locale

Comme nous avons évoqué ci-haut, jColibri2 propose des fonctions de similarité locale mais dans notre cas, nous n'allons utiliser que quelques-unes selon les attributs que nous avons pour nos cas et elles sont présentées dans le tableau suivant :

Mesures de similarité	Attributs
Euclidienne	Age Bp Chol MaxHr Depression
Equal	Sex ChestPain Fbs Ekg Angina Slope Thal
Interval	NbVessels

Tableau 3 - Mesures de similarité

- **Euclidienne :** C'est la distance euclidienne classique entre deux valeurs x et y.

$$D(x,y) = \sqrt{(x-y)^2}$$

Mais pour générer un résultat entre 0 et 1, nous avons la formule de normalisation suivante : $D_{normalisée}(x,y) = \frac{\sqrt{(x-y)^2}}{x+y}$

- **Equal :** C'est une fonction simple pour les attributs qui ont des valeurs fixes. Elle retourne 1 si deux attributs sont égaux, sinon elle retourne 0.
- **Interval :** C'est une fonction qui renvoie la similarité entre deux nombres x et y à l'intérieur d'un intervalle.

$$\text{Sim}(x,y) = 1 - \frac{|x-y|}{\max - \min + 1}$$

3.3.2.5.2 Calcul de similarité globale

La fonction de similarité globale calcule la distance entre deux cas en utilisant les différentes mesures de similarité locale citées en haut, entre les 13 attributs. jColibri utilise le principe de la méthode de k-ppv. Dans l'algorithme k-ppv, la similarité entre le nouveau cas et les autres cas dans la base de cas est calculée à base de la sommation pondérée des mesures de similarité locale. Le résultat est ensuite normalisé pour obtenir une valeur entre 0 et 1.

La formule de similarité globale utilisée est le suivant :

$Simglobale(C,S) = \sum_{i=1}^{n} Simlocale(Ci,Si) \Big/ \sum_{i=1}^{n} Wi$, où C est le nouveau cas, S le cas mémorisé depuis la base de cas, W le poids de chaque attribut, n le nombre d'attributs, i l'index de l'attribut et $Simlocale(Ci,Si)$ les similarités locales pour l'attribut i.

3.3.2.6 Adaptation des cas

jColibri2 laisse cette étape ouverte aux développeurs. Ils peuvent ainsi créer leurs propres méthodes d'adaptation. Quoiqu'il en soit, jColibri2 propose deux méthodes d'adaptation basiques :

- jcolibri.method.reuse.DirectAttributeCopyMethod : qui copie la valeur d'un attribut de la requête dans un attribut d'un cas.

- jcolibri.method.reuse.NumericDirectProportionMethod : qui effectue une proportion numérique directe entre les attributs de la requête et du cas.

La méthode NumericDirectProportionMethod est illustré dans la figure suivante :

```
public class NumericDirectProportionMethod {
    public static void directProportion(Attribute source, Attribute destination, CBRQuery query, Collection<CBRCase> cases)
    {
        Object qs = AttributeUtils.findValue(source, query);
        if(qs == null)
            return;
        if(!(qs instanceof Number))
            return;
        Number qsn = (Number)qs;
        for(CBRCase c: cases)
        {
            try {
                Object cs = AttributeUtils.findValue(source, c);
                Object cdcomp = AttributeUtils.findBelongingComponent(destination, c);
                Object cd = destination.getValue(cdcomp);
                if((cs == null)||(cd == null))
                    return;
                if(!(cs instanceof Number) || !(cd instanceof Number))
                    return;
                Number csn = (Number)cs;
                Number cdn = (Number)cd;
                Double dres = (cdn.doubleValue() / csn.doubleValue()) * qsn.doubleValue();
                if(cd instanceof Double)
                    destination.setValue(cdcomp, dres);
                else if(cd instanceof Integer)
                    destination.setValue(cdcomp, new Integer(dres.intValue()));
                else if(cd instanceof Float)
                    destination.setValue(cdcomp, new Float(dres.floatValue()));
                else if(cd instanceof Byte)
                    destination.setValue(cdcomp, new Byte(dres.byteValue()));
                else if(cd instanceof Short)
                    destination.setValue(cdcomp, new Short(dres.shortValue()));
            } catch (AttributeAccessException e) {
                org.apache.commons.logging.LogFactory.getLog(NumericDirectProportionMethod.class).error(e);
            }
        }
    }
}
```

Figure 14 - Méthode d'adaptation sur jColibri2

Une fois les cas adaptés, le système RàPC les propose comme des solutions suggérées du problème. L'utilisateur du système pourra ensuite réviser ces solutions dans la phase de révision.

3.3.2.7 Révision des cas

Dans cette étape, la solution proposée est testée, c'est-à-dire qu'elle sera appliquée dans le monde réel ou alors évaluée par un expert du domaine et réajusté en cas d'échec. Cette étape est dépendante de chaque domaine et peut changer selon les applications. jColibri2 inclut seulement une méthode, jcolibri.method.revise.DefineNewIdsMethod (Figure 15), pour définir des nouveaux identifiants Id aux cas au moment de leur enregistrement dans la base de données dans l'étape suivante et ils ne peuvent pas ainsi réutiliser les identifiants originaux des cas remémorés.

```
public class DefineNewIdsMethod {
    public static void defineNewIdsMethod(CBRCase _case, HashMap<Attribute, Object> componentsKeys)
            throws jcolibri.exception.ExecutionException
    {
            defineNewIds(_case.getDescription(), componentsKeys);
            defineNewIds(_case.getSolution(), componentsKeys);
            defineNewIds(_case.getJustificationOfSolution(), componentsKeys);
            defineNewIds(_case.getResult(), componentsKeys);
    }

    private static void defineNewIds(CaseComponent cc, HashMap<Attribute, Object> componentsKeys) throws
    jcolibri.exception.ExecutionException
    {
        if(cc == null)
            return;
        Attribute keyAtt = cc.getIdAttribute();
        Object newkeyvalue = componentsKeys.get(keyAtt);

        try {
            if(newkeyvalue != null)
                keyAtt.setValue(cc, newkeyvalue);

            for(java.lang.reflect.Field f: cc.getClass().getDeclaredFields())
            {
                Attribute at = new Attribute(f);
                Object o = at.getValue(cc);
                if(o instanceof CaseComponent)
                    defineNewIds((CaseComponent)o, componentsKeys);
            }
        } catch (Exception e) {
            org.apache.commons.logging.LogFactory.getLog(DefineNewIdsMethod.class).error(e);
        }

    }
}
```

Figure 15 - Méthode de révision sur jColibri2

3.3.2.8 Mémorisation des cas

Cette étape consiste à retenir les cas utiles dans la base de cas. Ils pourront être utilisés pour des futures utilisations, ce qui permet au système RàPC d'appendre une nouvelle expérience.

jColibri2 inclut la méthode jcolibri.method.retain.StoreCasesMethod (Figure 16) pour enregistrer des nouveaux cas dans la base de cas. Ces nouveaux cas seront stockés dans une persistante media selon l'implémentation choisie du CBRCaseBase. Dans notre cas, vu que nous avons une LinealCaseBase, les cas seront directement enregistrés sur le disque.

```
public class StoreCasesMethod {
    public static void storeCases(CBRCaseBase caseBase, Collection<CBRCase> cases)
    {
        caseBase.learnCases(cases);
    }

    /**
     * Simple method that add a case to the case base invoking caseBase->learnCases().
     */
    public static void storeCase(CBRCaseBase caseBase, CBRCase _case)
    {
        Collection<CBRCase> cases = new ArrayList<CBRCase>();
        cases.add(_case);
        caseBase.learnCases(cases);
    }
```

Figure 16 - Méthode de mémorisation sur jColibri2

3.3.3 Présentation de l'application

L'application que nous allons présenter dans cette partie a comme objectif d'aider les experts du domaine à prédire la présence ou l'absence de la maladie coronarienne chez les patients. Nous allons maintenant voir quelques interfaces qui permettent la communication entre l'application et ses différents utilisateurs et à travers ces interfaces, le cycle du RàPC sera exécuté en arrière-plan. Ensuite nous allons effectuer les différents tests de validation.

3.3.3.1 Interface élaboration du problème

Cette interface permet à l'utilisateur de l'application d'introduire les informations concernant le nouveau cas d'un patient dont il souhaite savoir s'il peut potentiellement avoir ou non la maladie coronarienne.

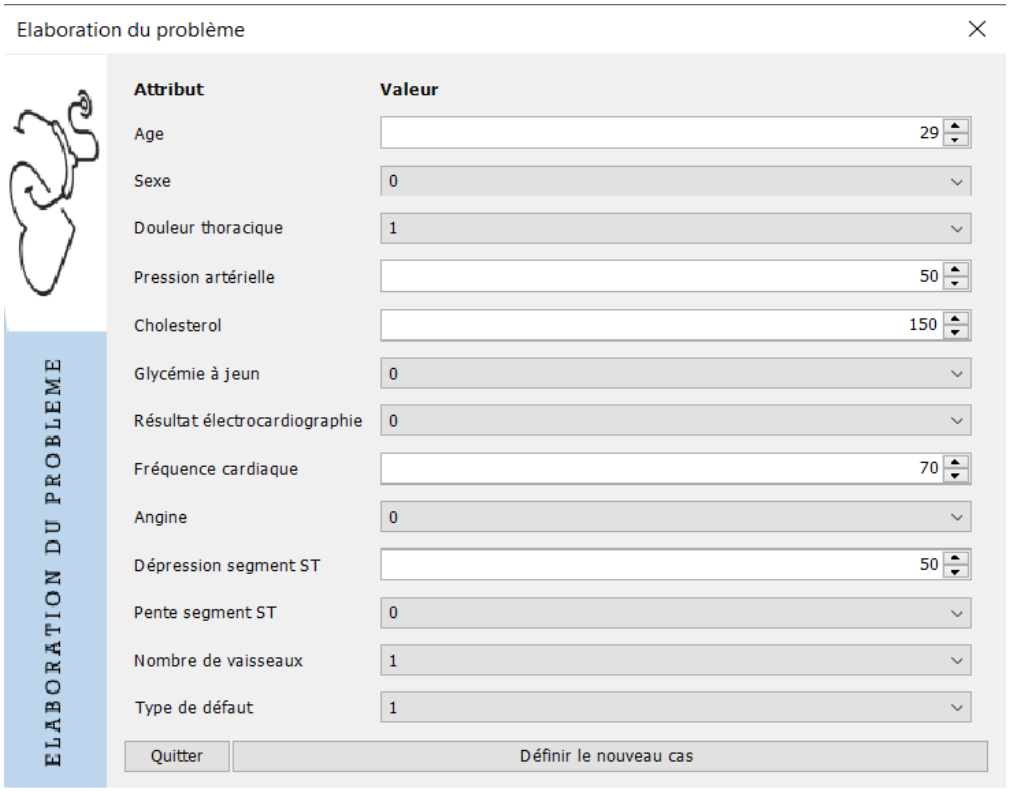

Figure 17 - Interface élaboration du problème

3.3.3.2 Interface de calcul de similarité

Cette interface permet aux utilisateurs de choisir les différentes fonctions utilisées par chaque attribut pour la recherche des cas similaires dans la base de cas.

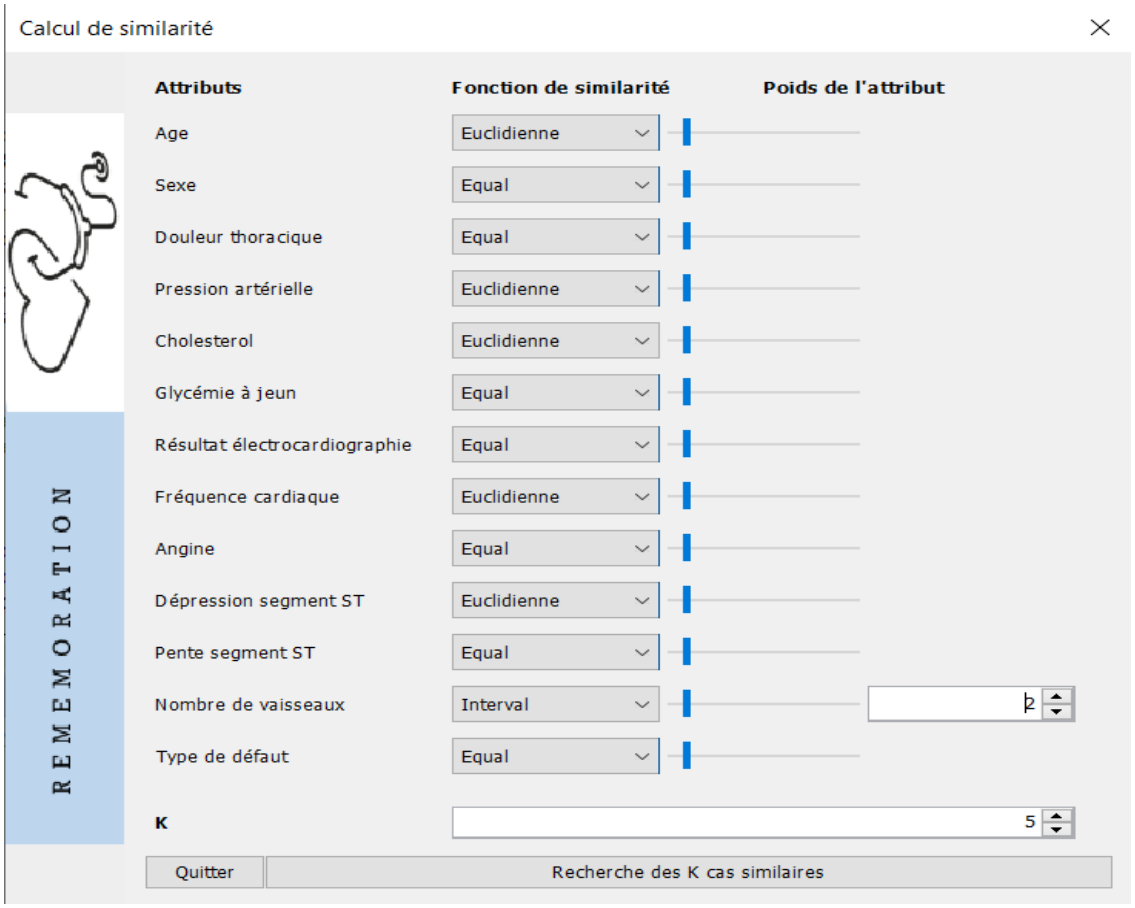

Figure 18 - Interface calcul de similarité

3.3.3.3 Interface cas similaires

Cette interface affiche les K cas similaires à notre nouveau cas.

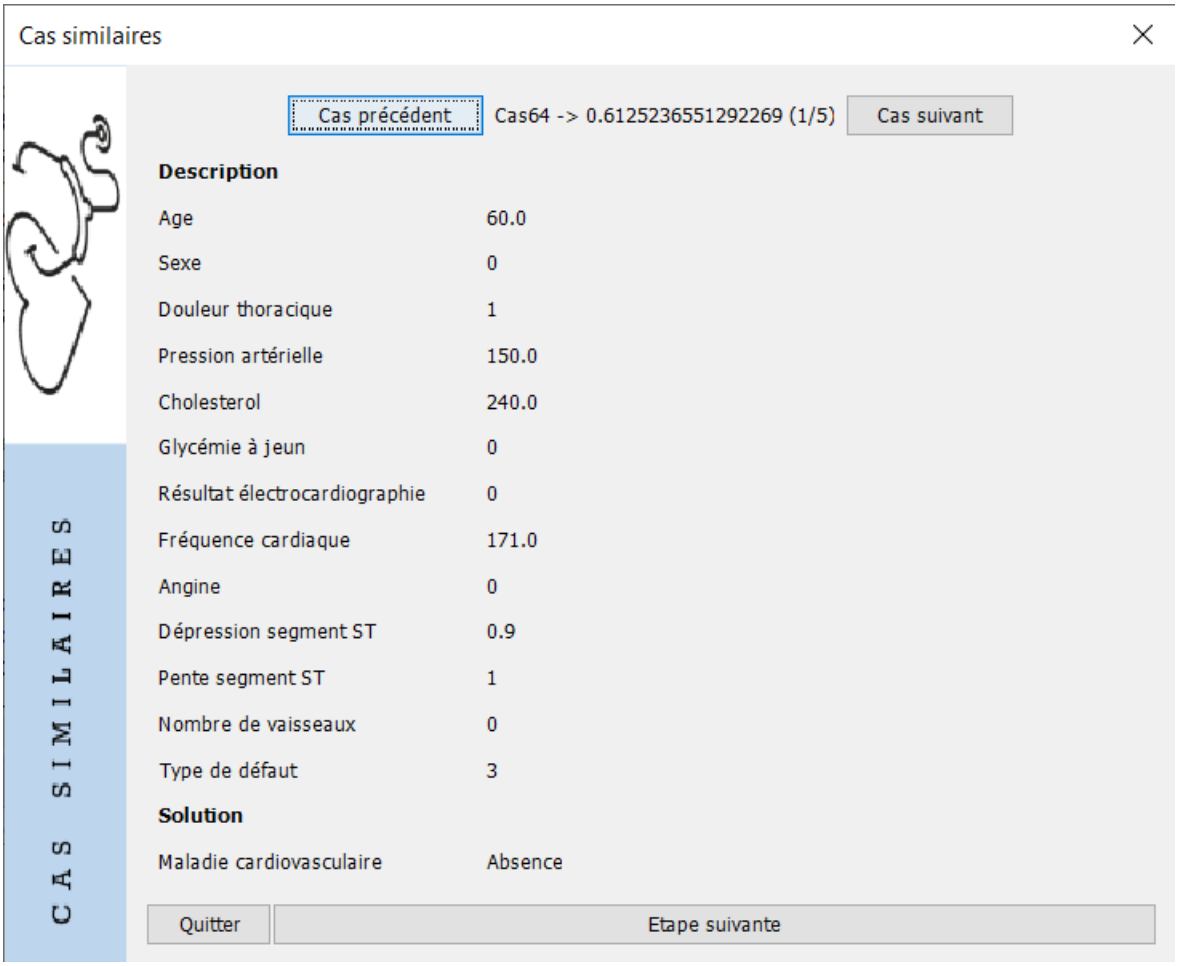

Figure 19 - Interface cas similaires

3.3.3.4 Tests et validation

Pour l'évaluation de la performance du système d'aide à la décision pour la prédiction de la maladie cardiovasculaire, nous avons repris les mêmes données d'apprentissage et de test utilisées par Henni et al. (2023) [28] dans leur travail. C'est-à-dire que la base de données initiale que nous avons utilisée contient 296 cas de patients. Notre base de test est composée de 30 patients dont les identifiants sont les suivants : 225, 111, 74, 50, 192, 214, 19, 275, 55, 188, 221, 274, 61, 113, 289, 98, 154, 41, 236, 121, 129, 6, 110, 66, 142, 296, 155, 36, 104, 102. Et les 266 patients restant constitue la base de cas.

Nous avons effectué deux différentes expérimentations pour chacun des cas en changeant la valeur de K. Nous avons testé avec K=3 et K=5. Seules les instances 74, 98 et 236 ont été mal classés.

Le résultat est donné dans le tableau suivant :

Identifiant	K=3	K=5
225	Bien prédit	Bien prédit
111	Bien prédit	Bien prédit
74	Mal prédit	Bien prédit
50	Bien prédit	Bien prédit
192	Bien prédit	Bien prédit
214	Bien prédit	Bien prédit
19	Bien prédit	Bien prédit
275	Bien prédit	Bien prédit
55	Bien prédit	Bien prédit
188	Bien prédit	Bien prédit
221	Bien prédit	Bien prédit
274	Bien prédit	Bien prédit
61	Bien prédit	Bien prédit
113	Bien prédit	Bien prédit
289	Bien prédit	Bien prédit
98	Mal prédit	Mal prédit
154	Bien prédit	Bien prédit

41	Bien prédit	Bien prédit
236	Mal prédit	Mal prédit
121	Bien prédit	Bien prédit
129	Bien prédit	Bien prédit
6	Bien prédit	Mal prédit
110	Bien prédit	Bien prédit
66	Bien prédit	Bien prédit
142	Bien prédit	Bien prédit
296	Bien prédit	Bien prédit
155	Bien prédit	Bien prédit
36	Bien prédit	Bien prédit
104	Bien prédit	Bien prédit
102	Bien prédit	Bien prédit

Tableau 4 - Résultat de la remémoration

Comme nous pouvons le voir, il y juste 3 cas parmi les 30 qui ont été mal classée par le système quand K=3 et même quand K=5.

Pour les cas mal prédits, nous les avons testés avec K=7 pour voir s'ils peuvent être bien classés en augmentant le nombre de voisins et nous avons observé les faits suivants:

- Pour le cas 74 : déjà qu'il a été bien prédit pour K=5, il en est de même pour K=7 ;
- Pour le cas 98 : il est toujours mal prédit.
- Pour le cas 236 : il est bien prédit.

3.4 Conclusion

Nous avons exposé dans ce chapitre l'approche que nous avons utilisé afin de mettre en place notre système d'aide à la décision. Nous avons implémenté une application RàPC pour la prédiction de la maladie cardiovasculaire sur jColibri, une plateforme qui a été développée spécialement pour l'implémentation des applications RàPC. Notre système a montré son efficacité et sa fiabilité à travers les différentes expérimentations que nous avons effectué.

Conclusion générale

En conclusion, ce manuscrit a exploré l'application de l'aide à la décision médicale dans le contexte spécifique des maladies coronariennes, en mettant l'accent sur l'implémentation du RàPC à l'aide de l'outil jColibri. L'objectif principal était d'enrichir le processus de remémoration des maladies coronariennes en utilisant la règle de Bayes pour le calcul de similarité. Il est important de souligner que seule la partie du manuscrit concernant l'implémentation du RàPC par jColibri a été présentée.

Le chapitre sur le RàPC par jColibri a démontré l'efficacité de cette approche dans le domaine médical, en particulier pour la prédiction des maladies coronariennes. L'intégration de l'outil jColibri a permis de mettre en place un processus de remémoration basé sur les k plus proches cas similaires, facilitant ainsi la prise de décision médicale.

Cependant, il est important de noter que l'intégration de la règle de Bayes dans le calcul de similarité constitue une perspective prometteuse pour améliorer encore davantage la précision du processus de remémoration. La partie restante, qui concerne la règle de Bayes, sera abordée et développée dans le cadre de la session 2 par l'étudiante Mecemma.

En perspective, ce projet ouvre la voie à des recherches futures visant à exploiter pleinement le potentiel de l'aide à la décision médicale en intégrant la règle de Bayes dans le processus de remémoration des maladies coronariennes. L'utilisation de l'outil jColibri pour implémenter le RàPC offre des perspectives intéressantes pour améliorer les résultats de prédiction et faciliter la prise de décision médicale dans le contexte des maladies coronariennes.

Bibliographie

[1] Hichem, B. (2022). Contribution à la recherche d'information guidée par apprentissage automatique : Prévention des accidents de la circulation chez l'enfant. Thèse de doctorat. Université d'Oran 1 Ahmed Benbella.

[2] Simon, H. A. : Administrative behavior: a study of decision-making processes in administrative organization. The MacMillan Company New York, 1957.

[3] Bouyssou, D., Dubois, D., Prade, H., and Pirlot, M., editors (2013). Decision making process : Concepts and methods. John Wiley & Sons.

[4] Moret-Bonillo, V., Fernández-Varela, I., Hernández-Pereira, E., Alvarez-Estévez, D., and Perlitz, V. (2018). On the automation of medical knowledge and medical decision support systems. In Advances in Biomedical Informatics, pages 187_217.

[5] Musen, M., Middleton, B., and Greenes, R. (2014). Clinical decision-support systems. In Biomedical informatics, page 643_674. Springer, London.

[6] Moreno, M. (2015). Développement des systèmes d'aide à la décision dans les cabinets de médecine générale en France Application avec l'AntibioVille 2015 de la formalisation d'une recommandation médicale et son intégration dans un logiciel métier. PhD thesis, Université de Lorraine.

[7] Fernandes, M., Vieira, S., Leite, F., Palos, C., Finkelstein, S., and Sousa, J. (2020). Clinical decision support systems for triage in the emergency department using intelligent systems : a review. Artifficial Intelligence in Medicine, 102:101762.

[8] Arora, T. and Soni, R. (2021). A pre-screening approach for covid-19 testing based on belief rule-based expert system. In K., S. and A, J., editors, COVID-19 : Prediction, Decision-Making, and its Impacts. Lecture Notes on Data Engineering and Communications Technologies, volume 60. Springer, Singapore.

[9] Sherimon, P. and Krishnan, R. (2016). Ontodiabetic : An ontology-based clinical decision support system for diabetic patients. Arabian Journal for Science and Engineering, 41(3) :1145.

[10] Davis, R. and Hamscher, W. (1988). Model-based reasoning : Troubleshooting. In Exploring articial intelligence, page 297346. Morgan Kaufmann.

[11] Parent, E. and Bernier, J. (2007). Le raisonnement bayésien : modélisation et inférence. Springer Science & Business Media.

[12] Sowa, J. F. (2014). Principles of semantic networks : Explorations in the representation of knowledge. Morgan Kaufmann.

[13] Maladie coronarienne https://www.ottawaheart.ca/fr/maladie-du-c%C5%93ur/maladie-coronarienne-ath%C3%A9roscl%C3%A9rose

[14] Kane-Gill, S., Archita, A., John, A. K., & Steven, M. H. (2016). *Clinical decision support for drug related events : Moving towards better prevention.* World journal of critical care medecine. doi:10.5492/wjccm.v5.i4.204

[15] Sward, K., & Newth, C. (2016). *Computerized decision support systems for mechanical ventilation in children.* Journal of pediatric intesive care, 5(3) :95.

[16] Adams, E., & Longhurst, C. (2016). *Clinical Decision Support for Pediatric Blood product Prescriptions.* Journal of Pedriatric Intensive Care. doi:10.1055/s-0035-1569996

[17] Leake, D. (2015). Problem Solving and Reasoning: Case-Based. *International Encyclopedia of the Social & Behavioral Sciences (Second Edition)*, 56-60.

[18] Benamar, F. & Bouguelmouna, F. (2013). *Raisonnement à partir de cas en utilisant le système jColibri.* Mémoire de Master, Département de mathématiques et d'Informatique. Université Abdelhamid Ibn Badis Mostaganem.

[19] Shang, Y. : *The Electrical Engineering Handbook*, Academic Press, 2005.

[20] Aamodt, A. and Plaza, E. (1994). Case-based reasoning : Foundational issues, methodological variations, and system approaches. AI communications, 7(1) :39-59.

[21] Algorithme des k plus proches voisins.
https://www.monlyceenumerique.fr/nsi_premiere/algo_a/a4_algo_knn.php#1

[22] Perez, B., Lang, C., Henriet, J., Philippe, L., & Auber, F. (2020). *Risk prediction in surgery using case-based reasoning and agent-based modelization.* Computers in biology and medicine, 104040

[23] Oyelade, O. and Ezugwu, A. (2020). Covid19 : A natural language processing and ontology oriented temporal case-based framework for early detection and diagnosis of novel coronavirus. preprints 2020, 2020050171.

[24] Rawson, T., Henrandez, B., Moore, L., Herrero, P., Charani, E., Ming, D., & Homes, A. : A real-world evaluation of a case-based reasoning algorithm to support antimicrobial prescribing decisions in acute care, *Clinical infectious diseases,* April 2020.

[25] Gu, D., Su, K., and Zhao, H. (2020). A case-based ensemble learning system for explainable
breast cancer recurrence prediction. Artifficial Intelligence in Medicine, 107 :101858.

[26] Mansoul, A., Atmani, B., Benamina, M., et Benbelkacem, S. (2019). Learning casebased reasoning solutions by association rules approach. In International Conference on Computing, page 111118, Cham. Springer.

[27] AlSagri, Hatoon S, Ykhlef, M., Al-Qutt, M., et al. An automatic Adaptive Case-Based Reasong System for Depressin Remedy Recommendation. International Journal of Advanced Computer Science and Applications, 2022, vol. 13, no 11.

[28] Henni, F., Atmani, B., Atmani, F., & Saadi, F. (2023). Improving Coronary Artery Disease Prediction: Use of Random Forest, Feature Importance and Case-Based Reasoning. *International Journal of Decision Support System Technology (IJDSST), 15*(1), 1-17.

[29] J. A.Recio-Garcìa, B. Dìaz-Agudo, P. Gonzalez-Calero, (2008). JCOLIBRI2 Tutorial. Group for artificial intelligence applications. Universidad Compultense de Madrid

[30] Heart Disease Dataset

https://archive.ics.uci.edu/dataset/45/heart+disease

[31] HSQLDB

https://fr.wikipedia.org/wiki/HSQLDB

[32] Hibernate

https://www.decodejava.com/hibernate-architecture.htm#:~:text=Hibernate%20is%20a%20middleware%20used,application%20and%20the%20database%20server