Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche
researchportal.unamur.be
University of Namur

THESIS / THÈSE

MASTER EN SCIENCES INFORMATIQUES

Système expert fiscal

Deketelaere, Cathy

Award date:
2009

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 04. May. 2026

Facultés Universitaires Notre-Dame de la Paix, Namur
Institut d'informatique
Année académique 2008-2009

SYSTEME EXPERT FISCAL

Cathy Deketelaere

Mémoire présenté en vue de l'obtention du grade de licencié en informatique

RESUME

Chaque année tout contribuable se doit de remplir sa déclaration fiscale. Le montant à rembourser par le fisc ou à verser au fisc est déterminé par une ensemble de règles précises publiées dans diverses revues (telles que l'Almanach du contribuable). Toutefois, l'application manuelle de ces règles est, en général, hors de portée du commun des mortels. En outre, dans certains cas de figure, il est possible de réduire l'impôt en déclarant d'une manière adéquate certains montants.

L'objectif du mémoire est de construire un système expert basé sur un codage en programmation logique par contraintes de règles fiscales. Un tel système contrastera avec les logiciels de type "feuille de calcul Excel", répandus par les organismes bancaires, en permettant des calculs multi-directionnels et en expliquant son raisonnement.

ABSTRACT

Every year, each citizen must fill his Tax declaration.

The final amount to pay to Tax authorities or to be reimbursed is derived from strict rules that are published in dedicated literature. By the way, the adequate use of these rules is not clear to common citizens. In some case, it's even possible to reduce the total tax pressure by declaring some amounts in one specific way and not in another way, even if both are legally admitted.

The goal of this thesis is to build an expert system based on constraints programming of these fiscal rules. Such system should make the difference regarding usual 'Excel spreadsheets', so in use in all bank companies. Such system should allow multi-directional calculation and explain its reasoning.

AVANT-PROPOS (ET REMERCIEMENTS)

Et nous y sommes : le mémoire de fin d'année...

J'ai choisi un mémoire applicatif sur un sujet qui m'attire beaucoup : un système expert fiscal.

Le domaine d'application, la fiscalité, est en effet un domaine familier. Ingénieur commercial de formation, j'ai une expérience professionnelle de plus de dix ans dans la comptabilité, dont près de cinq en tant que membre de l'Institut Professionnel des Comptables (IPC) avant la création d'un Institut de Fiscalistes Agréés.

Si je pensais être à même de traiter en tant qu'expert du domaine d'application, c'est sans compter sur la créativité de nos législateurs qui innovent annuellement en la matière. C'est dire, si plus de cinq ans après avoir abandonné la comptabilité au profit de l'informatique, la remise à niveau était importante.

L'objectif était de montrer que la programmation logique par contraintes pouvait s'adapter à ce domaine fiscal et permettre de raisonner comme un expert.

J'espère vous en convaincre, même si après tout, de nombreuses questions et pistes resteront encore à découvrir après la lecture...

Je tenais à profiter de cet avant-propos pour remercier mes supporters dans cette aventure :

Le noyau dur composé de Anne-France Brognaux, Olivier Locatelli et Pierre Bacquet.

Les collègues, anciens et nouveaux, pour leur compréhension concernant mes demandes de congés qui tombaient mal professionnellement mais qui étaient incontournables en fonction de l'année académique et particulièrement Kim Colard qui s'est proposé spontanément comme lecteur.

Je tiens aussi à saluer la patience, la gentillesse, la compréhension et la disponibilité de mon promoteur : Jean-Marie Jacquet. Il est certainement une des raisons majeures de son accomplissement.

Et je n'oublie pas ma famille, qui commence à s'impatienter de mes absences aux réunions familiales et qui a enduré mes différentes phases d'enthousiasme et de découragement pendant quatre longues années.

Finalement, en totale contradiction avec ma personnalité, j'ai décidé de suivre les recommandations énoncées dans la "somme des mémoires" et d'adopter le "nous" majestueux dans la suite de la rédaction. Loin de me prendre pour Napoléon, j'imagine que vous prends par la main et je "nous" emmène au travers de l'exposé de "mon" travail.

Ceci étant fait, le contexte étant posé, je vous souhaite une agréable lecture du fruit de ce labeur.

TABLE DES MATIERES (ET AUTRES TABLES)

Résumé 2

Abstract 2

Avant-propos (et remerciements) 3

Table des matières (et autres tables) 4

Glossaire 7

Introduction 9

Chapitre 1 : Le Système Expert 10

1.1 Définition 10

1.2 Architecture 11

1.3 La Représentation des Connaissances 12

1.3.1 Règles de production 13

1.3.2 Logique des prédicats 14

1.3.3 Réseaux sémantiques 15

1.3.4 Les frames 16

1.3.5 Stratégies d'évolution 17

1.4 Méthodologie 17

1.4.1 Choix du Domaine 17

1.4.2 Choix de l'outil 18

Chapitre 2 : Le Domaine d'Application 19

2.1 Place de l'IPP dans le Paysage Fiscal 19

2.2 Caractéristiques de l'IPP 21

2.3 Modélisation 24

2.3.1 Calcul de l'impôt 24

2.3.2 Autres règles 28

2.4 Tax on Web 31

Chapitre 3 : Application 33

3.1 Moteur d'Inférence 33

3.2 Base de Connaissance 35

3.3 Base de faits 35

3.3.1 Données utilisateurs 35

3.3.2 Paramètres annuels 37

3.4 Interface Utilisateur 38

3.5 Module d'Explication 38

3.6 Module d'Acquisition des Connaissances 39

3.7 Fonctionnement de l'application 40

Chapitre 4 : Modélisation des Contraintes 41

4.1 Problème d'optimisation 41

4.2 Détermination de l'impôt (calcul) 42
4.2.1 Procédure d'entrée : opti_IPP 42
4.2.2 Procédure calcule_SoldeImpot 43
4.2.3 Procédure calcule_RIG 44
4.2.4 Procédure calcule_RevenusImmobiliers 45
4.2.5 Procédure calcule_RCHabitationPropre 46
4.2.6 Procédure calcule_AbattementForfaitaireHabitation 46
4.2.7 Autres procédures du calcul d'impôt 47
4.3 Eléments d'optimisation (simulation) 48
4.3.1 Procédure calcule_SimulReductionImpot 48
4.3.2 Procédure calcule_SimulEconomieEnergie 49
4.4 Exemple d'utilisation 49
Chapitre 5 : Une autre approche 51
5.1 Critique de l'application développée 51
5.2 Les pistes d'évolution 53
5.3 Une autre approche 53
Chapitre 6 : Conclusions 55
Chapitre 7 : Bibliographie 57
Chapitre 8 : Annexes 59
8.1 Interface php 59
8.1.1 Index général 59
8.1.2 Module administration 60
8.1.3 Module déclaration 61
8.2 Fichiers Prolog créés par l'interface php 62
8.2.1 Index2008 62
8.2.2 Données de la déclaration 63
8.3 La base de connaissances : Prologique.pl 64
8.4 Simulation d'impôt du FiscoWeb via Dexia 75
8.5 Exemple de documents utilisateurs 79
8.5.1 Fiche 281.10 79

Liste des tables et graphiques

Figure 1 : Architecture traditionnelle d'un Système Expert 12
Figure 2 : Illustration d'un arbre sémantique par Quillian 15
Figure 3 : Paysage fiscal belge 19
Figure 4 : Les impôts sur les revenus en Belgique 20
Figure 5 : l'IPP est un impôt progressif par paliers 21
Figure 6 : Schéma du calcul d'impôt – partie 1/2 24
Figure 7 : Schéma du calcul d'impôt – partie 2/2 25
Figure 8 : l'IPP et les minima exonérés d'impôts 27
Figure 9 : Exemple de gestion de contrainte simple[27] 29
Figure 10 : Tax on Web – Exemple de wizard[28] 31
Figure 11 : Tax on Web – Exemple d'aide contextuelle[28] 32
Figure 12 : Schéma conceptuel de la base de données mySQL 36
Figure 13 : Exemple d'utilisation du système expert 50

GLOSSAIRE

[G-1] ***Décumul*** : Depuis la période imposable [G-8] 2004, chaque époux/cohabitant légal est imposé distinctement sur ses propres revenus professionnels, mobiliers, immobiliers (éventuellement pour la partie lui revenant) et divers [B-12].

[G-2] ***Ensemble des revenus nets*** : Ensemble des revenus de biens immobiliers, revenus des capitaux et biens mobiliers, revenus professionnels et revenus divers tels qu'ils sont déterminés avant déduction des dépenses communes et personnalisables [B-12, n°1107].

[G-3] ***Exercice d'imposition*** : Année à laquelle sont rattachées les règles relatives à l'établissement de l'imposition sur les revenus de la période imposable [G-8]. Il est désigné ordinairement par le millésime de l'année qui suit celle de la période imposable. Ex. : les revenus de l'année 2007 (période imposable [G-8]) sont imposés à l'IPP [G-6] pour l'exercice d'imposition 2008 [B-12, n°5].

[G-4] ***Impôt de base*** : Impôt théorique tel qu'il résulte de l'application du barème de l'impôt, avant déduction de l'impôt sur la quotité du revenu exemptée d'impôt et avant application de toute réduction, augmentation ou majoration d'impôt [B-12, n°1107].

[G-5] ***Impôt de base réduit*** : Impôt à répartir, diminué des réductions pour pensions et revenus de remplacement [B-12, n°1107].

[G-6] ***IPP*** : Impôt des personnes physiques (par opposition à : INR = impôt des non-résidents, IPM = impôt des personnes morales, ISOC = Impôt des sociétés) [B-12].

[G-7] ***Minimum exonéré d'impôt*** : Partie du revenu exonérée d'impôt. Le minimum exonéré d'impôt est appliqué dans les tranches inférieures de revenus et peut être majoré pour diverses raisons (enfants à charges, handicapés, ...) [B-12].

[G-8] ***Période imposable*** : Période dont les revenus doivent servir de base à l'imposition [B-12, n°5].

[G-9] ***Principal*** : Impôt de base [G-5] réduit après application des réductions pour revenus d'origine étrangère. La notion de "principal" et d'"impôt de base réduit" sont équivalentes lorsqu'il n'existe que des revenus taxables au taux plein [B-12, n°1107].

[G-10] ***Quotient conjugal*** : Partie des revenus professionnels du conjoint/cohabitant légal dont les revenus sont les plus élevés, est transféré à l'autre conjoint/cohabitant légal pour le calcul de l'impôt [B-12].

[G-11] ***Revenus imposables globalement (RIG)*** : Ensemble des revenus nets [G-1] diminués des dépenses déductibles [B-12, n°1107].

[G-12] ***ROI*** : Return On Investment : Méthode d'évaluation comptable qui permet de déterminer le bien fondé d'un investissement. La formule est la suivante : total des revenus nets / total des investissements. Une version améliorée de la formule considère la total actualisé des revenus nets / total des investissements [B-14].

[G-13] ***Actualisation*** : L'actualisation consiste à déterminer la valeur d'aujourd'hui de flux qui se produiront dans le futur : elle est donc l'inverse de la capitalisation. Elle permet de comparer des sommes reçues ou versées à des dates différentes. La formule est la suivante : $V_0 = V_n / (1+t)^n$, avec V_0 = valeur actuelle, V_n = flux futur (revenu par exemple), t = taux d'intérêt (que l'on obtiendrait si on plaçait l'argent au lieu de l'investir, par exemple), n = durée de l'investissement [B-15].

[G-14] ***Contribuable*** : Tout personne physique qui habite en Belgique et qui recueille des revenus [B-12].

[G-15] ***Cohabitant légal*** : Les personnes qui ont fait une déclaration de cohabitation légale devant l'officier de l'état civil du domicile commun, conformément à l'article 1476 du Code Civil, sont assimilées à des personnes mariées et un cohabitant légal est assimilé à un conjoint pour ce qui concerne l'impôt des personnes physiques [B-18][1].

[G-16] ***Taux moyen d'imposition*** : Ratio obtenu selon la formule suivant : Total Impôts dus / Revenus Imposables Globalement.

[G-17] ***Taux marginal d'imposition*** : Taux correspondant à la tranche de revenus imposables la plus élevée.

[1] Page 2.

INTRODUCTION

Le sujet de ce mémoire est l'élaboration d'un 'Système expert fiscal basé sur la programmation par contraintes'.

Ce sujet peut être vaste et le premier travail a été de déterminer les objectifs poursuivis par ce mémoire.

Pour commencer, dans le chapitre 1, nous allons définir les caractéristiques d'un système expert. Nous allons présenter une architecture type et la fonction de chaque composant ainsi que les différentes possibilités de les représenter. Nous allons aussi lister les conditions d'éligibilité d'un domaine d'application pour un système expert ainsi que les critères de choix technologiques.

Au chapitre 2, nous allons décrire le domaine d'application 'fiscal'. Le choix s'est opéré sur la partie 1 de la déclaration fiscale à l'impôt des personnes physiques (IPP [G-6]) et nous allons jeter les bases essentielles à la compréhension du domaine.

Le chapitre 3 va donner les détails de l'application développée et expliquer les choix qui ont été opérés.

Ensuite, le chapitre 4, va exposer la modélisation du domaine fonctionnel en programmation logique par contraintes et plus particulièrement comment organiser la descriptions des règles du domaine de connaissances afin de répondre aux objectifs. Comme vous le verrez, nous ne traiterons pas de tous les codes mais seulement des plus courants ou ceux qui servent à nous convaincre de l'adéquation de la programmation logique par contraintes à ce domaine.

Le chapitre 5 va s'arrêter sur le résultat obtenu et critiquer le système développé. Nous allons voir quels objectifs sont atteints ainsi que les nombreuses pistes d'évolution. A l'issue de cette réflexion, une autre approche s'est faite sentir, approche qui sera aussi exposée.

Finalement le chapitre 6 va présenter les conclusions de ce mémoire.

Chapitre 1 : LE SYSTEME EXPERT

Résumé du chapitre :
Définition d'un système expert et de ses composants. Présentation des conditions d'éligibilités d'un domaine d'application pour Système Expert ainsi que des critères de choix technologiques.

1.1 *Définition*

Nous allons commencer par définir le concept de "Système Expert".

Une première définition émane de Clyde W. Holsapple & Andrew B. Whinston[2] :

"Un Système Expert utilise une compétence empruntée à un expert humain sur la manière de résoudre un problème spécifique ou une classe de problèmes apparentés".

Loin de nous en tenir à une seule définition, regardons celle de J.N. Chatain & A. Dussauchoy[3]:

"Un Système Expert a pour but la modélisation du comportement d'un expert humain, accomplissant une tâche de résolution de problèmes pour laquelle on ne dispose d'aucun algorithme et ce dans un domaine bien précis".

Cependant, deux idées majeures y sont répétées :

- ✓ L'expert humain
- ✓ Le domaine précis (ou spécifique)

Par expert humain, on représente le but d'un système expert comme "*la recherche de la meilleure solution possible dans l'état des connaissances*"[4]. Il ne s'agit donc pas de donner la solution optimale, l'homme en est généralement incapable. Les Systèmes Experts utilisent essentiellement des méthodes empiriques avec des solutions heuristiques. De plus, on est autant intéressé par le raisonnement de l'expert que par la réponse finale à la question posée.

Le domaine précis est inhérent à l'expertise. On n'est jamais expert en tout, on est toujours expert "en" (un domaine, une fonction, etc.). Le domaine doit donc être suffisamment complexe afin de justifier l'intervention d'un expert.

Les Systèmes Experts sont des systèmes dans lesquels la base de connaissance (la partie spécialisée) est indépendante du moteur d'inférence qui utilise le contenu de cette base.

---

[2] Clyde W. Holsapple & Andrew B. Whinston [B-2], page 23

[3] J.N. Chatain & A. Dussauchoy [B-3], pages 11-23

[4] J.N. Chatain & A. Dussauchoy [B-3], pages 7-11

La base de connaissance doit présenter les caractéristiques suivantes[5] :

- ✓ Les éléments qui constituent la base de connaissance sont indépendants les uns des autres, c'est le principe de granularité des connaissances.
- ✓ L'ordre d'introduction des connaissances n'a aucune importance.
- ✓ Une modification ne change que la résolution du problème et non le déroulement du programme.
- ✓ La connaissance est essentiellement de nature symbolique (et non numérique).
- ✓ La base de connaissance a un volume important.

Un système expert se distingue d'un Système d'Aide à la Décision (SIAD) par le fait que le Système Expert est capable de résoudre lui-même les problèmes, de prendre des décisions, et de raisonner sur les problèmes. Pour les SIAD, l'essentiel de l'intelligence et du savoir faire est dans les mains de l'utilisateur (décideur)[6].

Un système Expert se distingue d'un Knowledge-based System par le fait que le Système Expert est capable d'expliquer le processus de raisonnement[7].

1.2 *Architecture*

Nous allons donc compléter la définition par la description d'une architecture type d'un Système Expert, telle que représentée à la *Figure 1*.

Un Système expert se compose de 3 éléments principaux[8] :

1. Une **base de connaissance** qui décrit les objets pris en compte et leurs relations. Elle comprend aussi les cas particuliers et les exceptions, les différentes stratégies de résolution
2. Une **base de faits** qui comprend les données utilisateurs ainsi que les résultats intermédiaires qui ne sont pas conservés.
3. Un **moteur d'inférence** qui construit le raisonnement. Il exécute les inférences (ou déductions) au cours du processus de résolution, soit par modification, soit par adjonction d'éléments à la base de faits. Il détecte les connaissances intéressantes.

Il est aussi généralement complété par les éléments suivants :

1. Une **interface utilisateur** (ou système de consultation) qui permet de dialoguer avec le moteur d'inférence.
2. Un **module d'explication** pour tracer le chemin du raisonnement. Il permet aussi de vérifier la cohérence du système.
3. Un **module d'acquisition des connaissances**.

---

[5] J.N. Chatain & A. Dussauchoy [B-3], pages 11-13

[6] J.M. Jacquet, cours LIHD, *Techniques d'intelligence artificielle*, chapitre 3 (pages non numérotées)

[7] J.P. Fichefet, cours LIHD, *Méthodes et outils d'aide à la décision médicale* (pages non numérotées)

[8] J.N. Chatain & A. Dussauchoy [B-3], pages 13-16

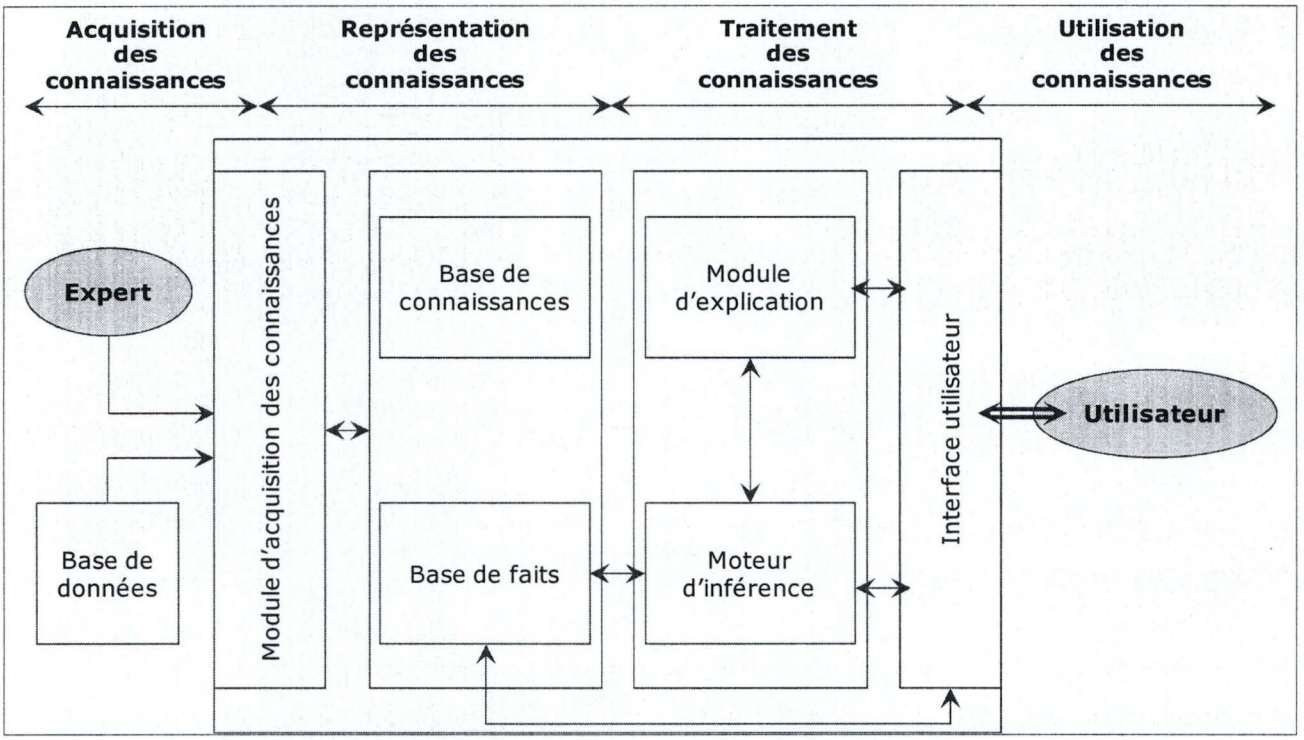


Figure 1 : Architecture traditionnelle d'un Système Expert[9]

1.3 *La Représentation des Connaissances*

Nous avons déjà déterminé que le domaine d'application d'un Système Expert doit être spécifique et complexe.

La représentation de ce domaine doit s'adapter à cette complexité et les méthodes utilisées peuvent être énumérées comme suit :

- ✓ Les règles de production
- ✓ Le calcul des prédicats
- ✓ Les réseaux sémantiques
- ✓ Les stratégies d'évolution

[9] J-N Chatain & A. Dussauchoy [B-3], page 14.

1.3.1 Règles de production

"Le principe de base de programmation en règles de production est que chaque règle est un morceau indépendant de connaissance (on dit: granule), c'est-à-dire qu'elle contient toutes les conditions de son application" [B-19]

L'approche par règles de production permet d'exprimer une connaissance de manière purement déclarative et de façon modulaire.

Une règle de production est une expression de la forme :

Membre de gauche (1) → membre de droite (2)

Ou

Si condition (1) alors action / conclusion (2)

Le membre de gauche, appelé "prémisse", peut se composer de :

- ✓ propositions logiques
- ✓ clauses
- ✓ suites de symboles
- ✓ expressions booléennes

Le membre de droite, appelé "action" ou "conclusion", peut se composer de :

- ✓ propositions logiques
- ✓ clauses
- ✓ suites de symboles
- ✓ fragments de programmes (pattern-directed invocation)

On peut aussi y associer un facteur de crédibilité, disons α, qui est aussi appelé le "facteur d'atténuation de la règle".

D'une façon générale, $\alpha \in [-1, +1]$:

- ✓ Les valeurs positives de α expriment la crédibilité plus ou moins forte que l'on attribue à la conclusion.
- ✓ Plus α est proche de 1, plus on affirme que la conclusion avec certitude.
- ✓ Les valeurs négatives de α expriment la crédibilité de la négation de la conclusion.

1.3.2 Logique des prédicats

La logique des prédicats est fréquemment utilisée comme langage de représentation des connaissances et comprend :

- ✓ Des variables
- ✓ Des fonctions
- ✓ Des constantes
- ✓ Des parenthèses
- ✓ Des prédicats

Exemple : TRAVAILLER(Cathy, LS&Co)[10]

Où

- ✓ TRAVAILLER est un prédicat et
- ✓ Cathy et LS&Co deux constantes qui sont les arguments de ce prédicat.

Un prédicat exprime donc une relation qui porte sur des objets, ces objets sont les arguments du prédicat.

La formule précédente signifie que Cathy travaille pour LS&Co.

La logique des prédicats est toujours capable de donner une interprétation à une formule syntaxiquement correcte (on parle de WFF pour Well Formed Formulas).

C'est ainsi qu'un prédicat pourra prendre les valeurs "vrai" ou "faux". Dans l'exemple précédent, TRAVAILLER(Cathy, LS&Co) est vrai.

L'interprétation consiste donc à faire correspondre les éléments figurant dans la réalité et leur écriture en termes de prédicat.

Il est évident qu'une même réalité peur s'écrire de différentes manières.

Exemple : TRAVAILLER (Cathy, LS&Co)

EMPLOYE (Cathy, LS&Co)

FONCTION (Employe, Cathy, LS&Co)

Il est également possible d'utiliser des quantificateurs universels :

Exemple : Tous les corbeaux sont noirs, peut s'écrire

$[(\forall x)\ (\text{Corbeau}(x) \rightarrow \text{Couleur}(x, \text{noir})]$

La logique du premier ordre n'autorise pas le fait que les quantificateurs portent sur autre chose que des variables ($(\forall P)\ P(A) \rightarrow B$ est interdit).

Toute WFF fermée peut être exprimée sous forme de clauses.

Une *clause* est une construction de la forme :

$L_1 \vee \ldots \vee L_n$ où chaque L_i est un atome ou la négation d'un atome.

Les variables d'une clause sont implicitement quantifiées universellement en tête de la clause.

Une *clause de Horn* est une clause contenant au plus un atome non nié.

---

[10] LS&Co est pour Levi Strauss & Co préféré à Levi's® qui est une marque de la société.

La restriction au sous-ensemble des clauses de Horn signifie que seules les formules du type :

$$H1 \wedge H2 \wedge .... \wedge H_n \rightarrow C1, \text{ sont permises.}$$

C'est-à-dire une conjonction d'hypothèses qui entrainent une seule conclusion. Une clause de cette forme peut donc s'interpréter comme une proposition conditionnelle:

$$\text{Si H1 et H2.... et } H_n \text{ alors C1.}$$

Remarquons au passage l'équivalence entre cette forme de représentation des connaissances et les règles de production.

1.3.3 Réseaux sémantiques

Les réseaux sémantiques sont des graphes dont les nœuds représentent des objets, des concepts, des actions, des événements et les arcs représentent leurs interrelations (*Figure 2*).

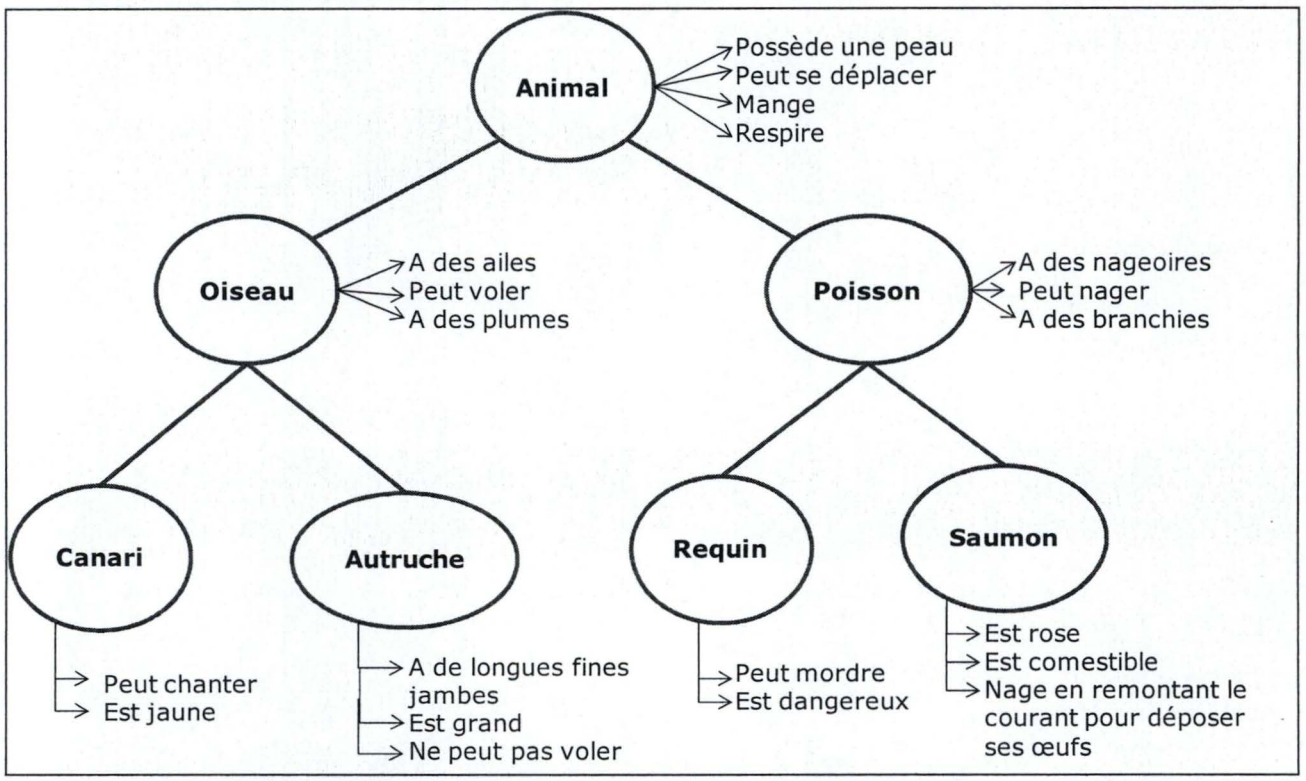


Figure 2 : Illustration d'un arbre sémantique par Quillian[11]

Les réseaux sémantiques permettent une représentation des connaissances granulaires.

Les objets fortement interconnectés peuvent y être aisément reportés et retrouvés (grâce aux nombreux algorithmes de recherche dans les graphes).

Cette représentation permet aussi d'inclure des héritages de propriétés et des définitions de prédicats dans la description des nœuds.

---

[11] M.R. Quillian [B-20], pages 216-270.

Cependant, la sémantique exacte des connaissances reste difficile à appréhender et surtout, ce formalisme est inadéquat pour la génération des explications.

1.3.4 Les frames

Les frames, ou objets structurés, sont une évolution des réseaux sémantiques qui permettent d'ajouter des extensions procédurales aux nœuds[12].

Les frames sont des classes d'objets réunissant les propriétés communes à tous les éléments de la classe.

En général, les frames se composent de plusieurs niveaux hiérarchiques. Un niveau supérieur fixe (entités fixes) et des niveaux inférieurs aussi nommés éléments terminaux ou variables

Les frames peuvent être représentés par une structure de record reprenant une liste de paires attributs-valeurs. Les attributs peuvent dénoter des objets très divers comme une propriété, une relation, une procédure, une règle de production, un prédicat, etc.

Marvin Minsky[12] préconise trois méthodes d'organisation de l'information :

1. La ***reconnaissance*** consiste à repérer pour un individu donné le ou les concepts (on dit aussi types génériques) auxquels cet individu appartient (on dit alors que l'individu est une instance du concept) : l'individu Pierre peut ainsi être reconnu comme étant un étudiant, ou une personne ou..., selon les propriétés déclarées pour l'individu Pierre.
2. L'***agrégation*** consiste à relier un individu à d'autres individus : l'objet Pierre peut ainsi être relié à l'objet symbolisant sa jambe... Il faut évidemment que la représentation soit telle qu'une jambe ne puisse appartenir qu'à une unique personne. Lors des mises à jour (on décide par exemple d'ôter l'individu Pierre de la base de connaissances), il faut aussi supprimer tout ce qui en est directement dépendant (la jambe de Pierre en l'occurrence).
3. La ***classification*** consiste à hiérarchiser un ensemble de concepts. Un concept peut ainsi être considéré comme sous-concept d'un autre (oiseau par rapport à animal). Classer un concept C sous un autre concept C', c'est indiquer que toutes les propriétés de C' sont aussi des propriétés de C (on parle dans ce cas d'héritage de propriétés). La sémantique des relations d'héritage est particulièrement complexe dès que l'on cherche à rendre compte des propriétés par défaut, des exceptions, ou à masquer des valeurs de propriétés.

Au vu de ces principes, on comprend que les frames peuvent être représentés par des classes de langage orienté objet comme par exemple Java[13].

Les nœuds sont des classes ou des objets et les relations sont des références d'objets contenues dans les objets ou les variables de classe.

La hiérarchie est définie en utilisant les fonctions du langage comme le "sousClasse extends classe" en Java.

Le rattachement d'un objet à une classe est opéré au moyen de la construction d'objet : "objet = new classe".

Les extensions procédurales sont représentées par les méthodes définies pour la classe.

---

[12] Marvin Minsky [B-21].

[13] Java est un langage de programmation créé et édité par Sun Microsystems.

1.3.5 Stratégies d'évolution

Les stratégies d'évolution (Evolutionary Computation) est une branche des techniques d'intelligence artificielle (autrement dit TIA) qui s'inspire de modèles de la nature et qui regroupe les techniques associées aux réseaux de neurones, aux algorithmes génétiques, etc.

Notamment, les systèmes experts traitant des sujets issus du domaine financier doivent traiter de très nombreuses séries temporelles[14]. De plus, les marchés financiers sont des marchés humains qui ne suivent pas une règle physique qui peut être représentée par une formule. Au contraire, les marchés évoluent dans le temps et les opportunités de faire du profit dépendent largement des changements de stratégie des adversaires.

Un système basé sur les stratégies d'évolution devrait permettre de s'adapter en ce qui concerne les données de références à considérer, les méthodes d'évaluation, etc.

1.4 *Méthodologie*

1.4.1 Choix du Domaine

Nous allons définir les caractéristiques du domaine d'application qui justifient la création d'un système expert :

- ✓ Les problèmes ne peuvent pas se résoudre par des méthodes numériques classiques, requièrent de nombreux paramètres ou sont difficilement formalisables
- ✓ Les problèmes requièrent l'intervention d'un expert humain afin d'être résolus
- ✓ Les connaissances évoluent vite
- ✓ Les capacités de l'expert reposent sur des connaissances, du jugement, de l'expérience et non sur du bon sens
- ✓ La résolution du problème nécessite plusieurs heures humaines
- ✓ Les experts humains sont peu disponibles, onéreux ou, dans le cadre d'une entreprise, la rotation du personnel est élevée

[14] Roy Rada [B-17].

1.4.2 Choix de l'outil

Différentes options sont à considérer. La première concerne le moteur d'inférence, à savoir s'il faut employer un moteur d'inférence existant ou construire le sien.

Dans les outils existants, le choix du moteur s'opère en fonction du choix de la représentation des données, elle-même dictée par le domaine d'expertise que l'on veut modéliser.

Nous n'allons pas énumérer les outils existants au moment de la rédaction du présent mémoire, mais plutôt surligner quelques éléments de décision à considérer lorsque qu'on désire développer un système expert en pratique.

La première condition sous-tendant le choix de l'outil est le profil et l'expertise que l'on dispose.

En effet, la plupart des représentations de données appellent aux techniques d'intelligence artificielle, domaine qui n'est pas la spécialité de tous les informaticiens.

Un autre facteur est le degré de support de l'éditeur de l'outil : l'outil est-il toujours maintenu, le support efficace, peut-on trouver des spécialistes qui peuvent aplanir les difficultés qui seront rencontrées, etc.

La tentation de développer son propre moteur d'inférence perd d'intensité ces dernières années. De plus en plus, on préconise davantage un système ouvert et intégré à différentes technologies au détriment d'un système, peut-être mieux adapté au cas étudié mais, tellement spécifique qu'il est peu réutilisable [B-17].

Cette ouverture permet de combiner les avantages des différentes technologies employées. Une partie du système permet de raisonner et répondre aux questions, une autre permet un traitement des réponses plus sophistiqué et leur distribution aux personnes concernées.

La plupart de ces considérations sont à prendre si l'objectif est produire un système qui est vraiment destiné être utilisé et nous serons donc moins regardant dans le cadre de ce mémoire qui a une vocation somme toute très théorique.

Chapitre 2 : Le Domaine d'Application

Résumé du chapitre :
Situation du domaine dans le paysage fiscal belge.
Présentation de ses caractéristiques et principes fondamentaux.

2.1 *Place de l'IPP dans le Paysage Fiscal*

Les recettes de l'Etat sont constituées par les revenus générés par la fiscalité.

La source principale est constituée des impôts.

Le domaine d'application choisi est l'IPP, puisque c'est ainsi que les experts en fiscalité appellent l'Impôts des Personnes Physiques.

Comme nous le voyons dans la *figure 3* ci-dessous, l'IPP est un impôt direct, puisqu'il est prélevé sur les bénéfices et les revenus.

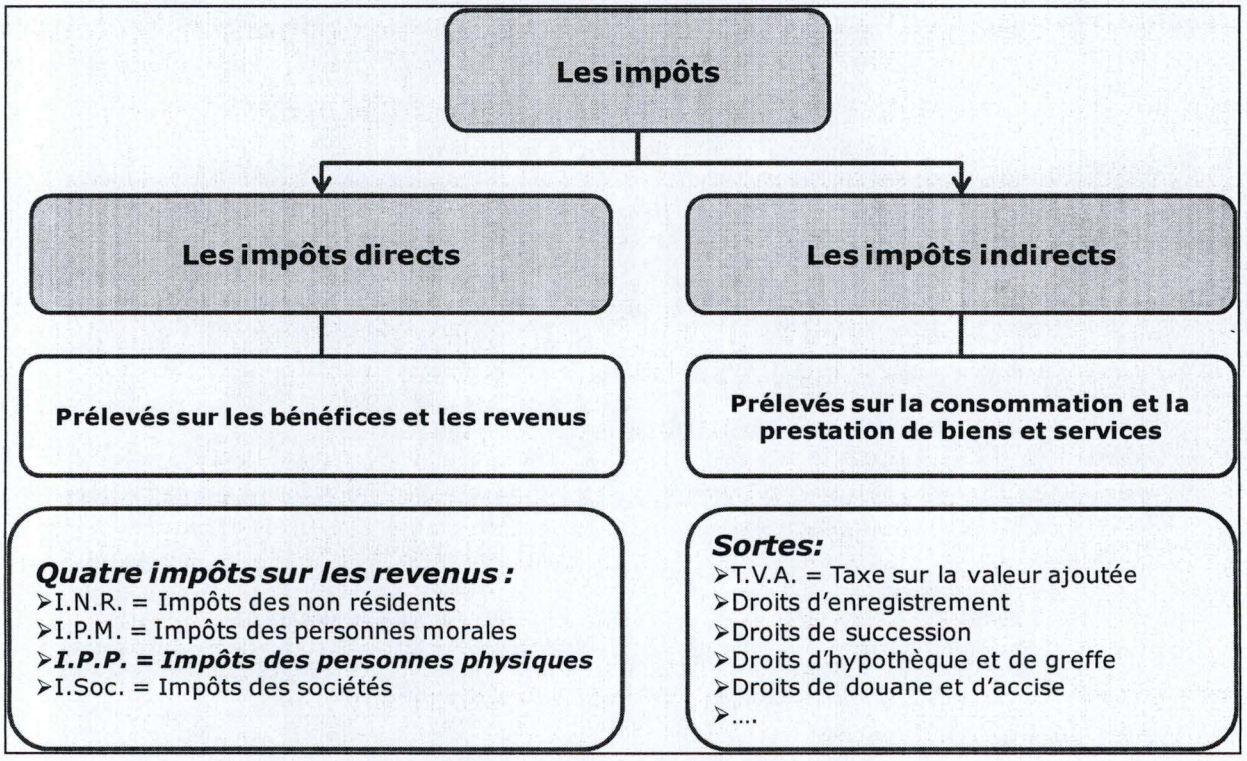


Figure 3 : Paysage fiscal belge [15]

[15] Cours IPP de la Dexia University [B-12].

L'IPP est un impôt levé sur les revenus des personnes physiques résidant en Belgique et il est calculé sur les revenus mondiaux nets imposables, réalisés au cours d'une année civile.

Les revenus nets sont les revenus bruts immobiliers, mobiliers, professionnels et divers, dont sont déduits, en principe, les frais supportés pour acquérir ou conserver ces revenus.

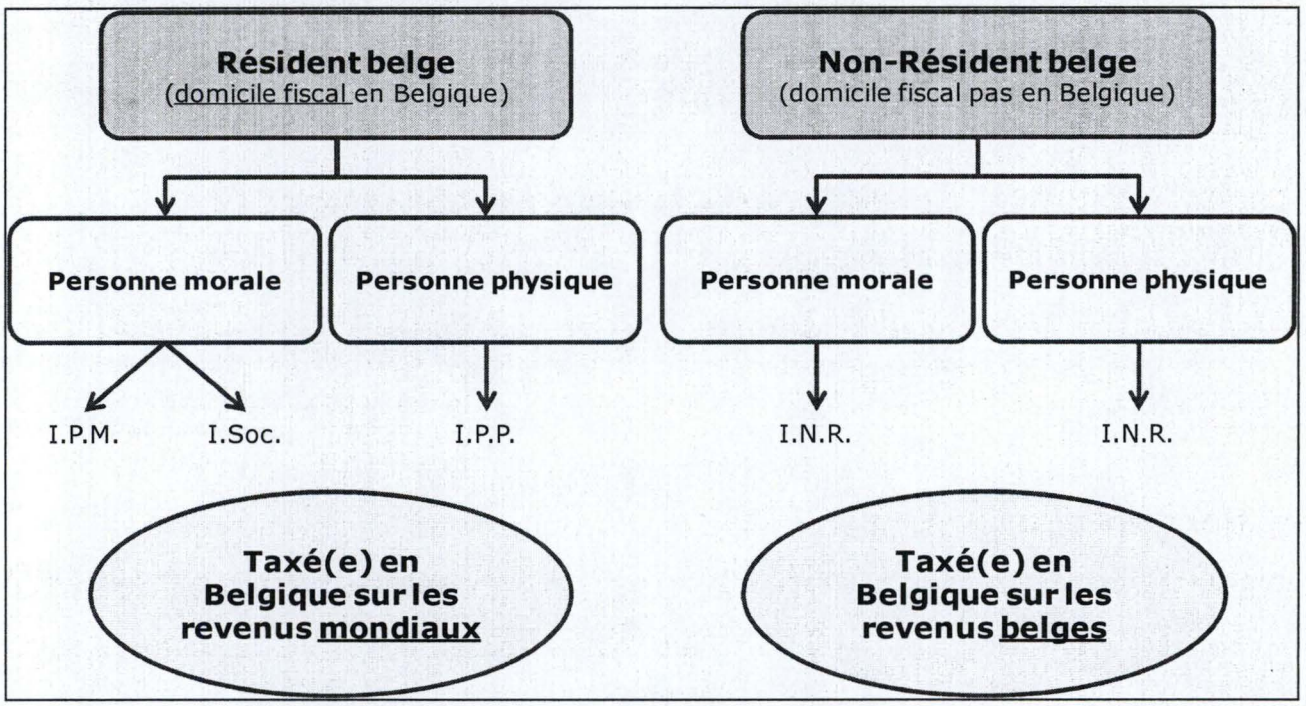


Figure 4 : Les impôts sur les revenus en Belgique [16]

Contrairement à l'impôt des sociétés qui est structuré selon une logique fiscale claire, l'IPP est un impôt que l'on peut qualifier de 'politique'. Il est composé d'une série disparate de mesures fiscales issues du pouvoir politique. Cela le rend donc peu structuré et plus complexe.

Pour nous en convaincre, comparons seulement la taille des déclarations à remplir ainsi que celle des cours professionnels délivré par le Service Public Fédéral Finances :

Déclaration à l'Impôt des Sociétés, 6 pages[17] avec un cours professionnel de 211 pages [18]

Déclaration à l'Impôt des Personnes Physiques, partie 1 (10 pages) + partie 2 (8 pages) avec un cours professionnel de 657 pages.

---

[16] Cours IPP de la Dexia University [B-12].

[17] Formulaire 275-1, disponible en ligne : http://ccff02.minfin.fgov.be/finformportlet/AppelPdf?id=35&mode=inline

[18] Cours du Ministère des Finances, disponible en ligne : http://ccff02.minfin.fgov.be/KMWeb/

2.2 *Caractéristiques de l'IPP*

Avant d'entrer dans le vif du sujet, il nous a semblé utile de donner sommairement quelques caractéristiques générales de l'impôt des personnes physiques qu'il est conseillé de ne pas perdre de vue[19] :

1. **Impôt de synthèse**, car il est perçu sur l'ensemble des revenus du contribuable.
2. **Impôt subjectif**, car il est établi en tenant compte de la situation personnelle du contribuable, situation qui a une répercussion sur ses facultés contributives. C'est ainsi qu'interviennent notamment les charges de familles, certaines dépenses personnelles, la hauteur et la nature des revenus, etc.
3. **Impôt progressif**, car le pourcentage de l'impôt augmente dans la mesure où le revenu s'accroît. La structure actuelle des taux d'imposition comporte 5 tranches de revenus et l'impôt est calculé suivant un tarif progressif allant de 25 à 50%.

Exemple :

Le contribuable A a un revenu imposable de 18 750,00 EUR. Il paie 6 185,25 EUR d'impôts, soit une moyenne de 33%.

Le contribuable B a un revenu imposable de 37 750,00 EUR. Il paie 14 953,00 EUR d'impôts, soit une moyenne de 40%.

Bien que le contribuable B gagne deux fois plus que le contribuable A, il paye plus du double d'impôts.

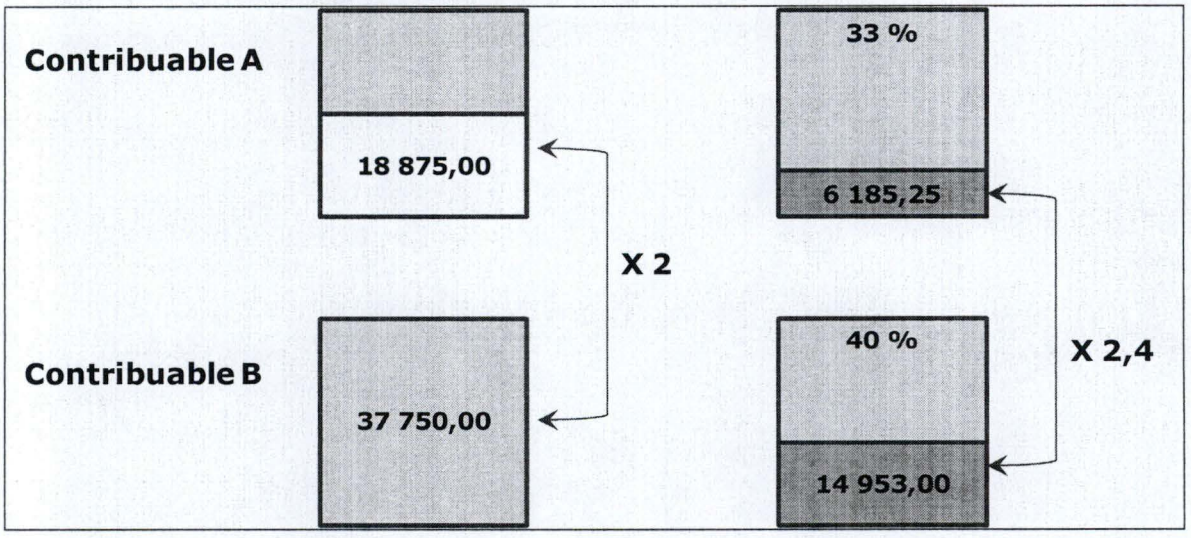


Figure 5 : l'IPP est un impôt progressif par paliers [20]

[19] Roland Rosoux, [B-16], pages 2 à 5.

[20] Cours IPP de la Dexia University [B-12].

4. **Cumul ou décumul des revenus**, car tous les revenus imposables d'un même contribuable sont cumulés (on parle de "globalisation" des revenus). Il existe toutefois une exception fondamentale à ce principe en ce qui concerne les "conjoints"[21], pour lesquels le revenu imposable est fixé séparément.
5. **Imputation des revenus professionnels**, car une quote-part des revenus professionnels d'un des conjoints peut être attribuée à l'autre en vertu du "quotient conjugal" et/ou de l'attribution au conjoint aidant.
6. **Annualité de l'impôt**, car l'IPP frappe les revenus d'une année civile déterminée. La période dont les revenus doivent servir de base à l'imposition s'appelle la période imposable [G-8].

   L'exercice d'imposition [G-3] correspond à l'année à laquelle sont rattachées les règles relatives à l'établissement de l'imposition sur les revenus de la période imposable. Il est désigné ordinairement par le millésime de l'année qui suit celle de la période imposable.
7. **Non-déductibilité des impôts**, car l'impôt *belge* qui atteint les revenus passibles de l'impôt des personnes physiques ne peut être déduit à titre de frais grevant ces revenus.
8. **Territorialité de l'impôt**, car cet impôt atteint les personnes physiques qui ont un lien avec le territoire de l'Etat belge; il s'agit en l'occurrence des habitants du Royaume. En principe, il s'agit des personnes physiques qui sont inscrites au Registre national des personnes physiques. Cependant, est réputé "habitant du Royaume" celui qui a établi dans le pays son domicile ou le *siège de sa fortune*.
9. **Assiette de l'impôt**, car le principe de territorialité ne porte cependant pas atteinte à certaines règles particulières telles que :
   - ✓ L'exonération d'impôt prévue dans les conventions préventives de la double imposition que la Belgique a souscrit avec des états étrangers. Les revenus exonérés sont toutefois pris en considération pour la détermination de l'impôt (et surtout du taux d'imposition), mais l'impôt afférent à ces revenus est porté en diminution de l'impôt total (exonération sous réserve de capacité contributive).
   - ✓ La réduction de moitié de l'impôt qui se rapporte proportionnellement à certains revenus d'origine étrangère.
10. **Classification des revenus d'après leur nature**, car la loi classe les revenus imposables en quatre catégories, à savoir :
    - ✓ Les revenus de biens immobiliers
    - ✓ Les revenus des capitaux et bien mobiliers
    - ✓ Les revenus professionnels
    - ✓ Les revenus divers

[21] Ou cohabitant légal [G-15]

11. **Classification des revenus d'après leur origine**, car les revenus se répartissent aussi en :
    - ✓ Revenus belges
    - ✓ Revenus étrangers, dont il convient de faire une distinction en vertu de la caractéristique exposée sous 9.

    Il faut aussi remarquer que les traitements de certains fonctionnaires d'organisation internationales ou supranationales déterminées sont exonérés d'impôts belges, qu'ils soient ou non des habitants du Royaume.
12. **Revenus nets**, car le revenu imposable à l'IPP est constitué par la somme des revenus nets des quatre catégories énumérées sous 10, diminuée des dépenses déductibles.

    Pour calculer le revenu imposable, il faut donc en premier lieu rechercher séparément le montant net des revenus de chaque catégorie, suivant les règles propres à chacune des catégories.

    Rappelons que l'impôt belge qui frappe un revenu ne peut pas être déduit de ce revenu (voir caractéristique 7).

2.3 *Modélisation*

2.3.1 Calcul de l'impôt

L'objectif de ce mémoire réside dans la modélisation du domaine en contraintes et donc, plutôt qu'une longue description narrative, nous avons opté pour une présentation schématique de ce calcul plutôt qui peut se représenter comme suit [B-12] :

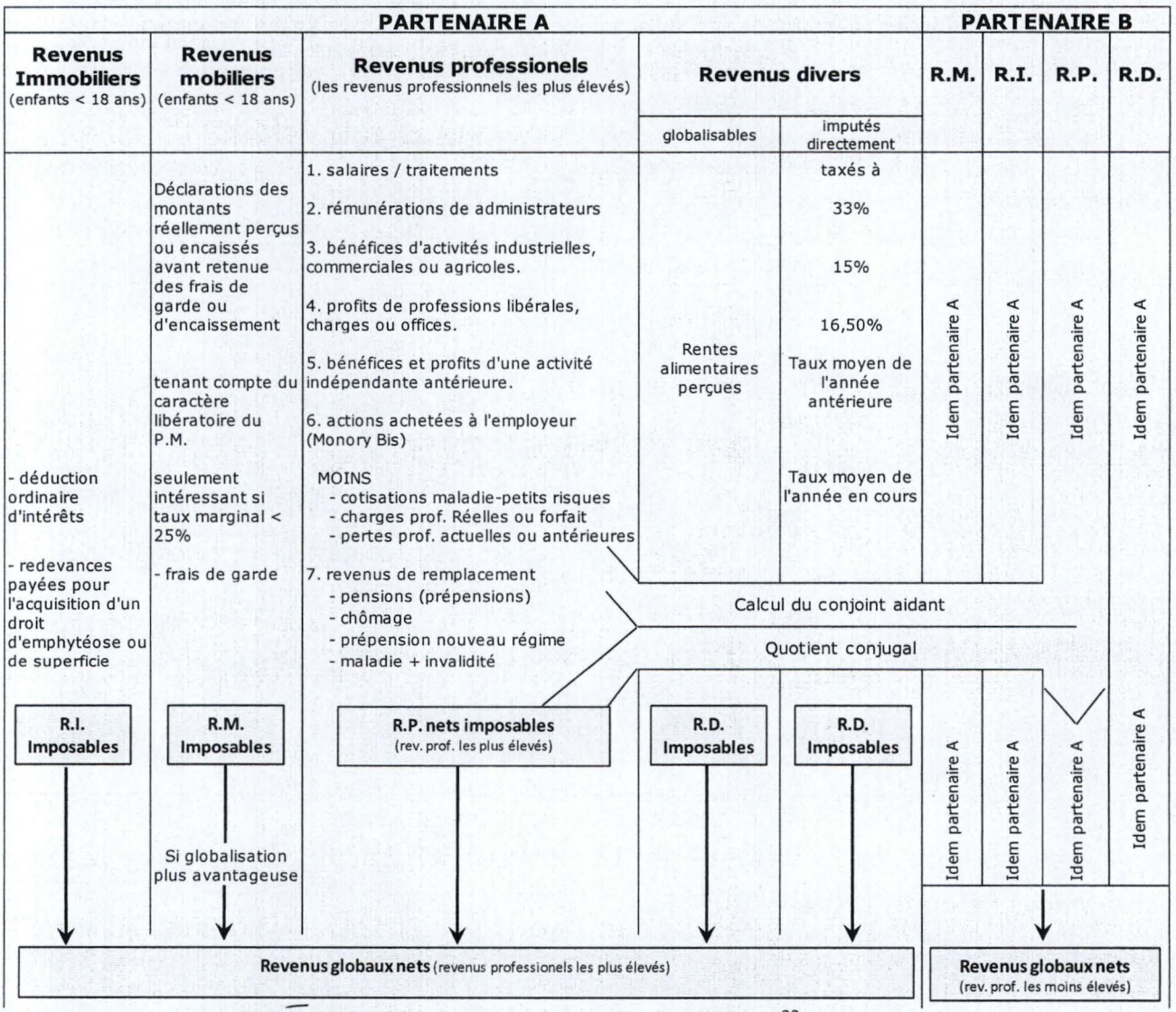


Figure 6 : Schéma du calcul d'impôt – partie 1/2 [22]

[22] Cours IPP de la Dexia University [B-12].

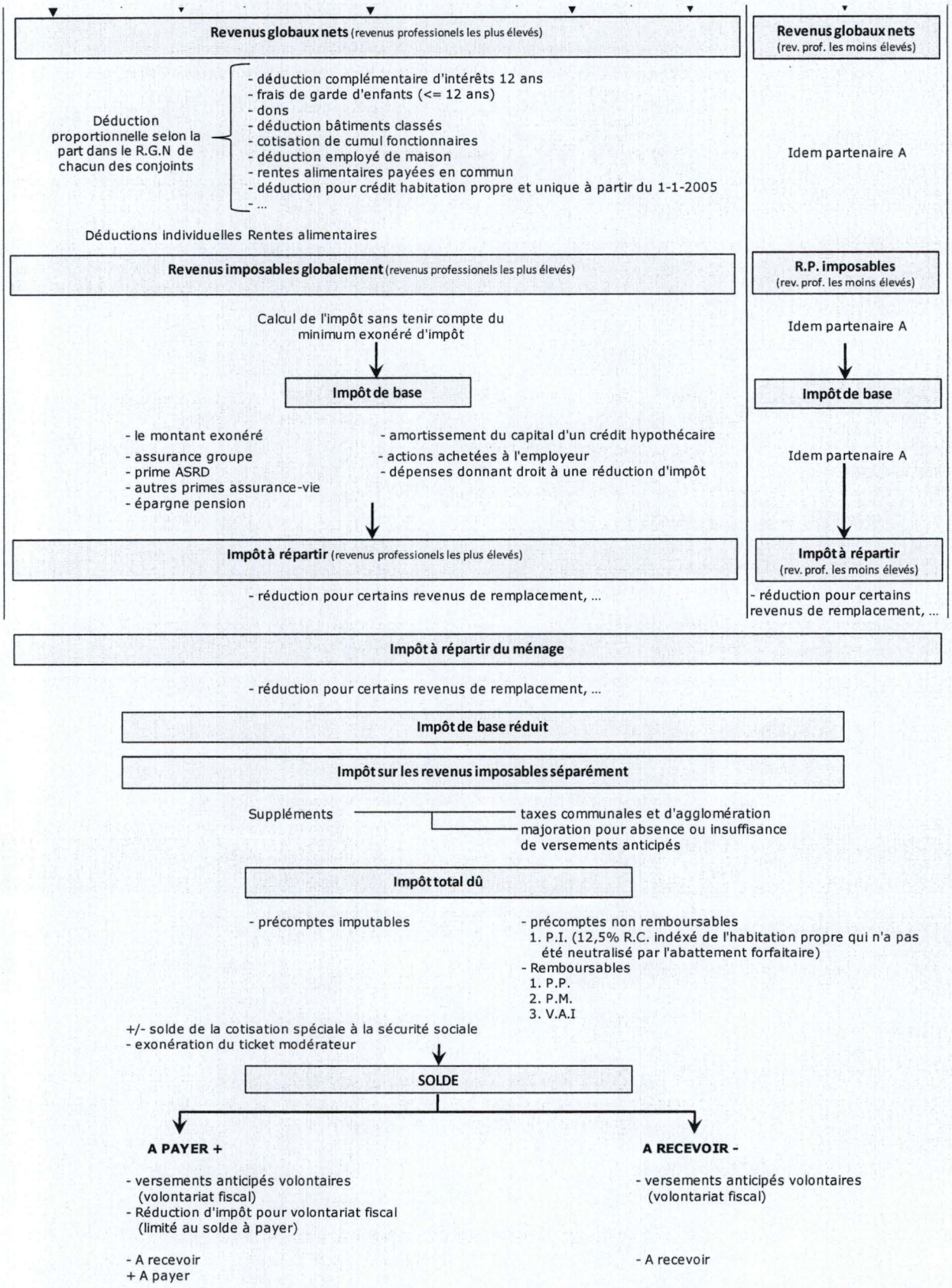


Figure 7 : Schéma du calcul d'impôt – partie 2/2 [23]

[23] Cours IPP de la Dexia University [B-12].

Nous allons cependant nous attarder sur quelques particularités qu'il nous a semblé utile de surligner.

Tout d'abord, le calcul d'impôt s'effectue ***par palier et par tranche***.

Ainsi, le premier taux d'imposition de 25% s'applique sur la première tranche de revenu imposable de 7 420 EUR (IPP 2008 sur les revenus 2007).

Si les revenus sont supérieurs à cette première tranche, le supplément, jusqu'à la limite de la tranche suivante, est taxé au taux supérieur par exemple de 30%. Même dans ce cas, les revenus de la première tranche restent toujours taxés à 25%.

Afin de mieux visualiser ce calcul par palier et par tranche, une illustration graphique est présentée en *Figure 8*.

Le taux moyen d'imposition n'est ni de 25, ni de 30% mais un ratio entre 25 et 30%[24].

Il faut donc faire la distinction entre le taux moyen d'imposition [G-16], à calculer, et le taux marginal d'imposition [G-17] qui correspond au taux appliqué à la tranche la plus élevée des revenus.

Nous avons aussi la notion de ***minima exonérés d'impôts***.

Il s'agit d'une partie des revenus qui est exonérée d'impôt et qui dépend de la situation familiale.

Les minima exonérés d'impôts sont toujours appliqués sur les tranches de revenu les plus basses. L'avantage fiscal que l'on réalise de cette manière n'est donc pas un avantage marginal (c'est-à-dire au taux le plus élevé) mais un avantage dans les tranches inférieures (c'est-à-dire au taux le plus bas). Cette particularité est aussi illustrée dans la *Figure 8*.

Exemple (voir *Figure 8*) :

Un couple dispose des revenus professionnels nets imposables suivants :

- ✓ Personne A : 19 400,00 EUR
- ✓ Personne B : 13 500,00 EUR

Ils ont trois enfants à charge dont un de moins de trois ans et pour lesquels ils ne déduisent pas de frais de garde.

[24] Mais comme nous le verrons dans l'exemple, il est même inférieur à 25% grâce à l'application des minima exonérés d'impôts.

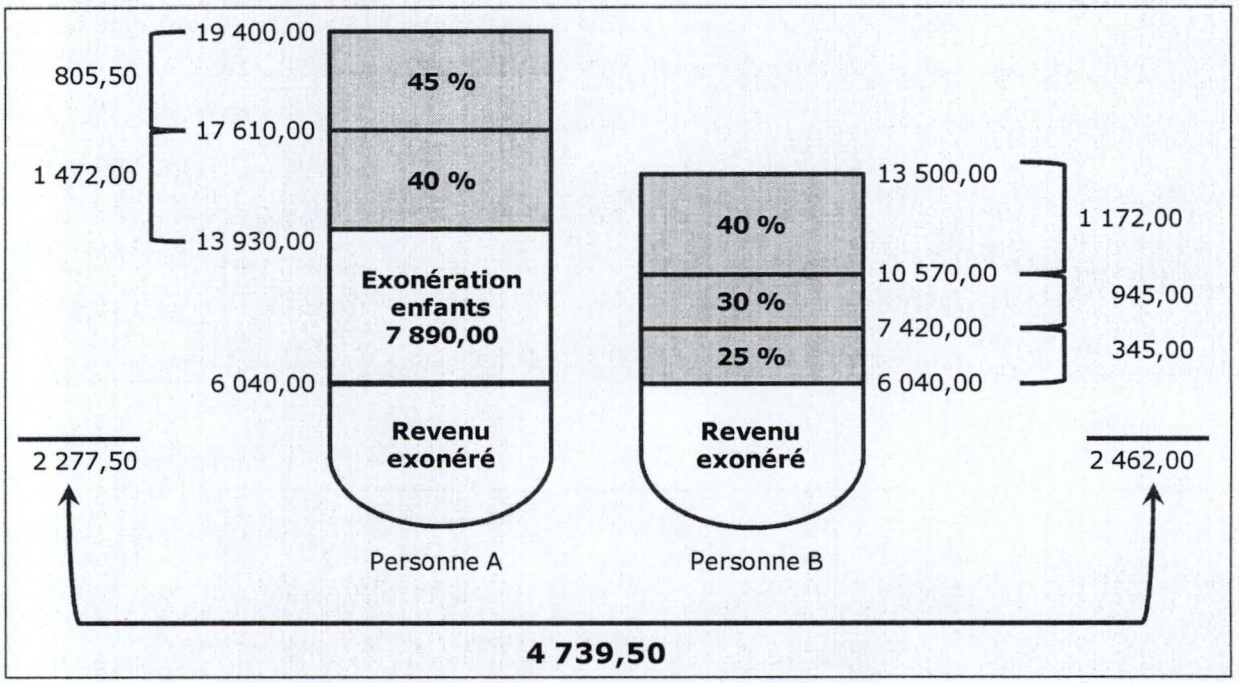


Figure 8 : l'IPP et les minima exonérés d'impôts [25]

Nous pouvons ainsi remarquer que l'avantage fiscal lié à la majoration du minimum exonéré d'impôt de 7 890,00 EUR pour enfants à charge est de 2 634,00. Soit (1 380,00 x 25%) + (3 150,00 x 30%) + (3 360,00 x 40%).

Dans cet exemple, le taux marginal d'imposition est donc de 45 %, tandis que le taux moyen d'imposition est de 14,41 %[26]

[25] Cours IPP de la Dexia University [B-12].

[26] Soit 4 739,50 / (19 400,00 + 13 500,00)

2.3.2 Autres règles

Cependant, nous ne pouvons pas limiter le domaine au calcul de l'impôt.

D'autres règles ont pu être identifiées :

- ✓ Règles simples d'exclusion
- ✓ Règles de condition d'application

2.3.2.1 Règles simples d'exclusion

La déclaration fiscale est truffée de règles d'exclusion que nous qualifions de simple.

Par exemple, si on est célibataire, on n'est pas marié, ni cohabitant, ni veuf, etc.

La case "vous vous êtes marié en 2007..." qui porte le numéro 1003, ne peut être cochée que si la case portant le numéro 1002 "marié" est cochée.

Cela concerne principalement les cases du cadre 2 (Renseignements d'ordre personnel et charges de famille) et aucune de ces règles ne sera modélisée dans le système expert et ce pour les raisons suivantes :

- ✓ Elles n'ont pas de valeur en tant que variable. En effet, la situation familiale n'est pas une variable mais une donnée. Si on veut vraiment appréhender l'incidence d'un changement sur les impôts, il suffira d'utiliser le programme avec les nouvelles données comme pour n'importe quel programme de calcul d'impôt.
- ✓ Elles peuvent facilement se gérer dans l'interface utilisateur. Le programme de calcul d'impôt de la banque Dexia[27] cache ce genre de case ou les fait apparaître selon les cases précédentes qui ont été sélectionnées (la case 1003 n'apparait que si la case 1002 a été préalablement cochée).

[27] Disponible sur leur site à l'adresse suivante (visiblement l'application Sofisk des éditions Standaard) :
http://www.dexia.be/Fr/Particulier/DOSSIERS/Fiscoweb/Calculate/

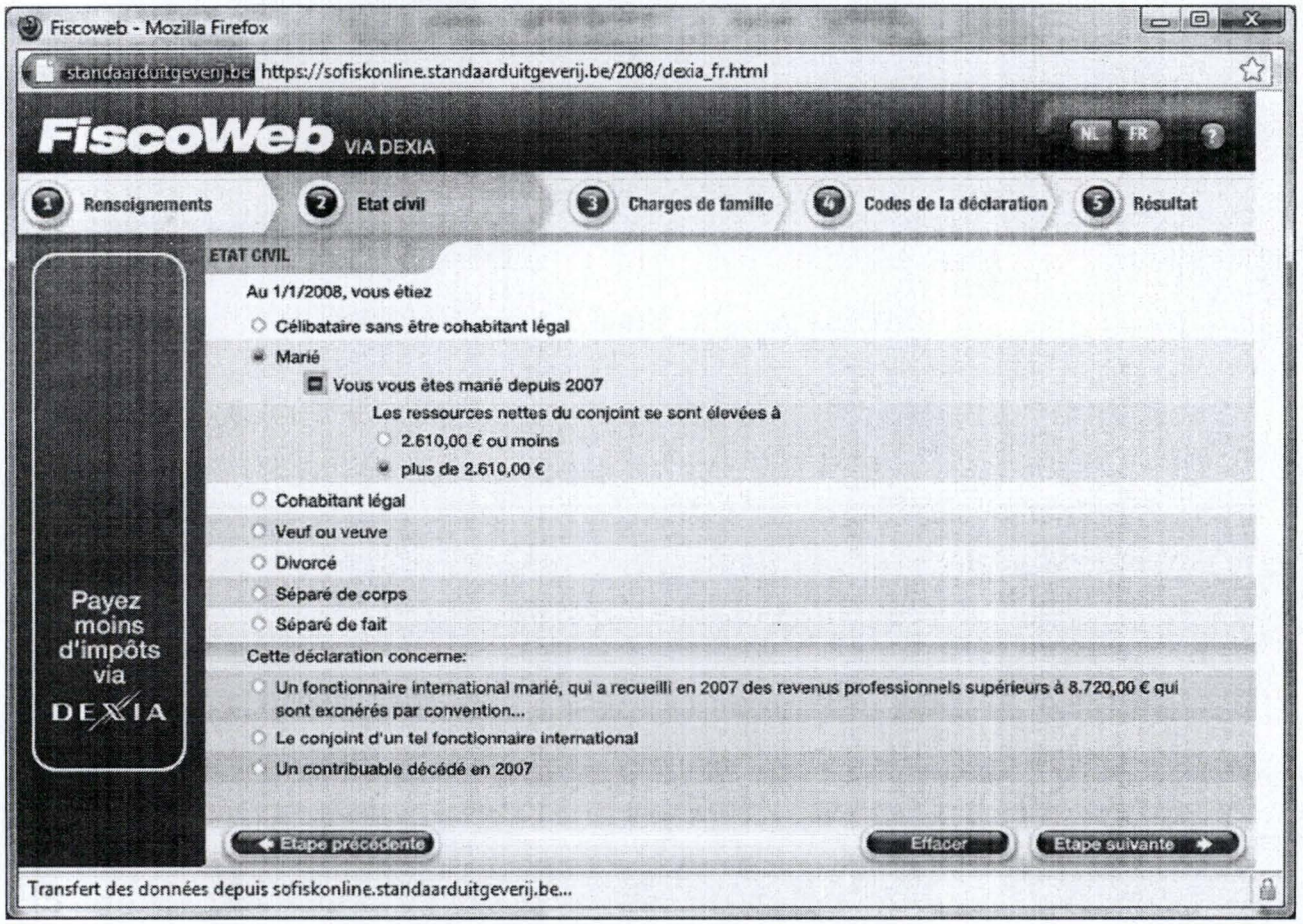


Figure 9 : Exemple de gestion de contrainte simple[27]

Par contre, nous pouvons aussi classifier de "règle d'exclusion simple" les possibilités de cumul ou non de différentes mesures de dépenses déductibles (cadre 7) ou de dépenses donnant droit à un avantage fiscal (cadre 8) ou de dépenses donnant droit à des réductions d'impôt (cadre 9) ou autres crédit d'impôt (cadre 10).

Par exemple, on ne peut pas cumuler la réduction d'impôt liée à l'achat d'action propre à l'employeur (cases 1362-2362 du cadre 9) et celle liée à l'épargne-pension (cases 1361-2361 du même cadre 9).

Ce genre de règle par contre sera modélisé dans le système car l'alternative est typiquement le genre de question qu'on doit pouvoir poser et comprendre.

L'idéal sera aussi de permettre au système de raisonner avec ces alternatives en tant que variable et ainsi donner des pistes afin de permettre un payement optimal (donc minimal) de l'impôt dans le respect de la légalité.

2.3.2.2 *Règles de condition d'application*

Pour de nombreuses cases des cadres 7 à 10, il existe des conditions d'application qui conditionnent la possibilité de demander une réduction d'impôt (ou un avantage fiscal ou un crédit d'impôt, etc.).

Par exemple, si nous reprenons le cas de l'achat de part de l'employeur (cases 1362-2362 du cadre 9), il ne s'agit pas d'acheter n'importe quelles parts de son employeur, mais uniquement des nouvelles actions émises lors de la fondation de l'entreprise ou lors d'une augmentation de capital. De plus, l'entreprise doit être établie dans un pays membre de l'EEE et le travailleur doit les conserver pendant minimum 5 ans. Si l'entreprise est une filiale, le travailleur peut aussi acquérir des parts de la maison mère.

Ces conditions sont énumérées dans la législation ainsi que dans la notice explicative envoyée avec la déclaration.

Ces contraintes liées aux conditions d'application ne seront pas modélisées dans le système expert pour les raisons suivantes :

- ✓ Elles sont généralement liées à des caractéristiques techniques que tout le monde ne maitrise pas. Par exemple, pour pouvoir bénéficier d'une réduction pour économie d'énergie (cases 1363-2363 du cadre 9), il faut que les travaux soient payés pendant la période imposable, réalisés par un entrepreneur enregistré qui atteste que les travaux *répondent aux caractéristiques techniques requises*. Les modéliser n'a aucun sens puisque la majorité des utilisateurs du système ne peuvent pas les comprendre.
- ✓ Elles sont généralement garanties par le fournisseur. Pour les travaux en vue d'économiser l'énergie, il suffit d'exiger de l'entrepreneur que ceux-ci répondent aux conditions fiscales et celui-ci se chargera de l'attester. Pour les parts de l'employeur, c'est en général l'entreprise qui se charge de remplir ces conditions et propose donc au travailleur de les acquérir en mettant en avant la possibilité de bénéficier de cette réduction fiscale (un peu comme un argument publicitaire).

2.4 *Tax on Web*

Nous ne pouvons pas parler de déclaration fiscale ou de calcul d'impôt sans mentionner le célèbre outil du S.P.F. Finances "Tax on Web"[28].

Cet outil permet d'abord au contribuable de rentrer sa déclaration de façon électronique, à condition de pouvoir s'identifier au moyen de codes fournis par le S.P.F. Finances (token) ou grâce à sa carte d'identité électronique.

Le contribuable doit ainsi remplir les codes et il peut obtenir le calcul de l'impôt. Malgré les avertissements d'usage que l'impôt calculé est une approximation, nous pouvons quand même présumer que cette simulation sera très proche de la réalité et la plus fiable qu'on pourra obtenir.

Certains codes nécessitent un calcul préalable, à réaliser par le contribuable avant d'être introduits. Tax on Web propose de nombreuses aides (wizards) afin d'aider le contribuable dans ces calculs.

Par exemple, si l'on reprend l'exemple des dépenses faites en vue d'économiser l'énergie, le montant à introduire au code 1363 (ou 2363) du cadre 9 doit correspondre à 40% de la facture (de remplacement d'une ancienne chaudière par exemple) limité à 2 600,00 EUR par habitation et par an. Le wizard, qui est illustré dans la *Figure 10*, demande le montant total payé et renvoie 40% de ce montant (éventuellement limité à 2 600,00 EUR).

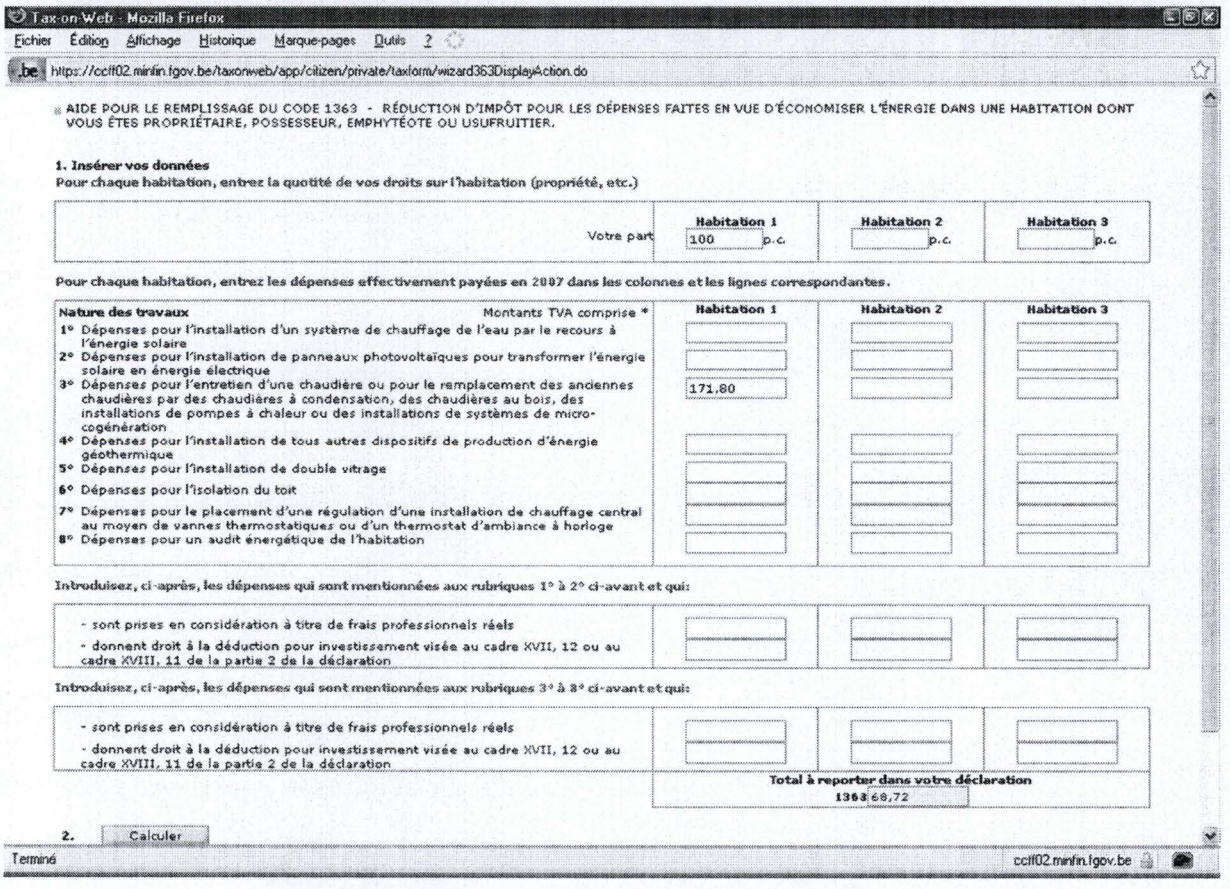


Figure 10 : Tax on Web – Exemple de wizard[28]

[28] Voir http://www.taxonweb.be

Ceci est un exemple de calcul simple, il est autrement plus complexe d'établir le montant à déclarer comme amortissement en capital d'emprunts hypothécaires (cadre 8). Il faut calculer le prorata d'amortissement total par rapport au montant total emprunté et un plafond dépendant de la situation particulière du contribuable et de l'année de conclusion du contrat. Le même prorata doit s'appliquer pour les primes d'assurance-vie garantissant ce même emprunt hypothécaire. Tax on Web ne propose d'ailleurs pas de wizard pour remplir ces codes.

Les règles d'exclusions simples sont intégrées dans l'interface utilisateur. En effet, lors de la sauvegarde, le respect de ces règles est contrôlé et validé.

Les règles de condition d'application sont représentées par une aide contextuelle qui montre les explications contenues dans la brochure, comme on peut le voir dans la *Figure 11*.

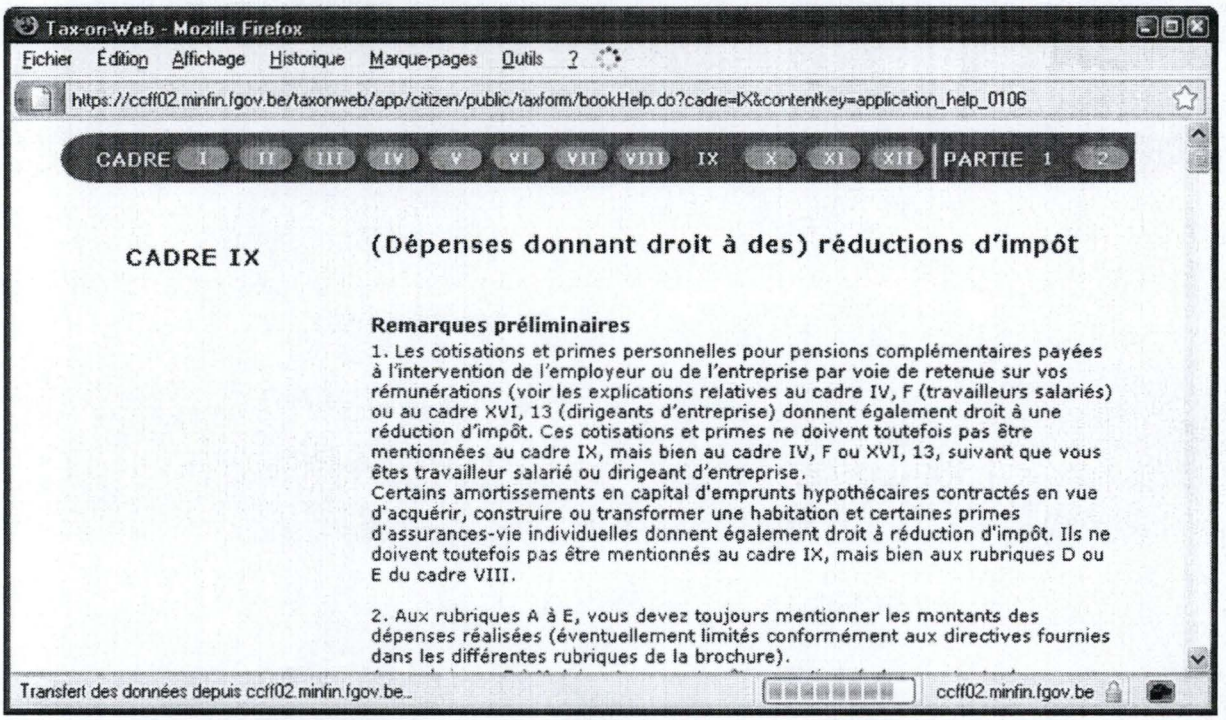


Figure 11 : Tax on Web – Exemple d'aide contextuelle[28]

Mentionnons aussi le petit frère de Tax on Web, Tax-Calc, qui permet une simulation de façon anonyme (c'est-à-dire sans posséder d'identifiants). Tax-Calc est disponible à l'adresse suivante : http://ccff02.minfin.fgov.be/taxcalc/app/anonymous/public/calcbox/home.do. La présentation et les fonctionnalités sont totalement identiques à Tax on Web.

Chapitre 3 : APPLICATION

Résumé du chapitre :
Description de l'application développée.
Description des motivations concernant les choix technologiques opérés.

Nous allons reprendre l'architecture type énoncée au chapitre 1 :

1. Moteur d'inférence
2. Base de connaissance
3. Base de faits
4. Interface utilisateur
5. Module d'explication
6. Module d'acquisition des connaissances

3.1 *Moteur d'Inférence*

Nous avons opté pour un moteur d'inférence Prolog qui applique le principe de résolution de Robinson (du en fait à J. Herbrand dans les années 30).

La recherche d'une solution d'un problème énoncé en Prolog peut être considérée comme la recherche de la démonstration d'un théorème. En terme de recherche de démonstration automatique, on utilise une procédure dite d'*unification*.

Etant donné deux arbres comportant éventuellement des variables, l'unification consiste à trouver, si elles existent, les valeurs que doivent prendre les variables présentent dans l'un ou l'autre des arbres pour que ceux-ci coïncident. L'ensemble de ces couples (variable, valeur) est appelé *substitution*.

Un programme Prolog correspondant à l'exemple ci-dessous pourrait être :

mortel(X) → homme(X);

homme(socrate) →;

Cette dernière clause traduit une affirmation, rien ne démontre que Socrate est un homme car ceci est toujours vrai.

L'utilisateur pourrait alors poser la question: "Mortel(socrate)." à l'interpréteur Prolog (Socrate est-il mortel ?).

L'interpréteur va alors unifier Mortel(socrate) et Mortel(X), par la substitution (X = socrate). Il va ensuite chercher à démontrer Homme(socrate) ce qui est immédiat puisque cette affirmation est présente en tant que règle. L'interpréteur va alors répondre par l'affirmative à la question posée.

Cet exemple illustre le fait que la résolution de problèmes s'effectue en *chaînage arrière*. En partant du but à atteindre, le moteur d'inférence remonte continuellement jusqu'aux sous-buts résolus directement par les assertions initiales. Cette résolution correspond à la réduction de problèmes puisque le but initial est décomposé en sous-buts qui à leur tout sont eux-mêmes décomposés.

L'ensemble des clauses définissant le domaine d'un problème constitue un programme. Chaque programme est composé de une ou plusieurs procédures contenant le même littéral de tête.

Cette tête définit donc les conditions dans lesquelles cette règle peut être utilisée lors d'un appel, ainsi que les substitutions à effectuer.

Exemple :

plat(P) → viande(P); (1)

plat(P) → poisson(P); (1)

viande(steak) →;

viande(poulet_roti) → ;

Cet exemple exprime le fait qu'un plat peut être une viande ou un poisson et qu'une viande peut être un steak ou un poulet rôti. Donc, si l'on cherche des plats, il faudra chercher des viandes et des poissons.

On peut donc dire que les deux clauses (1) définissent les différentes façons de résoudre un problème, chaque clause définissant un de ces manières.

L'activation d'une clause (plat par exemple) consiste alors à choisir une clause et à l'activer (viande par exemple).

La stratégie de contrôle de cette exécution est statique puisqu'elle impose la résolution de la clause le plus à gauche de la queue de règle ainsi que la sélection des clauses dans leur ordre d'écriture (dans l'exemple, on commencerait par examiner la viande, puis le steak, etc.).

Cet aspect statique nuit bien évidemment au non déterminisme du langage.

Par ailleurs, le contrôle du choix des règles est effectué par un mécanisme de retour en arrière (*backtracking*) qui en cas d'échec ou pour trouver d'autres solutions, choisi la clause suivante.

Donc, le principe de recherche dans l'arbre Prolog est la recherche en profondeur d'abord (Depth First Search ou DFS).

Parmi les avantages de Prolog, on peut citer l'uniformité de la représentation des connaissances (faits, règles, structure de contrôle, ...) qui entraîne une simplicité de programmation. Cette uniformité possède un inconvénient, toutes les connaissances sont au même niveau, alors qu'une différenciation peut procurer certains avantages (performance accrue, lisibilité accrue, ...).

C'est ainsi que, du fait de la non détermination de la définition des procédures, la simplicité de lecture d'un programme Prolog n'est qu'apparente (seulement au niveau des prédicats). Par ailleurs, Prolog est relativement efficace et permet l'écriture du contrôleur d'exécution en Prolog même : On peut utiliser des clauses comme arguments pour des prédicats de niveau supérieur, ce qui donne accès aux calculs en logique du deuxième ordre.

Nous pouvons dire que prolog permet d'écrire et de résoudre des problèmes définis dans un formalisme mathématique simple, la difficulté essentielle est de trouver les prédicats convenables à l'énoncé du problème.

Dans les moteurs Prolog disponibles à usage gratuit, nous avons considéré l'utilisation de :

- ✓ SWI Prolog (http://www.swi-prolog.org/) qui offre un environnement Prolog gratuit sous licence GNU. Il propose de nombreuses librairies, un solveur de contrainte sur domaine fini ou réel, des possibilités d'intégration en C et C++ et une intégration avec Java (librairie JPL)
- ✓ Moteur AMZI! (http://www.amzi.com/) qui offre des plug-in pour intégrer avec de nombreux langages tels que le C++, Java, .NET, Delphi, etc. Il possède aussi un composant à intégrer dans l'outil de développement Eclipse. La version 7.6.9. considérée était offerte gratuitement pour usage personnel. Depuis lors, la dernière version est la 8.0 et même la version personnelle est devenue payante. Cependant, il faut noter qu'AMZI! ne propose pas de solveur de contrainte sur domaine fini ou réel.

Nous avons donc opté pour le moteur Prolog offert par SWI-Prolog (version 5.6.64) non seulement pour sa gratuité, mais aussi pour sa complétude (et surtout pour son solveur de contrainte sur domaine fini ou réel) et pour la continuité de développement (la version 5.7. est en cours de développement depuis 2006 et ravivé en 2008).

3.2 *Base de Connaissance*

La représentation de la base de connaissance découle du choix du moteur d'inférence, c'est-à-dire que nous avons opté pour un ensemble de règles construites avec la logique des prédicats.

La connaissance correspond à la modélisation de l'expertise fiscale et le choix des prédicats, la partie la plus sensible de l'application, sera détaillé dans le chapitre 4.

3.3 *Base de faits*

La base de faits correspond à deux types de données :

- ✓ Les données utilisateurs qui sont les données de la déclaration fiscale
- ✓ Les paramètres annuels qui correspondent aux index et aux taux de la taxe communale

Pour représenter ces données, nous avons opté pour une base de données relationnelle en MySQL. Ce choix a aussi été dicté par la gratuité de l'outil, mais aussi par sa complétude et les ressources continuellement mises à jours disponibles.

Le schéma conceptuel de cette base de données correspond au schéma montré en *Figure 12*.

3.3.1 Données utilisateurs

L'administration des contributions nous a facilité la tâche quand à la représentation des données utilisateur.

En effet, la déclaration fiscale est identifiée par un numéro de répertoire unique et un exercice d'imposition (tranDeclarations).

Comme le numéro de répertoire est identique d'année en année, nous avons aussi permis d'attacher plusieurs exercices à un répertoire (descDossiers).

Les différentes données de la déclaration sont identifiées par un code unique : la case de la déclaration (tranCode).

La description des codes (descCode) devait permettre d'associer une description aux codes et faciliter les adaptations annuelles en construisant les pages web d'introduction des données de façon plus dynamique. Mais cette possibilité n'a pas été exploitée dans le cadre de ce mémoire.

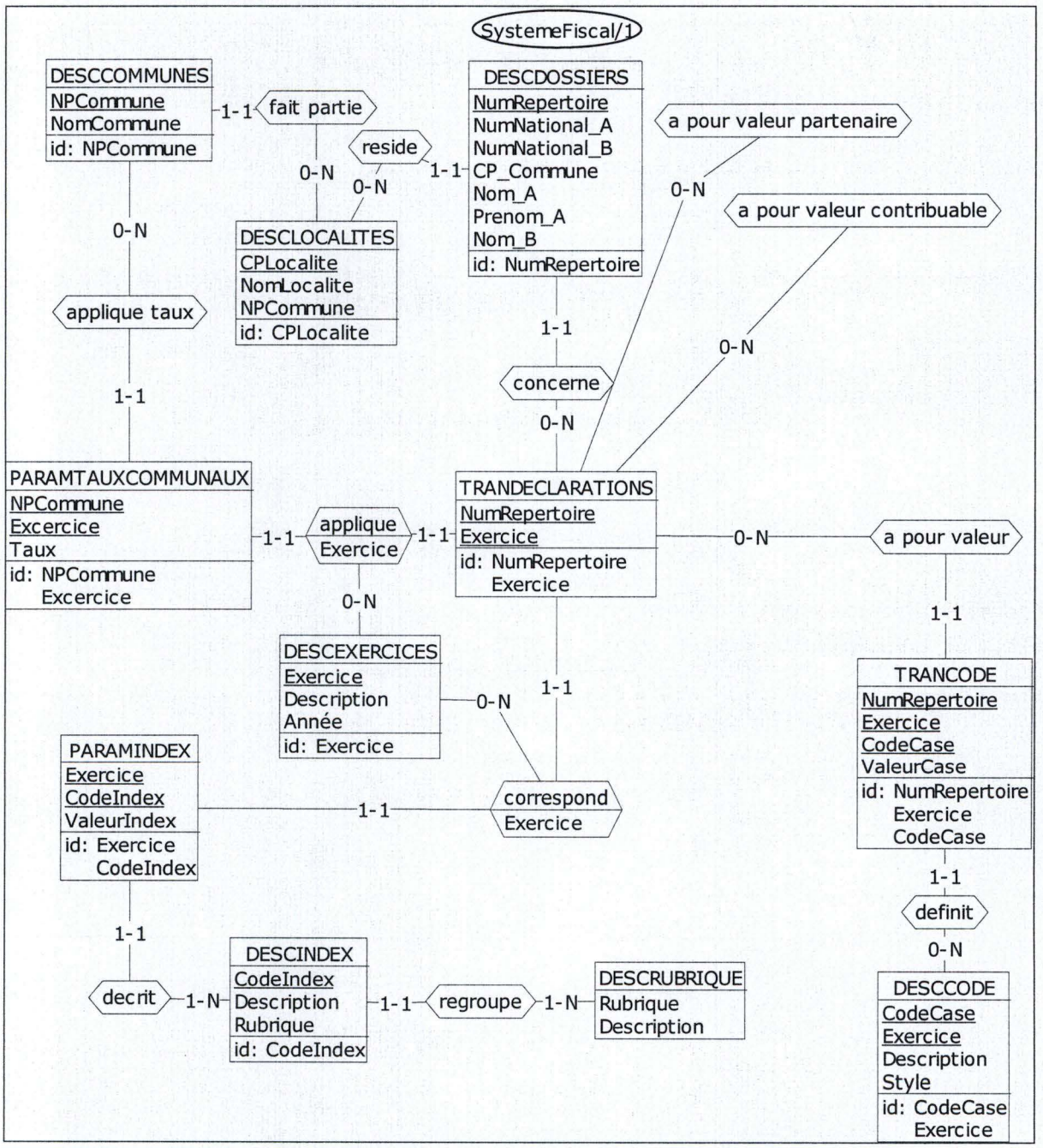


Figure 12 : Schéma conceptuel de la base de données mySQL[29]

[29] Réalisé au moyen de l'outil DBMain disponible à l'adresse suivante : http://www.db-main.com/

3.3.2 Paramètres annuels

Le calcul de l'impôt est basé sur un grand nombre de montant fixés par le législateur.

Par exemple, les minima exonérés d'impôts dont le montant de base est fixé à 4 095,00 EUR [30]. Cependant, ce montant est indexé chaque année et devient 6 040,00 EUR pour les revenus de l'année 2007.

Les sources pour obtenir ces index annuels sont multiples :

- ✓ Via le Fisconet, qui est la base de données fiscale du S.P.F. Finances disponible en ligne à l'adresse suivante : http://ccff02.minfin.fgov.be/KMWeb/
- ✓ Via le site de l'Administration de la fiscalité des entreprises et des revenus, à l'adresse suivante : http://fiscus.fgov.be/interfaoiffr/Vragen/Indexering/indexering_inleiding.htm

Ces montants sont référencés par l'article correspondant dans le code des impôts sur le revenu 92. Nous avons donc opté pour une indexation libre (descIndex) suivant la logique des rubriques telles que présentées dans le document pdf mentionné ci-dessus (descRubrique).

Ceci permet, même si les modalités du calcul de l'impôt et donc la base de connaissance resteraient inchangés, d'adapter le système expert de façon simple, juste en introduisant ces index (paramIndex).

A côté des index annuels, il faut aussi considérer les taux de taxe communale. Ceux-ci ne se modifient pas systématiquement chaque année, mais en pratique, nous pouvons constater que chaque année, l'une ou l'autre commune modifie son taux d'application.

La liste des taux communaux est aussi disponible sur le site de l'Administration de la fiscalité des entreprises et des revenus à l'adresse suivante : http://fiscus.fgov.be/interfaoiffr/ippTC/ippTC.htm (paramTauxCommunaux).

Il reste cependant à faire le lien entre la commune mentionnée sur ce site et le code postal qui lui correspond. Il faut aussi noter qu'une commune regroupe plusieurs codes postaux (depuis la fusion des communes) et donc nous avons décrit les localités (chaque code postal qui existe et pourrait être introduit par l'utilisateur – descLocalités) et nous les avons reliées à leur commune (descCommunes). Les codes postaux sont disponibles... sur le site de la Poste à l'adresse suivante : http://www.post.be/site/fr/residential/customerservice/search/postal_codes.html. Et pour nous faciliter la vie, les codes postaux sont disponibles en format Excel facilement importables dans une base mySQL.

La version de mySQL qui a été utilisée est la 5.0.5 et est disponible sur le site : http://www.mysql.com/.

Signalons encore que les informations contenues dans la base de données doivent être rendues accessible au moteur d'inférence et donc l'application permet de créer les fichiers Prolog "donnéesYYYY.pl" (reprenant les données utilisateurs pour l'exercice YYYY) et "indexYYYY.pl" (reprenant les index annuels correspondant à l'exercice)[31].

---

[30] Article 131, alinea 1er du code des impôts sur le revenu 92.

[31] Voir annexes, au chapitre 8.2, Fichiers Prolog créés par l'interface php

Finalement, le système expert est structuré autour des exercices fiscaux, qui comme décrit au chapitre 2, ne correspondent pas à l'année de recueillement des revenus (desExercices).

3.4 *Interface Utilisateur*

Pour rappel (voir chapitre 1), l'interface utilisateur permet de dialoguer avec le moteur d'inférence.

Nous allons utiliser l'interface de dialogue fournie par SWI-Prolog.

Nous avons envisagé une interface de dialogue plus conviviale, notamment en testant les intégrations de Prolog avec Java. Malheureusement, la librairie JPL fournie avec SWI-Prolog est très basique et plus très à jours (les deux premières versions datent de 1999, la dernière mise à jour de 2003 et depuis 2004 annonce les objectifs de la prochaine version qui n'est pas encore disponible).

Comme mentionné au chapitre 3.1., SWI-Prolog est certainement plus facilement intégrable dans un programme écrit en C ou C++ mais nous avons décidé de nous concentrer sur les objectifs du mémoire, à savoir la modélisation en programmation logique par contraintes plutôt que d'investir dans l'apprentissage d'un nouveau langage de programmation.

3.5 *Module d'Explication*

Ce module est essentiel dans un système expert.

SWI-Prolog propose différentes fonctions qui permettent de tracer l'exécution du programme et d'expliquer comment il arrive aux solutions proposées.

Cependant, cette fonction est illisible pour le commun des mortels et donc nous avons préféré de commenter l'exécution nous-mêmes.

Nous avons essayé d'établir les différentes règles de façon granulaire et donc nous imprimons un commentaire à chaque entrée dans la règle (quand celle-ci est considérée), un commentaire si la règle échoue et un commentaire si règle réussi.

Ces commentaires ont plusieures utilités :

- ✓ Pour le programmeur, mieux qu'une fonction de traçage ou un débogage, il permet de vérifier si le programme réagit conformément aux attentes.
- ✓ Pour l'expert, il permet de vérifier la conformité du raisonnement du système, surtout dans les configurations classique où l'expert n'est pas le programmeur (comme cela devrait être le cas pour tous les systèmes experts).
- ✓ Pour l'utilisateur qui n'est que secondairement intéressé par le résultat final du nouvel impôt calculé pour chaque solution mais surtout désire connaître les raisons justifiant les solutions proposées.

Nous pouvons ainsi suivre le cheminement du système expert et ce dans un langage naturel que nous avons établi.

3.6 *Module d'Acquisition des Connaissances*

Le module d'acquisition des connaissances qui permet d'acquérir les éléments de la base de fait a été réalisé en PHP pour la facilité de connexion avec mySQL et la combinaison aisée Apache – MySQL – PHP.

Ce module se compose donc de deux grandes parties : l'introduction des déclarations (données utilisateur) et l'administration (pour insérer les paramètres annuels).

En ce qui concerne l'introduction des données de la déclaration, la présentation a été largement copiée de Tax on Web (voir chapitre 2.4

Tax on Web) et ce pour des raisons d'ergonomie. Cette présentation est en effet familière car elle est très proche de la déclaration, ou tout du moins de son brouillon.

En ce qui concerne l'introduction des paramètres annuels, seuls ceux qui évoluent chaque année et qui doivent être introduits ont fait l'objet d'une page d'introduction. En Effet, ceux qui sont récupérables électroniquement pourront être introduits par import direct dans la base mySQL (comme les codes postaux, par exemple).

Des exemples des écrans développés se trouvent en annexe (voir 8.1 Interface php).

Les versions utilisées sont :

- ✓ Apache serveur 2.2.11 [32]
- ✓ mySQL 5.0.5 [33]
- ✓ PHP 5.3.0 [34]

En ce qui concerne la base de connaissances en elle-même, celle-ci s'alimente via l'éditeur fourni par SWI-Prolog.

3.7 *Fonctionnement de l'application*

Le fonctionnement de l'application a délibérément été réduit au plus simple :

- ✓ L'interface php permet d'introduire les données utilisateurs (et les paramètres annuels) et génère les deux fichiers Prolog composant la base de fait 'IndexYYYY.pl' et 'DonneesYYYY.pl' (où YYYY est l'exercice fiscal) dans le répertoire c:\SystemeFiscal
- ✓ Ensuite, il faut appeler le programme Prolog 'Prologique.pl' situé dans le même répertoire. SWI-prolog s'ouvre, les fichiers sont chargés et compilés, le système est prêt à recevoir la question.
- ✓ Pour calculer l'impôt et obtenir les conseils, il faut appeler la commande 'opti_IPP'. Le raisonnement apparaît alors dans la fenêtre de dialogue de SWI-Prolog.

---

[32] Voir http://www.apache.org/

[33] Voir http://www.mysql.com/

[34] Voir http://www.php.net/

Chapitre 4 : MODELISATION DES CONTRAINTES

Résumé du chapitre :
Description des règles de la base de connaissance.

Tout d'abord, il convient de déterminer le type de questions que l'on désire soumettre au système expert et dont on espère une réponse satisfaisante.

Le premier objectif du système expert fiscal est, comme nous le souhaitons tous, payer le moins d'impôts possible. Cette question revient donc à solutionner un problème d'optimisation.

Cependant, réduire le système expert à cette seule question serait dommage. En effet, parmi toutes les contraintes servant à établir le solde d'impôts, certaines dépendent d'une décision du contribuable : souscrire une épargne pension ou une assurance vie, acheter des actions de l'employeur, emprunter pour acheter une maison, investir dans cette maison, utiliser les titres-services ou les chèques ALE, etc.

Inutile de développer un système expert pour savoir qu'il suffit de dépenser le maximum dans ces rubriques donnant droit à une réduction d'impôts pour en minimiser le total. Nous souhaitons plutôt avoir la réponse suivante : ces investissements valent-ils la peine ? Comment en estimer la valeur ou le fameux ROI [G-12] ?

4.1 *Problème d'optimisation*

Un problème d'optimisation, aussi noté (C,f), consiste en une contrainte C et une fonction objective f qui est une expression sur les variables de C et qui évalue en un nombre réel [B-1].

Une évaluation, θ, est préférée à une évaluation, θ', si la valeur de la fonction objective f sur θ est moindre que la valeur sur θ'. En d'autres mots, $\theta(f) < \theta'(f)$.

Une solution optimale, θ, de (C,f) est une solution de C telle qu'il n'existe aucune autre solution de C qui est préférée à θ.

Dans le cas présent, la fonction objective f consiste à trouver le minimum de "solde impôt", soit :

$$f = \min(\text{SoldeImpot})$$

La contrainte C est le calcul de ce solde d'impôt basé sur les données communiquées.

La contrainte C est aussi ce que ce solde d'impôt aurait pu être sur les données qui font partie des décisions du contribuable.

Nous avons décidé de séparer ces deux connaissances. Le première est déterministe, basée exclusivement sur les données utilisateurs et le cheminement ne permet pas de retour en arrière (backtracking). La deuxième par contre est destinée à explorer les possibilités offertes au contribuable qui ont une incidence sur le calcul des impôts.

Sans cette séparation, le système, tel que décrit au chapitre 3.1, explore les différentes contraintes dans l'ordre de leur position dans le programme, exige de réclamer de nombreuses solutions, voire toutes les solutions avant d'obtenir cette réponse basique : Combien dois-je payer a priori ?

Ainsi, la seconde partie n'explore que les pistes que nous (mais rappelons que nous nous sommes substitué à l'expert) jugeons digne d'intérêt. Cette position nous a été permise dans la définition même d'un système expert telle que présentée au chapitre 1.1, c'est-à-dire non pas de donner la solution optimale mais bien de modéliser le comportement d'un expert humain.

Les différentes possibilités exposées en seconde partie et le résultat du calcul d'impôt avec ces options peut ainsi plus facilement se comparer au résultat initial obtenu en fin de première partie (que l'on pourra ainsi considérer comme l'impôt de $base_0$).

4.2 *Détermination de l'impôt (calcul)*

Pour déterminer le calcul de l'impôt, nous avons suivi le schéma décrit au chapitre 2.3.1... en partant de la fin.

Le code complet se trouve en annexe 8.3.

Le programme Prolog commence par charger la base de faits, ce qui est fait en chargeant les 'modules' index2008 et donnees2008 (lignes 7 et 8).

Notons au passage que Prologique.pl modélise les impôts de l'année fiscale 2008 (revenus 2007) et qu'il devrait être totalement remanié pour traiter l'année fiscale 2009 (revenus 2008).

4.2.1 Procédure d'entrée : opti_IPP

La procédure d'entrée est *opti_IPP* (ligne 15).

Tout d'abord, on place le contexte, à savoir si la déclaration concerne un contribuable isolé ou un couple. Le calcule est en effet différent, non seulement dans les données à considérer (considérer les cases dont le numéro commence par 2 ou pas) mais il existe des règles spécifiques pour les couples comme l'attribution du quotient conjugal [G-10].

Dans le cadre de ce mémoire, seul le calcul de l'impôt pour un contribuable isolé a été implémenté, le calcul étendu au couple pouvant se dériver de ce modèle.

Cette procédure permet de scinder la résolution du problème en deux parties distinctes :

- ✓ Le calcul du solde impôt de $base_0$, *calcule_SoldeImpot* : le résultat attendu de cette procédure est donc une solution unique.

- ✓ La recherche de la voie la moins imposée, *calcule_SimuleReductionImpot* : le résultat attendu est donc multiple et dépend directement des règles de réduction d'impôt qui seront modélisées.

4.2.2 Procédure calcule_SoldeImpot

La première partie, *calcule_SoldeImpot* commence à la ligne 34 pour le contribuable isolé et peut être détaillée comme suit :

Procédure : calcule_SoldeImpot(SoldeImpot,0)

Type : SoldeImpot = entier

IsCouple = 0 pour le contribuable isolé (sera 1 pour un couple)

Relation :

1. Calculer le RIG (revenu imposable globalement) et le taux marginal : par appel de la procédure *calcule_RIG* afin de déterminer RIG1.
2. Le RIG permet le calcul de l'impôt sans tenir compte du minimum exonéré d'impôt, soit ce qui est renseigné comme l'impôt de base dans le schéma (voir *Figure 7*) : par appel de la procédure *calcule_TotalImpot* afin de déterminer TotalImpot1 et TauxMarginal1.
3. Ensuite, nous déterminons le minimum exonéré d'impôt : par appel de la procédure *calcule_TotalExonere* afin de déterminer TotalExonere.
4. Nous pouvons ainsi calculer la réduction d'impôt découlant de ce minimum exonéré d'impôt : par appel de la procédure *calcule_ExonerationImpot* afin de déterminer ExonerationImpot.
5. Nous calculons la réduction d'impôt découlant des dépenses qui en donne droit : par appel de la procédure *calcule_ReductionImpot* afin de déterminer TotalReduction.
6. L'impôt de base réduit (tel que décrit à la *Figure 7*) est finalement calculé (TotalImpot1 – ExonerationImpot – TotalReduction), éventuellement limité à son minimum, soit 0 (*cherche_Max* entre l'impôt de base réduit et 0).
7. De ce montant, les précomptes remboursables sont déterminés : par appel de la procédure *calcule_SoldeEtat* afin de déterminer SoldeEtat.
8. Ainsi que le montant de la taxe communale : par appel de la procédure *calcule_Commune* afin de déterminer Commune.
9. Le solde final de l'impôt (à payer ou à recevoir) peut finalement être calculé (SoldeEtat + Commune). Ce montant sera celui renvoyé à opti_IPP.

Cette relation traduit le calcul granulaire mais inverse à celui présenté à la *Figure 7*. C'est un appel récursif à une sous-procédure qui réalise un sous-calcul et qui est bien fondé car le niveau minimal sera atteint quand on reprendra les données de la base de faits, c'est-à-dire les affirmations qu'il ne faudra plus démontrer.

Conditions d'application :

In (any, ground) : Out (ground, ground).

Nous attirons l'attention sur les cuts insérés après chaque étape.

Le cut est une fonction Prolog qui permet de figer les choix des clauses opérés et donc la résolution à son point d'insertion. C'est grâce à ce procédé que nous avons pu rendre la résolution de la première partie (calcul du solde impôt de base) déterministe.

4.2.3 Procédure calcule_RIG

La procédure *calcule_RIG* commence à la ligne 289 pour le contribuable isolé et peut être détaillée comme suit :

Procédure : calcule_RIG(RIG,1)

Type : RIG = entier

IsCouple = 1 pour le contribuable isolé (sera 2 pour un couple)

Relation :

1. Calculer le total des revenus immobiliers : par appel de la procédure *calcule_RevenusImmobiliers* afin de déterminer RevenusImmobiliers.
2. Calculer le total des revenus professionnels : par appel de la procédure *calcule_RevenusProfessionnels* afin de déterminer RevenusProfessionnels.
3. Le RIG total peut finalement être calculé (RevenusImmobiliers + RevenusProfessionnels). Ce montant sera celui renvoyé à calcule_SoldeImpot.

Cette relation traduit le calcul tel que présenté à la *Figure 6* : On calcule les revenus imposables par type. Comme nous l'avons mentionné, il existe quatre grands types, seuls deux sont implémentés. Il manque les revenus mobiliers et divers, qui peuvent s'établir selon une modélisation similaire.

Conditions d'application :

In (any, ground) : Out (ground, ground).

4.2.4 Procédure calcule_RevenusImmobiliers

La procédure *calcule_RevenusImmobiliers* commence à la ligne 306 pour le contribuable isolé et peut être détaillée comme suit :

Procédure : calcule_RevenusImmobiliers(RevenusImmobiliers,1)

Type : RevenusImmobiliers = entier

IsCouple = 1 pour le contribuable isolé (sera 2 pour un couple)

Relation :

1. Reprendre les valeurs introduites comme RC pour habitations propres[35] (C1100 et C1101).
2. Tester si un revenu immobilier pour habitations propres doit être calculé.
3. Si le test réussi, calculer le total des revenus immobiliers pour habitations propres : par appel de la procédure *calcule_RCHabitationPropre* afin de déterminer RCHabibationPropre.
4. Reprendre les données nécessaires pour le calcul de l'abattement pour occupation personnelle : I01002 (Déduction pour habitation - montant de base), le nombre d'enfants renseignés dans la case 1104 et I02003 (Déduction pour habitation – majoration). Nous pouvons ainsi déterminer l'abattement maximum auquel nous pouvons prétendre (MaxAbattement).
5. Calculer l'abattement réel à considérer: appelle la procédure *calcule_CalculeAbattementForfaitaireHabitation* afin de déterminer AbattementHabitation.
6. Le total des revenus immobiliers (imposables) est calculé (RCHabitationPropre - AbattementHabitation). Ce montant sera celui renvoyé à calcule_RIG.

Cette relation traduit le calcul des revenus immobiliers imposables tels que défini dans le code fiscal et expliqué dans le cours établi par le S.P.F. Finances [B-13]. Elle n'apparait pas dans le schéma présenté à la *Figure 6* : nous traitons ici du niveau de détail qui n'a pas de raison d'être représenté dans un schéma récapitulatif.

Seuls les revenus pour habitations propres situés en Belgique ont été implémentés, nous pourrons représenter les autres types de revenus immobiliers selon la même logique.

Conditions d'application :

In (any, ground) : Out (ground, ground).

---

[35] Propre dans le sens habitation occupée personnellement par le contribuable (quel que soit sont niveau de propreté).

4.2.5 Procédure calcule_RCHabitationPropre

La procédure *calcule_RCHabitationPropre* commence à la ligne 426 et elle sera identique pour le contribuable isolé ou le couple et peut être détaillée comme suit :

Procédure : calcule_RCHabitationPropre(RCHabitationPropre,Case100,Case101)

Type : RCHabitationPropre = entier

Case100 = entier (correspondra en fait à la case 1100 ou 2100 selon que l'on se trouve dans le calcul pour le partenaire A ou B)

Case101 = entier (correspondra en fait à la case 1101 ou 2101 selon que l'on se trouve dans le calcul pour le partenaire A ou B)

Relation :

1. Reprendre l'index I15004 (Indexation des RC et de la déduction ordinaire pour habitation).
2. Indexer les RC et les arrondir par le bas (floor()).
3. Le total des RCHabitationPropre est calculé et renvoyé à calcule_RevenusImmobiliers.

Cette relation descend encore plus dans le détail et a été isolée dans un but de factorisation pour chaque partenaire.

Conditions d'application :

In (any, ground, ground) : Out (ground, ground, ground).

4.2.6 Procédure calcule_AbattementForfaitaireHabitation

La procédure *calcule_AbattementForfaitaireHabitation* commence à la ligne 442 et peut être détaillée comme suit :

Procédure : calcule_AbattementForfaitaireHabitation(AbattementHabitation, RCHabitationPropre, MaxAbattement)

Type : AbattementHabitation, RCHabitationPropre, MaxAbattement = entiers

Relation :

Cette relation traduit le fait que l'abattement ne peut pas être supérieur aux revenus immobiliers. Nous aurions pu utiliser la fonction cherche_Min (dans les fonctions générales qui sont définies à partir de la ligne 557) car, fonctionnellement, nous aurions obtenu le même résultat. Si nous avons préféré faire une procédure spécifique dans ce cas précis, c'est uniquement afin de pouvoir expliquer le raisonnement plus spécifiquement. Ainsi le message d'explication sera 'Limité au RC total pour habitation propre' en lieu et place de 'On choisi le minimum'.

Ainsi, nous avons illustré la décision du programmeur qui peut soit optimiser le code (en réutilisant la procédure *cherche_Min*), soit enrichir le module d'explication à destination de l'utilisateur.

Conditions d'application :

In (any, ground, ground) : Out (ground, ground, ground).

4.2.7 Autres procédures du calcul d'impôt

La même logique a été appliquée pour les autres procédures destinées à détermine ce premier calcul de l'impôt de $base_0$.

Nous avons aussi implémenté la détermination des **revenus professionnels imposables**, bien que sommairement : le revenu professionnel imposable est le salaire brut tel que déclaré à la case 1250. Nous calculons les frais professionnels forfaitaires et les comparons aux frais réels tels qu'introduits dans la case 1258. Seul le maximum des deux montants sont repris dans le calcul. Nous avons été tenté de permettre au système d'établir deux solutions, selon le montant de frais professionnels admis (frais réels ou forfait légal ?) mais nous avons finalement opté pour un comportement d'expert, à savoir considérer directement l'alternative la plus avantageuse.

Les autres procédures servent au calcul de l'impôt et ont peu de valeur en tant qu'expertise. Nous allons donc juste les énumérer afin de les localiser dans le code, nous vous invitons à vous référer aux lignes correspondantes de l'annexe 8.3 pour prendre connaissance des détails:

- ✓ Procédure *calcule_TotalImpot* (ligne 94 et suivantes)
- ✓ Procédure *calcule_TotalExonere* (ligne 225 et suivantes) qui appelle :
  - Procédure *calcule_Exonere_Contribuable* (ligne 243 et suivantes)
  - Procédure *calcule_Exonere_Famille* (lignes 252 et suivantes) qui appelle :
    - Procédure *calcule_ExonerationEnfants* (lignes 257 et suivantes)
- ✓ Procédure *calcule_ExonerationImpot* (ligne 164 et suivantes)
- ✓ Procédure *calcule_ReductionImpot* (ligne 75 et suivantes) qui appelle :
  - Procédure *calcule_EconomieEnergie* (lignes 531 et suivantes). Nous soulignons ici que nous nous distinguons de Tax On Web par le fait que Tax On Web demande de renseigner 40% de la facture, c'est-à-dire la réduction déjà calculée. A l'instar de notre modèle, nous n'avons pas créé de wizard mais nous calculons nous même la réduction. Le montant à renseigner dans notre système est donc bien le total de la facture.

  Les autres possibilités de réduction d'impôts (cadre 7 à 10) n'ont pas été implémentées mais pourrons l'être selon la même logique.
- ✓ Procédure *calcule_SoldeEtat* (ligne 83 et suivantes)
- ✓ Procédure *calcule_Commune* (ligne 88 et suivantes)

4.3 *Eléments d'optimisation (simulation)*

La deuxième partie du système expert permet de déterminer les différentes options qui s'offrent au contribuable afin de bénéficier d'une réduction d'impôt.

C'est à ce stade qu'il convient de laisser le système offrir toutes les solutions possibles, au contribuable de choisir celle qui lui convient le mieux.

Pour ce faire, *opti_IPP* fait appel à la procédure *calcule_SimulReductionImpot* qui modélise les différentes possibilités offertes par le législateur.

Ensuite, une fois ces réductions d'impôts déterminées, le nouvel impôt est recalculé.

Nous aurions pu aussi calculer la différence entre ce nouvel impôt et l'impôt de $base_0$ afin de surligner le gain de la solution proposée.

4.3.1 Procédure calcule_SimulReductionImpot

La procédure *calcule_SimulReductionImpot* commence à la ligne 75 et peut être détaillée comme suit :

Procédure : calcule_SimulReductionImpot(TotalReduction)

Type : TotalReduction = entier

Relation :

Cette relation doit appeler le calcul du maximum non utilisé de chaque réduction d'impôt possible. Seule la réduction pour économie d'énergie a été implémentée (calcule_SimulEconomieEnergie), les autres réductions présentes dans les cadres 7 à 10 peuvent être ajoutées selon la même logique.

Conditions d'application :

In (any) : Out (ground).

4.3.2 Procédure calcule_SimulEconomieEnergie

La procédure *calcule_SimulEconomieEnergie* commence à la ligne 544 et peut être détaillée comme suit :

Procédure : calcule_SimulEconomieEnergie(SimulEconomieEnergie)

Type : SimulEconomieEnergie = entier

Relation :

Il est important de souligner que cette procédure n'est pas identique à la procédure *calcule_EconomieEnergie* qui est utilisée dans la première partie pour déterminer l'impôt de $base_0$.

La différence se justifie par un raisonnement juste inverse :

Dans *calcule_EconomieEnergie*, nous calculons la réduction à partir du total de la facture renseigné au cadre 1363. Nous vérifions que nous ne dépassons pas le maximum autorisé et donc retenons le minimum entre la réduction acquise (par les factures payées et renseignées au système) et le maximum légal admis.

Dans *calcule_SimulEconomieEnergie*, nous regardons la différence entre le total légal admis et la réduction effective (telle que calculée en première partie). Cette différence est une réduction supplémentaire auquel nous pourrions prétendre. Ensuite, nous reconstituons le montant des investissements qui correspondent à cette réduction complémentaire possible (en inversant le calcul de la réduction).

Le montant total des investissements possible est imprimé afin d'expliquer la solution.

Conditions d'application :

In (any) : Out (ground).

4.4 *Exemple d'utilisation*

Nous avons introduit un exemple simple afin de constater que système réagit conformément à nos attentes, c'est-à-dire :

- ✓ Calcule convenablement l'impôt de $base_0$, une solution unique basée sur les données introduites.
- ✓ Propose les différentes options suivant les possibilités de réduction d'impôt modélisées.

Les résultats sont montrés dans la *Figure 13*.

L'impôt de $base_0$ est correctement calculé en tant que solution unique et le résultat peut être comparé aux simulations données par l'outil Dexia de simulation (voir annexe 8.4).

Les différentes solutions, selon que l'on utilise la réduction d'impôt implémentée ou non, sont aussi déterminées par le système (Solution 1 et Solution 2).

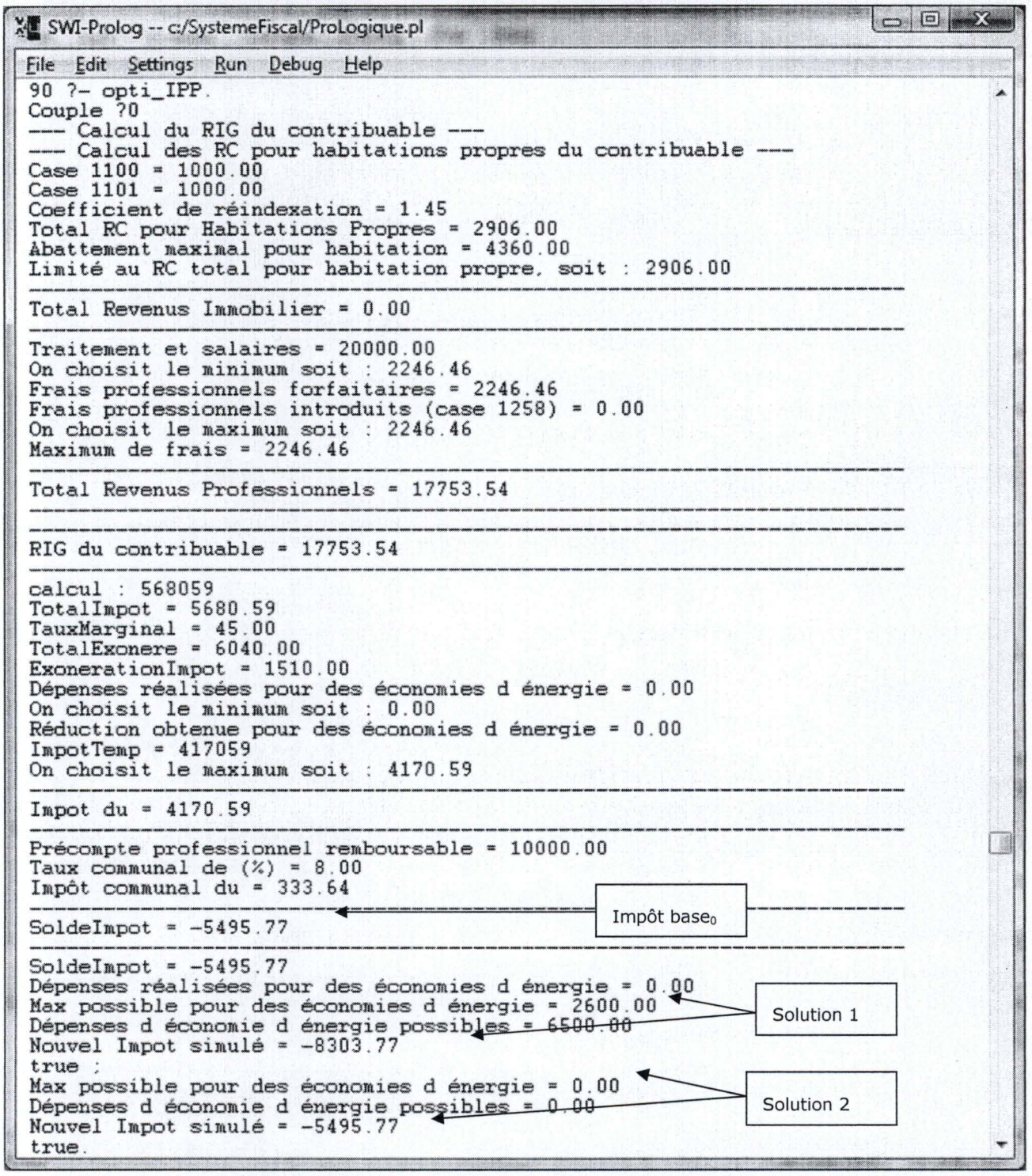


Figure 13 : Exemple d'utilisation du système expert

Chapitre 5 : UNE AUTRE APPROCHE

Résumé du chapitre :
Critique de l'application développée.
Les pistes d'évolution.
Présentation d'une autre approche.

Après avoir obtenu les résultats présentés au chapitre 4, nous avons voulu nous demander si les objectifs initiaux ont bien été atteints.

Nous allons donc analyser ce qui a été développé jusqu'alors et nous demander s'il n'y avait pas une autre approche, plus intéressante pour l'utilisateur lambda et toute aussi conforme aux objectifs.

5.1 *Critique de l'application développée*

Si l'on reprend la description fait d'un système expert au chapitre 1, nous pouvons dire que le domaine d'application est adéquat pour un système expert.

Reprenons les caractéristiques énoncées au chapitre 1.4.1 :

- ✓ Les problèmes ne peuvent pas se résoudre par des méthodes numériques classiques, requièrent de nombreux paramètres ou sont difficilement formalisables.
  - → Lors de l'analyse du domaine, nous avons souligné la complexité du domaine. Les paramètres sont nombreux : plus de 300 cases en ce qui concerne les données de la déclaration, près de 200 index par exercice fiscal, sans compter les taux communaux. Aussi certaines cases de la déclaration sont le résultat d'un calcul préalable, dont certains sont suffisamment complexes pour qu'aucune aide ne soit proposée dans l'application de référence sur le domaine, à savoir Tax on Web.
- ✓ Les problèmes requièrent l'intervention d'un expert humain afin d'être résolus
  - → En effet, chaque année les contribuables se tournent massivement vers les experts pour remplir leur déclaration: soit les experts-comptables et fiscalistes payants ou tout du moins vers les inspecteurs des contributions qui organisent des permanences (cette année même en dehors de leurs bâtiments et de leurs horaires habituels puisqu'ils sont allés à la rencontre des citoyens dans des centres commerciaux lors de la journée d'affluence qui est le samedi).

- ✓ Les connaissances évoluent vite
  - → Pour ce point aussi, on constate de nombreuses différences d'année en année. Les connaissances doivent, au moins une fois par an, être compétées avec les nouvelles mesures. Si l'investissement est rentable pour les experts, il est lourd pour le citoyen qui ne va l'appliquer que pour son unique déclaration.
- ✓ Les capacités de l'expert reposent sur des connaissances, du jugement, de l'expérience et non sur du bon sens
  - → Nous espérons vous avoir convaincu que les connaissances requises pour appréhender le domaine sont très étendues. Pour rappel, le cours officiel du S.P.F. Finances concernant l'IPP fait 657 pages.
- ✓ La résolution du problème nécessite plusieurs heures humaines
  - → Tout dépend évidemment de la situation de chacun, mais pour peu que l'on doive remplir plus que les codes mentionnés dans sa fiche salariale récapitulative ou un emprunt hypothécaire pour l'achat de sa maison, cela peut vite nécessiter plus d'une heure. Si l'on envisage de bénéficier de mesure fiscale, les calculs peuvent vite rebuter le citoyen lambda.
- ✓ Les experts humains sont peu disponibles, onéreux ou, dans le cadre d'une entreprise, la rotation du personnel est élevée
  - → Les experts sont nombreux mais les questions fiscales sont concentrées sur une période courte de maximum un mois (temps moyen entre la possession de tous les documents nécessaire à la tâche et la date limite de remise de la déclaration). Les experts sont souvent surchargés et les listes d'attente pour une consultation sont toujours très longues, surtout que les contribuables ont la propension à attendre l'arrivée de l'échéance pour s'acquitter de ce devoir.

La modélisation du domaine en programmation logique par contraintes ouvre des perspectives intéressantes.

Nous sommes convaincus qu'avec la modélisation en deux partie, la première calculant l'impôt de $base_0$ correspondant aux données introduites et la seconde en permettant au système de déduire toutes les combinaisons exploitant les réductions permises, on atteint l'objectif du mémoire : on canalise la connaissance du calcul d'impôt en forçant une et une seule résolution, on permet d'envisager les différentes possibilités comme si on s'entretenait avec un expert, c'est-à-dire sur un nombre défini de possibilités à envisager.

L'utilisation de Prolog a permis une programmation d'apparence assez simple : le problème a pu être divisé (granulé) successivement en éléments de plus en plus simples, jusqu'à atteindre le niveau des assertions données dans la base de faits.

Il faut cependant reconnaitre la faiblesse énoncée au chapitre 3.1., à savoir que la simplicité du programme n'est qu'apparente. Nous nous interrogeons d'ailleurs sur la capacité d'un tel système à être mis à jour annuellement. Si les index annuels ne poseront aucun problème, grâce au module d'acquisition de données partie 'Administration', les changements dans les règles représentées en logique des prédicats ne seront pas aisés. Comme élément de complication, il faut aussi mentionner le délai relativement long entre les changements (un an, ce qui peut suffire pour en perdre la maîtrise).

5.2 *Les pistes d'évolution*

L'application développée avait pour objectif d'illustrer la capacité théorique d'un tel système expert.

En pratique, il faudrait investir en temps d'intégration et d'ouverture. Les questions au système ne devraient pas nécessiter d'utiliser l'interface de dialogue de SWI-Prolog mais devraient pouvoir se poser par le biais d'une interface utilisateur conviviale et plus accessible.

Les réponses devraient pouvoir bénéficier d'une meilleure représentation, voire pouvoir se présenter sous forme d'un graphique illustrant les bénéfices attendus de chaque solution proposée, à l'instar de la *Figure 8*.

Mais surtout, la partie implémentée peut se comparer à la partie émergente d'un iceberg. Il faut aussi continuer à implémenter les règles relatives aux deux catégories de revenus ignorées (revenus divers et mobiliers) ainsi que compléter celles des catégories traitées (revenus immobiliers et professionnels). Il faut aussi implémenter le calcul pour un couple, et pas seulement pour un contribuable isolé.

Last but not least, comme on dit, il faudrait implémenter toutes les réductions d'impôt possibles avant de l'appeler réellement 'système expert'.

L'objectif était de démontrer comment on pourrait articuler ce genre d'application à un domaine fiscal et donc nous pouvons considérer ceci comme un prototype que nous qualifierons d'apéritif : suffisant pour attiser l'appétit et laisser présager un bon repas mais insuffisant pour nous rassasier.

5.3 *Une autre approche*

Quand nous sommes arrivés à ce stade du travail, nous nous sommes demandé si vraiment le travail réalisé était la bonne approche.

Nous ne dirons pas que l'approche était mauvaise : nous pensons que les premières conclusions sont encourageante mais nous pensons néanmoins qu'une autre option peut s'envisager.

Nous avons pris comme point de départ la déclaration fiscale à remplir et donc comme modèle, sa version électronique appelée Tax on Web.

Ceci a deux inconvénients majeurs :

- ✓ Nous avons vu que les données à renseigner nécessitent souvent des calculs préalables. Si notre système prend quelques libertés facilitatrices pour certaines cases (comme par exemple l'économie pour énergie modélisée où notre système admet le total des dépenses et n'exige pas de calculer au préalable la réduction), nous n'avons pas abordé les réductions d'impôts complexes telle que les calculs des déductions des emprunts hypothécaires...
- ✓ Nous n'avons pas modélisé les règles de condition d'application (voir chapitre 2.3.2.2.). La justification énoncée alors nous avait convaincu, mais on peut se demander pourquoi créer un système expert et toujours devoir lire ces conditions dans les brochures explicatives ou de devoir se les faire confirmer par un expert.

Selon notre opinion, nous avons pêché : nous avons trop analysé le point de vue de l'expert. L'expert qui est familier avec cette déclaration fiscale.

Nous aurions peut-être du nous positionner plus au point de vue du citoyen lambda qui pourrait substituer notre système à une consultation d'expert.

L'idée serait donc de ne pas partir de la déclaration fiscale mais des documents en notre possession, nous, pauvres contribuables.

Prenons un exemple, pour les traitements et salaires, l'interface d'introduction des données devrait plutôt présenter l'aspect des fiches récapitulatives que nous recevons, à savoir :

- ✓ La fiche 281.10 pour les rémunérations ordinaires, pécules de vacances et indemintés
- ✓ La fiche 281.11 pour les pensions et rentes
- ✓ La fiche 281.13 pour les allocations de chômage et les prépensions
- ✓ La fiche 281.14 pour les revenus de remplacement
- ✓ La fiche 281.16 pour les indemnités légales en cas d'incapacité permanente.

Prenons la fiche 281.10 qui est celle que tous les salariés reçoivent de leur employeur, un exemple se trouve en annexe 8.5.1.

Avec ce premier exemple, la valeur ajoutée de cette approche peut ne pas apparaître au premier abord.

En effet, cette fiche est déjà prévue pour faciliter le travail du contribuable, puisque les codes de la déclaration y sont déjà renseignés.

Par à rapport à la déclaration traitée (exercice 2008 – revenus de l'année 2007), la grande différence est l'ordre d'introduction. On peut noter qu'on pourrait ainsi modéliser les règles concernant les frais de déplacement et déterminer le montant de l'exonération à renseigner (case 255).

Notons aussi que cette approche commence à devenir évidente depuis l'exercice 2009 (revenus de l'année 2008).

En effet, depuis cette année, la déclaration officielle ne demande que de renseigner les codes et leur montant et qu'elle ne porte aucune autre indication. La déclaration 'traditionnelle' n'existe plus qu'à l'état de brouillon et, gageons, uniquement pour un temps délimité d'acclimatation.

Pour les revenus immobiliers, on pourrait structurer l'entrée des données sur base de l'invitation à payer le précompte immobilier et sur base de l'attestation des intérêts payés pour un emprunt hypothécaire. Le système se chargera alors de répartir les montants à travers les différentes cases (exemple : un emprunt hypothécaire conclu entre 1989 et 2005 peut conduire à remplir des montants dans les cases 146, 355, 351 et 353 du cadre 8).

Une telle approche exigerait une interaction accrue entre la base de connaissance et le module d'acquisition des données utilisateurs.

Si une telle interaction est possible, alors nous pourrions reléguer le calcul de l'impôt pur à un programme procédural plus classique et consacrer la base de connaissances aux règles fiscales, qui, somme toute, ne serait pas moins imposante puisque les règles de condition d'application seraient ajoutées à cette base de connaissance.

Pour remplir ces objectifs, il faudrait reconsidérer les choix opérés et se tourner vers une combinaison Java couplé avec le moteur Prolog AMZI! ou C++ et le moteur Prolog SWI_Prolog (voir chapitre 3.1 pour les références de ces outils).

Chapitre 6 : CONCLUSIONS

" Chaque année tout contribuable se doit de remplir sa déclaration fiscale. Le montant à rembourser par le fisc ou à verser au fisc est déterminé par une ensemble de règles précises publiées dans diverses revues (telles que l'Almanach du contribuable). Toutefois, l'application manuelle de ces règles est, en général, hors de portée du commun des mortels. En outre, dans certains cas de figure, il est possible de réduire l'impôt en déclarant d'une manière adéquate certains montants.

L'objectif du mémoire est de construire un système expert basé sur un codage en programmation logique par contraintes de règles fiscales. Un tel système contrastera avec les logiciels de type "feuille de calcul Excel", répandus par les organismes bancaires, en permettant des calculs multi-directionnels et en expliquant son raisonnement."

Tels sont les objectifs de ce mémoire lors de la remise du sujet au BEX, tels sont les termes de son résumé.

Nous avons commencé par étudier les caractéristiques d'un système expert afin de comprendre les différences avec autres types d'application.

Ensuite, nous avons donné les fondements du domaine d'application afin de déterminer les connaissances qui seront modélisées et donner un schéma directeur pour cette modélisation.

Nous avons continué par présenter l'application développée, sa structure, les outils utilisés et justifier leur choix.

Nous avons regardé l'agencement et la structure de la partie concernant la programmation logique par contraintes. Nous avons obtenu un compromis entre la détermination de l'impôt et les propositions déduites en vue de réduire l'impôt à payer. Les informations calculées peuvent permettre au contribuable de mieux appréhender les possibilités qui restent offertes et donc de choisir au mieux les options.

Arrivé à ce stade, nous nous sommes retourné sur objectifs atteints par cette approche et nous nous sommes interrogés sur une autre approche : nous avons envisagé le contenu d'un système expert qui ne suivrait plus le point de vue de l'expert mais le point de vue du contribuable. Au lieu de prendre la déclaration comme point de départ du module d'acquisition de données, nous pourrions envisager les documents courants collectés par le contribuable, sans tenir compte de la structure de la déclaration. Le système expert se chargerait de donner en plus des explications sur les conditions d'application des différentes règles fiscales. Le système se chargerait de la répartition sur les différents codes, voire les différents cadres.

Finalement, comme nous l'avons répété à plusieurs reprises, nous avons surtout créé un prototype qui permet d'appréhender la portée théorique et les possibilités d'un tel système expert.

En pratique, si nous voulons réellement développer un tel système expert, nous devons absolument privilégier l'ouverture et l'intégration de diverses technologies. Certaines tâches du système seraient plus facilement maintenable avec de la programmation procédurale alors que la programmation logique par contraintes peut certainement modéliser les règles fiscales, pas toujours très structurées et ainsi trouver la voie la moins taxée dans ce qui ressemble bien à un dédale.

Si on ajoute un traitement plus élaboré des solutions trouvées, nous pourrions les présenter de manière plus compréhensible, éventuellement accompagnée de représentations graphiques, à l'instar de l'illustration de la *Figure 8*.

Ce travail n'est certainement pas un achèvement en soi, mais une porte ouverte sur un type d'application un peu passé de mode : le système expert.

Après un engouement dans les années 70 et 80, le sujet du 'système expert' tend à disparaitre des articles scientifiques publiés [B-17] au profit de techniques résolument plus ancrées encore dans la branche des techniques d'intelligence artificielle : les stratégies d'évolution et la formation de Knowledge-based Systems.

Nous revenons donc à une caractéristique essentielle du système expert : l'explication du raisonnement.

Dans ce domaine, et avec la programmation logique par contraintes avec Prolog comme langage, nous pouvons jalonner le processus de résolution de commentaires ad hoc, en langage naturel, utiles pour toutes les parties prenantes du système : du programmeur à l'utilisateur en passant par l'expert.

Chapitre 7 : BIBLIOGRAPHIE

[B-1] Kim Marriott and Peter J. Stuckey, *Programming with Constraints – An Introduction*, The MIT Press Cambridge Massachusetts, United States of America, 1999.

[B-2] Clyde W. Holsapple et Andrew B. Whinston, *guru® - l'utilisation des systèmes experts dans l'entreprise*, Les éditions d'organisation, Paris, France, 1987.

[B-3] Jean-Noël Chatain et Alain Dussauchoy, *Systèmes experts – méthodes et outils*, Eyrolles, Paris, France, 1987.

[B-4] William F. Clocksin and Christopher S. Mellish, *Programming in Prolog*, Springer-Verlag Berlin Heidelberg New York, Germany, 1981.

[B-5] Leon Sterling and Ehud Shapiro, *The Art of Prolog - Advanced Programming Techniques*, The MIT Press Cambridge Massachusetts, United States of America, 1986.

[B-6] Jan Wielemaker, *SWI-Prolog 5.6 – Reference Manual (updated for version 5.6.35, June 2007)*, http://www.swi-prolog.org , University of Amsterdam, The Netherlands, 2007.

[B-7] David Frédrich, *Conception et développement d'un système basé sur des connaissances juridiques extensibles*, Mémoire/Facultés universitaires N.D. de la Paix Namur. Informatique, 2004.

[B-8] François Gruselin et Julien Vilz, *Vérification et validation d'un système expert pour la mesure fonctionnelle (Cosmicxpert)*, Mémoire/Facultés universitaires N.D. de la Paix Namur. Informatique, 2003.

[B-9] G.F. Luger, *Artificial Intelligence: Structures and Strategies for Complex Problem Solving (Fourth Edition)*, Addison-Wesley, England, 2002.

[B-10] Julien Lefèvre, *Création de librairies Prolog pour le Web*, Mémoire/Facultés universitaires N.D. de la Paix Namur. Informatique, 2004.

[B-11] Stéphane Dereppe, *Système expert fiscal*, Mémoire/Facultés universitaires N.D. de la Paix Namur. Informatique, 2007.

[B-12] Dexia University, *e-Learning : Impôt des personnes physiques*, version 16.00, copyright Dexia N.V./S.A., 04/2008.

[B-13] Ministère des Finances, *Impôt des personnes physiques exercice d'imposition 2005*, http://www.fisconet.fgov.be/fr/?frame.dll&root=v:/sites/FisconetFraAdo.2/&versie=04&type=VRB08!vakcpb05.htm&, Fisconet, 2006.

[B-14] Steven M. Bragg, *Business Ratios and Formulas : A Comprehensive Guide* (second edition), John Wiley & Sons Inc., Hoboken, New Jersey, 2007.

[B-15] Editions Fever sarl, Glossaire financier de Vernimmen.net, http://www.vernimmen.net/html/glossaire/index.html, Paris, consulté le 26-2-2009.

[B-16] Roland Rosoux, *La Pratique de l'Impôt des Personnes Physiques*, syllabus du cours de Maîtrise en Pratique de l'Expertise Comptable et du Revisorat organisés par l'Ordre des Experts Comptables et Comptables Brevetés de

Belgique en collaboration avec l'Institut d'Administration des Entreprises de Belgique sous le haut patronage de la Fondation Joseph Colleye, Bruxelles, 1995.

[B-17] Roy Rada, *Expert systems and evolutionary computing for financial investing : A review*, available online at www.sciencedirect.com , doi: 10.1016/j.eswa.2007.05.012, 2007

[B-18] Service Public Fédéral Finances, *Notice explicative annexée à la déclaration à l'impôt des personnes physiques, Exercice d'imposition 2009 (revenus de l'année 2008)*, Belgique, 2009.

[B-19] Alain Bonnet, *L'intelligence artificielle. Promesses et Réalités*, InterEditions, Paris, 1984.

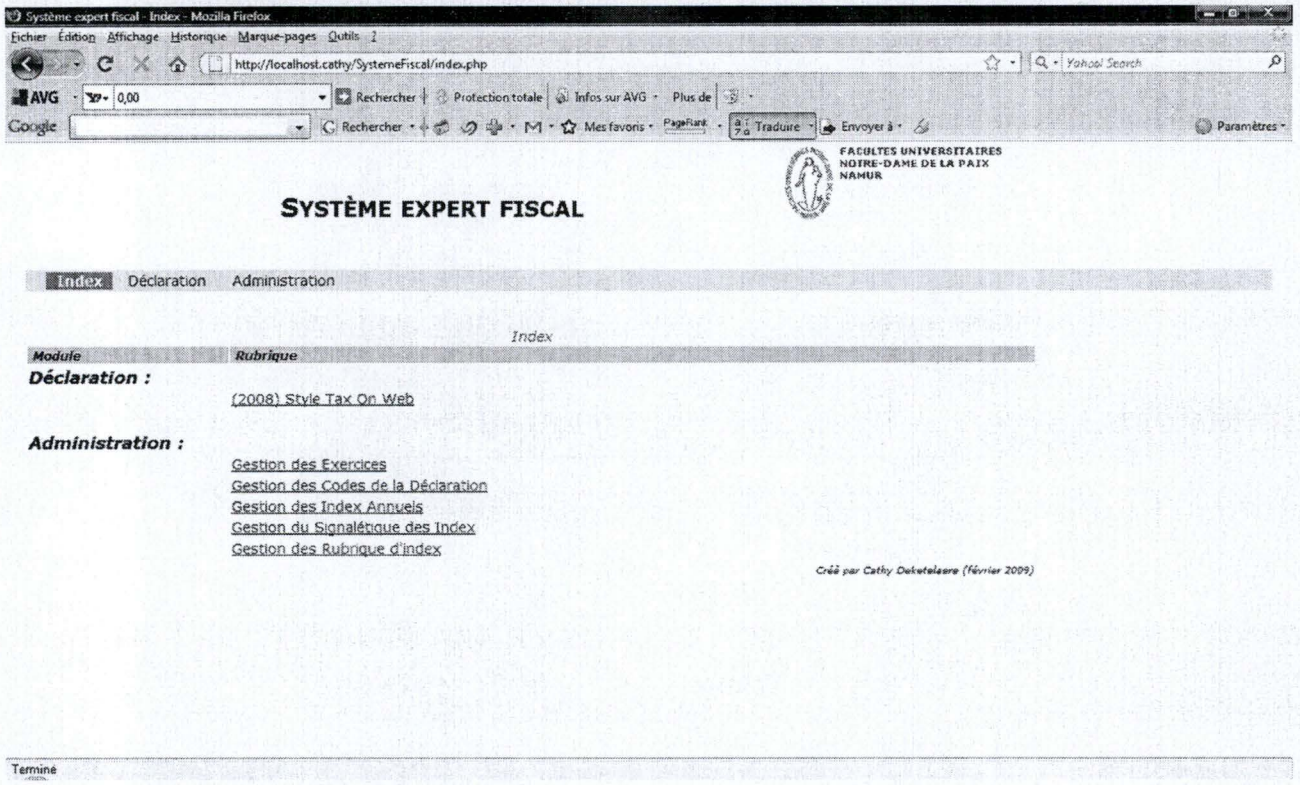
[B-20] M.R. Quillian, *Semantic Memory*, in Semantic Information Processing, Ed. M.Minsky/ MIT Press, Cambridge, 1968.

[B-21] Marvin Minsky, *A Framework for Representing Knowledge*, MIT-AI Laboratory Memo 306, June, 1974. (aussi disponible en ligne : ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-306.pdf)

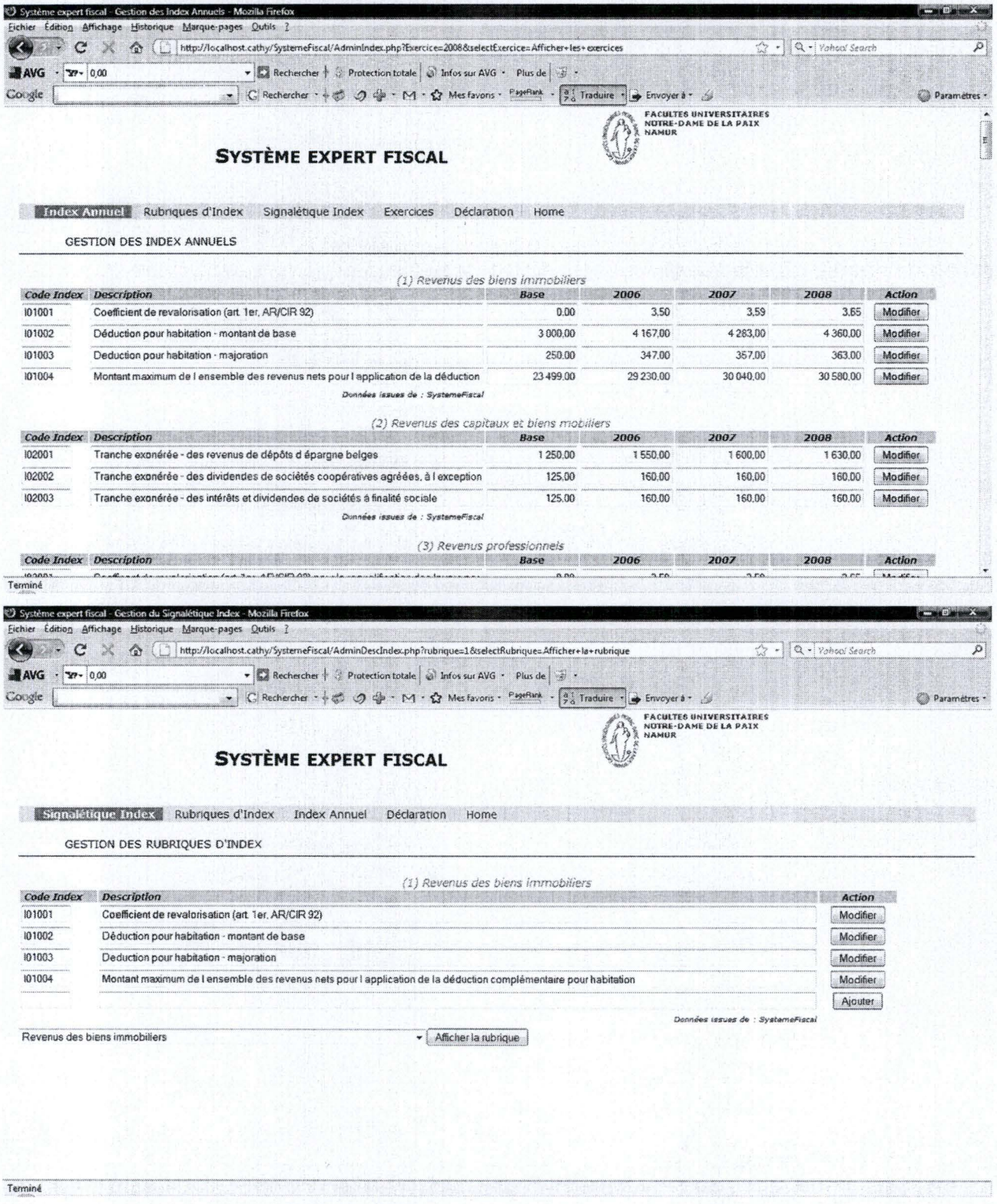
Chapitre 8 : ANNEXES

8.1 *Interface php*

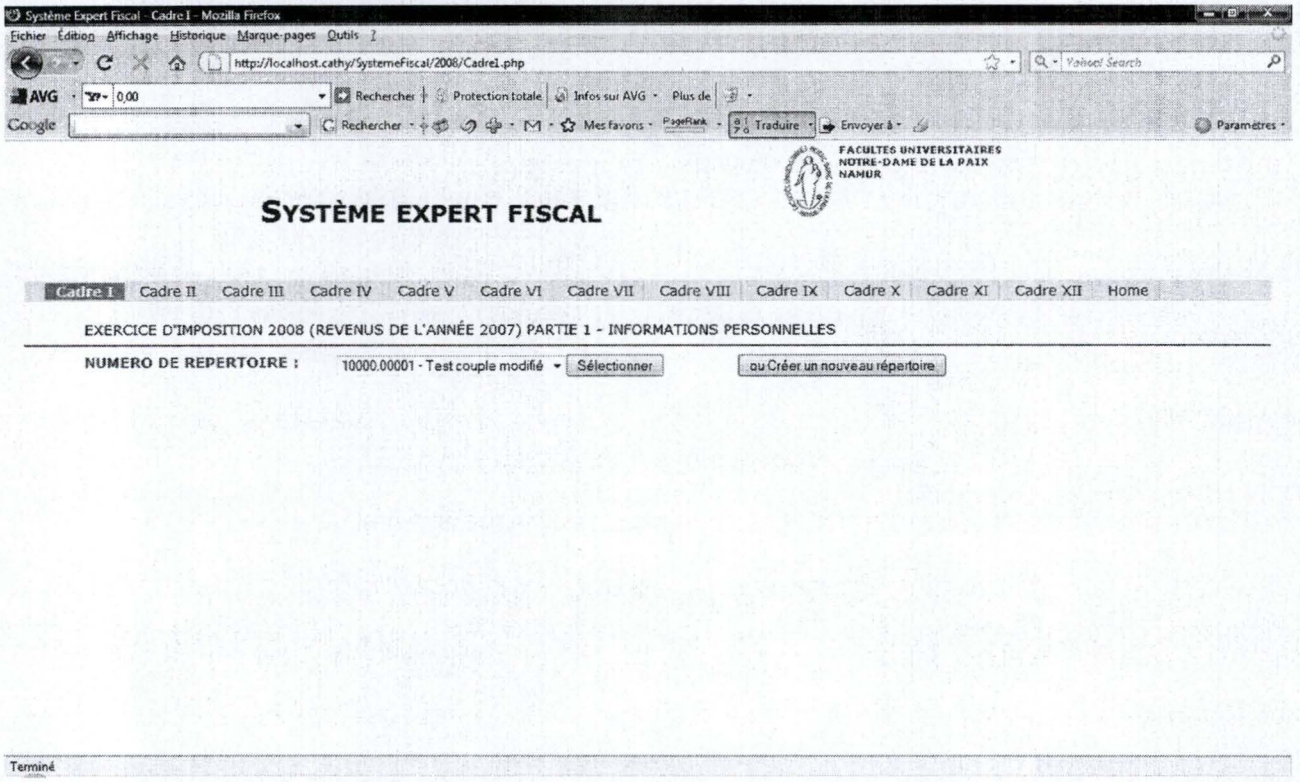
8.1.1 Index général

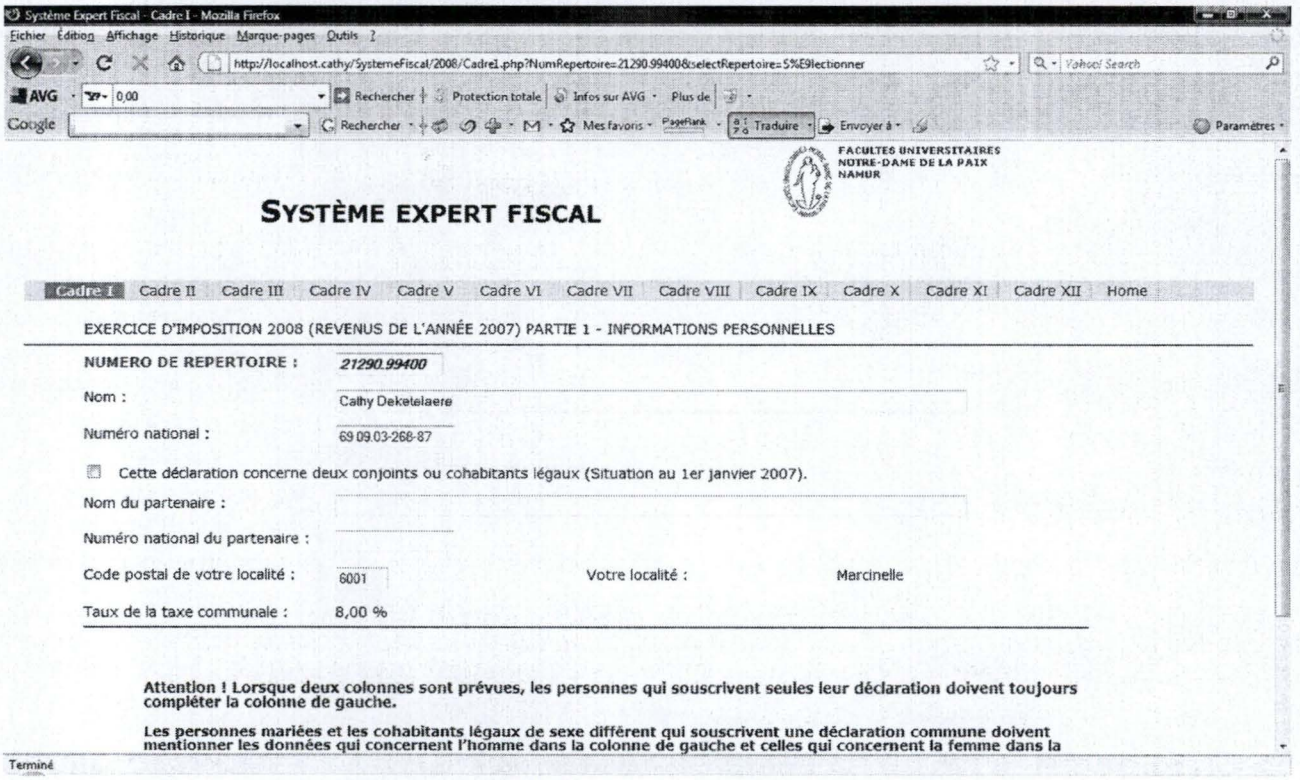


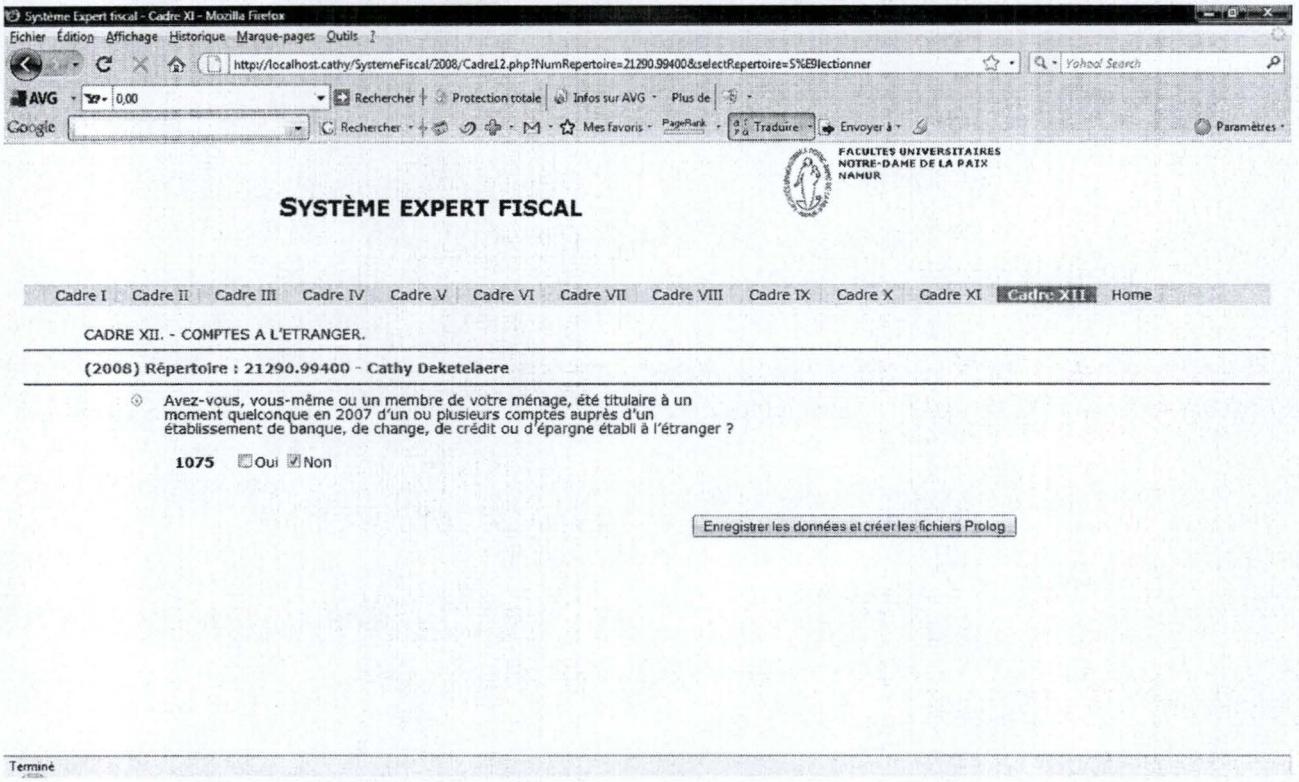
8.1.2 Module administration



8.1.3 Module déclaration







8.2 *Fichiers Prolog créés par l'interface php*

8.2.1 Index2008

```
1 :- module(index2008, [getIndex/2]).
2
3 :- use_module(library(bounds)).
4
5 % ________Reprise des Index 2008______
6
7 getIndex(Index,NumIndex):-
8 NumIndex = i01001,
9 Index #= 365.
10 getIndex(Index,NumIndex):-
11 NumIndex = i01002,
12 Index #= 436000.
13 getIndex(Index,NumIndex):-
14 NumIndex = i01003,
15 Index #= 36300.
16 getIndex(Index,NumIndex):-
17 NumIndex = i01004,
18 Index #= 3058000.
19 getIndex(Index,NumIndex):-
20 NumIndex = i02001,
21 Index #= 163000.
22 getIndex(Index,NumIndex):-
23 NumIndex = i02002,
24 Index #= 16000.
```

Etc.

8.2.2 Données de la déclaration

```
1 :- module(case, [getCase/2]).
2
3 :- use_module(library(bounds)).
4 % % ________Reprise des Codes déclaration (10000000000) - (2008) __________
5
6 getCase(Case,NumCase):-
7 NumCase = isCouple,
8 Case #= 0.
9 getCase(Case,NumCase):-
10 NumCase = tauxCommunal,
11 Case #= 800.
12 getCase(Case,NumCase):-
13 NumCase = c1001,
14 Case #= 100.
15 getCase(Case,NumCase):-
16 NumCase = c1002,
17 Case #= 0.
18 getCase(Case,NumCase):-
19 NumCase = c1003,
20 Case #= 0.
21 getCase(Case,NumCase):-
22 NumCase = c1004,
23 Case #= 0.
24 getCase(Case,NumCase):-
25 NumCase = c1005,
26 Case #= 0.
27 getCase(Case,NumCase):-
28 NumCase = c1006,
29 Case #= 0.
30 getCase(Case,NumCase):-
31 NumCase = c1007,
32 Case #= 0.
33 getCase(Case,NumCase):-
34 NumCase = c1008,
35 Case #= 0.
36 getCase(Case,NumCase):-
37 NumCase = c1009,
38 Case #= 0.
39 getCase(Case,NumCase):-
40 NumCase = c1010,
41 Case #= 0.
42 getCase(Case,NumCase):-
43 NumCase = c1011,
44 Case #= 0.
45 getCase(Case,NumCase):-
46 NumCase = c1012,
47 Case #= 0.
48 getCase(Case,NumCase):-
49 NumCase = c1013,
50 Case #= 0.
51 getCase(Case,NumCase):-
52 NumCase = c1014,
53 Case #= 0.
54 getCase(Case,NumCase):-
55 NumCase = c1015,
56 Case #= 0.
```

Etc.

8.3 *La base de connaissances : Prologique.pl*

C:\SystemeFiscal\ProLogique.pl

```
1 %_____
2 % Déclaration des librairies nécessaires au système.
3 %_____
4
5 :- use_module(library(clpr)).
6 :- use_module(library(bounds)).
7 :- use_module(index2008).
8 :- use_module(donnees2008).
9
10
11 %_____
12 % Minimisation de l'impôt.
13 %_____
14
15 opti_IPP :-
16 getCase(IsCouple,isCouple),
17 print('Couple ?'), print(IsCouple),nl,!,
18 % trace,
19 calcule_SoldeImpot(SoldeImpot,IsCouple),
20 % notrace,
21 % minimize(SoldeImpot),
22 print('SoldeImpot = '), format('~2f~n', [SoldeImpot/100]),
23 getCase(TauxCommunal,tauxCommunal),!,
24 calcule_SimulReductionImpot(SimulTotalReduction),
25 RedCommune #= SimulTotalReduction * TauxCommunal / 10000,
26 ImpotSimule #= SoldeImpot - SimulTotalReduction - RedCommune,
27 print('Nouvel Impot simulé = '), format('~2f~n', [ImpotSimule/100]).
28
29
30 %_____
31 % Calcul final de l'impôt.
32 %_____
33
34 % _____Contribuable seul_____
35
36 calcule_SoldeImpot(SoldeImpot,0) :-
37 calcule_RIG(RIG1,1), !,
38 print('-----------------------------------------------------------------------'),nl,
39 print('RIG du contribuable = '), format('~2f~n', [RIG1/100]),
40 print('-----------------------------------------------------------------------'),nl,
41 calcule_TotalImpot(TotalImpot1,RIG1,TauxMarginal1),!,
42 print('TotalImpot = '), format('~2f~n', [TotalImpot1/100]),
43 print('TauxMarginal = '), format('~2f~n', [TauxMarginal1/100]),
44 calcule_TotalExonere(TotalExonere,0),!,
45 print('TotalExonere = '), format('~2f~n', [TotalExonere/100]),
46 calcule_ExonerationImpot(ExonerationImpot,TotalExonere),
47 print('ExonerationImpot = '), format('~2f~n', [ExonerationImpot/100]),
48 calcule_ReductionImpot(TotalReduction),!,
49 ImpotTemp #= (TotalImpot1 - ExonerationImpot - TotalReduction),
50 print('ImpotTemp = '), print(ImpotTemp), nl,
51 cherche_Max(MaxImpot,ImpotTemp,0),
```

```
52 ImpotDu #= MaxImpot,!,
53 print('------------------------------------------------------------------------'),nl,
54 print('Impot du = '), format('~2f~n', [ImpotDu/100]),
55 print('------------------------------------------------------------------------'),nl,
56 calcule_SoldeEtat(SoldeEtat,ImpotDu),!,
57 calcule_Commune(Commune,ImpotDu),!,
58 SoldeImpot #= SoldeEtat + Commune ,!,
59 print('------------------------------------------------------------------------'),nl,
60 print('SoldeImpot = '), format('~2f~n', [SoldeImpot/100]),
61 print('------------------------------------------------------------------------'),nl.
62
63 % _______Couple_____________________________
64
65 calcule_SoldeImpot(SoldeImpot,1) :-
66 SoldeImpot #= 0,
67 print('------------------------------------------------------------------------'),nl,
68 print('Les couples ne sont pas encore modélisés'),
69 print('------------------------------------------------------------------------'),nl,!.
70
71 % _______Calcul des simulations de réductions d'impot_____________
72
73 calcule_SimulReductionImpot(SimulTotalReduction) :-
74 calcule_SimulEconomieEnergie(SimulEconomieEnergie),
75 SimulTotalReduction #= SimulEconomieEnergie.
76
77 % _______Calcul de réductions d'impot_____________________________
78
79 calcule_ReductionImpot(TotalReduction) :-
80 calcule_EconomieEnergie(EconomieEnergie),
81 TotalReduction #= EconomieEnergie.
82
83 % _______Calcul impot max_________________________________
84
85 calcule_SoldeEtat(SoldeEtat,ImpotDu) :-
86 getCase(C1286,c1286),
87 print('Précompte professionnel remboursable = '), format('~2f~n', [C1286/100]),
88 SoldeEtat #= (ImpotDu - C1286).
89
90 calcule_Commune(Commune,ImpotDu) :-
91 getCase(TauxCommunal,tauxCommunal),
92 print('Taux communal de (%) = '), format('~2f~n', [TauxCommunal/100]),
93 Commune #= ImpotDu * TauxCommunal / 10000,
94 print('Impôt communal du = '), format('~2f~n', [Commune/100]).
95
96 % _______Calcul impot par palier (et taux marginal)_____________________________
97
98 calcule_TotalImpot(TotalImpot,RIG,TauxMarginal) :-
99 getIndex(I14001,i14001),
100 getIndex(I14002,i14002),
101 RIG @=< I14001,!,
102 TotalImpot #= (RIG * I14002 / 10000),
103 print('calcul : '),print(TotalImpot), nl,
104 TauxMarginal #= I14002.
105
106 calcule_TotalImpot(TotalImpot,RIG,TauxMarginal) :-
107 getIndex(I14001,i14001),
108 getIndex(I14002,i14002),
```

```
109 getIndex(I14003,i14003),
110 getIndex(I14004,i14004),
111 RIG @> I14001, RIG @=< I14003,!,
112 TotalImpot #= (I14001 * I14002 / 10000) + ((RIG-I14001) * I14004 / 10000),
113 print('calcul : '),print(TotalImpot), nl,
114 TauxMarginal #= I14004.
115
116 calcule_TotalImpot(TotalImpot,RIG,TauxMarginal) :-
117 getIndex(I14001,i14001),
118 getIndex(I14002,i14002),
119 getIndex(I14003,i14003),
120 getIndex(I14004,i14004),
121 getIndex(I14005,i14005),
122 getIndex(I14006,i14006),
123 RIG @> I14003, RIG @=< I14005,
124 TotalImpot #= (I14001 * I14002 / 10000) +
125 ((I14003-I14001) * I14004 / 10000) +
126 ((RIG-I14003) * I14006 / 10000),
127 print('calcul : '),print(TotalImpot), nl,
128 TauxMarginal #= I14006.
129
130 calcule_TotalImpot(TotalImpot,RIG,TauxMarginal) :-
131 getIndex(I14001,i14001),
132 getIndex(I14002,i14002),
133 getIndex(I14003,i14003),
134 getIndex(I14004,i14004),
135 getIndex(I14005,i14005),
136 getIndex(I14006,i14006),
137 getIndex(I14007,i14007),
138 getIndex(I14008,i14008),
139 RIG @> I14005, RIG @=< I14007,!,
140 TotalImpot #= (I14001 * I14002 / 10000) +
141 ((I14003-I14001) * I14004 / 10000) +
142 ((I14005-I14003) * I14006 / 10000) +
143 ((RIG-I14005) * I14008 / 10000),
144 print('calcul : '),print(TotalImpot), nl,
145 TauxMarginal #= I14008.
146
147 calcule_TotalImpot(TotalImpot,RIG,TauxMarginal) :-
148 getIndex(I14001,i14001),
149 getIndex(I14002,i14002),
150 getIndex(I14003,i14003),
151 getIndex(I14004,i14004),
152 getIndex(I14005,i14005),
153 getIndex(I14006,i14006),
154 getIndex(I14007,i14007),
155 getIndex(I14008,i14008),
156 getIndex(I14010,i14010),
157 RIG @> I14007,!,
158 TotalImpot #= (I14001 * I14002 / 10000) +
159 ((I14003-I14001) * I14004 / 10000) +
160 ((I14005-I14003) * I14006 / 10000) +
161 ((I14007-I14005) * I14008 / 10000) +
162 ((RIG-I14007) * I14010 / 10000),
163 print('calcul : '),print(TotalImpot), nl,
164 TauxMarginal #= I14010.
165
```

```
166 %_____Calcul Exonération impot par palier_____
167
168 calcule_ExonerationImpot(ExonerationImpot,TotalExonere) :-
169 getIndex(I14001,i14001),
170 getIndex(I14002,i14002),
171 TotalExonere @=< I14001,!,
172 ExonerationImpot #= (TotalExonere * I14002 / 10000),!.
173
174 calcule_ExonerationImpot(ExonerationImpot,TotalExonere) :-
175 getIndex(I14001,i14001),
176 getIndex(I14002,i14002),
177 getIndex(I14003,i14003),
178 getIndex(I14004,i14004),
179 TotalExonere @> I14001, TotalExonere @=< I14003,!,
180 ExonerationImpot #= (I14001 * I14002 / 10000) +
181 ((TotalExonere-I14001) * I14004 / 10000),!.
182
183 calcule_ExonerationImpot(ExonerationImpot,TotalExonere) :-
184 getIndex(I14001,i14001),
185 getIndex(I14002,i14002),
186 getIndex(I14003,i14003),
187 getIndex(I14004,i14004),
188 getIndex(I14005,i14005),
189 getIndex(I14006,i14006),
190 TotalExonere @> I14003, TotalExonere @=< I14005,!,
191 ExonerationImpot #= (I14001 * I14002 / 10000) +
192 ((I14003-I14001) * I14004 / 10000) +
193 ((TotalExonere-I14003) * I14006 / 10000),!.
194
195 calcule_ExonerationImpot(ExonerationImpot,TotalExonere) :-
196 getIndex(I14001,i14001),
197 getIndex(I14002,i14002),
198 getIndex(I14003,i14003),
199 getIndex(I14004,i14004),
200 getIndex(I14005,i14005),
201 getIndex(I14006,i14006),
202 getIndex(I14007,i14007),
203 getIndex(I14008,i14008),
204 TotalExonere @> I14005, TotalExonere @=< I14007,!,
205 ExonerationImpot #= (I14001 * I14002 / 10000) +
206 ((I14003-I14001) * I14004 / 10000) +
207 ((I14005-I14003) * I14006 / 10000) +
208 ((TotalExonere-I14005) * I14008 / 10000),!.
209
210 calcule_ExonerationImpot(ExonerationImpot,TotalExonere) :-
211 getIndex(I14001,i14001),
212 getIndex(I14002,i14002),
213 getIndex(I14003,i14003),
214 getIndex(I14004,i14004),
215 getIndex(I14005,i14005),
216 getIndex(I14006,i14006),
217 getIndex(I14007,i14007),
218 getIndex(I14008,i14008),
219 getIndex(I14010,i14010),
220 TotalExonere @> I14007,!,
221 ExonerationImpot #= (I14001 * I14002 / 10000) +
222 ((I14003-I14001) * I14004 / 10000) +
```

```
((I14005-I14003) * I14006 / 10000) +
((I14007-I14005) * I14008 / 10000) +
((TotalExonere-I14007) * I14010 / 10000),!.

% _____Calcul des montants exonérés_____

calcule_TotalExonere(TotalExonere,0) :-
getCase(C1028,c1028),
calcule_Exonere_Contribuable(ExonereContribuable,C1028,1),
calcule_Exonere_Famille(ExonereFamille),
TotalExonere #= ExonereContribuable + ExonereFamille.

calcule_TotalExonere(TotalExonere,1) :-
getCase(C1028,c1028),
calcule_Exonere_Contribuable(ExonereContribuable,C1028,1),
%__Contribuable, fonction de l'handicap, le 1 est inutile ?
getCase(C1029,c1029),
calcule_Exonere_Contribuable(ExonereConjoint,C1029,1),
%__Conjoint, fonction de l'handicap, le 1 est inutile ?
calcule_Exonere_Famille(ExonereFamille),
TotalExonere #= ExonereContribuable + ExonereConjoint + ExonereFamille.

calcule_Exonere_Contribuable(ExonereContribuable,0,1) :-
getIndex(I06002,i06002),
ExonereContribuable #= I06002.

calcule_Exonere_Contribuable(ExonereContribuable,1,1) :-
getIndex(I06002,i06002),
getIndex(I06003,i06003),
ExonereContribuable #= I06002 + I06003.

calcule_Exonere_Famille(ExonereFamille) :-
getCase(C1030,c1030),
calcule_ExonerationEnfants(ExoEnfants1, C1030),
ExonereFamille #= ExoEnfants1.

% _____Calcul de l'exoneration enfants_____

calcule_ExonerationEnfants(ExonerationEnfants, 0) :-
ExonerationEnfants #= 0,!.

calcule_ExonerationEnfants(ExonerationEnfants, 1) :-
getIndex(I06004,i06004),
ExonerationEnfants #= I06004,!.

calcule_ExonerationEnfants(ExonerationEnfants, 2) :-
getIndex(I06005,i06005),
ExonerationEnfants #= I06005,!.

calcule_ExonerationEnfants(ExonerationEnfants, 3) :-
getIndex(I06006,i06006),
ExonerationEnfants #= I06006,!.

calcule_ExonerationEnfants(ExonerationEnfants, 4) :-
getIndex(I06007,i06007),
ExonerationEnfants #= I06007,!.
```

```
280 calcule_ExonerationEnfants(ExonerationEnfants, NbEnfants) :-
281 NbEnfants > 4,
282 getIndex(I06007,i06007),
283 getIndex(I06008,i06008),
284 ExonerationEnfants #= I06007 + (NbEnfants - 4) * I06008,!.
285
286 %_____
287 % Calcul du RIG
288 %_____
289
290 calcule_RIG(RIG,1) :-
291 print('--- Calcul du RIG du contribuable ---'), nl,
292 calcule_RevenusImmobiliers(RevenusImmobiliers,1),!,
293 calcule_RevenusProfessionnels(RevenusProfessionnels,1),
294 RIG #= RevenusImmobiliers + RevenusProfessionnels.
295
296 calcule_RIG(RIG,2) :-
297 print('--- Calcul du RIG du conjoint ---'), nl,
298 calcule_RevenusImmobiliers(RevenusImmobiliers,2),!,
299 RIG #= RevenusImmobiliers.
300
301 %_____
302 % Détermination des revenus immobiliers
303 %_____
304
305 %_____Contribuable_____
306
307 calcule_RevenusImmobiliers(RevenusImmobiliers,1):-
308 getCase(C1100,c1100),
309 getCase(C1101,c1101),
310 C1100 @> 0,!,
311 print('--- Calcul des RC pour habitations propres du contribuable ---'), nl,
312 print('Case 1100 = '), format('~2f~n', [C1100/100]),
313 print('Case 1101 = '), format('~2f~n', [C1101/100]),
314 calcule_RCHabitationPropre(RCHabitationPropre, C1100, C1101),
315 % Index abattement ordinaire
316 getIndex(I01002,i01002),
317 % Enfants à charge à prendre en considération
318 getCase(C1104,c1104),
319 % Index majoration pour enfant
320 getIndex(I01003,i01003),
321 MaxAbattement #= I01002 + (C1104 * I01003),
322 print('Abattement maximal pour habitation = '), format('~2f~n', [MaxAbattement/
100]),
323 calcule_AbattementForfaitaireHabitation(AbattementHabitation, RCHabitationPropre,
MaxAbattement),
324 RevenusImmobiliers #= RCHabitationPropre - AbattementHabitation,
325 print('-----------------------------------------------------------------'),nl,
326 print('Total Revenus Immobilier = '), format('~2f~n', [RevenusImmobiliers/100]),
327 print('-----------------------------------------------------------------'),nl.
328
329 calcule_RevenusImmobiliers(RevenusImmobiliers,1):-
330 getCase(C1100,c1100),
331 getCase(C1101,c1101),
332 C1100 == 0,
333 C1101 @> 0,!,
334 print('--- Calcul des RC pour habitations propres du contribuable ---'), nl,
```

```
335 print('Case 1100 = '), format('~2f~n', [C1100/100]),
336 print('Case 1101 = '), format('~2f~n', [C1101/100]),
337 calcule_RCHabitationPropre(RCHabitationPropre, C1100, C1101),
338 % Index abattement ordinaire
339 getIndex(I01002,i01002),
340 % Enfants à charge à prendre en considération
341 getCase(C1104,c1104),
342 % Index majoration pour enfant
343 getIndex(I01003,i01003),
344 MaxAbattement #= I01002 + (C1104 * I01003),
345 print('Abattement maximal pour habitation = '), format('~2f~n', [MaxAbattement/
100]),
346 calcule_AbattementForfaitaireHabitation(AbattementHabitation, RCHabitationPropre,
MaxAbattement),
347 RevenusImmobiliers #= RCHabitationPropre - AbattementHabitation,
348 print('-----------------------------------------------------------------------'),nl,
349 print('Total Revenus Immobilier = '), format('~2f~n', [RevenusImmobiliers/100]),
350 print('-----------------------------------------------------------------------'),nl.
351
352 calcule_RevenusImmobiliers(RevenusImmobiliers,1):-
353 getCase(C1100,c1100),
354 print('Case 1100 = '), print(C1100),nl,
355 getCase(C1101,c1101),
356 C1100 == 0,
357 C1101 == 0,!,
358 print('--- Pas de RC pour habitations propres pour le contribuable ---'), nl,
359 RevenusImmobiliers #= 0,
360 print('-----------------------------------------------------------------------'),nl,
361 print('Total Revenus Immobilier = '), format('~2f~n', [RevenusImmobiliers/100]),
362 print('-----------------------------------------------------------------------'),nl.
363
364 %_______Conjoint_____________________________________________
365
366 calcule_RevenusImmobiliers(RevenusImmobiliers,2):-
367 getCase(C2100,c2100),
368 getCase(C2101,c2101),
369 C2100 @> 0,!,
370 print('--- Calcul des RC pour habitations propres pour le conjoint ---'), nl,
371 print('Case 2100 = '), format('~2f~n', [C2100/100]),
372 print('Case 2101 = '), format('~2f~n', [C2101/100]),
373 calcule_RCHabitationPropre(RCHabitationPropre, C2100, C2101),
374 % Index abattement ordinaire
375 getIndex(I01002,i01002),
376 % Enfants à charge à prendre en considération
377 getCase(C1104,c1104),
378 % Index majoration pour enfant
379 getIndex(I01003,i01003),
380 MaxAbattement #= I01002 + (C1104 * I01003),
381 print('Abattement maximal pour habitation = '), format('~2f~n', [MaxAbattement/
100]),
382 calcule_AbattementForfaitaireHabitation(RCHabitationPropre, AbattementHabitation,
MaxAbattement),
383 RevenusImmobiliers #= RCHabitationPropre - AbattementHabitation,
384 print('-----------------------------------------------------------------------'),nl,
385 print('Total Revenus Immobilier = '), format('~2f~n', [RevenusImmobiliers/100]),
386 print('-----------------------------------------------------------------------'),nl.
387
```

```
388 calcule_RevenusImmobiliers(RevenusImmobiliers,2):-
389 getCase(C2100,c2100),
390 getCase(C2101,c2101),
391 C2100 == 0,
392 C2101 @> 0,!,
393 print('--- Calcul des RC pour habitations propres pour le conjoint ---'), nl,
394 calcule_RCHabitationPropre(RCHabitationPropre, C2100, C2101),
395 print('Case 2100 = '), format('~2f~n', [C2100/100]),
396 print('Case 2101 = '), format('~2f~n', [C2101/100]),
397 % Index abattement ordinaire
398 getIndex(I01002,i01002),
399 % Enfants à charge à prendre en considération
400 getCase(C1104,c1104),
401 % Index majoration pour enfant
402 getIndex(I01003,i01003),
403 MaxAbattement #= I01002 + (C1104 * I01003),
404 print('Abattement maximal pour habitation = '), format('~2f~n', [MaxAbattement/
100]),
405 calcule_AbattementForfaitaireHabitation(RCHabitationPropre, AbattementHabitation,
MaxAbattement),
406 RevenusImmobiliers #= RCHabitationPropre - AbattementHabitation,
407 print('-----------------------------------------------------------------------'),nl,
408 print('Total Revenus Immobilier = '), format('~2f~n', [RevenusImmobiliers/100]),
409 print('-----------------------------------------------------------------------'),nl.
410
411
412 calcule_RevenusImmobiliers(RevenusImmobiliers,2):-
413 getCase(C2100,c2100),
414 getCase(C2101,c2101),
415 C2100 == 0,
416 C2101 == 0,!,
417 print('--- Pas de RC pour habitations propres pour le conjoint ---'), nl,
418 RevenusImmobiliers #= 0,
419 print('-----------------------------------------------------------------------'),nl,
420 print('Total Revenus Immobilier = '), format('~2f~n', [RevenusImmobiliers/100]),
421 print('-----------------------------------------------------------------------'),nl.
422
423 %_____
424 % Détermination des RC pour habitation propre
425 %_____
426
427 calcule_RCHabitationPropre(RCHabitationPropre, Case100, Case101) :-
428 getIndex(I15004,i15004),
429 print('Coefficient de réindexation = '), format('~2f~n', [I15004/100]),
430 Tmp10 is Case100 * I15004 / 10000,
431 Tmp11 is floor(Tmp10),
432 TmpRC100 #= Tmp11 * 100,
433 Tmp20 is Case101 * I15004 / 10000,
434 Tmp21 is floor(Tmp20),
435 TmpRC101 #= Tmp21 * 100,
436 RCHabitationPropre #= TmpRC100 + TmpRC101,
437 print('Total RC pour Habitations Propres = '),format('~2f~n', [RCHabitationPropre
/100]).
438
439 %_____
440 % Détermination de l'habattement forfaitaire pour habitation propre
441 %_____
```

```
442
443 calcule_AbattementForfaitaireHabitation(AbattementHabitation,RCHabitationPropre,
MaxAbattement) :-
444 RCHabitationPropre @> MaxAbattement,
445 AbattementHabitation #= MaxAbattement.
446
447 calcule_AbattementForfaitaireHabitation(AbattementHabitation,RCHabitationPropre,
MaxAbattement) :-
448 RCHabitationPropre @=< MaxAbattement,
449 AbattementHabitation #= RCHabitationPropre,
450 print('Limité au RC total pour habitation propre, soit : '), format('~2f~n', [
AbattementHabitation/100]).
451
452 %_____
453 % Détermination des revenus professionnels
454 %_____
455
456 %_____Contribuable_____
457
458 calcule_RevenusProfessionnels(RevenusProfessionnels,1) :-
459 calcule_TotalSalaire(TotalSalaire,1),
460 print('Traitement et salaires = '), format('~2f~n', [TotalSalaire/100]),
461 calcule_ChargesForfaitaires(ForfaitSalaire,TotalSalaire),
462 print('Frais professionnels forfaitaires = '), format('~2f~n', [ForfaitSalaire/
100]),
463 getCase(C1258,c1258),
464 print('Frais professionnels introduits (case 1258) = '), format('~2f~n', [C1258/
100]),
465 cherche_Max(Max,ForfaitSalaire,C1258),
466 print('Maximum de frais = '), format('~2f~n', [Max/100]),
467 RevenusProfessionnels #= TotalSalaire - Max,
468 print('-----------------------------------------------------------------'),nl,
469 print('Total Revenus Professionnels = '), format('~2f~n', [RevenusProfessionnels/
100]),
470 print('-----------------------------------------------------------------'),nl.
471
472 %_____Calcule Total Salaire_____
473
474 calcule_TotalSalaire(TotalSalaire,1) :-
475 getCase(C1250,c1250),
476 TotalSalaire #= C1250.
477
478 % _____Calcule Charges forfaitaires sur salaire_____
479
480 calcule_ChargesForfaitaires(ForfaitSalaire,TotalSalaire) :-
481 getIndex(I10001,i10001),
482 getIndex(I10002,i10002),
483 TotalSalaire @=< I10001,!,
484 ForfaitSalaire #= (TotalSalaire * I10002 / 10000),!.
485
486 calcule_ChargesForfaitaires(ForfaitSalaire,TotalSalaire) :-
487 getIndex(I10001,i10001),
488 getIndex(I10002,i10002),
489 getIndex(I10003,i10003),
490 getIndex(I10004,i10004),
491 TotalSalaire @> I10001, TotalSalaire @=< I10003,!,
492 ForfaitSalaire #= (I10001 * I10002 / 10000) +
```

```
493 ((TotalSalaire-I10001) * I10004 / 10000),!.
494
495 calcule_ChargesForfaitaires(ForfaitSalaire,TotalSalaire) :-
496 getIndex(I10001,i10001),
497 getIndex(I10002,i10002),
498 getIndex(I10003,i10003),
499 getIndex(I10004,i10004),
500 getIndex(I10005,i10005),
501 getIndex(I10006,i10006),
502 TotalSalaire @> I10003, TotalSalaire @=< I10005,!,
503 ForfaitSalaire #= (I10001 * I10002 / 10000) +
504 ((I10003-I10001) * I10004 / 10000) +
505 ((TotalSalaire-I10003) * I10006 / 10000),!.
506
507 calcule_ChargesForfaitaires(ForfaitSalaire,TotalSalaire) :-
508 getIndex(I10001,i10001),
509 getIndex(I10002,i10002),
510 getIndex(I10003,i10003),
511 getIndex(I10004,i10004),
512 getIndex(I10005,i10005),
513 getIndex(I10006,i10006),
514 getIndex(I10007,i10007),
515 getIndex(I10008,i10008),
516 getIndex(I10013,i10013),
517 TotalSalaire @> I10005, TotalSalaire @=< I10007,!,
518 ForfaitTemp #= (I10001 * I10002 / 10000) +
519 ((I10003-I10001) * I10004 / 10000) +
520 ((I10005-I10003) * I10006 / 10000) +
521 ((TotalSalaire-I10005) * I10008 / 10000),
522 % Attention à ne pas dépasser le forfait
523 cherche_Min(Min,ForfaitTemp,I10013),
524 ForfaitSalaire #= Min,!.
525
526 %_____
527 % Recherche des réductions d'impôt possibles
528 %_____
529
530 % _____Economie d'énergie_____
531
532 calcule_EconomieEnergie(EconomieEnergie) :-
533 getCase(C1363,c1363),
534 print('Dépenses réalisées pour des économies d énergie = '), format('~2f~n', [
C1363/100]),
535 Economie #= C1363 * 40 /100,
536 getIndex(I06027,i06027),
537 cherche_Min(MinEco,Economie,I06027),
538 EconomieEnergie #= MinEco,
539 print('Réduction obtenue pour des économies d énergie = '), format('~2f~n', [
EconomieEnergie/100]).
540
541 %_____
542 % Recherche des réductions d'impôt possibles
543 %_____
544
545 % _____Economie d'énergie_____
546
547 calcule_SimulEconomieEnergie(SimulEconomieEnergie) :-
```

```
548 getCase(C1363,c1363),
549 print('Dépenses réalisées pour des économies d énergie = '), format('~2f~n', [
C1363/100]),
550 getIndex(I06027,i06027),
551 print('Max possible pour des économies d énergie = '), format('~2f~n', [I06027/
100]),
552 MaxEconomie #= I06027 / 40 * 100,
553 DiffEconomie #= MaxEconomie - C1363,
554 SimulEconomieEnergie #= DiffEconomie * 40 / 100,
555 print('Dépenses d économie d énergie possibles = '), format('~2f~n', [
DiffEconomie/100]).
556
557 %_____
558 % Fonctions générales
559 %_____
560
561 cherche_Max(Max,Montant1,Montant2) :-
562 Montant1 >= Montant2,
563 Max #= Montant1,
564 print('On choisit le maximum soit : '), format('~2f~n', [Montant1/100]).
565
566 cherche_Max(Max,Montant1,Montant2) :-
567 Montant1 < Montant2,
568 Max #= Montant2,
569 print('On choisit le maximum soit : '), format('~2f~n', [Montant2/100]).
570
571 cherche_Min(Min,Montant1,Montant2) :-
572 Montant1 > Montant2,
573 Min #= Montant2,
574 print('On choisit le minimum soit : '), format('~2f~n', [Montant2/100]).
575
576 cherche_Min(Min,Montant1,Montant2) :-
577 Montant1 =< Montant2,
578 Min #= Montant1,
579 print('On choisit le minimum soit : '), format('~2f~n', [Montant1/100]).
580
```

8.4 *Simulation d'impôt du FiscoWeb via Dexia*

FiscoWeb VIA DEXIA

Version : O7.3

Commune : 6001 MARCINELLE
Date de naissance : Déclarant : 03/01/1960
Taux communal : 8,00 %

Date du calcul : 31/8/2009

Taux moyen année antérieure : 45,0 %

Rubriques remplies :
Isolé(e)

VA1	0,00	VA2	0,00	VA3	0,00	VA4	0,00
1001	1	1250	20.000,00	1286	10.000,00		

Impôt à vous rembourser		5.495,76
Cotisation spéciale pour la sécurité sociale		
Montant dû sur	17.753,54	
Déjà retenu	0,00	
Montant qui vous sera remboursé		**5.495,76**

Une (des) assurance(s)-vie pour le montant maximum = € 392,66 d'économie d'impôt supplémentaire.

Le montant maximum d'épargne pension (810,00 EUR) = € 262,44 d'économie d'impôt supplémentaire.

Cadre IV TRAITEMENTS-SALAIRES-ALLOCATIONS DE CHOMAGE-...

Traitements et salaires	20.000,00	
	20.000,00	
Charges forfaitaires	-2.246,46	
	17.753,54	
Précompte professionnel	10.000,00	
Taux plein	17.753,54	

RESUME REVENUS PROFESSIONNELS

Traitements et salaires	17.753,54	
Total des revenus professionnels	17.753,54	
Revenus professionnels nets	17.753,54	

DEPENSES DEDUCTIBLES - REVENU IMPOSABLE

Revenus immobiliers	0,00	
Revenus mobiliers	0,00	
Revenus divers impos. globalement	0,00	
Total des revenus professionnels	17.753,54	

Total des revenus			17.753,54
Revenu imposable			17.753,54

CALCUL DE L'IMPOT

Revenu imposable			17.753,54
Quotités exemptées			
Impôt de base			6.040,00
Impôt de base			5.680,59
Réduction pour quotités exemptées			-1.510,00
Impôt à répartir			4.170,59
Précompte professionnel			-10.000,00
Solde état			-5.829,41
Taux communal	4.170,59	x 8,00% =	333,65
Impôt à vous rembourser			5.495,76

Cotisation spéciale pour la sécurité sociale

Montant dû sur	17.753,54		
Déjà retenu		0,00	
Montant qui vous sera remboursé			**5.495,76**

RENSEIGNEMENTS COMPLEMENTAIRES

Calcul selon globalisation partielle -5.495,76

Total de l'imposition : 4.504,24

Taux moyen de l'exercice :
Déclarant : 23,4 %

Taux moyen année antérieure : 45,0 %

Composition des quotités exemptées

Quotités exemptées	
Impôt de base	6.040,00

UTILISEZ-VOUS LES POSSIBILITES D'IMMUNISATION ?

	Déclarant	
épargne-logement	0,00	45,0 %
épargne à long terme	0,00	30,0 %
maximum	1.211,91	
prime supplémentaire	1.211,91	
économie d'impôt supplémentaire	**392,66**	**32,40 %**
épargne-pension	0,00	30,0 %
maximum	810,00	
montant supplémentaire	810,00	
économie d'impôt supplémentaire	**262,44**	**32,40 %**

FiscoWeb VIA DEXIA

Version : O7.3

Commune : 6001 MARCINELLE
Date de naissance : Déclarant : 03/01/1960
Taux communal : 8,00 %

Date du calcul : 31/8/2009

Taux moyen année antérieure : 45,0 %

Rubriques remplies :

Isolé(e)

VA1	0,00	VA2	0,00	VA3	0,00	VA4	0,00
1001	1	1250	20.000,00	1286	10.000,00	1363	2.600,00

Impôt à vous rembourser			8.303,76
Cotisation spéciale pour la sécurité sociale			
Montant dû sur	17.753,54		
Déjà retenu		0,00	
Montant qui vous sera remboursé			**8.303,76**

Une (des) assurance(s)-vie pour le montant maximum = € 392,66 d'économie d'impôt supplémentaire.

Le montant maximum d'épargne pension (810,00 EUR) = € 262,44 d'économie d'impôt supplémentaire.

Cadre IV TRAITEMENTS-SALAIRES-ALLOCATIONS DE CHOMAGE-...

Traitements et salaires	20.000,00	
	20.000,00	
Charges forfaitaires	-2.246,46	
	17.753,54	
Précompte professionnel	10.000,00	
Taux plein	17.753,54	

RESUME REVENUS PROFESSIONNELS

Traitements et salaires	17.753,54	
Total des revenus professionnels	17.753,54	
Revenus professionnels nets	17.753,54	

DEPENSES DEDUCTIBLES - REVENU IMPOSABLE

Revenus immobiliers	0,00	
Revenus mobiliers	0,00	
Revenus divers impos. globalement	0,00	
Total des revenus professionnels	17.753,54	

Total des revenus			17.753,54
Revenu imposable			17.753,54

CALCUL DE L'IMPOT

Revenu imposable			17.753,54
Quotités exemptées			
Impôt de base			6.040,00
Impôt de base			5.680,59
Réduction pour quotités exemptées			-1.510,00
Réd. pour investissements économiseurs d'énergie			-2.600,00
Impôt à répartir			1.570,59
Précompte professionnel			-10.000,00
Solde état			-8.429,41
Taux communal	1.570,59	x 8,00% =	125,65
Impôt à vous rembourser			8.303,76

Cotisation spéciale pour la sécurité sociale

Montant dû sur	17.753,54		
Déjà retenu		0,00	
Montant qui vous sera remboursé			**8.303,76**

RENSEIGNEMENTS COMPLEMENTAIRES

Calcul selon globalisation partielle -8.303,76

Total de l'imposition : 1.696,24

Taux moyen de l'exercice :
Déclarant : 8,8 %

Taux moyen année antérieure : 45,0 %

Composition des quotités exemptées

Quotités exemptées	
Impôt de base	6.040,00

UTILISEZ-VOUS LES POSSIBILITES D'IMMUNISATION ?

	Déclarant	
épargne-logement	0,00	45,0 %
épargne à long terme	0,00	30,0 %
maximum	1.211,91	
prime supplémentaire	1.211,91	
économie d'impôt supplémentaire	**392,66**	**32,40 %**
épargne-pension	0,00	30,0 %
maximum	810,00	
montant supplémentaire	810,00	
économie d'impôt supplémentaire	**262,44**	**32,40 %**

8.5 *Exemple de documents utilisateurs*

8.5.1 Fiche 281.10

FICHE DE REMUNERATIONS N° 281.10 - ANNEE ....

1. N° ..... (à reprendre au relevé 325.10) | 2. Date de l'entrée : ..... de la sortie : .....

3. **Débiteur des revenus** :
NN ou NE : .....
.....
.....

4. Expéditeur :
.....
.....
....

Destinataire :
.....
.....
.....
....

Nom et prénoms de l'époux ou du cohabitant légal :
.....

5. Situation de famille

Cjt.	Enf.	Autres	Divers

6. Etat civil : .....

7. N° commission paritaire : .....

8. N° d'identification : .....

9. a) Rémunérations (1) :		..... , . .
b) Pécule de vacances : (..... jours)		..... , . .
c) Avantages de toute nature (2) : Nature :		..... , . .
d) TOTAL :	250	..... , . .
10. Options sur actions % : ..... % : ..... % : ..... ☐ Société étrangère (3)	249	..... , . .
	248	..... , . .
11. Revenus taxables distinctement : a) Pécule de vacances anticipé :	251	..... , . .
b) Arriérés :	252	..... , . .
c) Indemnités de dédit :	253	..... , . .
d) Indemnité de reclassement :	245	..... , . .
e) Prime régionale de remise au travail (4) :	291	..... , . .
12. Intervention dans les frais de déplacement : a) Transport public en commun :		..... , . .
Véhicule mis à disposition Nombre de kilomètres : ..... b) Transport collectif organisé : ☐ OUI ☐ NON		..... , . .
c) Autre moyen de transport :		..... , . .
d) TOTAL :	254	..... , . .
13. Retenues pour pensions complémentaires :		
a) Cotisations et primes normales :	285	..... , . .
b) Cotisations et primes pour la continuation individuelle :	283	..... , . .
Caisse ou société :		
14. Heures supplémentaires qui donnent droit à un sursalaire :		
a) Nombre total d'heures supplémentaires effectivement prestées :	246	.....
b) Base de calcul du sursalaire relatif aux heures donant droit à une réduction		
de : - 24,75 % (..... heures)	247	..... , . .
- 66,81 % (..... heures)	233	..... , . .
- 57,75 % (..... heures)	234	..... , . .
15. Précompte professionnel :	286	..... , . .
16. Cotisations spéciales pour la sécurité sociale :	287	..... , . .
17. Personnel du secteur public sans contrat de travail (5) :	290	☐ OUI
18. Renseignements divers : a) Déplacements à vélo : Km ..... Indemnité totale :		..... , . .
b) Dépenses propres à l'employeur :		..... , . .
c) Pourboires : Code (6) ..... Forfait Séc. Soc. :		..... , . .

N° 281.10 - DTP - 2007

(1) à (6) : voir renvois au verso.