II.1 Introduction

Dans le domaine de réseaux électriques, on constate que l'application de la MOO (Modélisation orientée objets) à beaucoup d'avantages vue que la structure physique d'un réseau électrique est adaptable à une structure de classes [8]. La MOO d'un réseau électrique ne vise pas seulement la construction d'une bonne structure de classes de ses éléments, mais également les méthodes d'analyse de ce système et les GUI. La construction des GUI est devenue une partie importante du génie logiciel [8,9 ,1]. En effet les phénomènes physiques ainsi que les données d'un réseau électrique sont mieux assimilées si l'information est représentée sous forme graphique contrairement à la forme numérique. Pour des systèmes complexes, où les interactions homme-machine sont nombreuses, la spécification de la GUI par une approche orientée objets offre des avantages indéniables en terme de génie logiciel.

Dans ce chapitre, on essaye de combiner la MOO, la simulation des réseaux électriques et la GUI pour dégager la structure générale de l'outil informatique développé. Mais avant tout, nous passons en revue les principales caractéristiques et propriétés de tout système logiciel. Et enfin on présente les grandes structures de classes adaptées pour le développement de cet outil.

II.2 Développement des systèmes logiciels

Le développement de systèmes logiciels étant une industrie relativement neuve, il n'a pas encore atteint le niveau de maturité des branches plus traditionnelles de l'industrie. Par conséquent, les produits que l'on développe sur la base de la technologie du logiciel souffre du manque de pratiques établies nécessaires à leur développement et à leur exploitation en tant que produits commerciaux [4].

II.2.1 Cycle de vie d'un système logiciel

Tous les systèmes évoluent au cours de leur cycle de vie. On doit garder cela à l'esprit lorsque l'on développe des systèmes qui sont censés évoluer au delà de la première version. Normalement, on développe un système au fil des modifications, de version en version. La première version représente une partie mineure de la consommation de ressources totale qui aura lieu durant le cycle de vie d'un système. A chaque nouvelle version, on effectue les mêmes activités de développement que pour le développement d'un nouveau système. La différence vient du fait que les données d'entrée décrivent des besoins de modification.

Historiquement, dans la plupart des projets à connotation informatique, on spécifie les besoins du système comme un tout. Après les spécifications viennent l'analyse, la conception et le test du système complet. Cette méthode peut fonctionner si on connaît tous les besoins du système final dès le début, mais c'est rarement le cas. D'habitude, on ne connaît pas complètement les besoins au démarrage du projet pour les systèmes techniques aussi bien que pour les systèmes d'information. On accroît sa connaissance du système au fur et à mesure que le travail progresse. Lorsque la première version du système est opérationnelle, apparaissent de nouveaux besoins, et les anciens changent. On ne peut donc pas développer le système dans son ensemble en croyant que la spécification des besoins restera la même durant le temps de développement, qui peut atteindre plusieurs années pour les gros systèmes.

Dans la plupart des cas, il vaut mieux développer le système étape par étape en commençant par quelques unes de ses fonctions clés. On peut ajouter de nouvelles fonctions ensuite jusqu'à ce que l'on ait atteint le niveau désirer.

II.2.2 Réutilisation et composants

Un désir commun à tous les travaux de développement est de pouvoir réutiliser les résultats des travaux précédents. La nécessité de réutilisabilité est applicable durant le codage, puisqu'elle peut avoir une influence significative sur la productivité. C'est dans ce contexte que les gens du logiciel parlent de réutilisabilité. La réutilisation au niveau des autres phases du développement peut améliorer encore plus la productivité. Le problème de réutilisation est de trouver et de comprendre ce que l'on doit réutiliser, et de juger de la pertinence de la réutilisation. L'orientée objets offre une technique totalement nouvelle qui

fournit une aide efficace pour résoudre ces problèmes. Etre capable de réutiliser des parties déjà développées (composants) dans un produit constitue un facteur significatif de diminution du coût du cycle de vie du produit.

Les composants logiciels sont traditionnellement disponibles sous la forme de procédures et de fonctions pour les applications numériques et statiques. Il faut y rajouter les composants logiciels permettant entre autre de gérer des tampons de mémoire, des files d'attente, des listes et des arbres, dont on a souvent besoin lorsque l'on programme des algorithmes. On trouvera aussi des fenêtres, des icônes et des barres de défilement pour les interfaces homme/machine.

II.2.3 Stratégie de développement

L'une des propriétés clés d'un système logiciel est sa structure interne. Une bonne structure rend le système facile à comprendre, modifier, tester et maintenir. C'est pourquoi les propriétés de l'architecture d'un système déterminent la manière dont on le traitera au cours de son cycle de vie. Les plus gros systèmes seront sujets à modification au cours de leur cycle de vie. Une approche industrielle du génie logiciel doit prendre ce fait en compte. En fait, le développement de systèmes est un processus de modifications successives qui dure aussi longtemps que l'on impose de nouveaux besoins au produit. Le processus de développement logiciel est un ensemble d'activités reliées à la création, réalisation et maintenance des systèmes logiciels. Il n'existe pas de méthode universelle applicable à tous les processus de développement.

En pratique, l'approche orientée objets s'inscrit dans un processus de développement itératif. Le système logiciel n'est pas construit étape par étape mais bien itération en itération. Un sous ensemble d'exigences est analysé, des solutions de conception sont proposées, la conception et la validation des solutions sont réalisées dans chacune des itérations. De cette façon le système logiciel est construit par incrément et sa complexité devient gérable. De plus, à la fin de chaque itération, il est possible d'apporter des modifications et corriger les erreurs d'analyse et de conception.

II.3 Développement de logiciels de réseaux électriques

La nécessité de modéliser les réseaux pour en simuler le fonctionnement remonte sans doute aux origines des réseaux électriques eux-mêmes. Les premiers simulateurs étaient des simulateurs analogiques où des modèles réduits de réseaux permettaient de mieux prévoir ou de connaître le comportement du système. Ils permettaient de couvrir certains besoins comme la mise au point ou le test d'équipements de régulation et de protection des ouvrages. De nombreux simulateurs analogiques de cette génération, fonctionnant en temps réel sont toujours utilisés. Aujourd'hui, il est de plus en plus question de simulateurs numériques.

II.3.1 Complexité des logiciels de réseaux électriques

Les problèmes que les logiciels de réseaux électriques doivent résoudre comportent souvent des éléments extrêmement complexes et qui cachent de multitudes exigences. Aujourd'hui, les réseaux d'énergie électrique sont de plus en plus complexes alimentant des charges elles même de plus en plus exigeantes. En parallèle, la maîtrise des régimes de fonctionnement perturbés et la conception de protections sûres et sélectives contribuent à augmenter cette complexité. Donc, le fonctionnement des réseaux électriques est déjà difficile à cerner, pourtant il faut y ajouter des exigences (non fonctionnelles) telles que la facilité d'utilisation et la maintenance de ses logiciels. Cette complexité du problème lui même entraîne donc une complexité du logiciel.

La complexité du logiciel provient généralement de la façon dont les utilisateurs et les développeurs voient les choses. Les gens de réseaux électriques trouvent des difficultés à fournir une expression précise de leurs besoins sous une forme que les développeurs (informaticiens) peuvent comprendre. Cette incompréhension n'est due ni aux utilisateurs ni aux développeurs, mais plutôt au fait que chacun de ces groupes manque d'expertise dans le domaine de l'autre. En plus, cette complexité est augmentée par le fait que les spécifications d'un logiciel changent souvent en cours de développement. C'est pourquoi, d'après la littérature, une grande majorité de logiciels de réseaux électriques sont développées par les gens de réseaux électriques eux mêmes.

II.3.2 Acheter ou développer les logiciels à utiliser

La grande question souvent posée dernièrement se résume à ceci : vaut-il mieux investir et acheter l'outil de simulation ou développer son propre outil ? Les deux options sont valables et le choix se fait généralement selon les contraintes de temps, des ressources et le but visé.

L'achat de l'outil entraîne souvent des coûts imprévus. De plus, vu que la désuétude des ordinateurs est de plus en plus rapide, le choix de dépréciation devrait être de cinq ans sans quoi la technologie risque d'être périmée. Une entreprise peut se permettre de gros investissement dans cette direction contrairement à un établissement académique où développer convient le mieux.

II.4 Structure générale de l'outil développé

Afin de tirer parti des avantages des TOO (Technique Orientée Objets), une modélisation orientée objets selon OMT (Object Modeling Technique) à été développée pour la conception des composants logiciels intervenant dans le processus de développement. La stratégie retenue quant à l'architecture des simulateurs de réseaux électriques a conduit à quatre grandes parties. La figure II.1 montre ces principales parties qui sont :

- Un éditeur graphique est spécialement développé pour visualiser les diagrammes unifilaires des réseaux électriques avec fenêtres de boites de dialogue. Il utilise des symboles graphiques pour représenter les éléments du réseau électrique tels que les jeux de barres, les lignes de transmission, les charges, les générateurs, etc.
- Une base de données visuelle est développée pour que l'utilisateur puisse faire entrer et modifier les données avec souplesse sur écran. Les données sont liées au diagramme unifilaire et aux applications à exécuter.
- Les applications qui simulent le fonctionnement d'un réseau électrique, les applications réalisées actuellement dans cet outil sont la matrice de générateur de donnée et calcul la fiabilité.

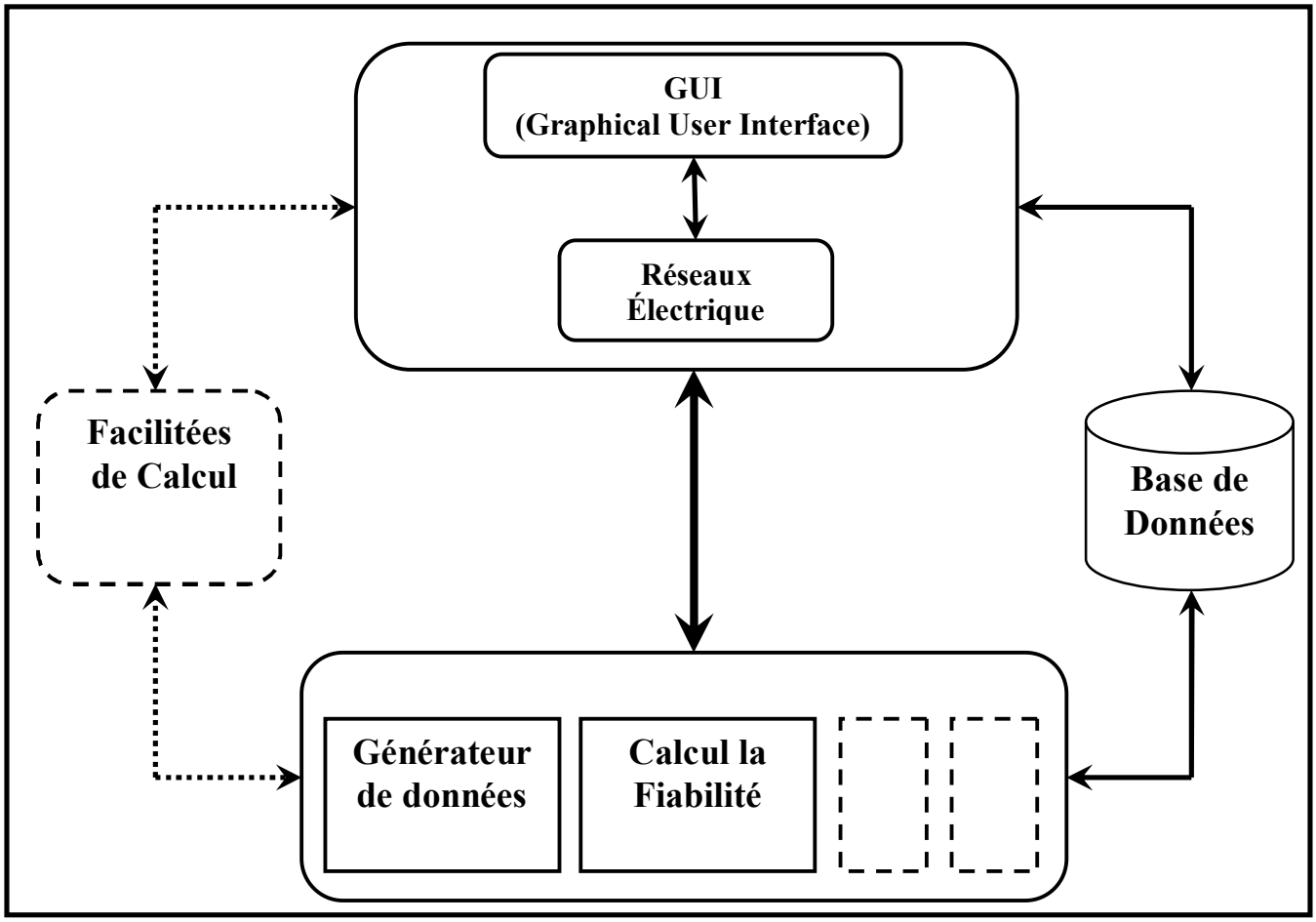

Figure II.1 Structure générale

Toutes les parties sont développées en utilisant Microsoft Visual Studio 2008 version 9.0 [10] et le langage de programmation C# [11]. Le matériel utilisé est un PC Intel IV. Pour implémenter les fonctionnalités de la GUI, l'outil développé utilise deux hiérarchies d'objets, l'une dérivant de l'objet TForm de Microsoft Visual Studio 2008 pour représenter les fenêtres elles mêmes et l'autre dérive de l'objet TGraphicControl de Microsoft Visual Studio 2008 pour représenter les éléments graphiques sur ces fenêtres.

II.5 Interface graphique usager GUI

La construction des interfaces usagers graphiques est devenue une partie importante du génie logiciel. Au cours des dernières années, nous avons vu apparaître de nombreuses méthodes et outils permettant de réduire la charge de travail des développeurs et des concepteurs et permettant de produire des interfaces de meilleurs qualité. Depuis leur apparition, les interfaces graphiques n'ont pas cessé d'évoluer en apportant plus de souplesse à l'utilisateur, en réduisant sa charge de travail et en répondant efficacement à ses besoins. L'acceptation de ces nouvelles interfaces n'est pas uniquement due à

l'avancée technologique de l'informatique que ce soit au niveau matériel ou logiciel, mais elle revient aussi aux résultats des sciences cognitives qui s'intéressent à l'étude du comportement humain. Par l'intégration des règles ergonomiques (facilité d'utilisation, concision, cohérence, flexibilité, etc.), ces interfaces ont mérité l'appréciation de l'utilisateur. Une autre raison principale de ce succès provient du fait que l'utilisateur est maître de l'interaction avec l'application pendant toute la session de travail [8].

L'avancée technologique au niveau hardware et software a fait bondir les applications interactives au premier plan. Actuellement le développement de la GUI prend une grande place dans le développement des applications informatiques. Certaines évaluations [8] ont révélé que plus de 50% du temps de développement est consacré à la réalisation de la GUI, et que plus de 70% du coût d'un produit logiciel s'accumule dans sa maintenance où la maintenance de la GUI représente la grande partie.

Pour des systèmes industriels complexes, où les interactions homme-machine sont nombreuses, la spécification de la GUI par une approche orientée objets offre des avantages indéniables en terme de génie logiciel. En effet, cette technique repose sur les notions de classe et d'héritage. L'utilisation et la combinaison d'un ensemble de techniques facilitent la maîtrise de la complexité, l'extensibilité et la réutilisabilité des objets pour la construction des GUI, et assurent la cohérence et l'homogénéité dans les modes de présentation et les dialogues. Le marché informatique actuel offre un éventail varié d'environnements graphiques facilitant la réalisation et la modification des GUI en réduisant l'écriture de code et en permettant la réutilisabilité des programmes.

Les phénomènes physiques ainsi que les données d'un réseau électrique sont mieux assimilées si l'information est représentée sous forme graphique contrairement à la forme numérique. Plusieurs outils graphiques dans le domaine de réseaux électriques ont été développés pour l'enseignement, la recherche et l'entraînement [8,9,10,11]. Dans ce travail de mémoire, on essaye de combiner la MOO, la simulation des réseaux électriques et la GUI dans un seul outil. L'interaction principale entre l'utilisateur et la simulation est faite à travers la GUI. Pour que l'outil développé soit une plate forme pour l'étude des réseaux électriques, il est important que la GUI puisse fournir à l'utilisateur des accès faciles à toutes les informations concernant le réseau électrique, même pour de grands systèmes. Pour accomplir ceci, la GUI est conçue en utilisant la TOO. La partie la plus importante de la GUI concerne la représentation des diagrammes unifilaires ou l'éditeur graphique. La TOO permet à l'utilisateur d'interagir avec tous les objets dans les différentes fenêtres. Les

dernières années, le monde du développement de logiciels s'est trouvé profondément modifié par le succès des outils de développement rapide d'applications (RAD) tels que Microsoft Visual Studio 2008, C++ Builder, Delphi, Java et autres et l'arrivée de la TOO.

II.6 Modélisation orientée objets des réseaux électriques

La phase de modélisation est une étape très difficile, c'est celle qui fait le plus appel au « sens de l'ingénieur », elle suppose que l'on ait préalablement fourni une réponse claire à ce qu'on veut représenter et avec quel objectif. Dans le domaine de réseaux électriques, on constate que l'application de la MOO a beaucoup d'avantages vue que la structure physique d'un réseau électrique est adaptable à une structure de classes [2,4,8].

La MOO d'un réseau électrique ne vise pas seulement la construction d'une bonne structure de classes de ses éléments, mais également des méthodes d'analyse (applications ou fonctions de calcul) de ce système électrique et son GUI. Il serait bien normal de penser que les méthodes d'analyse devraient être implémentées comme méthodes (fonctions membres) dans la structure représentative du système. Cependant, cette vision limite la flexibilité de la structure. Actuellement la MOO fait sortir deux grandes structures de classes : la structure des éléments physiques du système électrique et les méthodes d'analyse appliquées au système ou sur ses éléments. Une troisième structure utilitaire concerne les facilitées de calcul (matrices, algorithmes, etc.).

La structure des éléments physiques est représentée en premier lieu, ensuite la structure des méthodes d'analyse est formée indépendamment de la première structure représentative du réseau électrique. La deuxième structure agit sur la première en l'utilisant comme données d'entrée et de sortie pour ses méthodes. La représentation des méthodes d'analyse par une telle structure indépendante de la structure physique apporte de la flexibilité à l'outil informatique. Dans la construction de la structure des éléments physiques on ignore les détails caractérisant les méthodes d'analyse du réseau électrique, et vice versa. Ce qui permet d'avoir un projet plus modulaire.

La définition des structures de classes est le point de départ dans un projet orienté objets. Le modèle objet statique qui représente les éléments du monde réel et ses relations est la base des prochaines étapes du projet [4,8]. Par conséquent, une attention particulière doit être adressée à cette phase du projet qui n'est pas une tâche facile. En effet, dans des problèmes complexes, le cas des réseaux électriques, la possibilité d'avoir plusieurs représentations de classes est largement évidente. Donc le choix d'une forme ou d'une

autre pour la représentation de la structure, selon la philosophie de la MOO, peut impliquer des dissimilitudes dans l'exécution des applications bien déterminées.

D'une manière générale, la conception d'un bon logiciel orienté objets n'est pas une tâche facile. Un bon logiciel doit également avoir de bonnes performances sur le plan temps de calcul. Les travaux les plus récents [1,2,3,11] ont donné des résultats satisfaisants pour des réalisations orientées objets (en utilisant le langage de programmation C#). Ces améliorations dans le temps de calcul sont dues aux développements de compilateurs qui supportent l'orientée objets et qui ont évolué suffisamment ces dernières années. D'un autre coté une grande évolution sur le plan matériel a permis de disposer de machines de grande capacité de traitement de données.

Dans ce travail, le processus de création des structures de classes représentatives des entités diverses du réseau électrique dans son ensemble sont divisées en abstractions distinctes et représentées par des packages. La figure II.3 montre ces abstractions et leurs dépendances. On considère trois abstractions principales : le réseau électrique (classe Power System), les fonctions de calcul (classes des Applications) et les facilitées de calcul (classe Computational Facilities). Cette dernière n'est pas décrite dans ce travail parce qu'elle n'est pas vraiment implémentée orientée objets, mais elle est utilisée dans les différents calculs.

En principe, l'outil informatique est constitué par un objet réseau électrique et une ou plusieurs applications peuvent être exécutées sur cet objet. Les abstractions ne sont pas fermées, elles admettent l'inclusion de nouvelles entités dans leurs propres limites.

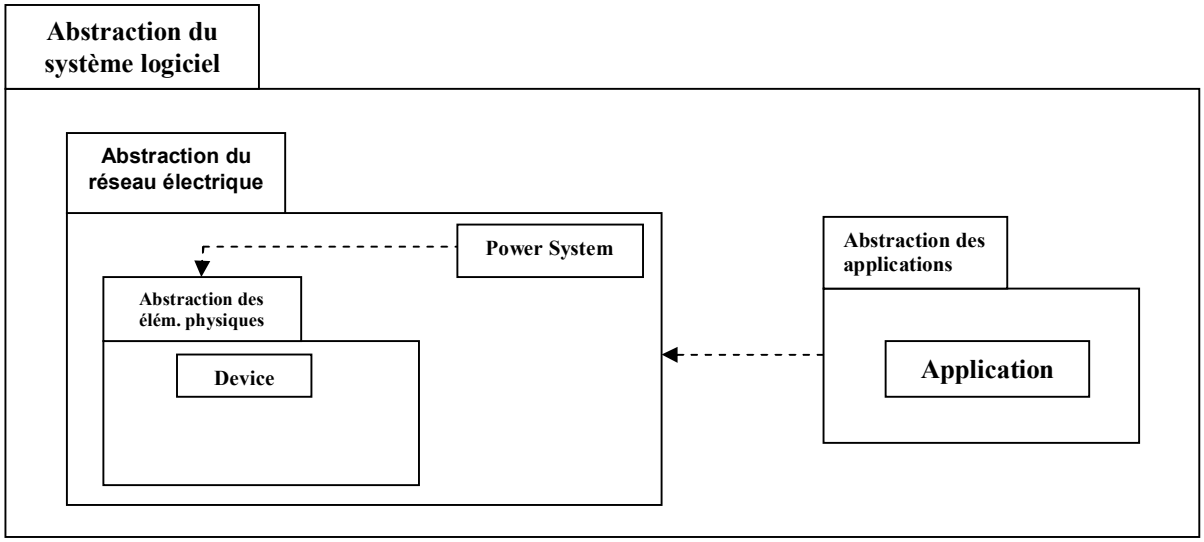

Figure II.2 Abstractions du système logiciel

L'abstraction du système électrique délimite la modélisation des éléments physiques du réseau électrique. Cette abstraction est illustrée par la figure II.3.

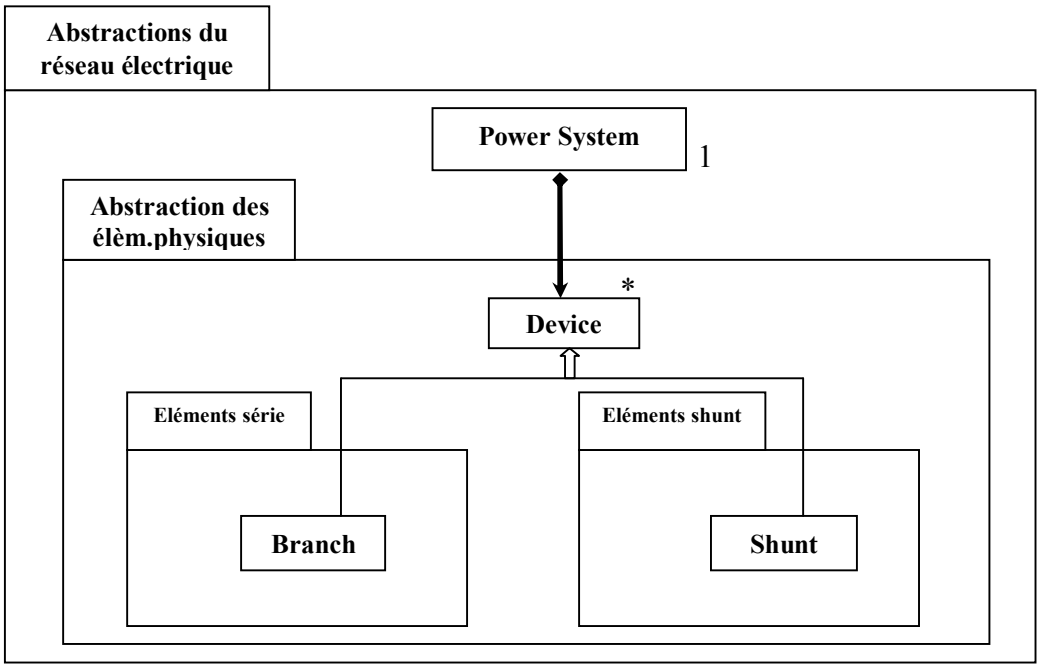

Figure II.3 Abstraction du réseau électrique

Dans cette abstraction deux entités sont identifiées : la classe (Power System) et la classe (Device).La classe (Power System) forme le réseau électrique dans son ensemble, elle englobe les classes qui forment ce réseau et qui représentent les éléments physiques. Cette abstraction contient les classes des jeux de barres, des lignes de transmission, des transformateurs, des charges, des générateurs etc., qui sont représentées dans la figure II.4 par la classe de base (Device). Cette classe est la classe de base de tous les éléments physiques du réseau électrique. Dans cette abstraction apparaît la classe constituée par la composition de ses éléments physiques divers.

II.6.1 Modélisation des éléments physiques

Un point clef dans la conception d'une structure qui représente le réseau électrique est le critère de classification hiérarchique des classes représentatives de ce système. On entend par élément physique tout dispositif qui ; relié d'une manière ou d'une autre au réseau électrique fait partie de sa constitution. Dans ce travail actuel de mémoire, la classification des éléments physiques est basée sur la structure réelle du système électrique. Cette classification prend en compte dans sa limite inférieure le nombre de connections aux différents jeux de barres que chaque élément présente, identifiant là trois classes :

— Jeux de barres.

— Eléments en série (ou branche, avec deux connections).

— Eléments en dérivation (ou shunt, avec une connexion).

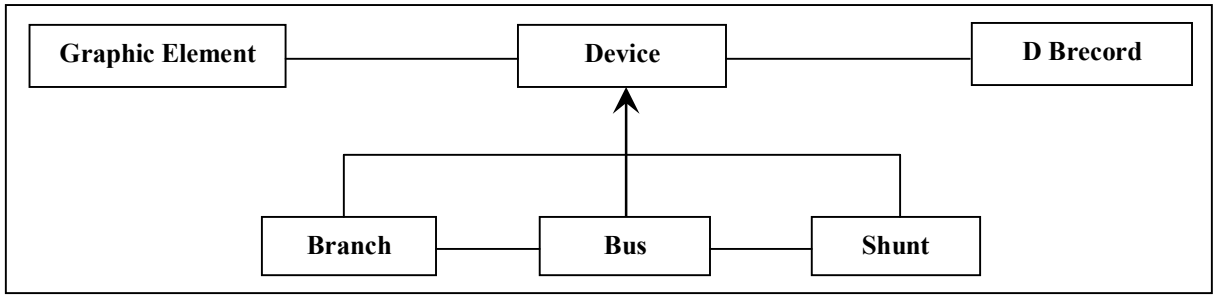

Figure II.4 Eléments physiques

II.6.2 Modélisation des applications

L'abstraction des applications représente les méthodes d'analyse des réseaux électriques. Ces applications dérivent toutes de la classe PowerSystem. Les applications dépendent du système électrique, donc les algorithmes d'analyse seront exécutés sur la base des données représentant le système à étudier.

L'abstraction des applications définit les classes représentant les différentes méthodes d'analyse et de synthèse appliquées aux réseaux électriques. Cette abstraction est illustrée par la figure II.5. Dans cette abstraction la classe PowerSystem sert de base dans la structure hiérarchique des classes représentant toutes les applications qui peuvent exister.

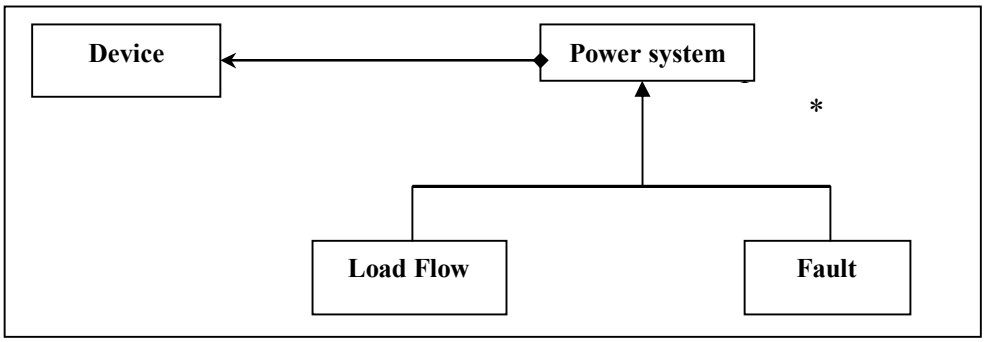

Figure II.5 Eléments physiques, réseau électrique et applications

II.6.3 Modélisation des facilitées de calcul

Généralement toutes les applications réseaux électriques ont besoin de certaines structures de données bien précises tels que les complexes, les vecteurs, les matrices, les matrices creuses, la factorisation, les listes chaînées etc. Ainsi plusieurs auteurs [6,7] ont pensé à réaliser une sorte de bibliothèque mathématique spéciale pour les réseaux électriques qui englobe ces structures et en utilisant la TOO.

Egalement ici la question qui se pose : faut-il développer ses propres structures ou utiliser les bibliothèques standards qui existent sur marché ? Ces dernières ont une taille énorme et sont d'un usage général. Autrement dit laquelle de ces deux situations est la plus juste :

- Utiliser des « boites noires » avec une garantie totale de leur bon fonctionnement sans comprendre les détails internes des algorithmes utilisés.
- Développer les structures spéciales pour des applications particulières même si ces programmes ne sont pas assez performant que ceux sur marché.

Les deux situations sont justes. Mais récemment, dans le domaine des réseaux électriques, chacun à tendance à développer ses propres outils tant qu'il peut [1,2,6,7]. La raison est que les structures de données à usage général ont comme majeure conséquence un gaspillage de performances quant elles sont appliquées à un problème particulier.

Actuellement, l'utilisation des Template semble répondre à toutes les exigences vu qu'ils ont un aspect général et particulier en même temps.

II.7 Conclusion

Dans ce chapitre, nous avons présenté les grandes abstractions pour la plate forme logicielle de l'outil développé. Une grande partie des fonctionnalités de ces abstractions ont été mises en œuvre dans une architecture orientée objets (mise à part la partie base de données). L'objet de cette étude que nous détaillons dans les chapitres suivants, vise à fournir des composants logiciels pour la représentation, la simulation et la manipulation de données des réseaux électriques.