N° d'ordre : **2790**

THÈSE

présentée à

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DES SCIENCES PHYSIQUES ET DE L'INGÉNIEUR

par **Fabien LEGRAND**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : **ÉLECTRONIQUE**

****

MODÉLISATION DE CIRCUITS ÉLECTROTECHNIQUES EN VUE DE LEUR SIMULATION - RÉALISATION D'UN SIMULATEUR.

****

Soutenue le : 29 janvier 2004

Après avis de :

J. BOUCHER	Professeur Emérite	INP-Toulouse	**Rapporteur**
J.J. CHARLOT	Professeur	ENST Paris	**Rapporteur**

Devant la commission d'examen formée de :

A. TOUBOUL	Professeur	Université Bordeaux I	**Président**
J. BOUCHER	Professeur Emérite	INP Toulouse	**Rapporteur**
J.J. CHARLOT	Professeur	ENST Paris	**Rapporteur**
M. DELEST	Professeure	Université Bordeaux I	**Rapporteuse de la soutenance**
H. LEVI	Professeur	Université Bordeaux I	**Directeur de Thèse**
N. COUTURE	Maître de conférences	ESTIA	**Co-directrice de Thèse**
J. PERE-LAPERNE	PDG	Algo'Tech Informatique	**Invité**
R. BRIAND	Maître de conférences	ESTIA	**Invité**

– 2004 –

à Cristiana

Remerciements

Je tiens à remercier André TOUBOUL, directeur du laboratoire IXL de l'université de Bordeaux 1, Jean-Rodolphe PUIGGALI et Pascal WEIL les directeurs scientifiques du laboratoire LIPSI - ESTIA de m'avoir accueilli au sein de leurs laboratoires et pour l'intérêt qu'ils ont su porter à mes travaux. Merci également à Jean-Roch GUIRESSE, directeur de l'ESTIA, pour ses encouragements et son soutien.

Je souhaite remercier tout particulièrement Hervé LEVI, professeur à l'IXL et Nadine ROUILLON-COUTURE, responsable déléguée du LIPSI, directeur et co-directrice de cette thèse, pour leur écoute bienveillante, leur conseils avisés. Ils ont su, avec une constante bonne humeur, me faire partager l'approche de leur domaines respectifs et participer ainsi à ma culture scientifique.

Un merci amical et particulier pour Renaud BRIAND qui c'est impliqué activement dans ce travail de thèse malgré ses disponibilités réduites.

Je n'oublie pas le personnel d'Algo'Tech Informatique, pour son concours, sa sympathie et de m'avoir fait partager sa connaissance du domaine de la schématique électrique. Je remercie plus particulièrement Jacques PERE-LAPERNE, PDG d'Algo'Tech et initiateur de ce projet de recherche, pour sa disponibilité et l'attention qu'il a su me porter.

Ma gratitude est également adressée à Guy VERNHERES, chef des travaux du lycée Louis de Foix de Bayonne ainsi qu'à l'équipe enseignante, spécialement Rolland POULET, pour leur participation active à la validation du simulateur et pour nos collaborations à venir.

Je n'oublierai pas Nicolas DEPAIL, élève ingénieur de L'ENSEIRB, ni Henri ABITBOL, élève ingénieur de l'ESTIA, pour leur brillante contribution à mes travaux. Merci, à Sabeur JEMMALI, Doctorant à l'ENST, pour sa disponibilité, sa patience et son aide à la compréhension des arcanes de VHDL-AMS.

Enfin, j'exprime mes plus vifs remerciements à M. Jean-Jacques CHARLOT, Professeur à l'ENST et M. Jacques BOUCHER, Professeur Emérite de l'INP Toulouse qui m'ont fait l'honneur de s'intéresser à ces travaux et d'avoir accepté d'en être les rapporteurs.

Sommaire

Chapitre 2 : Les modes d'analyses.

Chapitre 3 : modélisation.

Chapitre 4 :Validation

Introduction générale

Les simulateurs de circuits électriques sont apparus dans les années 1970. Leur précurseur, SPICE (Simulation Program with Integrated Circuit Emphasis) a connu un succès mondial en raison du fort développement de l'industrie électronique. Son cœur de simulation développé par l'université de Berkeley [SP1] est dans le domaine public et est intégré dans de nombreux simulateurs. On le retrouve dans sa dernière version, Spice3f5 [SP2] notamment dans des logiciels de CAO électronique tels que OrCAD [ORC] (avec Pspice) ou Protel DXP.

L'évolution rapide des composants électroniques, que chacun a pu mesurer avec l'augmentation de performance des microprocesseurs et la baisse du prix des ordinateurs, nécessite des moyens de test et de contrôle de plus en plus efficaces. En raison de la densité et de la complexité croissante des circuits électroniques, la simulation est devenue une étape incontournable du cycle de conception des circuits et des composants. Il est indispensable de vérifier à priori le comportement électrique du circuit qui va être soumis au fabricant. La simulation permet ainsi une réduction des coûts de développement en évitant en partie la réalisation de prototypes relativement coûteux.

Certains composants électroniques, transistor, thyristor ou triacs (IGBT, GTO…) sont utilisés pour la commutation de courant élevé comme pour certains convertisseurs statiques. Malgré les fortes puissances mises en jeu et l'utilisation de ce type de composants pour le pilotage et l'alimentation d'équipement électrotechniques, on considère généralement que ces composants électroniques appartiennent de l'électronique de puissance, domaine charnière entre l'électronique et l'électrotechnique. Ceux-ci pourront être modélisé et simulé par des simulateurs orientés vers l'électronique.

L'évolution technologique des composants de l'électrotechnique se fait à une toute autre échelle. Certains composants conçus il y a plus de 10 ans sont toujours commercialisés. On ne connaît pas ce phénomène d'augmentation de la densité des circuits de l'électronique. L'évolution des coûts de conception et de fabrication restent relativement stables par rapport à ceux de l'industrie électronique. De plus, l'utilisation d'outils de CAO électrique ne s'est généralisée que depuis peu de temps, notamment grâce à l'accessibilité plus grande des microordinateurs.

Toutes ces raisons font que l'utilisation de la simulation en électrotechnique ou en électricité des courants forts est assez peu répandue. Il n'y existe pas de culture de la simulation. Parmi les logiciels de schématiques électriques, pour l'électrotechnique ou l'électricité, disponibles sur le marché français, il n'existe aucun simulateur intégré. Il existe certains outils de calcul qui permettent une aide au choix et au dimensionnement du matériel. Ces outils ne sont pas des simulateurs et sont généralement basés sur la capitalisation des règles métier et de règles de calculs définies par des normes. On trouve notamment des logiciels de calcul pour les

installateurs électriciens qui permettent de déterminer les valeurs de courant de court-circuit et de proposer un dimensionnement des câbles et des appareils de protection. Ces logiciels parmi lesquels on peut citer Caneco ou Ecodial sont basés entre autre sur la norme NFC 15-100 [NFC98] qui définit les règles de réalisation d'installations électriques basse tension et la norme NFC 15-500 qui détaille les principes de calculs à appliquer pour ces installations.

L'utilisation de simulateurs apparentés à Spice se retrouve toutefois dans des domaines périphériques de l'électrotechnique, tels que l'électronique de puissance. On peut également en trouver des applications à l'automatique, notamment pour la régulation et l'étude de la dynamique des moteurs comme avec Simplorer [SIM], bien que l'on privilégie plutôt l'utilisation de logiciels plus dédiés comme MatLab – Simulink [MATL]. Mais ces logiciels semblent mal adaptés aux circuits électrotechniques. En effet, ces circuits sont soumis à de fréquentes modifications de leur structure en raison du relayage ou du sectionnement qui peuvent mettre en service ou non certaines parties d'une installation. D'une part, le temps qui s'écoule entre chacune de ces modifications suit le rythme des activités humaines ou du fonctionnement des machines et peut s'exprimer en minutes, en heures ou plus. D'autre part, les temps de réponse des composants sont relativement faibles, de l'ordre de la milliseconde à quelques secondes. Si on souhaite simuler l'évolution d'une telle installation au cours du temps, il faut observer son comportement sur une durée relativement longue mais avec un pas de calcul suffisamment fin pour se rapprocher des temps de réponse des appareils. L'utilisation de logiciels de type Spice dans ce genre de configuration conduit rapidement à des temps de calcul prohibitifs même pour des installations relativement simples.

Bien que les besoins de simulation en électrotechnique soient différents dans leur aspect industriel de ceux des électroniciens, le but de la simulation reste relativement proche, bien qu'elle soit mise en oeuvre différemment. C'est par exemple l'aide au dimensionnement du matériel, la vérification du fonctionnement séquentiel des installations, la simulation de pannes ou de courts-circuits, le contrôle des équipements de protection. Ces besoins sont d'ailleurs renforcés par de nouvelles normes et décrets plus exigeants quant à la fiabilité et à la sécurité des installations.

La simulation en électrotechnique ne concerne pas uniquement les industriels ou les installateurs électriciens. L'apport pédagogique de la simulation est indiscutable et s'applique relativement bien à ce domaine. Les équipements électrotechniques sont relativement coûteux et dangereux étant donné les courants élevés qu'ils utilisent. En utilisant un simulateur, un enseignant peut faire comprendre à ses élèves le fonctionnement de certaines installations sans avoir à disposer du matériel. La simulation permet de se placer dans des situations difficiles à réaliser ou dangereuses pour les personnes ou le matériel. Cet outil peut apporter également un complément pédagogique plus visuel à des filières n'ayant ni le budget ni le temps d'investir dans du matériel.
Il existe des logiciels à vocation purement pédagogique tels que Schémaplic [FIT] ou Crocodile Clips [CRO] qui permettent d'observer le comportement des circuits. Mais ces derniers ne prennent en considération que les aspects logiques du fonctionnement, tels que les états logiques, la continuité du courant, etc. De plus, ils ne proposent qu'une schématique sommaire, très éloignée des standards de la CAO électrique actuelle.

Le travail présenté dans ce mémoire concerne le développement d'un simulateur de circuit électrotechnique et de modèles en vue de leur simulation. Il a débouché sur la création du simulateur « Simul'Elec » commercialisé par l'éditeur de logiciels de CAO électrique Algo'Tech Informatique [ALG].

Les dirigeants d'Algo'Tech Informatique sont présents dans le domaine de la CAO électrique depuis plus de 15 ans. Le marché de la schématique électrique en France est fortement concurrentiel. La démarche d'Algo'Tech Informatique est axée sur le développement de logiciels innovants et performants dans le but de satisfaire à des besoins émergeants ou correspondants à des niches technologiques. Dans la lignée de son logiciel de schématique électrique « Elec'View », Algo'Tech Informatique propose des solutions originales et inédites comme la génération automatique de schémas ou la reconnaissance intelligente de plan papier.
De sa forte connaissance des besoins de ses clients et de son goût de l'innovation, Algo'Tech Informatique a initié, en 1998, le développement d'un prototype de simulateur de circuit électrique SiCi. Fort de l'intérêt suscité auprès des industriels et du corps enseignant, cette voie de recherche a été explorée plus profondément par F. Seyler [SEY99] en 2000. Ce travail de recherche a permis de définir les besoins et les axes de recherche majeurs qui ont été mis en application dans les travaux présentés dans ce document et développés en partenariat avec Algo'Tech Informatique à partir de 2001.

Avec Simul'Elec, nous avons voulu proposer un outil de simulation complet mais simple à manipuler qui puisse être utilisé aussi bien dans un cadre industriel que pédagogique. Son but est de permettre d'étudier le fonctionnement d'un circuit avec des mesures de tension et de courant ainsi que de pouvoir suivre l'évolution séquentielle de son fonctionnement en reproduisant les états logiques des différents composants. Pour cela, nous avons développé un noyau de simulation se basant sur une approche originale et adaptée aux circuits électrotechniques et aux utilisateurs visés. Le simulateur Simul'Elec offre de plus l'avantage d'être intégré à la solution de schématique d'Algo'Tech Informatique et de pouvoir traiter les schémas réalisés en CAO avec son logiciel Elec'View, comme Pspice est intégré à la famille OrCAD.

Le premier chapitre de ce document présente un état de l'art de la simulation dans son application à la simulation de circuits électriques ou de systèmes discrétisés. Il recense certaines techniques et concepts que nous avons mis en œuvre dans la réalisation du simulateur. Nous évoquons notamment les principes de simulation de Spice ainsi que différents outils de simulation existant. Nous traitons également de la simulation par événements discrets dans ses différentes approches et techniques que nous avons exploitées dans l'implémentation de notre mode d'analyse événementielle. Finalement, nous introduisons les notions majeures de la programmation orientée objet, nécessaires à une meilleure compréhension des arguments développés dans les chapitres suivants.

Le chapitre 2 décrit les modes d'analyses de Simul'Elec. Ces trois modes d'analyse, en régime permanent, fréquentiel et événementiel, se démarquent de ceux que l'on rencontre habituellement dans les simulateurs plus orientés vers l'électronique. Nous y justifions nos choix de conception liés au contexte particulier de la simulation de circuits électrotechniques. Nous insistons particulièrement sur le mode d'analyse événementielle pour souligner l'originalité de son approche qui permet de simuler aussi bien les valeurs de tension et de courant dans le circuit que de pouvoir suivre l'évolution séquentielle de l'installation.

Le troisième chapitre traite de l'importance de la modélisation dans ce travail de création d'un simulateur. Nous abordons tout d'abord la démarche de modélisation que nous avons adoptée et qui s'inscrit, comme le développement des modes d'analyse, dans le contexte particulier de l'électrotechnique. Nous mettons en valeur l'emploi de la programmation orientée objet dans

l'implémentation des modèles et en particulier des primitives du simulateur. Nous abordons également le cas de modèles exploités par l'analyse événementielle tels que les modèles de relais et de disjoncteur. Nous discutons aussi de l'intérêt de l'utilisation d'un langage de modélisation et nous justifions notre choix du langage VHDL-AMS. Nous détaillons, enfin, les étapes de notre démarche de traduction de nos modèles en VHDL-AMS.

Ce mémoire s'achève, dans le chapitre 4, par la validation de l'outil que nous avons développé. La validation, nous permet de confirmer la pertinence des modèles de composants et des modes d'analyses. Ce travail important à reçu le soutien de la région Aquitaine à travers le pôle technologique EITICA. La contribution importante de deux élèves ingénieurs N. Depail et H. Abitbol à permis une comparaison et une analyse fine des résultats. Nous confrontons les résultats de Simul'Elec à des mesures effectuées sur des bancs d'essais et à des résultats issus de simulateurs de référence tels que Spice. Nous en tirons des conclusions sur la justesse de nos modèles et les améliorations qu'il faudra leur apporter.

Chapitre 1

Etat de l'art.

Avant l'apparition des logiciels de CAO durant les années 60-70, la conception et la fabrication d'un système passait par la réalisation de prototypes successivement testés jusqu'à la version finale destinée à la production. Le recours à la simulation a permis de prévoir, de vérifier et d'optimiser le comportement d'un circuit ou d'un système en économisant la fabrication coûteuse de prototypes.

Dans un environnement CAO, on dispose de divers outils pour passer du concept à la réalisation des circuits. Ces outils regroupent les interfaces de saisie de schéma, les simulateurs, les interfaces graphiques qui permettent de visualiser les résultats obtenus , les bibliothèques de modèles, les outils de vérification, etc. Le simulateur est un programme qui calcule les caractéristiques électriques d'un circuit dans des conditions de fonctionnement définies par l'utilisateur, et cela à partir de modèles définissant le comportement des composants.

Le simulateur Simul'Elec, but de notre étude, doit pouvoir traiter des circuits électrotechniques en prenant en considération le comportement analogique du circuit (variations de tensions et de courants) ainsi que son évolution séquentielle au court du temps. Nous nous sommes donc intéressés de près aux différentes techniques de modélisation et de simulation classiquement utilisées en électricité ainsi qu'au principe de la programmation objet nécessaire à la création de l'outil de simulation.

Le programme de l'Université de Berkeley, SPICE (Simulation Program with Integrated Circuit Emphasis), est considéré comme un standard en matière de simulation analogique [Nag75]. Nous présenterons donc tout d'abord dans ce chapitre les principes utilisés par ce simulateur, puis un panorama des simulateurs du commerce les plus utilisés aujourd'hui, avec leurs avantages et leurs inconvénients. Enfin nous présentons un panorama des techniques et de concepts que nous avons employés ou auxquels nous faisons référence aussi bien dans le développement des modes d'analyse que dans la modélisation des composants électriques.

1 Simulateurs analogiques de type SPICE.

La plupart des simulateurs analogiques actuels sont basés sur le logiciel SPICE [SP1]. Ce type de simulateur permet de simuler des circuits à base de composants exclusivement analogiques ayant des caractéristiques linéaires ou non. SPICE est aujourd'hui un standard de fait international pour la simulation de circuits électroniques utilisé aussi bien par les universitaires que par les industriels.

1.1 Démarche de résolution.

Le but d'une telle simulation est d'obtenir les valeurs de tous les courants et tensions en tous points d'un circuit électrique. Pour cela, le simulateur dispose d'un modèle du circuit basé sur les modèles de chacun de ses composants et de leurs interconnections. Il exploite ce modèle pour en extraire un système d'équations qu'il sera à même de résoudre.

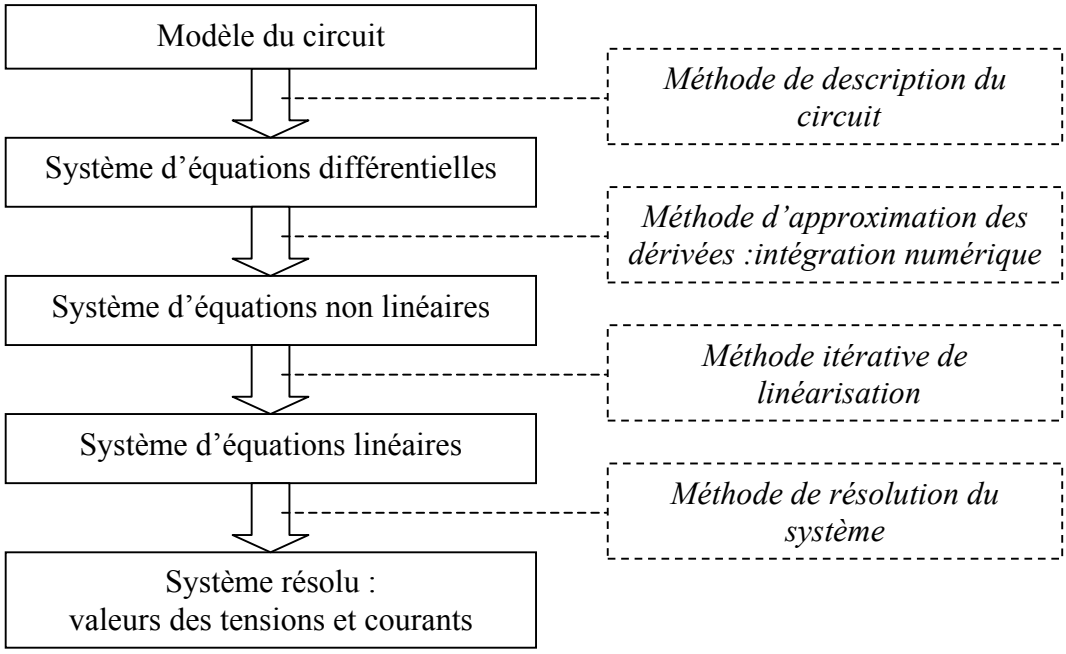


Fig. 1 Démarche de résolution, modèle vers solution.

1.2 Principe de description des circuits.

Un circuit électrique peut être décrit par un système d'équations comportant autant d'équations que d'inconnues (généralement les tensions). Les composants non linéaires sont modélisés par des éléments simples ou par une fonction de transfert qui doivent être intégrés dans ce système. L'approche classique de ce type de problème est l'Analyse Nodale (AN). La méthode utilisée par Spice en est une adaptation pour la description de circuit électrique. On parle alors d'Analyse Nodale Modifié (ANM) [LIT97].

1.2.1 Analyse Nodale.

On peut définir la topologie d'un circuit par une succession de nœuds et de branches :
- Les branches portent les composants et sont connectées à 2 nœuds (1 à chaque extrémité).
- Les nœuds réalisent les connections entre les différentes branches.

Chaque branche est traversée par un courant et chaque nœud est porté à un certain potentiel. Un des nœuds est choisi comme nœud de référence et porte le potentiel 0.

L'approche de l'analyse nodale, consiste à exprimer les relations courants – tensions de chaque branche au moyen les lois de Kirchhoff (loi des mailles et loi des nœuds) et de la loi d'Ohm sous la forme, dans le cas d'une branche résistive :

$$G_{AB}.(v_A - v_B) = i_{AB} \tag{1.1}$$

ou dans le cas de source de courant.

$$I=i_{12}. \tag{1.2}$$

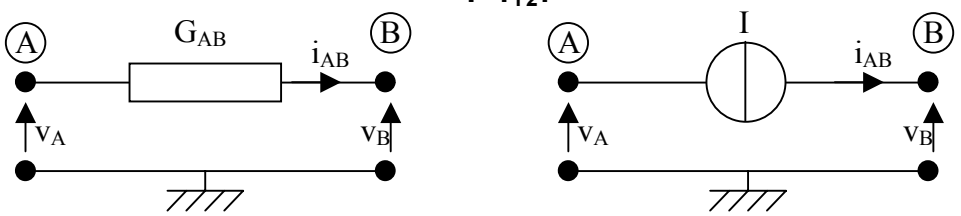


Fig. 2 Expression des relations nœuds – branches : admittance et source de courant

On peut exprimer cette relation sous forme matricielle :

$$G \,.\, v = i \tag{1.3}$$

Soit :

$$\begin{bmatrix} G_{AB} & -G_{AB} \\ -G_{AB} & G_{AB} \end{bmatrix} \times \begin{bmatrix} v_A \\ v_B \end{bmatrix} = \begin{bmatrix} i_{AB} \\ i_{AB} \end{bmatrix} \tag{1.4}$$

Pour chaque composant, on réalise cette description en regroupant les facteurs communs dans 3 matrices globales qui représentera le système d'équations décrivant tout le circuit. Ce système d'équations comporte N équations à N inconnues. N est la dimension des matrices et le nombre de nœuds du circuit –1 car le nœud de référence n'est pas pris en compte (il sert de référence aux autres nœuds). Chaque ligne des matrices correspond à un nœud du circuit.
Ainsi, les courants constituent l'excitation du circuit qui est connue alors que les tensions sont les valeurs à déterminer par la résolution. Toutefois cette méthode de description n'est valide que si on peut décrire les composants à partir d'éléments des matrices G et i, c'est à dire des termes d'admittances et d'excitations (courants).
Pour décrire des sources réelles de tension, il faut les transformer en sources réelles de courant pour quelle soit intégrés comme des excitations.

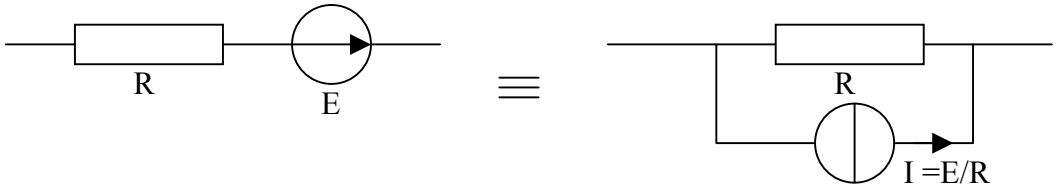


Fig. 3 Source de tension réelle et son circuit équivalent.

1.2.2 Analyse Nodale Modifiée.

Le but de la simulation de circuits est de pouvoir simuler le plus grand nombre de composants différents. L'analyse Nodale permet de décrire correctement des circuits à base de source de courant et d'admittance, mais elle se heurte à quelques difficultés lorsqu'il s'agit de décrire des sources de tensions parfaites.
Une source de tension est décrite par la relation :

$$v_A - v_B = E_{AB} \tag{1.5}$$

On ne trouve aucune variable de courant dans cette équation et on ne peut donc pas formuler la loi des nœuds pour établir le système d'équations.
On a vu précédemment que l'on peut modéliser une source de tension parfaite sous forme de source de courant équivalente à condition de l'associer à une admittance en série. Cela ne peut être fait que si on dispose déjà de cette admittance dans le circuit. Une solution alternative est l'introduction d'un giratoire idéal de rapport de proportionnalité 1.

Définition du girateur idéal :
Bi-porte passif non réciproque conservant la puissance instantanée et tel que la tension instantanée à chaque porte est proportionnelle au courant instantané à l'autre porte. L'impédance vue d'une porte est proportionnelle à l'admittance de fermeture de l'autre porte.
Cela se traduit par les relations électriques suivantes :

$$v1 = K.i2 \quad \text{et} \quad v2 = K.i1 \tag{1.6}$$

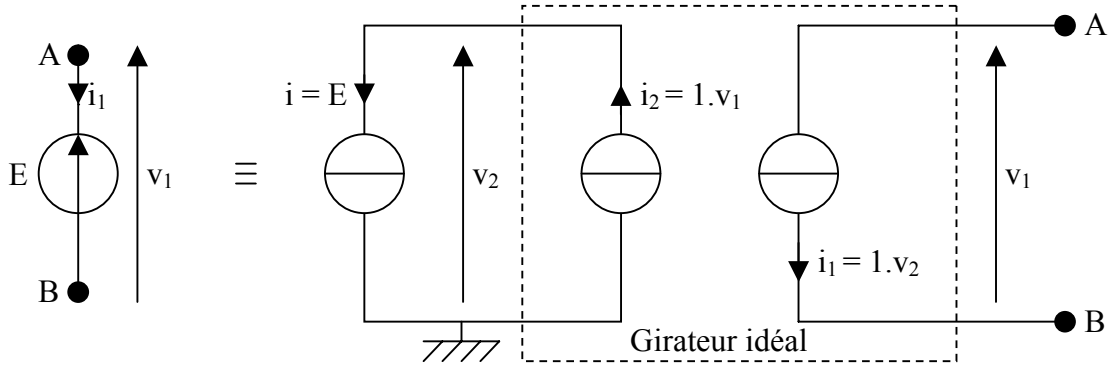


Fig. 4 Equivalence d'une source de tension parfaite et d'un girateur +source de courant .

On peut ainsi, remplacer une source de tension parfaite par ce circuit équivalent ce qui permet d'introduire une nouvelle variable de courant dépendante de la tension de la source.
Cette idée décrite par [LIT97] est appelée analyse nodale modifiée. Elle constitue la méthode la plus couramment employée dans ces programmes.

Suite à cette modification de l'analyse, la structure des matrices se trouve modifiée car on a introduit de nouvelles branches et de nouveaux nœuds. Le système n'est plus de dimension N (N+1 = nombre de nœuds). On passe alors d'un système :

$$G.v = i \tag{1.7}$$

A un système :

$$A.x = z \tag{1.8}$$

où

$$\mathbf{A} = \begin{bmatrix} \mathbf{G} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}, \mathbf{x} = \begin{bmatrix} \mathbf{v} \\ \mathbf{j} \end{bmatrix} \text{ et } \mathbf{z} = \begin{bmatrix} \mathbf{i} \\ \mathbf{e} \end{bmatrix} \quad (1.9)$$

Soit

$$\mathbf{G.v + B.j = i} \quad \text{et} \quad \mathbf{C.v + D.j = e} \quad (1.10)$$

Où :

- **G, v et i** sont respectivement les matrices admittances (dimension NxN), vecteurs tensions (Nx1) et courants (Nx1) que l'on obtient avec une analyse nodale classique.
- **B, C et D** sont respectivement les matrices de dimension NxM, MxN et MxM où M est le nombre de nouvelles variables de courant. Leurs éléments réalisent la liaison entre les anciennes variables tensions des nœuds et variables de courants et avec les nouvelles variables de courants et les nouveaux nœuds.
- **j** est le vecteur (Mx1) des courants des nouvelles branches.
- **e** est le vecteur (Mx1) des excitations dues aux sources de tensions.

Ce principe est applicable non seulement pour décrire une source de tension parfaite mais également pour tous les composants qui nécessitent d'introduire de nouvelles variables de courants (certaines sources contrôlées, transformateurs …).

1.3 Modèles de composants

Pour mettre en œuvre une telle analyse, il faut définir pour chaque composant un modèle sous forme matricielle. Ce modèle permet de déterminer les cases du systèmes de matrices que va modifier le composant. On utilise une représentation sous forme de table appelée tampon qui ne présente que les valeurs que l'on va ajouter aux matrices.

1.3.1 Source de tension parfaite E_{kj}.

On retrouve ici le tampon exprimant les relations courants - tensions résultant de la transformation de la source au moyen d'un gyrateur (fig 3). Sa structure est la suivante :

- Les cases blanches correspondent aux données que l'on va ajouter dans les matrices et les cases grises indiquent leur position.
- Les deux première lignes (A et B) indiquent des cases relatives aux nœuds A et B du composant.
- La dernière ligne (Brch) spécifie qu'une nouvelle branche a été rajoutée afin de pouvoir introduire une variable de courant.
- Les 3 premières colonnes concernent la matrices admittances (nœuds A, B et branche).
- La dernière colonnes représente le vecteur second membre (excitations).

	v_A	v_B	i_1	VSM
A			1	
B			-1	
Brch	1	-1		E_{kj}

Tab. 1 tampon d'une source de tension parfaite.

1.3.2 Source de courant parfaite I_{AB}.

Une source de courant parfaite n'introduit aucun terme d'admittance, uniquement des valeurs d'excitation pour chacun de ses 2 nœuds (courants entrant et sortant).

	v_A	v_B	VSM
A			$-i_{AB}$
B			i_{AB}

Tab. 2 tampon d'une source de courant parfaite.

1.3.3 Admittance.

Ce tampon permet de définis les trois types de composants fondamentaux : résistance, condensateur et inductance en adaptant la valeur de G_{AB} au composant :

- Pour une résistance R : $\mathbf{G_{AB} = 1/ R.}$ (1.11)
- Pour une inductance L : $\mathbf{G_{AB} = 1 / (pL)}$ (1.12)
- Pour un condensateur C : $\mathbf{G_{AB} = pC.}$ (1.13)

	v_A	v_B	VSM
A	G_{AB}	$-G_{AB}$	
B	$-G_{AB}$	G_{AB}	

Tab. 3 tampon d'une admittance quelconque.

1.3.4 Transformateur idéal.

Si m est le rapport de transformation, ce tampon traduit les relations suivantes :

$$\mathbf{V_1 = V_2 / m} \quad \text{et} \quad \mathbf{I_1 = - I_2.m} \tag{1.14}$$

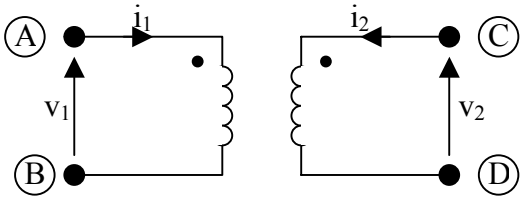


Fig. 5 Transformateur idéal.

	v_A	v_B	v_C	v_D	i_1	i_2	VSM
					1		
					-1		
						1	
						-1	
A	1	-1	-1/m	1/m			
B					1/m	1	

Tab. 4 tampon d'un transformateur idéal.

1.4 Modes d'analyses de SPICE.

Dans chaque mode d'analyse [VLA81], on doit obtenir la solution d'un système d'équations décrivant le comportement d'un circuit. Selon le mode d'analyse et les options retenues par l'utilisateur, on devra calculer plusieurs solutions correspondantes à différentes approximations ou états du circuit. Dans tous les cas, la démarche pour obtenir le résultat en partant du modèle du circuit est la même (**Fig. 1**). La différence entre les modes d'analyse réside dans l'approximation que l'on va faire pour obtenir un système d'équations non linéaire à partir d'un système d'équations différentielles.

1.4.1 Analyse DC-OP : point de fonctionnement.

Le simulateur calcule uniquement les tensions et courants continus. Ce type d'analyse est particulièrement adapté au cas des circuits électroniques. Elle permet notamment de trouver les points de fonctionnement de composants non-linéaires tels que les transistors au moyen de méthodes itératives (Newton-Raphson, relaxation…) . Pour ce mode d'analyse, les matrices ne sont renseignées qu'avec des valeurs réelles. Les selfs sont remplacées par des Courts-circuits et les condensateurs par des circuits ouverts. Cette analyse est notamment utilisée afin de déterminer les conditions initiales d'un circuit pour l'analyse transitoire et avant l'analyse AC (petits signaux) pour calculer les modèles linéarisés de composants non linéaires.

1.4.2 Analyse Balayage DC.

Pour ce type d'analyse, on fait varier une source de tension (ou de courant) continue selon une plage de valeurs et un pas définis par l'utilisateur. Pour chaque point de l'analyse, le simulateur calcule un point de fonctionnement différent (DC-OP) en utilisant une méthode itérative, ce qui en fait une analyse coûteuse en temps de calcul.

1.4.3 Analyse AC : étude fréquentielle linéaire.

Après un premier calcul d'un point de fonctionnement, le simulateur effectue un balayage fréquentiel en petit signaux. Les modèles des composants sont remplacés par des modèles linéarisés correspondant à ce point de fonctionnement. Pour ce type d'analyse, les matrices sont renseignées avec des valeurs complexes qui prennent en compte les notions d'amplitude et de phase. Le simulateur résout le système d'équations pour chaque fréquence de l'analyse en recalculant toutes les admittances des composants liées à la pulsation.
Cette analyse étant linéaire, elle ne nécessite pas de méthode itérative ce qui la rend bien plus rapide qu'une analyse balayage DC ayant le même nombre de points de calcul. Le résultat de cette analyse est généralement une fonction de transfert (gain en tension). Cette analyse permet également de prendre en compte certains bruit, en simulant notamment la génération de bruit blanc par des résistances ou des semi-conducteurs.

1.4.4 Analyse transitoire TRAN.

Le terme d'analyse transitoire est généralement utilisé pour le calcul de la réponse d'un circuit électronique dans le domaine temporel. Cette désignation provient de l'utilisation classique d'une fonction échelon pour caractériser un circuit dans le domaine temporel. Avec ce type d'excitation, le circuit bascule d'un état stable à un autre état stable d'où le terme transitoire. Or c'est un abus de langage que de l'employer pour une excitation arbitraire. On devrait plutôt parler d'analyse dans le domaine temporel mais ce terme est toutefois devenu standard.

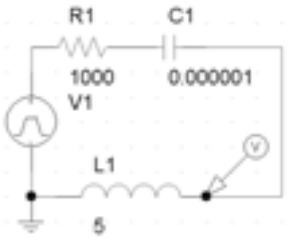
Cette analyse permet d'obtenir l'évolution temporelle de chaque tension et courant sur une durée prédéterminée. L'analyse commence par le calcul d'un point de fonctionnement correspondant à l'instant t= 0 et qui permet de traduire les conditions initiales du circuit. Ensuite, on obtient une série de points à intervalle défini par l'utilisateur afin de constituer une sortie fonction du temps. Pour les points suivants, le traitement d'analyse est différent. On intègre les impédances des condensateurs et des selfs (dépendantes du temps) sous forme de circuits équivalents résistifs . Ces circuits sont décrits par des équations différentielles et sont simplifiés par des méthodes d'approximation des dérivées (voir 1-4) dont la plus courante est « Euler arrière ».

2 Quelques simulateurs de circuit électrique.

Il existe une gamme de simulateurs qui permet de simuler différents types de circuits électriques. Cette offre couvre essentiellement la simulation de circuits électroniques aussi bien analogique que numérique ou mixte. On rencontre également des logiciels dont la vocation est plutôt pédagogique, faciles à utiliser, mais ne présentant pas un intérêt scientifique évident. Avant de démarrer tout travail de développement, nous avons dressé un inventaire des outils de simulation disponibles afin de pouvoir nous donner un ensemble de logiciels existants, de leurs défauts et avantages pour pouvoir en tirer les axes de développement de Simul'Elec.

Nom du logiciel	Développeur	Description
PSpice	Orcad	Simulateur Spice de circuits électroniques analogiques ou numériques http://www.orcad.com/products/pspice/
Psim	Powersim	Simulateur de circuits électroniques de puissance et de contrôle moteur http://www.powersimtech.com/
Winecad	Micrelec	Simulateur Spice de circuits électroniques analogiques ou numériques http://glao.dezai.free.fr/
Simplorer	Ansoft	Simulateur de circuits électroniques et électromécaniques http://www.ansoft.com/products/em/simplorer/
DXP	Protel	Simulateur de circuits électroniques analogiques ou numériques http://www.protel.com/
Tkgate	Jeffery P. Hansen	Logiciel de simulation électronique gratuit, sources libres (license GNU) http://www-2.cs.cmu.edu/~hansen/tkgate/
Electric	Steven M. Rubin	Logiciel de simulation électronique et électrique créé en 1982 par Steven M. Rubin et développé à l'origine sous Linux, sources libres (license GNU) http://www.gnu.org/software/electric/
Schemaplic	Fitec	Logiciel de conception et de simulation de schémas électriques http://www.fitec.fr/interactif.htm

2.1 PSpice (Orcad) :



Pspice utilise le noyau de calcul Spice sous Windows avec Orcad [ORC]. Le module de simulation permet de visualiser les tensions, courants, différences de potentiels en mode temporel et de visualiser les diagrammes de Bode, Black et Nyquist en mode fréquentiel.

Fig. 6 Montage sous PSpice

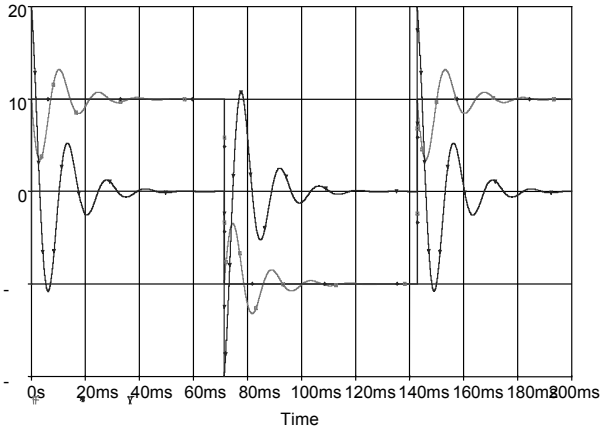


Fig. 7 Analyse temporelle sous PSpice

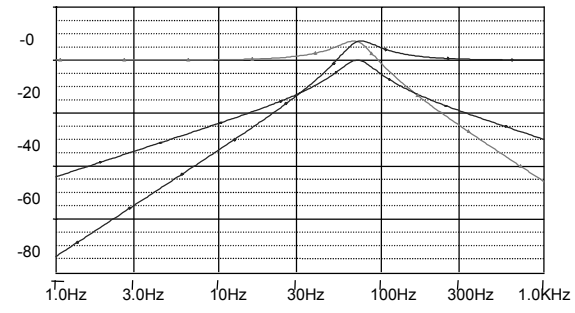


Fig. 8 Analyse fréquentielle sous PSpice

2.2 Psim (Powersim) :

Psim est un logiciel qui possède un noyau Spice, son utilisation est exactement la même que celle de PSpice, ses défauts sont que seule la simulation temporelle est disponible et que l'outil de visualisation des résultats est très rudimentaire.

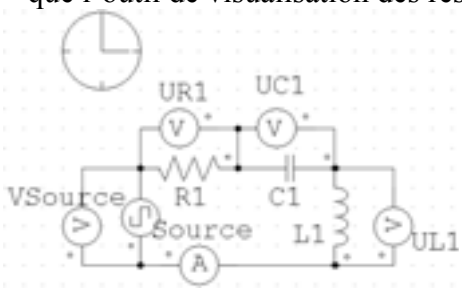


Fig. 9 Montage sous Psim

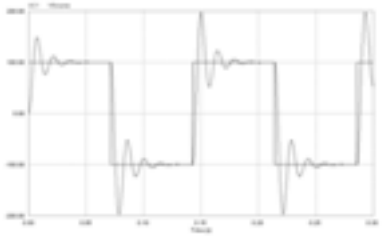


Fig. 10 Analyse temporelle sous Psim

2.3 Simplorer (Ansoft) :

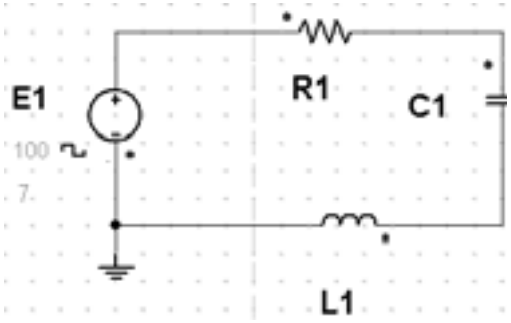


Fig. 11 Montage sous Simplorer

Simplorer [SIM] est un logiciel qui comprend plusieurs modules. Schematic permet de réaliser des schémas de circuits électriques qui peuvent être simulés dans la même feuille. Différents types d'analyse sont disponibles dans ce module (diagramme de Bode ou Nyquist) mais il est également possible d'afficher les valeurs de ces simulations dans un tableau.

La simulation lancée, les résultats s'affichent au fur et à mesure dans les cadres adéquats comme le montrent les figures pages suivantes.

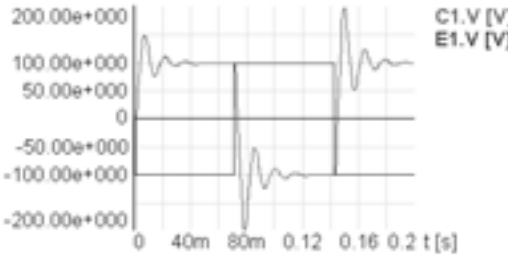


Fig. 12 Analyse temporelle sous Simplorer

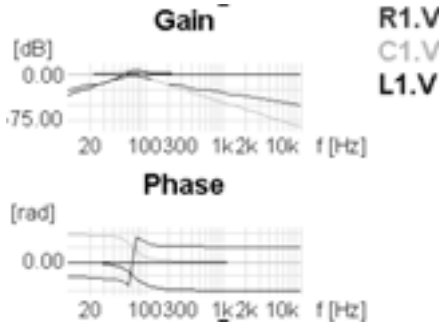


Fig. 13 Analyse fréquentielle sous Simplorer

Pour récupérer les points de calcul des courbes et avoir une meilleure visualisation des résultats, on lance le module "Day" qui permet l'analyse de tous les circuits.
L'exportation de ces résultats pour les visualiser dans un tableur comme par exemple Excel est possible sous le module "Day".

2.4 WinEcad (Micrelec) :

Micrelec rassemble plusieurs logiciels : Winschem (logiciel de DAO), Xsymbol (permet de modifier ou créer des composants), WinEcad (logiciel de simulation).
Sur le modèle de Spice, on visualise une forme générale de courbe, mais les résultats sont difficilement exploitables à cause de l'imprécision des courbes.

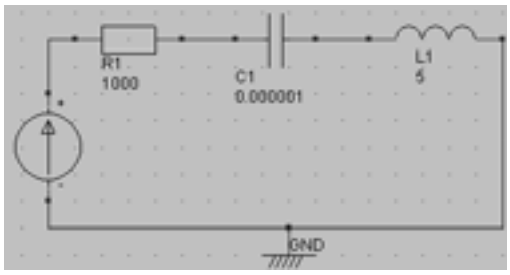


Fig. 14 Montage sous Micrelec

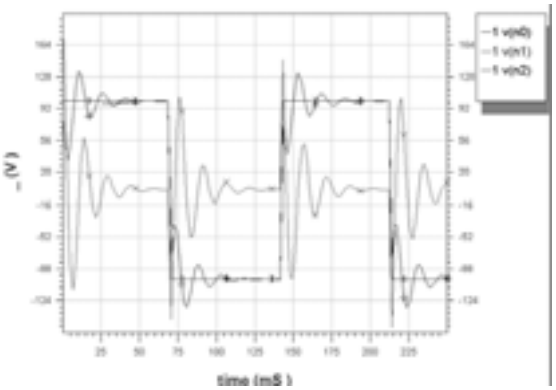


Fig. 15 Analyse temporelle sous Micrelec

2.5 DXP (Protel)

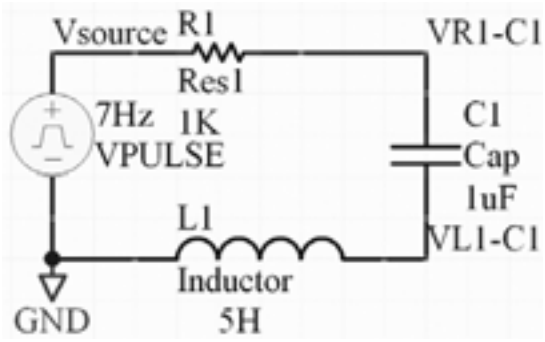


Fig. 16 Montage sous DXP (Protel)

Ce logiciel semble très complet. En effet, il propose une solution intéressante pour la conception et la simulation de circuits électroniques. Il offre une interface graphique conviviale et originale.
Son module de simulation est basé sur Spice3f5 et Xspice.

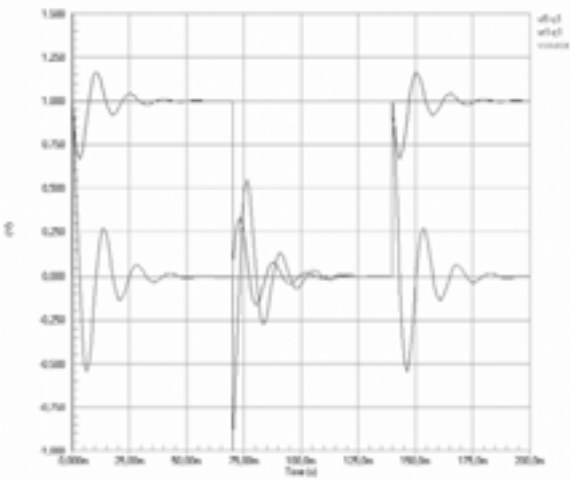


Fig. 17 Analyse temporelle sous DXP (Protel)

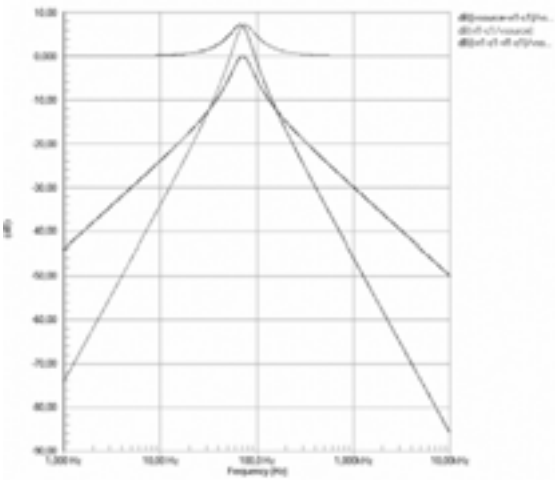


Fig. 18 Analyse fréquentielle sous DXP (Protel)

2.6 Schemaplic (Fitec)

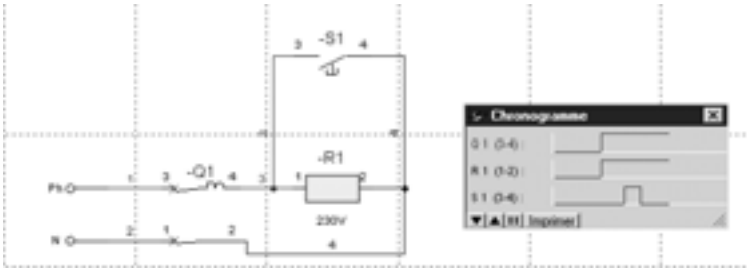


Fig. 19 Montage sous Schemaplic

Schemaplic est un logiciel pédagogique de visualisation d'états logiques et non pas un logiciel de simulation analogique.
Les composants ne peuvent pas être paramétrés, on ne peut pas choisir la valeur des résistances par exemple. Il est impossible de crée un court circuit sans que le logiciel n'affiche un message d'erreur.

3 Simulation par Evénements Discrets.

La SED traite des instants particuliers du temps en le discrétisant. On considère qu'entre deux instants successifs du temps discret l'état du système n'évolue pas. De nombreux simulateurs sont basés sur ce principe réducteur et obtiennent des résultats satisfaisants. Il existe trois approches qui correspondent chacune à un point de vue sur le système simulé. Nous nous sommes intéressés plus particulièrement aux deux plus répandues : l'approche orientée événements dont nous nous sommes inspirés et l'approche orientée processus qui est utilisée dans de nombreux simulateurs électroniques mixtes et dont nous avons intégré certains principes dans la solution que nous proposons dans le Chapitre 2**.**

3.1 Approches de la simulation par événements discrets.

Pour la simulation par événements discrets, un système va être constitué d'objets, d'acteurs et d'activités (ou tâches) [ERA96]. Les acteurs réalisent certaines activités pendant lesquelles ils manipulent les objets. L'activation des tâches est définie par des événements. Les processus d'un système correspondent à ses différentes tâches actives. L'activité d'une de ces tâches peut être dépendante de l'activité d'une autre comme dans une relation client serveur.
Pour prendre un exemple simple, nous pouvons considérer une station de lavage de voiture. Les objets manipulés sont les véhicules, les acteurs sont les différentes machines de la station de lavage et son personnel. Les activités sont les différentes opérations que chaque machine ou opérateur peut réaliser. Pour simuler un tel système, on peut prendre différents points de vue en fonction du type d'élément (objet, acteur ou activité) que l'on place au centre de la simulation.

Il existe donc 3 approches majeures qui correspondent chacune à ces différents points de vue sur le système à simuler [GAR01] :

- Balayage des activités.
- Planification d'événements.
- Interaction de processus (tâches actives).

Les différences entre ces approches sont liées à la manière dont on va décomposer le système en différents blocs qui représentent l'unité conceptuelle de la simulation. Le tableau **Tab. 5** ci-dessous résume les principales nuances existant entre ces techniques.

Approche de simulation	Elément traité	Unité conceptuelle de la simulation	Actions liées	Exécution de l'action
Balayage des activités.	Activité	Activité	Séquence d'activité	Conditionnelle
Planification d'événements.	Objet	Evénement	Notice d'événement	Selon les événements planifiés
Interaction de processus.	Acteur	Processus interactifs	Tache active	Activation du processus

Tab. 5 Différentes approches de la simulation par événement discret.

Ces concepts peuvent paraître relativement proches, car ils permettent chacun de simuler un système constitué des mêmes éléments (activité, acteur et outils). C'est essentiellement leur implémentation qui utilisera des techniques informatiques différentes. Dans la pratique, on rencontre presque uniquement les deux dernières techniques que l'on qualifiera de simulation orientée événement ou orientée processus.

3.2 Balayage des activités

Dans cette approche, l'élément clé de la simulation est l'activité. Le modèle d'un système est décomposé en un ensemble d'activités qu'il peut réaliser à chaque événement de la simulation. L'action attachée à une activité est une séquence d'activité. Celle ci sera exécutée si certaines conditions sont remplies. Ainsi pour chaque événement, les conditions de réalisation de chaque activité sont testées en fonction des variables d'état du système. Si l'ensemble des conditions est vrai, l'action est réalisée, comme on peut le vérifier en **Fig. 20**.

3.3 Planification d'événements.

L'approche par planification d'événement est traitée par des simulateurs dit **orientés événements**. L'unité conceptuelle du système est alors l'événement. Une routine d'événement est associée à chaque type d'événement et décrit les actions qui doivent être réalisées. La réalisation de ces événements peut provoquer par la suite la planification de nouveaux événements. Ici, le point de vue est centré sur les objets manipulés par les acteurs du système. Un échéancier gère la planification et l'activation séquentielle de chaque événement. Lorsque l'action liée à un événement est réalisée, le temps de la simulation est suspendu. Le principe de cette approche dite orientée événement est résumé en **Fig. 21**.

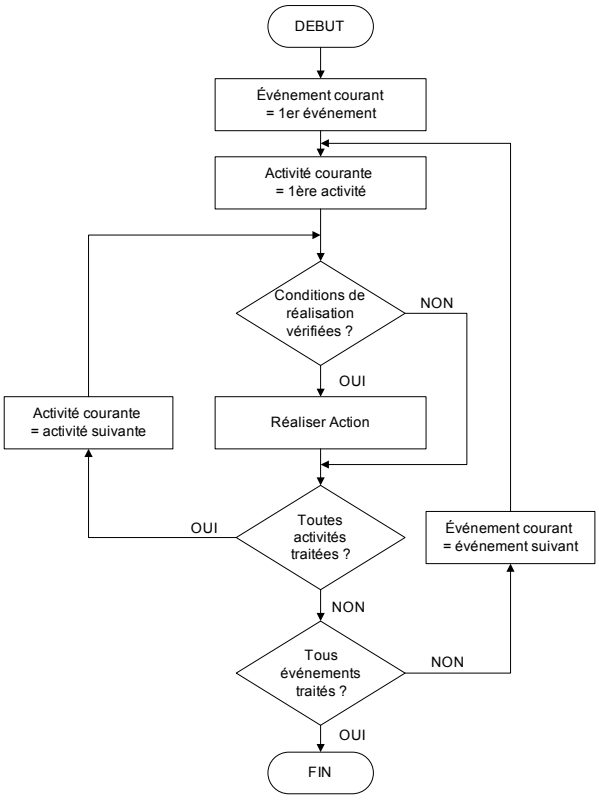


Fig. 20 Simulation par balayage des activités

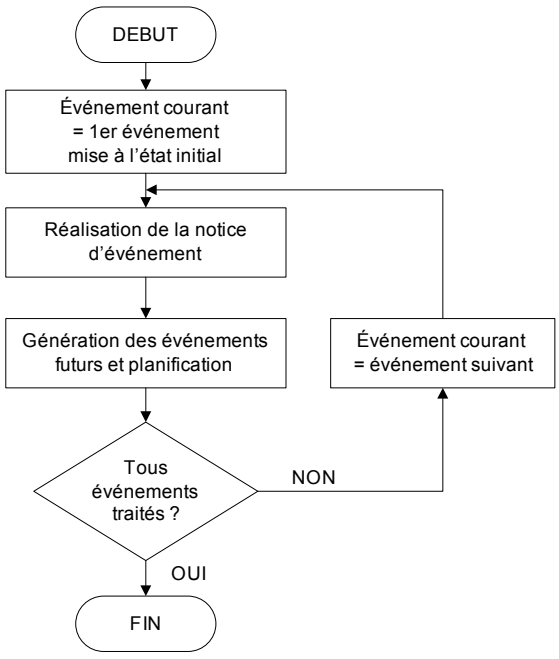


Fig. 21 Simulation par planification des événements

Cette approche repose sur un principe de file d'attente. Cette file va contenir les événements et les organiser en fonction de leur date de réalisation. Cela constitue un calendrier d'événements. Le premier événement se déroule au temps t=0. Il modifie l'état du système simulé pour le mettre dans son état initial. De nouveaux événements peuvent être planifiés traduisant la réaction de chacun des acteurs à ce nouvel état du système. Ensuite la date du simulateur jusqu'au prochain événement afin de réaliser la notice d'événement associée. Lorsque les actions planifiées dans cette notice ont été réalisées, on obtient un nouvel état du système qui provoque la planification de nouveaux événements futurs. L'avance du temps reprend alors jusqu'au prochain événement et ainsi de suite jusqu'à arriver à la fin du temps alloué à la simulation.

3.4 Interaction de processus.

On considère ici que le système est constitué de plusieurs tâches actives qui peuvent s'exécuter simultanément. Chaque tâche est représentée par un processus qui interagit avec les autres. La simulation est alors qualifiée d'orientée processus. Les tâches sont exécutées par ces processus qui sont activés ou désactivés par un échéancier. Contrairement aux approches précédentes, ce ne sont pas les actions ou les objets manipulés qui sont au centre de la simulation. En effet, chaque processus représente un acteur du système. Lorsqu'il est activé par l'échéancier, il réalise sa tâche jusqu'à ce qu'il soit désactivé par un autre événement. Ici la notion d'événement existe aussi et indique une intervention sur un ou des processus (activation, suspension…) ou l'échange de message. Contrairement à l'approche par événement, le temps de simulation continue de se dérouler pendant l'exécution de la tache. Ainsi plusieurs processus peuvent être actifs en même temps et fonctionner en parallèle. Ce principe est plus naturel et plus réaliste qu'une modélisation séquentielle du comportement. Il est généralement mis en œuvre par une programmation de processus parallèles ou répartis. C'est un principe qui s'applique particulièrement bien à la simulation de tout type de flux (personnes, véhicules, pièces d'assemblage, etc.). Son implémentation est aujourd'hui grandement facilitée par l'utilisation de langage de programmation orienté objet (C++, Java, Delphi…) ou de simulation. C'est l'approche qui est la plus utilisée en raison de son réalisme plus élevé et de performance plus grande pour la simulation de système qui ne sont pas séquentiels. Elle est notamment employée pour l'implémentation de nombreux simulateurs électroniques mixtes dont Saber [SAB].

3.4.1 Processus logiques et physiques.

Nous avons affaire ici à deux notions distinctes de processus suivant que l'on considère l'activité d'un système ou un processus informatique qui intervient en programmation parallèle. Ainsi Erard et al. [ERA96] définissent comme processus logique une tâche active d'un système, alors que les composantes logicielles d'un système réparti sont des processus physiques. Les processus logiques fonctionnent de manière autonome mais communiquent avec les autres processus au travers de messages.

3.4.2 Gestion et états des processus logiques.

Un processus logique peut prendre 4 états fondamentaux. Ces états dépendent de leur positionnement dans l'échéancier. Les processus peuvent être actifs, différés, passifs ou terminés [ERA96].

Etat	Description
Actif	Son heure d'activation est identique à celle du système. Il réalise sa tâche.
Différé	En attente de l'activation par l'échéancier.
Passif	Suspendu dans l'attente d'un message d'activation d'un autre processus.
Terminé	La tâche active est terminée.

Tab. 6 Etat des processus logiques.

Pour gérer le passage entre les différents états des processus, le simulateur doit pouvoir permettre les actions de création, activation, réactivation, suspension, passivation et annulation.

Action	Description	Etat résultant
création	Crée un nouveau processus et mise en file d'attente.	Passif
activation	Rend actif un processus immédiatement ou après un délai.	Actif
réactivation	Modifie l'heure d'activation d'un processus existant.	Différé
suspension	Suspend un processus actif jusqu'à une date ultérieure.	Différé
passivation	Interrompt l'activité d'un processus en attente d'une activation par un autre processus.	Passif
annulation	Interrompt la tache active.	Passif

Tab. 7 Action de modification de l'état des processus.

3.4.3 Synchronisation des processus.

Quand plusieurs processus travaillent en même temps, ils réalisent des tâches d'une manière autonome les uns des autres. Ils doivent toutefois conserver des liens entre eux car leurs tâches peuvent être dépendantes. Il est nécessaire de communiquer des informations afin de maintenir des données partagées dans un état cohérent [ERA96]. Par exemple, deux processus peuvent traiter un même objet alternativement et doivent pour cela savoir quand l'objet est pris par un processus et quand il est rendu disponible. Dans certains cas, plusieurs processus peuvent réaliser une tâche conjointement. On parle alors de coopération synchrone [GAR01].

Bien qu'ayant décomposé le système en tâches autonomes, les processus qui les réalisent ont donc nécessairement besoin d'échanger des messages et de se synchroniser. Cette synchronisation constitue une difficulté de ce type de simulation car les différentes techniques doivent faire un compromis entre la performance du simulateur et la limitation des risques d'erreurs. Chaque processus logique peut être implémenté par un programme séquentiel qui va interagir avec les autres à des instants donnés (événements). Les techniques de synchronisation retenues dépendent notamment du principe de programmation et de l'architecture matérielle choisie comme le résume le tableau **Tab. 8** ci dessous.

Système	Architecture matérielle	Répartition des tâches	Note
Co-routines	1 processeur	1 seule co-routine active à la fois	Changement de tâche par une activation explicite de la co-routine
Commutation de processus	1 processeur	Plusieurs tâches simultanément en attente	Un gestionnaire de tâche définit les priorités et arbitre la répartition des ressources
Réparti	Plusieurs processeurs ou ordinateurs	Tâches réparties sur les différents processeurs	Il faut gérer des problèmes d'environnement différents (systèmes d'exploitation, performances) et le temps de transmission des messages

Tab. 8 Gestion des processus selon différents systèmes.

Le choix entre ces différents systèmes dépendra de l'architecture matérielle retenue (un ou plusieurs processeurs et ordinateurs) ainsi que des instructions et fonctions supportées, les co-routines et la gestion des processus physiques (ex :Objets thread en Java ou Delphi) notamment.

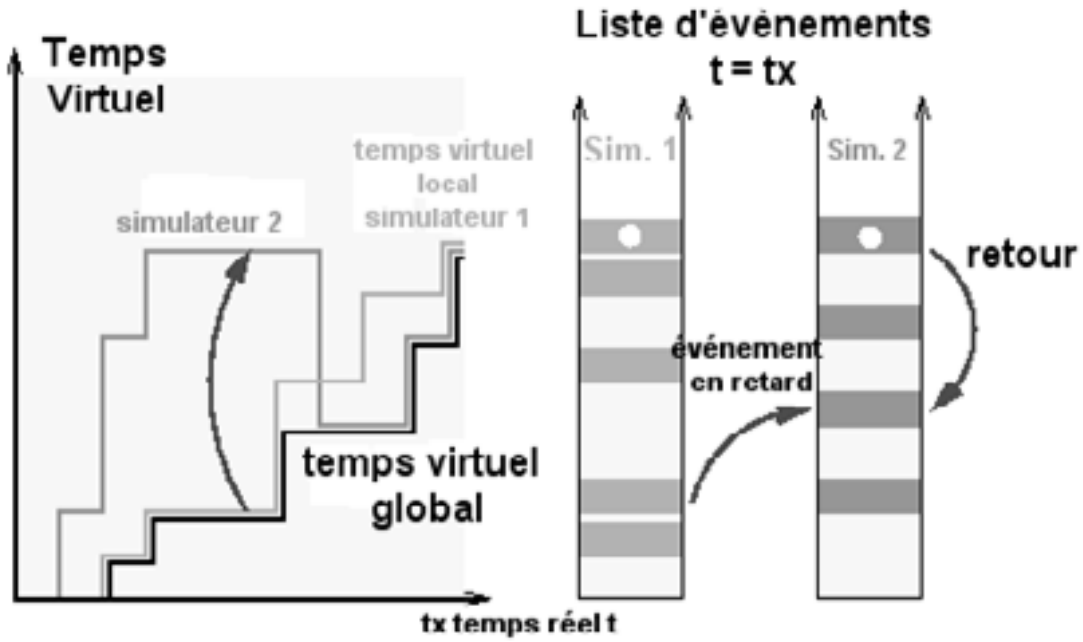
3.4.4 Synchronisation sûre, risquée et erreur de causalité.

Les différentes tâches de chaque processus peuvent être plus ou moins coûteuses en temps de calcul. Un processus dont la tâche est simple la traitera rapidement et pourra passer à une autre tâche. Ainsi le temps virtuel de la simulation y évolue rapidement par rapport au temps de calcul. De même, pour un processus qui réalise une tâche très complexe, le temps de traitement sera plus long et cela ralentira l'avancement du temps virtuel pour ce processus. On se retrouve alors avec deux processus n'ayant pas d'horloge commune et au sein desquels le temps avance à une vitesse différente. Si ces processus sont dépendants (un client et un serveur par exemple) il est nécessaire qu'ils puissent échanger des informations pour se synchroniser.
Un des points les plus délicats consiste à définir quels sont les instants de la simulation ou la synchronisation doit se faire. Si on synchronise trop fréquemment, le simulateur va être rendu moins performant par une prolifération de messages superflus. Dans le cas contraire, si on ne synchronise pas suffisamment les processus, on perd de l'information et le simulateur peut alors produire des résultats erronés. Le principal risque est l'erreur de causalité. Elle se produit lorsqu'un processus réalise une tâche à partir d'une date donnée alors que d'autres tâches qui doivent être réalisées à une antérieure mais que leur message d'activation arrive ultérieurement. C'est notamment le cas lorsqu'un processus est plus rapide qu'un autre et manque de ce fait un message de synchronisation que le processus lent n'a pas encore délivré.

Deux approches majeures reflètent ces options de réalisation. L'approche sûre ou conservative privilégie la justesse des résultats au détriment de l'efficacité du simulateur. L'approche risquée ou optimiste va s'orienter davantage vers la rapidité d'exécution en incluant des mécanismes de suppression d'erreur.

Pour résumer, l'approche sûre multiplie les messages entre les processus afin de ne pas manquer de synchronisation. Ainsi certains processus se suspendent dans l'attente de message d'un autre processus avec lequel il interagit. On va éviter alors toute erreur de causalité. Par contre on peut se trouver dans des situations bloquées où plusieurs processus sont suspendus et en attente d'une réponse de l'autre. Cette impasse peut être éliminée par des mécanismes de détection.

L'approche risquée implémentée par des techniques de distorsion du temps (time warp **Fig. 22**) [IED] permet à chaque processus logique de réaliser sa tâche sans se préoccuper des autres. Le simulateur détectera les erreurs de causalité qui amènera l'annulation des traitements réalisés après cette erreur. Le processus « en avance » fait alors un retour en arrière (Roll back) qui permet d'effacer ces erreurs et de reprendre sa tâche à partir d'un état correct du système. De nombreuses techniques, adaptées aux différents domaines de simulation, réalisent cette synchronisation plus ou moins efficacement. La difficulté est ici de détecter suffisamment tôt, voire d'anticiper l'erreur de causalité afin que le retour en arrière n'annule pas le bénéficie qu'apporte l'approche risquée en terme de performance.



Source www.eda.ei.tum.de

Fig. 22 Technique de synchronisation risquée par distorsion du temps.

3.5 SED : Outils et techniques de développement

Quelque soit l'approche de SED retenue, on peut utiliser différents outils et techniques développement. Nous décrivons ici des outils spécialisés comme des langages et des paquets logiciels pour la SED. Une solution alternative consiste à utiliser un langage à usage général (comme C++, Java, Delphi …) sans l'aide des ces outils. Nous présentons donc ces deux possibilités en comparant les avantages qu'elles peuvent présenter.

3.5.1 Langages et paquets de simulation.

Un paquet logiciel (ou package) de simulation est une bibliothèque de fonctions (programmation structurée) ou de classes (programmation orientée objet) à destination d'un langage de programmation à usage général. Le développeur de l'application exploitera alors ces fonctionnalités prédéfinies pour réaliser son simulateur.
Les langages de simulation qui sont généralement dérivés de langages de programmations plus classiques (ADA, FORTRAN, C, Java…). L'emploi d'un langage de simulation amène un niveau d'abstraction et une implémentation des simulateurs plus simple qu'avec un langage classique grâce à sa spécialisation.
Il existe de nombreux paquets logiciels ou langages de simulation. Ils permettent à l'utilisateur de réaliser un logiciel de simulation par événements discrets. Pour cela, ils présentent un certain nombre de fonctionnalités requises pour la plupart de ces simulateurs :

- Génération de nombres et de valeurs aléatoires ou selon des distributions statistiques.
- Gestion de l'avance des temps.
- Détermination du prochain événement.
- Ajout et suppression d'enregistrements à une liste.
- Collecte et analyse statistique des données.
- Report des résultats.
- Détection d'erreur.

Ces langages ou paquets de simulation traitent les approches orientées événement et/ou processus. On peut toutefois très bien implémenter un simulateur par événements discrets sans utiliser ni langage ni paquet de simulation.

Le tableau **Tab. 9** ci dessous synthétise l'inventaire des paquets et langages de simulation dressé par Garrido [GAR01].
Notations :
(1) Approche : A (orientée activité), E (orientée événement) et P (orientée processus).
(2) Paquet (P) ou Langage (L)

	Approche possible (1)	Paquet, Langage (2)	Langage de base	Note, public	Auteur
SimPack	E	P	C++	Chercheurs, enseignants	P. Fishwick
Sim++	E	P	C++		P. Fishwick
CSIM	P	P	C++	Programmeurs	Mesquite software
ModSim	P	L	Modula2		Armée US
GPSS	P	L		Tout usage	IBM
Simula	E, P	L	Algol	Premier langage orienté objet	
SimScript	E, P	L		Usage general	CACI
GASP	E, P	P	Fortran		P.J. Kiviat
SLAM	A, E, P	L	Fortran	Combinaison des approches possible	
Task library	P	P	C++	Environnement UNIX	
Psim	P	P	C++		
Psim-J	P	P	Java		
PSimL	P	L	C++		

Tab. 9 Principaux paquets et langages de simulation disponibles.

Cette liste n'est pas exhaustive mais souligne déjà la multiplicité des solutions possibles. La majorité des langages ou paquets sont basés sur les langages C++, Fortran ou Java. Notons que Law [LAW91] propose d'autres implémentations de simulateurs pilotés par événements en langage classique C, Fortran ou Pascal n'utilisant pas de paquet de simulation.

3.5.2 Choix entre un langage classique ou un langage de simulation.

L'emploi d'un langage à usage général sera par définition moins adapté que celui d'un langage ou d'un paquet de SED. Il présente toutefois de multiples avantages justifiant que l'on s'intéresse à cette option. L'utilisation d'un paquet ou d'un langage pour la SED n'est pas inévitable et peut se révéler mal adapté dans certains cas.

Les tableaux **Tab. 10** et **Tab. 11** reprennent la synthèse de Law [LAW91] sur les avantages que présentent ces deux options sans toutefois les comparer directement. Le choix entre ces deux solutions dépend des fonctions finales attendues. De plus, pour une implémentation plus sûre, on privilégiera un langage de simulation, qui restera moins souple qu'un langage classique. L'emploi d'un paquet logiciel représente une option intermédiaire mais assez proche de l'utilisateur d'un langage de simulation. En effet, il permet d'utiliser un langage standard qui peut être déjà connu par l'utilisateur en intégrant des fonctions prédéfinies pour la SED.

Langages et paquets de simulation
Les fonctions nécessaires sont intégrées : réduction du temps de programmation
Environnement de travail spécialisé pour le développement de simulateur
Modifications plus simples des modèles
Meilleure détection d'erreurs car les différents types d'erreurs sont déjà identifiés.

Tab. 10 Avantages des paquets et langages de simulation

Langage de programmation classique
Langages plus répandus, peut être déjà connu par l'utilisateur.
Langage disponible sur plusieurs types de plateformes (Fortran, C, Java…).
Un programme bien écrit peut être plus efficace car mieux spécialisé.
Plus grande souplesse dans la programmation.

Tab. 11 Avantages des langages classiques

3.6 Mécanismes d'avance temporelle.

En SED, par définition, le temps est discrétisé. Il est constitué d'un nombre fini de points précis du temps qui constituent l'horloge de la simulation. Comme le temps ne se déroule pas de manière continue mais par saut d'un instant au suivant, un des rôles essentiels d'un simulateur piloté par événements est de gérer l'avance du temps quelque soit l'approche retenue.
Law [LAW91] décrit les deux techniques principales dont les tâches seront de piloter l'horloge du simulateur et traiter le prochain événement :

- Avance à pas fixe.
- Saut au prochain événement.

Pour ces deux méthodes, on n'a généralement pas de relation entre le temps virtuel qui s'écoule selon l'horloge de la simulation, et la durée effective des opérations que réalise le simulateur. Ceci pose un certain nombre de problèmes de synchronisation lorsque l'on utilise une modélisation orientée processus.

Le choix entre ces deux techniques sera essentiellement déterminé par la périodicité des comportements à modéliser.

3.6.1 Avance temporelle à pas fixe.

Avec l'avance à pas fixe, le simulateur dispose d'une horloge dont la période est déterminée. Elle est avancée d'un pas constant Δt. A chaque instant, le simulateur traite les événements qui ont pu être planifiés. Si des événements se produisent pendant l'intervalle Δt, il faudra attendre la fin de l'intervalle pour qu'il soit pris en compte par le simulateur. Cette solution présente plusieurs inconvénients :

- L'attente de fin de l'intervalle peut provoquer des erreurs.
- Lorsque plusieurs événements sont inclus dans un intervalle, il faudra décider à la fin quel sera l'ordre de priorité.
- Le simulateur va prendre en compte des instants pour lesquels il ne peut y avoir aucun événement.

Cette technique est toutefois intéressante si on sait que les événements se produisent toujours à intervalle régulier (multiple de Δt). Elle sera d'autant plus efficace si on trouve un événement pour chaque coup de l'horloge. On peut l'appliquer à des systèmes périodiques comme par exemple des simulations en économie avec une période de 1 an.

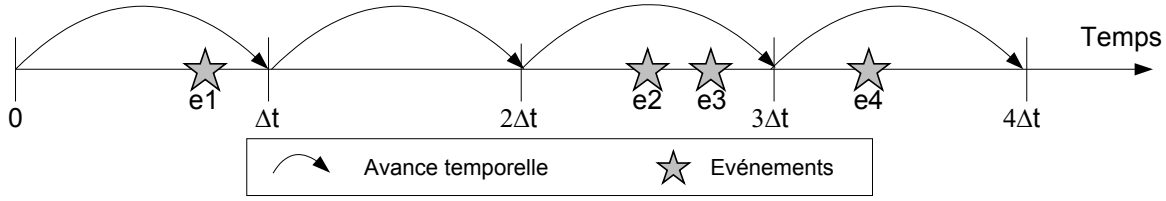


Fig. 23 Exemple d'avance temporelle à pas fixe

3.6.2 Avance temporelle par saut au prochain événement.

La technique par saut au prochain événement ne considère pas de période particulière dans l'arrivée des événements. C'est la plus couramment utilisée dans les simulateurs par événements discrets. Comme son nom l'indique l'horloge avance par saut d'un événement au prochain. L'avance est alors irrégulière. Pendant l'intervalle séparant deux événements, le simulateur ne traite pas le temps. Il ne s'occupe que des instants utiles. Ainsi, son implémentation se trouve plus efficace qu'une avance à pas fixe, aussi bien en terme de rapidité que d'occupation de la mémoire.

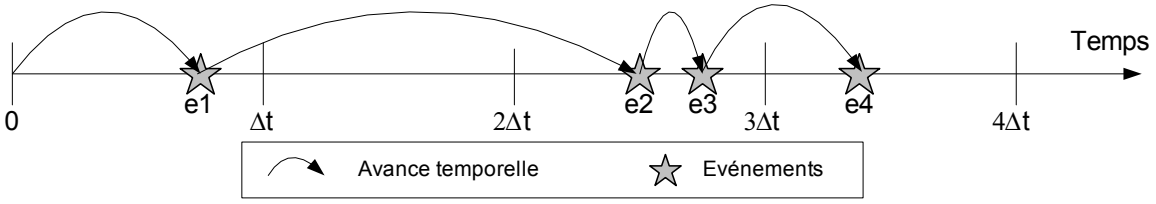


Fig. 24 Exemple d'avance temporelle par saut au prochain événement

4 Méthodes de synchronisation en simulation mixte.

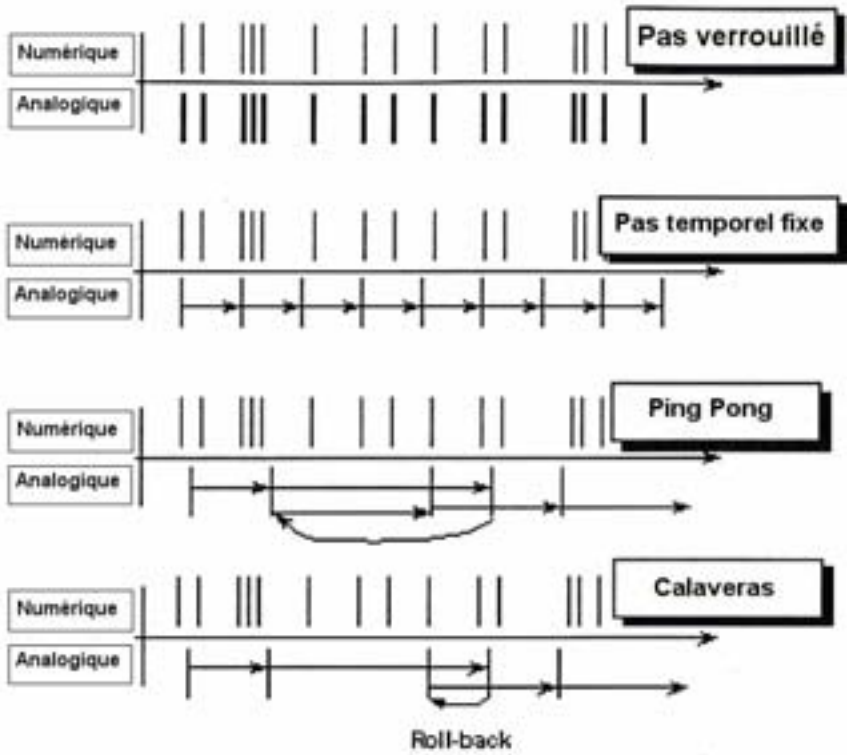
Dans les simulateurs mixtes analogique – numérique, on retrouve deux algorithmes pour la résolution analogique et numérique. Il est toujours nécessaire de synchroniser ces deux algorithmes pour qu'ils puissent échanger des informations de façon efficace et en minimisant le risque d'erreur. Chaque échange d'informations va provoquer des modifications des matrices de description de la partie analogique ayant pour conséquences de résoudre à nouveau ce système. Il existe différentes techniques de synchronisation qui sont des compromis entre la rapidité de la résolution et la précision du résultat. Quatre des plus courantes sont : Pas verrouillé, pas temporel fixe, ping-pong et Cavaleras.

4.1 Pas verrouillé.

Cette technique synchronise la simulation analogique à chaque événement de la simulation numérique. C'est l'un des premiers principes employés avec les simulateurs mixtes. Elle est assez grossière et synchronise systématique, que des données doivent être transférées ou non. C'est donc très coûteux en temps de simulation car pour chaque événement on fait une nouvelle résolution du système. Elle n'est utilisée que par peu de simulateurs.

4.2 Pas temporel fixe.

La synchronisation à pas temporel fixe utilise un fond de panier qui gère et déclenche les échanges de données à intervalles réguliers. Cette méthode a pour inconvénients de perdre les événements logiques entre deux intervalles et d'introduire des pas temporels supplémentaires qui n'ont pas lieu d'être et augmente inutilement le temps de simulation.



Source : www.analogy.com

Fig. 25 Techniques de synchronisation d'algorithmes analogique et numérique

4.3 Ping-pong.

Pour être le plus efficace possible, aussi bien en rapidité de simulation que pour la justesse des résultats, il est nécessaire que la synchronisation n'intervienne que lorsque l'échange de données est absolument nécessaire. De plus, les algorithmes analogiques doivent travailler indépendamment et avec chacun un pas de calcul optimisé.
Pour être certain que les échanges d'informations arrivent au moment opportun, on permet aux deux simulateurs de déclencher un échange, d'où le terme de ping-pong.
Comme les deux algorithmes fonctionnent indépendamment, l'un d'eux sera forcement plus rapide que l'autre et peut continuer ses calculs en manquant une synchronisation parce que le second algorithme n'est pas encore parvenu à cet instant. Lorsqu'il y arrive, le 1er doit revenir en arrière pour que la synchronisation ait lieu, afin de préserver la justesse des résultats.

4.4 Calaveras.

Cette méthode est assez proche de la précédente et est déposée par la société Analogy et est utilisée dans le simulateur Saber [SAB]. La différence par rapport à la méthode ping-pong réside dans le fait qu'elle enregistre le résultat de la précédente résolution analogique évitant ainsi de la recalculer. Elle privilégie ainsi la vitesse de simulation au détriment de l'occupation de la mémoire.

5 Notion de programmation orienté objet.

Notre objectif est de réaliser un simulateur de circuits électrique adaptés aux domaines de l'électrotechnique et des courants forts. Cela passe par la modélisation des comportements des différents composants mais aussi par l'implémentation de l'outil de simulation. La création de cet outil doit répondre à des contraintes définies par notre partenaire Algo'Tech Informatique. Le logiciel de simuler doit s'intégrer dans l'application de schématique existante. Pour favoriser l'efficacité, la réutilisabilité et la maintenabilité du code, il est impératif d'utiliser les techniques de programmation orientées objets.
Le langage de développement utilisé est Delphi (Pascal Objet). Lors de la création du prototype du simulateur en 1999, Algo'Tech travaillait avec la version 2 de Delphi qui ne supportait pas la totalité des concepts de la programmation orientée objet. Au cours des développement, nous avons fait évoluer le code du simulateur avec la progression de Delphi, de la version 2 à la version 7 en 2003 qui couvre désormais la presque totalité des concepts de la programmation orientée objets.

Les notions attachées à ce type de programmation sont utilisées aussi bien dans la description des modes d'analyse réalisé que dans la modélisation des composants. C'est pourquoi il est essentiel de définir ses concepts et son vocabulaire (Source Nadine Couture - ESTIA).

5.1 Les objets

Les caractéristiques fondamentales d'un objet sont : l'état, le comportement, l'identité.

5.1.1 Définitions.

Un **attribut** est une information qui qualifie l'objet qui le contient. Un attribut est à valeur dans un domaine de définition donné. **L'état**, regroupe les valeurs instantanées de tous les attributs d'un objet.
Le **comportement** regroupe toutes les compétences d'un objet et décrit les actions et les réactions de cet objet.
L'identité permet de distinguer tout objet de façon non ambiguë quelque soit son état.

Les objets communiquent en échangeant des **messages**.

Il existe 5 catégories de messages :
les **constructeurs**, qui créent les objets.
les **destructeurs**, qui détruisent les objets.
les **sélecteurs** qui renvoient tout ou partie de l'état d'un objet
les **modificateurs** qui changent tout ou partie de l'état d'un objet
les **itérateurs** qui visitent l'état d'un objet ou le contenu d'une structure de données qui contient plusieurs objets.

5.1.2 Exemple d'objet : MonVelo.

Dans le cas d'un objet *MonVelo*, nous pouvons exprimer ces attributs et ses méthodes :

Attributs :
- Couleur = jaune
- Poids = 8 kg
- Nombre de pignons = 5
- Pignon actuel = 3e
- Cadence de pédalage = 30 Tr/min
- Vitesse actuelle : 27 km/h

Méthodes :
- Changer de pignon (+1 ou -1)
- Afficher vitesse actuelle
- Tourner (droite ou gauche)
- Augmenter cadence (nouvelle cadence)
- Freiner
- Calculer vitesse

Remarque : pas de méthode pour augmenter directement la vitesse. Elle est fonction de la cadence et du pignon actuel. La méthode *Afficher vitesse* appelle *Calculer vitesse* avant d'afficher le résultat.

5.2 Les classes.

Pour maîtriser la complexité, on regroupe les objets qui se ressemblent. La classe décrit le domaine de définition d'un ensemble d'objets. Chaque objet appartient à une classe. Les généralités sont contenues dans la classe. Les particularités sont contenues dans l'objet.

5.2.1 Exemple de classe Bicyclette.

attributs :
- Couleur : couleur
- Poids (kg) : réel
- Nombre de pignons : entier
- Pignon actuel : entier
- Cadence de pédalage (tours/min) : entier
- Vitesse actuelle (km/h) : réel

méthodes :
- Changer de pignon (+1 ou -1)
- Afficher vitesse actuelle
- Tourner (angle)
- Augmenter cadence (incrément)
- Freiner (intensité)

5.2.2 L'instanciation d'un objet.

Les objets sont construits à partir de classe par un processus (une technique) appelée instanciation. Tout objet est une instance de classe.

Si nous reprenons l'exemple de la classe *Bicyclette*, on peut instancier deux objets de la classe bicyclettes avec des valeurs d'attributs différentes. On peut avoir par exemple *MonVelo* dont l'attribut couleur vaut Jaune et un autre objet *TonVelo* dont l'attribut couleur est rouge.

5.2.3 Spécification et réalisation.

On retrouve deux parties dans une classe : la spécification, la réalisation. La spécification (interface en Delphi) décrit le domaine de définition et les propriétés des instances de cette classe. La réalisation (implémentation en Delphi) décrit comment la spécification est réalisée et contient le corps des opérations et les données nécessaires à leur fonctionnement.

5.3 Héritage des classes.

En programmation objet la technique la plus utilisée pour classifier les objets est l'héritage.
Les classes enfants héritent des caractéristiques de leurs classes parents, c'est à dire que les attributs et les opérations déclarées dans la classe parent sont accessibles dans la classe enfant comme si ils avaient été déclarés localement.
Une sous classe peut étendre les fonctionnalités de sa super-classe en définissant de nouveaux attributs et/ou de nouvelles méthodes Une sous classe peut spécialiser les fonctionnalités de sa super-classe en redéfinissant (override) des méthodes dont elle hérite.

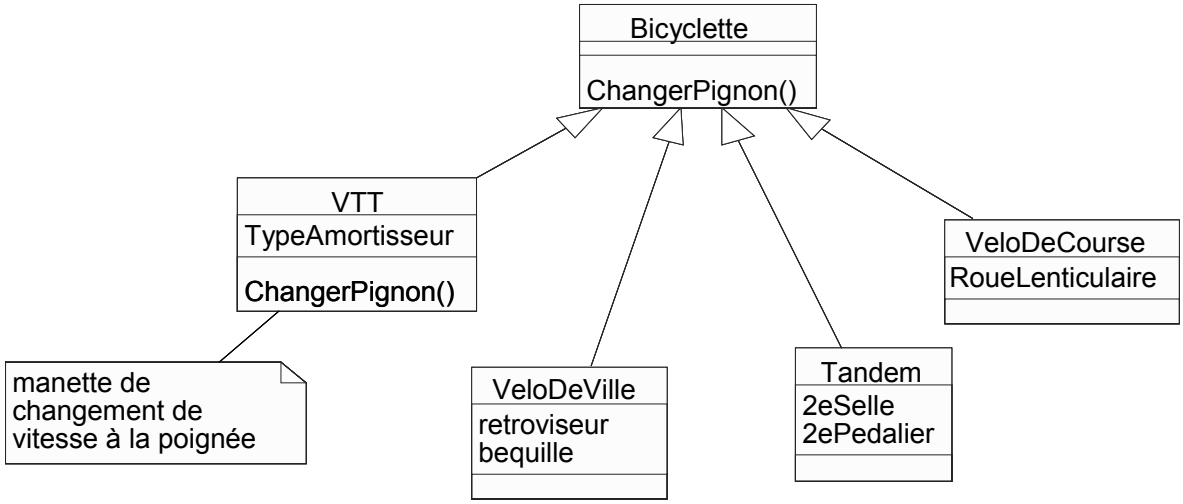


Fig. 26 Arbre d'héritage de la bicyclette

On parlera de classes concrètes lorsque qu'un objet peut être une instance de cette classe. Dans le cas contraire, la classe est qualifié d'abstraite

Si on prend le cas de la superclasse *Herbivoire* qui a deux sous classes *Lapin* et *Vache*, on considère que les classes *Lapin* et *Vache* sont concrètes car on va rencontrer des instances de *Lapin* (nain, lièvre) ou de *Vache* (Normande, Charollaise) dans la nature. Au contraire, la classe *Herbivore* est abstraite car on ne rencontera pas d'animal étant un *Herbivoire*.

Chapitre 2 :

Les modes d'analyse.

Pour répondre à un besoin croissant d'outil de simulation pour l'électrotechnique, nous avons développé Simul'Elec, un simulateur disposant de trois modes d'analyses complémentaires (régime permanent, événementiel et fréquentiel). Son but est de proposer une alternative aux outils actuels de simulation pour l'électrotechnique qui sont soit trop simplistes, soit trop complexes. Ces outils sont de plus dissocié de l'environnement de schématique électrique professionnel. Nous proposons un logiciel permettant à toute personne ayant des notions d'électricité (technicien, élève, enseignant) de simuler facilement l'évolution séquentielle de l'état d'un circuit. Pour cela nous utilisons notamment un mode d'analyse événementiel original et innovant avec lequel il est possible de mesurer aussi bien les tensions et courants du circuit que de visualiser l'évolution des états logiques des composants.

1 Contexte d'utilisation.

Simul'Elec est l'appellation commerciale du simulateur développé au cours de ce travail de thèse. Il est commercialisé par la société Algo'Tech Informatique [ALG]. Il est intégré au logiciel de DAO-CAO électrique « Elec'View » sous la forme d'un module complémentaire qui permet de simuler les installations dont on a précédemment réalisé le schéma. Elec'View est un logiciel de schématique orienté vers la représentation d'installation électrotechnique, d'automatisme et de distribution électrique basse tension notamment.

1.1 Les utilisateurs de Simul'Elec.

Il est principalement destiné à un public qui utilise le logiciel de schématique pour réaliser des dossiers techniques électriques ayant pour partie centrale les schémas. La simulation n'est pas la fonctionnalité première qu'ils recherchent mais constitue une valeur ajoutée non négligeable.

On peut cependant classer les utilisateurs en deux grandes catégories qui ont des approches différentes :

- Les enseignants, pour lesquels la simulation apparaît davantage comme un outil pédagogique permettant d'inculquer des notions fondamentales et ouvrant des possibilités d'étudier des phénomènes que l'on pourrait difficilement reproduire pratiquement (court-circuit, pannes…), d'un coût trop élevé ou dangereux pour les élèves.
- Les industriels : bureau d'étude ou de maintenance, électrotechnique, automatisme, distribution électrique, courants forts. La plupart n'ont pas encore la culture de la

simulation qui existe en électronique, hormis chez les concepteurs de circuits intégrant de l'électronique de puissance. Ceux là utilisent déjà des simulateurs du marché (SPICE ...) qui sont performants sur ces circuits et disposent de modèles fins fournis par les constructeurs de composants.

Simul'Elec est conçu dans l'optique de pouvoir être manipulé par des utilisateurs n'ayant pas la maîtrise des modèles ni des modes d'analyse. Il attendent des résultats fiables, à coup sûr, sans toutefois attendre un précision trop fine.
L'utilisation d'un logiciel de type SPICE avec ses problèmes de non-convergence et son paramétrage complexe semble donc mal adapté à ce public.

1.2 Choix d'une technique de modélisation.

De nombreux simulateurs de réseaux électriques orientés vers l'électronique et l'électronique de puissance existent sur le marché. Les travaux de recherche entrepris par F. Seyler en 2000 [SEY00], antérieurement à nos travaux de thèse, ont permis de faire l'inventaire des différentes techniques de simulation employées pour les circuits électriques notamment. Parmi ces techniques, on retrouvait la méthode d'analyse nodale [LIT97], la méthode des tableaux ou encore la méthode d'analyse sensitive. Il est apparu que la méthode la plus fiable et la plus éprouvée est la Méthode d'Analyse Nodale [LIT97]. Cette méthode est employée notamment par SPICE [NEW78] dont le code source est dans le domaine public. SPICE est de fait à la base de la majorité des simulateurs de circuits que l'on trouve dans les logiciels de CAO électronique. Il nous serait possible d'intégrer un noyau SPICE (la dernière version SPICE3F5 1995 [SPI2]). Mais intégrer SPICE ne présente pas forcément un grand intérêt scientifique ou technique. Toutefois, en faisant certaines restrictions et adaptations, à partir de cette méthode il est possible d'obtenir des résultats satisfaisants et relativement bien adaptés au domaine de l'électrotechnique.

1.3 Simulateur existant SiCi.

Suite à une demande de ses clients, Algo'Tech Informatique a développé le premier prototype de simulateur « SiCi » (**Si**mulateur de **Ci**rcuit), en se basant sur la méthode d'analyse nodale. Il permet de simuler des circuits analogiques linéaires comportant des composants élémentaires (sources sinusoïdales, résistances, inductances, condensateurs).
Ses caractéristiques principales sont les suivantes :
Simulation de sources sinusoïdales ou continues uniquement.
Analyse en régime établi sinusoïdal.
Analyse fréquentielle.
Voici ses limitations par rapport à Spice.
Pas d'algorithme de linéarisation de caractéristiques des composants.
Pas de modélisation de composant ayant des caractéristiques tension/courant non linéaires (ex : semi-conducteurs).
Pas de réduction d'équations différentielles. Pas d'algorithme d'intégration, dérivation numérique.
Pas d'analyse transitoire (TRAN) ni de point de repos en continu (DC).
Le développement de ce prototype à permit de valider la faisabilité du simulateur et de jeter les bases de ce travail de thèse.

1.4 Solutions retenues dans Simul'Elec.

Dans le domaine de la simulation de circuits électriques SPICE [SPI1], où les simulateurs intégrant son noyau, constituent l'immense majorité des outils disponibles. Ils sont avant tout destinés à la simulation de circuits électroniques où l'on trouve typiquement quantité de diodes, transistors et autres composants « semi-conducteurs ».
Ces composants ont un comportement non-linéaire, ce qui augmente notablement la complexité et la durée des calculs des simulateurs en introduisant en plus des risques de non-convergence. Pour éviter ces risques, l'utilisateur doit posséder une bonne connaissance des modèles qu'il utilise et de l'outil de simulation.

Dans la plupart des cas, les circuits de l'électrotechnique que l'on souhaite simuler ne comportent pas les composants non linéaires rencontrés fréquemment en électronique (diodes, transistor, etc.). Les non-linéarités que l'on doit traiter en électrotechnique (seuil, hystérésis…) sont davantage associées à des changement d'état logique ou à des comportements Tout Ou Rien, qui pourraient d'ailleurs être traités plus facilement par des simulateurs logiques plutôt qu'analogiques.

Ce contexte de travail, nous amène à conclure qu'il n'est pas nécessaire de traiter les non linéarités d'une manière aussi rigoureuse que Spice peut le faire. Pour s'affranchir du risque de non convergence, nous avons donc éliminé dès la conception la possibilité de traiter les non-linéarités par les méthodes classiques de linéarisation par itération (Newton-Raphson notamment [RAP98]). Il est toutefois essentiel de prendre en considération les non-linéarités associées à un état logique que l'on traitera grâce au mode d'analyse événementiel. Certaines installations ont un comportement séquentiel que le simulateur doit reproduire. Chaque étape de ce comportement séquentiel peut correspondre à un circuit différent du fait de la fermeture et à de l'ouverture de certaines branches liées par exemple au positionnement d'un contact de relais. De plus, les temps séparant deux étapes successives peuvent être relativement longs (secondes, minutes, heures…) devant les constantes du temps des systèmes électriques. Le suivi des évolutions du système avec un mode d'analyse transitoire classique (TRAN dans SPICE) se révèlerai alors trop coûteux en temps de calcul, complexe à exploiter et mal adapté à nos préoccupations.
Pour ces raisons, nous avons développé un mode d'analyse événementiel innovant et plus ciblé sur nos objectifs.

1.5 Les modes d'analyse développés.

Nous avons montré que la principale attente des utilisateurs de notre simulateur concerne principalement le suivi du comportement séquentiel du circuit étudié. Cela comprend d'une part l'étude de l'aspect discret de ces séquences (changements d'état logique, continuité électrique, présence de tension ou non…) qui donne des informations qualitatives sur l'état de l'installation. D'autre part, l'utilisateur souhaite aussi obtenir des informations précises concernant l'aspect continu traité classiquement par les simulateurs électriques analogiques. Ce sont en général les valeurs instantanées des tensions et des courants en tout point du circuit qui donnent des informations quantitatives sur l'état du circuit.

Pour permettre un étude adaptée à ces différents besoins, nous avons développés 3 modes d'analyse complémentaires :

- Régime permanent : Il donne des valeurs de tension et de courant correspondant à un état stable et établi du circuit. Il permet d'obtenir les résultats de la simulation de manière immédiate et sans aucun paramétrage.
- Fréquentiel : Il permet d'étudier le comportement harmonique du circuit dans une plage de fréquence donnée.
- Evénementiel : Il complète le mode d'analyse en régime permanent en suivant le comportement séquentiel des composants. Aux valeurs de tension et de courant, il ajoute la visualisation des états logiques et de leur impact électrique sur le circuit. Il facilite la création de stimuli complexes permettant d'intervenir sur le circuit pour observer ses réactions.

2 Analyse en régime permanent.

Le mode d'analyse en régime permanent (RP) donne pour un instant et un état donné du circuit les valeurs de tension et de courant en tout point. On considère que le circuit est dans un état stable, tout phénomène transitoire terminé. Ce mode d'analyse prend en compte les états logiques des composants et leur influence sur le comportement électrique du circuit, comme par exemple la position ouverte ou fermé d'un interrupteur. Par contre, il ne permet pas de déterminer l'évolution d'un état logique en fonction des résultats (tension et courant) de la simulation, tel que l'activation d'un relais ou le déclenchement d'un disjoncteur. Ce type de comportement séquentiel sera traité par le mode d'analyse événementiel qui est donc une extension séquentielle des possibilités du mode d'analyse en régime permanent.
Le mode d'analyse en régime permanent est bien adapté aux spécificités des circuits électrotechniques et aux composants que l'on y rencontre.

2.1 Principe de l'analyse en régime permanent

On va considérer qu'un schéma peut être simulé dès lors que :

- Tous ses composants sont associés à des modèles connus.
- Toutes les bornes des composants sont câblées.
- Il existe au moins une maille fermée dans le circuit.
- Il existe au moins une source dans le circuit et qu'elle n'est pas court-circuitée.

Ces conditions sont nécessaires quel que soit le mode d'analyse.

Lorsqu'un circuit simulable est ouvert dans le logiciel de schématique, il est possible d'activer le module de simulation. Dès qu'il est activé, le simulateur effectue une analyse RP. En cas d'échec de la résolution, l'utilisateur va devoir contrôler que les critères énoncés précédemment sont respectés. Si la résolution à réussi, l'utilisateur peut choisir les signaux de tension et de courant qu'il veut observer et obtenir immédiatement les résultats correspondants. Il peut ensuite modifier les caractéristiques des composants et en constater l'effet sur les courbes de tension et de courant qu'il a sélectionnées.

Il existe deux types de paramètres qui ont une influence sur le résultat de la simulation présenté à l'utilisateur. Alors que les caractéristiques des composants influencent la résolution, les paramètres de l'analyse modifient seulement l'affichage des résultats.

2.1.1 Caractéristiques des composants.

Les caractéristiques des composants vont renseigner leurs modèles et déterminer leurs comportements. Nous distinguons :

- Les paramètres tels que l'impédance, l'excitation, la fréquence, etc., liés à une valeur caractéristique du modèle.
- L'état logique qui permet de choisir entre deux modèles possibles pour un même composant. C'est par exemple le cas de l'interrupteur qui a un modèle de circuit ouvert lorsqu'il est au repos et un modèle de court-circuit quand il est activé. L'état logique n'existe que chez les composants où cette notion a une signification.

Une modification d'un paramètre va provoquer le changement d'un coefficient du système d'équations à résoudre alors que la modification de l'état logique peut amener à redéfinir complètement ce système d'équations pour refléter une modification de la structure du circuit

(branche ouverte ou court-circuitée). Dans tous les cas, la modification d'une de ces caractéristiques oblige l'utilisateur à relancer une simulation pour qu'elle soit prise en compte.

2.1.2 Paramètres de l'analyse.

Les paramètres de l'analyse sont :

- La durée de la simulation qui détermine l'intervalle de temps sur lequel les courbes de tension et de courant sont tracées.
- La finesse du tracé qui fixe le nombre de points visualisés sur la durée de la simulation.
- Le mode de représentation permet de choisir un tracé temporel ou sous forme de vecteurs de Fresnel [EDM97] (utilisable uniquement en régime sinusoïdal).

Contrairement aux caractéristiques des composants, ces paramètres n'ont aucune influence sur la résolution en RP du système. Cela apporte un bénéfice important au niveau du temps d'affichage car le simulateur redessine uniquement les courbes à afficher en utilisant les résultats de la dernière résolution.

2.2 Spécificités des circuits « électrotechniques ».

En électrotechnique, on utilise principalement deux types de courants:

- Courant alternatif sinusoïdal (de fréquence 50 Hz en Europe).
- Courant continu.

La majorité des appareils rencontrés fonctionne sous une tension nominale fixe (hormis les variateurs alimentant des moteurs par exemple) sinusoïdale ou continue. Dans un circuit à transistors, on trouve la notion de point de fonctionnement (voir Chapitre 1) qui correspond aux points autour desquels on linéarise la caractéristique des composants. Cette notion associée à la polarisation des composants n'existe pas en électrotechnique.
L'analyse RP peut traiter le cas du courant continu, du sinusoïdal et d'une manière générale des circuits composés de plusieurs sources sinusoïdales de fréquences différentes. Cela reste transparent à l'utilisateur qui choisit uniquement les signaux de tension et de courant qu'il souhaite afficher**.**

2.2.1 Régime sinusoïdal.

Lorsque l'on veut simuler un circuit, excité par une (ou des) sources sinusoïdales de fréquence donnée, l'expression des équations différentielles régissant le comportement du circuit peut alors se mettre sous la forme simplifiée d'une équations aux termes complexes.

Ex : cas d'un circuit RLC

$$u(t)=R.i(t)+L\frac{di(t)}{dt}+\frac{1}{C}\int_0^T i(t)dt$$

devient pour u(t) = A.cos(ωt+φ)

$$\underline{U}=\left(R+j\left(L.\omega+\frac{-1}{C.\omega}\right)\right)\underline{I}$$

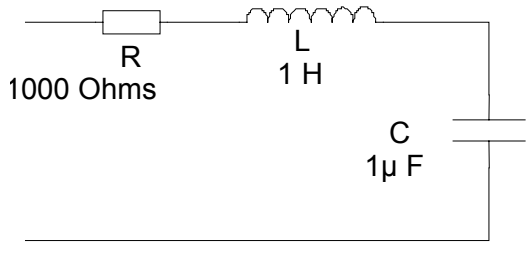


Fig. 1 Circuit RLC série

En application de ce principe, un circuit sera décrit par un système d'équations complexes. Pour la résolution, nous utiliserons la Méthode Nodale Modifiée [LIT97] comme le fait l'analyse AC de Spice. Mais dans notre cas, nous n'avons pas à effectuer de linéarisation, donc nous n'avons pas à calculer au préalable de point de fonctionnement comme dans l'analyse OP de Spice, opération qui peut se révéler coûteuse en temps de calcul lié aux multiples itérations.

Ce mode d'analyse nous permet d'obtenir, en une seule résolution du système d'équations, les valeurs complexes de tensions et de courants en tout point du circuit, donc de connaître très rapidement l'état du circuit en régime établi.

2.2.2 Régime continu.

Le prototype SiCi manipule des valeurs complexes d'impédance, de tension et de courant. Elles permettent d'obtenir à chaque résolution du système des signaux sinusoïdaux de tension et de courant à partir de leur amplitude, leur phase et leur fréquence.
Si les sources sinusoïdales permettent de traiter un grand nombre de cas des circuits électrotechniques, il est indispensable de pouvoir appliquer des excitations continues. En conservant le principe de résolution de SiCi, nous avons fait évoluer Simul'Elec pour pouvoir traiter des tensions et courants continus.
Afin d'être résolu numériquement, le système d'équations complexe est stocké dans des matrices. La fréquence d'excitation du circuit va influer sur les valeurs qui vont être inscrites dans ces matrices. Par exemple, pour un inductance L, on va insérer un valeur Zl =jL.ω correspondant à son impédance complexe qui dépend donc de la pulsation du circuit traité. Mais lorsque les matrices seront complètement construites pour traduire l'ensemble du système d'équations, il n'y aura plus que des valeurs numériques indépendantes de la pulsation.
Au niveau de la résolution, le simulateur considère que pour un signal continu, on traite un signal de fréquence nulle. Par contre, l'exploitation des résultats doit être adaptée. En effet, pour une fréquence donnée, le simulateur donne les valeurs de tension et de courant sous forme d'amplitude A et de phase φ. Dans le cas général, le simulateur va tracer un signal en appliquant la relation $s(t) = A.\sin(\omega.t+\varphi)$. Dans le cas du continu, ω et φ sont nuls, on aurait donc des signaux $s(t) = 0.\sin(0.t+\varphi) = 0$. Le simulateur utilise alors la relation s(t)=A.
Ainsi, pour traiter le cas d'une source continue, nous utilisons la même méthode d'analyse et de résolution, en considérant qu'il s'agit du cas particulier où la fréquence de travail du circuit est nulle. Alors au lieu de termes complexes, on obtient des termes réels dans le système d'équations, donc des solutions réelles.
Ayant intégré la simulation des courants continus, il a été nécessaire de modifier les modèles existants pour les rendre plus efficaces et faciliter la résolution numérique du système. Ainsi l'inductance idéale dont l'impédance Z_L est définie par la relation $Z_L=L.\omega$ devient un court circuit (fusion de 2 nœuds) plutôt qu'une impédance nulle.

2.2.3 Régime multifréquence :

Certains circuits peuvent être excités par des sources possédant plusieurs fréquences d'excitation. Le simulateur effectuera alors une résolution pour chacune des fréquences présentes dans le circuit, en éteignant les sources de fréquence différentes de celle de la résolution en cours. Ensuite, ayant obtenu autant de solutions que de fréquences, l'application du principe de superposition permettra de combiner les participations des différentes excitations et de retrouver ainsi l'évolution temporelle du signal.

Principe de superposition [EDM97] :
Pour obtenir les différents tensions et courants de branche d'un réseau électrique comportant deux ou plusieurs sources indépendantes, nous pouvons considérer chacune des sources comme agissant seule à la fois et ensuite superposer leurs effets. Ce principe peut s'appliquer vu le caractère linéaire des relations tension - courant. Lorsque les sources sont dépendantes, le principe ne s'utilise que lorsque les fonctions de commande sont extérieures au réseau comportant ces sources, ce qui assure que leur contrôle reste inchangé même si elles agissent une à une. Les sources de tension supprimées (lorsque qu'une seule source est maintenue) sont remplacée par des courts-circuits ; les sources de courant sont remplacées par des circuits ouverts.

Des cas simples peuvent être par exemple une source délivrant une tension sinusoïdale comportant un composante continue. Nous considérons alors que nous disposons d'une source continue (fréquence nulle) et d'une source sinusoïdale en série.
On peut encore traiter une source sinusoïdale représentant l'alimentation du réseau électrique perturbée par une seconde source sinusoïdale d'une fréquence plus élevée et d'amplitude bien plus faible qui modélise des harmoniques qui parasitent le réseau.
Ce principe de régime multifréquence nous permet notamment de modéliser des sources complexes issues de l'assemblage de plusieurs sources de fréquences différentes. Nous avons pu modéliser ainsi des sources de tension carrée ou triangulaire à partir de leur décomposition en série de Fourier.

Exemple d'une source multifréquence, source de tension carrée :

En se basant sur le principe décrit précédemment, il nous est possible de simuler la réponse d'un circuit à une tension carrée. Pour cela, nous modélisons cette source carré sous forme d'une superposition de sources sinusoïdales dont les caractéristiques (amplitude, pulsation et phase) nous sont données par une décomposition en série de Fourier [EDM97] .

Soit la source de tension carrée d'amplitude A, de pulsation ω, de rapport cyclique ½, de « déphasage » φ et de valeur moyenne nulle.

$$Ve(t)=\sum_{i=1}^{N} a_i.\cos(\omega_i.t+\varphi_i)$$

avec $a_i=\frac{4.A}{(2.i-1).\Pi.\sqrt{2}}$, $\omega_i=(2.i-1).\omega$ et $\varphi_i=\varphi.(2.i-1)$

Le modèle retenu apparaît en **Fig. 2**:

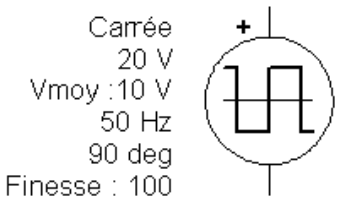


Fig. 2 Symbole de source carrée

TtensionCarree
Nom = NomSource
Amplitude = E
Moyenne = Valeur Moyenne
pulsation = 2.PI.Fréquence
Phase = Phase
Nbr harmoniques = Nombre harmoniques

Tab. 1 Paramètre de la source carrée

Les paramètres Amplitude, Pulsation et Phase caractérisent la source et permettent de déterminer les coefficients de Fourier pour les différentes pulsations.
Le paramètre « Nombre d'harmoniques » détermine le nombre N de pulsation que va contenir la source. Plus cette valeur sera élevée, plus la forme de la tension sera proche d'une tension carré idéale comme on le vérifie sur les **Fig. 3** à **Fig. 6** .

Par contre, l'amélioration de la finesse du signal provoque l'augmentation du temps de calcul dû à l'ajout d'une résolution par fréquence harmonique supplémentaire.

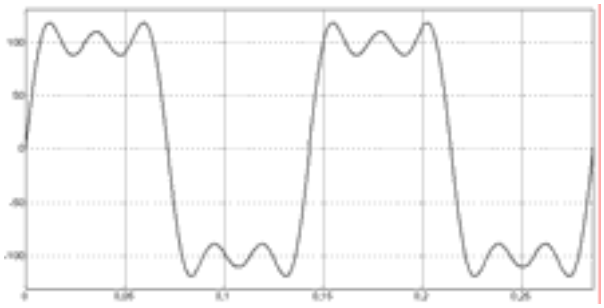


Fig. 3 Signal Pseudo carré pour N = 3

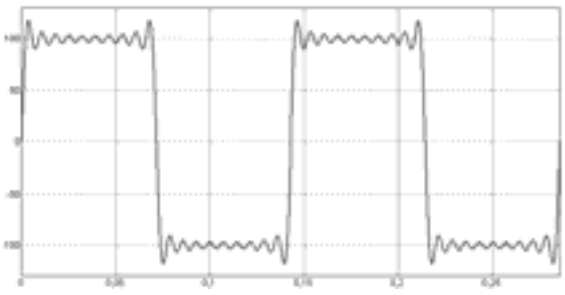


Fig. 4 Signal Pseudo carré pour N=10

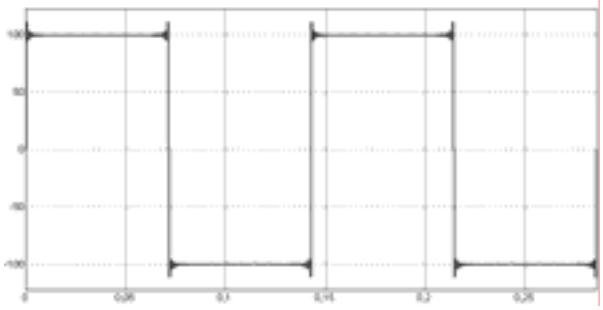


Fig. 5 Signal Pseudo carré pour N=100

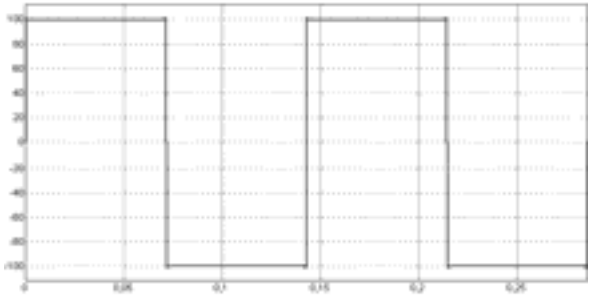


Fig. 6 Signal Pseudo carré pour N=1000

2.3 Utilisation de source multifréquence : réponse d'un circuit RLC série.

En excitant un circuit RLC série avec notre source carrée (**Fig. 7**), nous pouvons obtenir la réponse indicielle du circuit comme on le ferait expérimentalement. Ainsi, nous obtenons un résultat similaire à celui donné par à une analyse transitoire réalisée avec Spice.

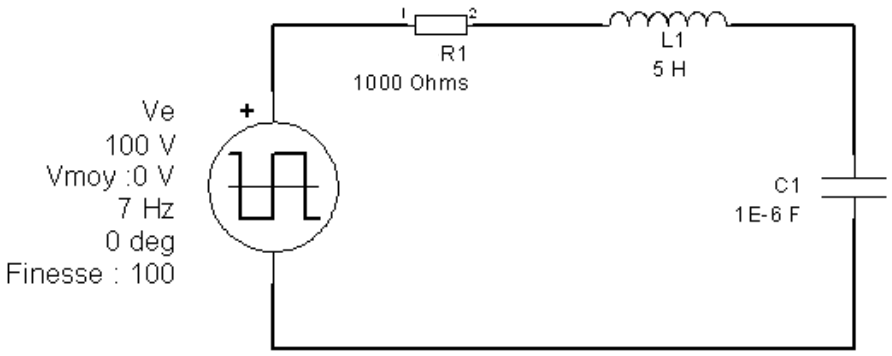


Fig. 7 Circuit RLC série alimenté par une source carrée

Lors de cette analyse transitoire, on va calculer l'évolution temporelle des signaux point après point en utilisant le système d'équations différentielles régissant le comportement du circuit. Il sera nécessaire pour chaque point de dériver ces équations et fonctions du point précédent (calcul de la dérivée partielle) puis de les linéariser.
Par contre, l'utilisation d'une source multifréquence pour appliquer une tension carrée, nous permet de résoudre directement, pour chaque fréquence, un système d'équations complexes linéaires. Dans ce cas, on n'évalue pas l'évolution temporelle des signaux lors de la résolution. C'est uniquement au moment du tracé de signaux sélectionnés que l'on va calculer point par point la courbe de tension ou de courant à visualiser (selon la finesse d'affichage voulue). Pour cette opération, on applique le principe de superposition pour ajouter les résultats obtenus pour les différentes pulsations.
Pour tracer une tension u(t), on disposera des données suivantes :

- TabTension : Tableau de valeurs complexes de taille N (nombre d'harmoniques) où TabTension[i].Mod désigne l'amplitude de la tension correspondante à l'harmonique i et TabTension[i].Arg la phase de cette même tension.
- TabPulsations : Tableau des N pulsations présentes dans le circuit. De même TabPulsations[i] contient la valeur réelle de la pulsation de rang i.

On peut alors déterminer la tension :

$$u(t)=\sum_{i=1}^{N} TabTension[i].Mod\times\cos(TabPulsations[i]\times t+TabTensions[i].Arg)$$

Cette opération est certes coûteuse en temps de calcul, mais elle ne s'applique qu'aux seuls signaux à visualiser. A l'opposé, lors d'une analyse transitoire Spice, on recalcule pour chaque point temporel l'ensemble des tensions et courants du circuit.

Le premier avantage qui apparaît est de permettre d'observer une réponse indicielle sans avoir à développer un mode d'analyse transitoire ainsi que les méthodes de réductions ainsi que les méthodes de « réduction » des équations différentielles et de linéarisation qui doivent y être associées. Ainsi, on s'affranchit également d'avoir à déterminer un pas de calcul temporel optimal.
Dans les deux cas, la durée de la simulation est directement liée à la finesse du résultat à obtenir. En analyse transitoire, celle ci est rattachée au pas de calcul temporel (dont dépend le nombre de points calculés). A l'opposé, pour l'analyse en régime permanent de notre simulateur, elle est dépendante du nombre d'harmonique existant dans le circuit.

Nous allons comparer, sur le cas de ce circuit RLC série, l'évolution de l'erreur entre le signal théorique et celui obtenu par la simulation en fonction du nombre de points de calcul.

2.3.1 Conditions de l'étude.

Nous allons comparer les résultats obtenus pour la tension Uc. L'analyse porte sur un temps allant de 0 à 5τ, τ étant la constante de temps du circuit. On considère que pour t=5τ le régime est établi. Sur cette durée de 5τ, nous allons calculer théoriquement 5000 points servant de référence par rapport aux résultats de simulation. Nous allons évaluer dans les 2 cas l'évolution de l'erreur pour un nombre de point allant de 1 à 1000.
On sait qu'en appliquant un échelon de tension unitaire, on obtient une tension au bornes du condensateur :

$$Uc(t)=(1-e(-t/\tau))\times\cos(t/To)$$

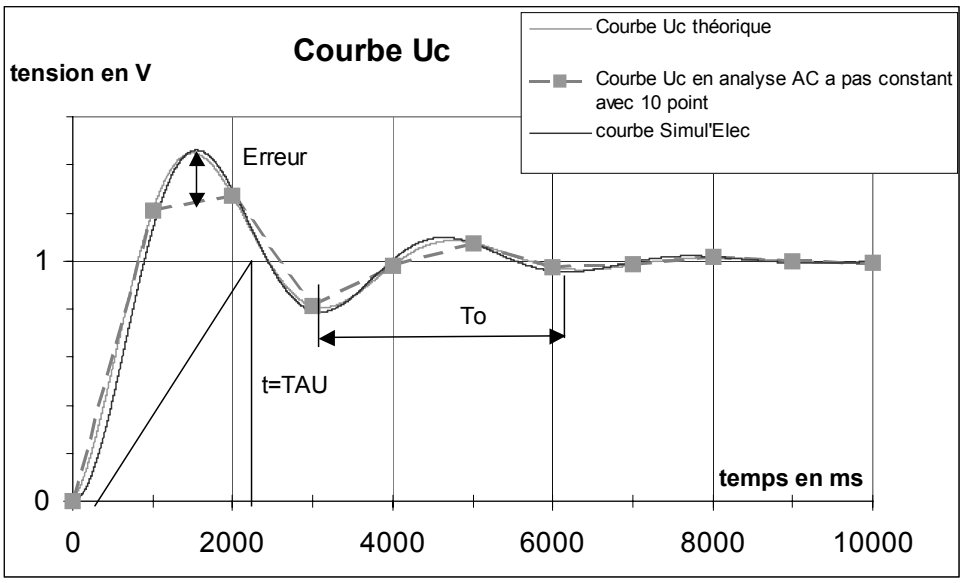


Fig. 8 Comparaison des réponses du circuit RLC

2.3.2 Erreur relevée en analyse transitoire Spice.

Pour chaque point de calcul, on obtient un résultat conforme à la courbe théorique (aux arrondis de calcul près). En effet, on applique un échelon de tension idéal et on résoud le système à partir des équations différentielles. Entre 2 points de calcul la courbe obtenue est linéaire, on a donc pour les temps intermédiaires une erreur que l'on va cumuler pour obtenir l'erreur globale sur toute la durée de l'analyse. On considère ici que le pas de calcul temporel est constant alors que dans la pratique on peut avoir un pas variable (optimisé selon la pente locale de la courbe notamment).

2.3.3 Erreur relevée avec une source multi-fréquence.

Contrairement au cas précédent, on aura toujours une erreur entre la valeur théorique et la valeur calculée pour un instant t, d'après la superposition des résultats obtenus pour chaque fréquence. L'égalité est vraie pour un nombre infini d'harmoniques. Dans la pratique, pour le cas d'une source carrée, l'erreur est quasiment nulle au delà de 3000 harmoniques. Nous allons calculer la valeur de Uc pour différent nombre d'harmoniques en appliquant le principe de superposition.

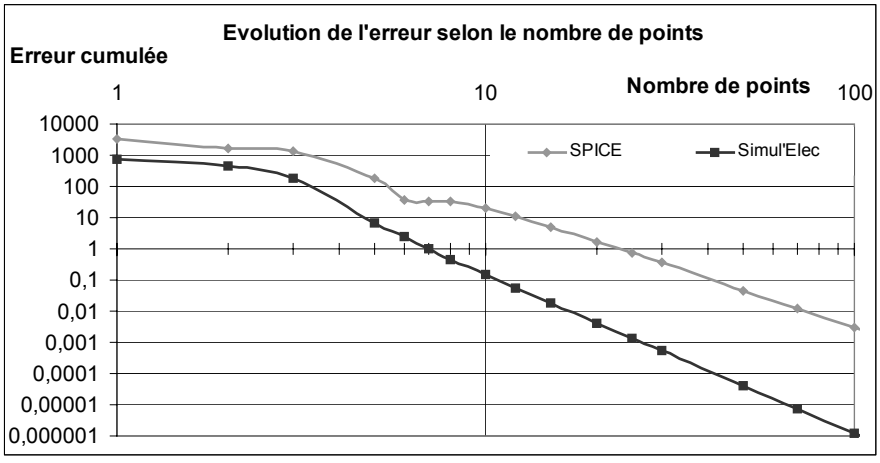


Fig. 9 Evolution de l'erreur en fonction du nombre de points

3 L'analyse fréquentielle.

Le but du mode d'analyse fréquentielle est de permettre de réaliser une étude harmonique d'un circuit. Ce mode d'analyse s'apparente d'assez près au mode d'analyse AC de SPICE. Nous verrons que les résultats sont quasiment identiques et présentent des performances satisfaisantes (chapitre 4 : validation). Ce mode d'analyse n'est pas réellement original dans son principe mais nous avons essayé de le développer dans l'esprit général de Simul'Elec axé sur le simplicité d'utilisation.
Précisons également que pour nos utilisateurs, c'est surtout l'analyse en régime permanent ou l'analyse événementielle qui présente une forte valeur ajoutée. L'analyse fréquentielle n'en constitue pas moins un complément intéressant de l'étude temporelle notamment dans un cadre pédagogique. Il faut tout d'abord noter que ce mode d'analyse va simuler des réponses harmoniques de circuits en correspondance avec les modèles qui ont pu être introduit. Or pour la plupart des modèles, c'est essentiellement sur le comportement temporel ou séquentiel que la modélisation s'appuie. De nombreux modèles ont donc des bandes passantes infinies ou donnent des réponses en fréquences. C'est par exemple le cas des disjoncteurs et des transformateurs qui ont un comportement identique quelle que soit la fréquence.
Un des champs d'application de ce mode d'analyse est l'étude de filtres.

3.1 Principe de l'analyse fréquentielle.

Il va être possible d'appliquer ce mode d'analyse sur un circuit excité par une source unique. Cela peut être par exemple le cas du circuit RLC série que l'on a traité par l'analyse RP en **2.3** et dont on détaille le comportement fréquentiel plus loin (voir en **3.2**).

L'utilisateur doit tout d'abord sélectionner l'entrée du circuit où l'on va placer une source sinusoïdale dont la fréquence va varier pour effectuer un balayage dans la gamme de fréquence choisie par l'utilisateur. Il d'agit de la source qui excite normalement le circuit. Pour notre circuit RLC (**Fig. 10**), on sélectionnera la source de tension carrée. L'utilisateur doit également sélectionner la tension de sortie (2 bornes ou un dipôle).
En fonction de la finesse désirée, le simulateur va déterminer un certain nombre de points dans la gamme de fréquence sélectionnée. La progression entre un point et le suivant se fait de manière logarithmique. Pour chaque valeur de fréquence, le système d'équations est résolu, après avoir été réévalué. En effet, toutes les valeurs caractéristiques de composants dépendantes de la pulsation sont recalculées comme par exemple l'admittance d'un condensateur $Yc = C.\omega$.
Le simulateur fait ainsi autant de résolutions que de points de fréquence souhaitée dans la plage définie par l'utilisateur.
Ce dernier a choisi une source associée à une tension complexe $\underline{Ve}$ et une sortie à laquelle on peut associer une autre tension complexe $\underline{Vs}$. Le résultat affiché représente le rapport $\underline{Vs}/\underline{Ve}$ qui sera tracé sur des diagrammes de Bode d'amplitude et de phase ou encore des diagrammes de Black.

3.2 Cas d'un circuit RLC série.

Un circuit RLC série est un cas typique pouvant être traité par l'analyse fréquentielle. A partir du même schéma qui nous a permit d'étudier son comportement temporel en réponse à un signal carré, on va pouvoir en faire l'étude harmonique.

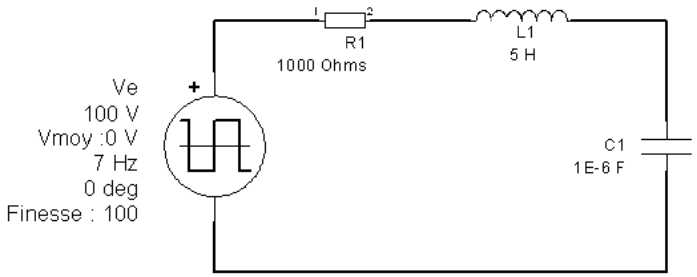


Fig. 10 Circuit RLC série alimenté par une source carrée

On peut ainsi mesurer des valeurs caractéristiques telles que le gain statique, la fréquence de coupure, de résonance, etc et vérifier l'influence des paramètres des composants du circuit sur ces valeurs. L'utilisateur sélectionne ici la source carrée à laquelle le simulateur substitue une source sinusoïdale de tension Ve. En sortie on mesurera la tension Vs aux bornes du condensateur C1. L'étude porte sur une plage de 1 à 1000 Hz qui compte 650 points.

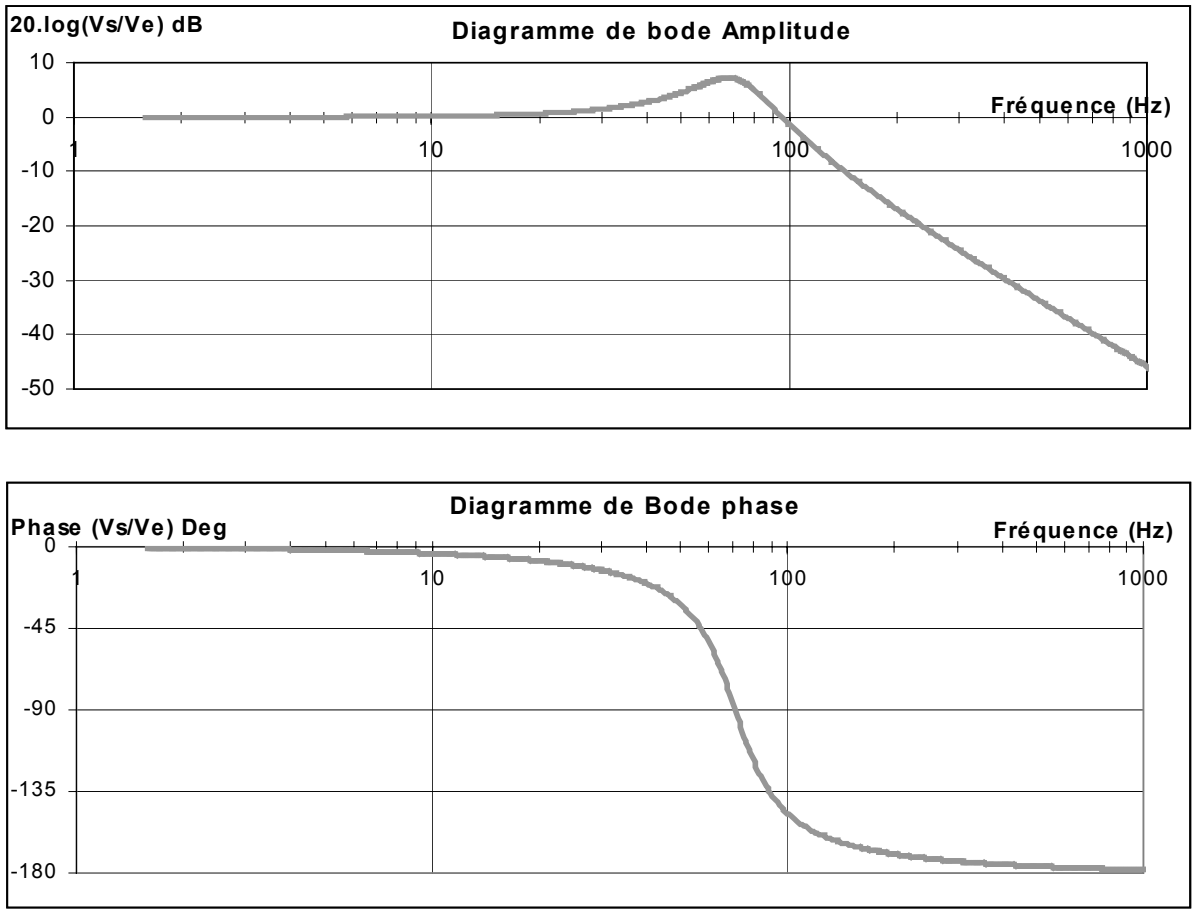


Fig. 11 Diagrammes de Bode du Circuit RLC

3.2.1 Correction automatique de phase.

Les résultats bruts issus du simulateurs sont des valeurs de tension complexes. On aura des valeurs de déphasage comprises entre –180° et 180°. Bien que correct, l'affichage ces données brutes peut être déstabilisant car on a un risque de discontinuité dans la courbe de phase du diagramme de Bode. Par exemple pour un filtre d'un ordre 4 ou pourra rencontrer une phase qui décroît lorsque la fréquence augmente pour passer de 0 à –360°. Comme le simulateur, ne donne que des valeurs comprise entre +/-180, toutes les valeurs inférieurs à -180° sont affichées comme valant 360° de plus que la valeur attendue, provoquant ainsi une rupture dans la continuité de la courbe. Nous avons donc mis en place un dispositif permettant d'éliminer l'affichage de ces discontinuités pour avoir un rendu plus familier pour l'utilisateur. On peut voir l'effet de cette correction automatique de phase en **Fig. 12**.

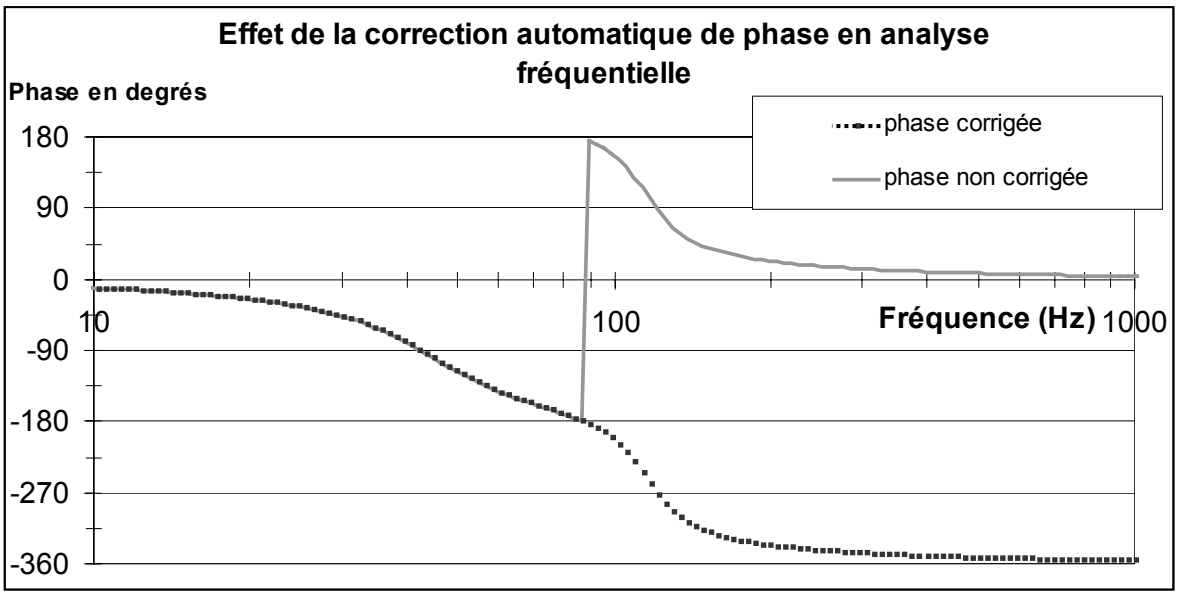


Fig. 12 Diagramme de Bode (Phase) avec et sans correction de phase

Tout d'abord, on doit détecter ces discontinuités autour des valeurs +/-180° et leur multiples. Pour cela, on parcours la courbe jusqu'à rencontrer deux points consécutifs caractéristiques d'une discontinuité (un point à –179° suivi d'un point à + 176 par exemple). Notons que cette correction automatique est possible si le nombre de points est suffisant pour que la discontinuité soit évidente. Si par exemple on a un point à –165° suivit d'un point à +160°, il devient moins évident que l'on a affaire à une discontinuité, et la correction ne sera pas faite.
Dès que ces discontinuités sont isolées, il est nécessaire de vérifier qu'il s'agit bien d'un phénomène lié au calcul du simulateur. Si ces deux points sont censés être placés côte à côte sur une courbe les pentes de leurs tangentes doivent être relativement proches. On mesure donc la pente de la tangente à la courbe pour les points à chaque extrémité de la rupture. Si les deux pentes sont suffisamment proches, on peut alors procéder à la correction. De la même manière que précédemment, si le nombre de points est insuffisant les tangentes peuvent être trop différentes et empêcher la correction.

4 L'analyse événementielle.

Les modes d'analyse en régime permanent et fréquentiel décrits précédemment permettent d'étudier le comportement d'un circuit dans un état stable. Or, dans un circuit électrotechnique le comportement de l'installation peut varier subitement à un instant donné. Le traitement de ces discontinuités peut être pris en compte par la Simulation par Evénements Discrets (SED). Après un bilan de l'utilisation de la SED en électricité et en électronique, nous proposons une adaptation de la SED électrotechnique. Nous terminons par une discussion des choix de conception que nous avons faits, suivi par des détails d'implémentation de l'analyse événementielle.

4.1 Analyse événementielle et simulation par événements discrets (SED).

Le simulateur propose des modes d'analyse qui offrent différents niveaux de détail dans l'étude d'un circuit. Les plus basiques sont le mode d'analyse en régime permanent (pour une étude temporelle) et le mode d'analyse fréquentiel (pour une étude harmonique). Ces modes d'analyses simulent le comportement continu du circuit. Or un circuit électrotechnique peut connaître des changements d'état soudains qui constituent des discontinuités dans son comportement. Ces discontinuités sont des événements discrets. Pour pouvoir les traiter, nous avons développé l'analyse événementielle qui utilise les principes de la simulation par événements discrets ou SED. Nous avons détaillé les différents concepts liés à la SED dans le chapitre 1.

4.1.1 Les limitations de l'analyse en régime permanent.

Le mode d'analyse en régime permanent permet de simuler l'état stable d'un circuit à un instant donné. On obtient des signaux de tension et de courant qui sont des sinusoïdes ou des compositions de sinusoïdes. Avec ce mode d'analyse, on peut représenter ces signaux stables sur une durée de temps choisie sans tenir compte des phénomènes transitoires ni de l'influence de cet état stable sur le circuit. Cette analyse affiche des résultats qui ne tiennent pas compte du comportement séquentiel du circuit électrotechnique traité. Il prend en compte l'état ouvert ou fermé des composants tels que des interrupteurs, disjoncteurs ou contacteurs, mais ne sait pas modifier leur état pour traduire leur comportement.
Par exemple, un disjoncteur magnétothermique qui est traversé par un courant trop élevé ne déclenchera jamais lors de l'analyse en régime permanent.
Il est donc nécessaire de développer un mode d'analyse qui puisse reproduire le comportement séquentiel des éléments du circuit au cours du temps.

4.1.2 Le Concept de l'analyse événementielle.

Pour répondre à ce besoin de simuler un comportement séquentiel, il faut pouvoir déterminer les différents états successivement pris par le circuit. Pour un état donné, l'analyse en régime permanent donne les valeurs de tension et de courant correspondantes. On peut donc obtenir pour les différents états possibles du circuit chaque résultat associé.
Il reste à définir les conditions de passage d'un état stable à un autre. Le résultat que l'on attend de l'analyse événementielle est un enchaînement d'états stables du circuit en faisant abstraction des phénomènes transitoires. Toutefois, il sera possible que chaque modèle d'élément du circuit possède un comportement événementiel qui simule ces transitoires.

4.1.3 Choix de la SED.

Avec l'analyse événementielle, nous introduisons une notion de discontinuité dans le comportement des composants. Des simulateurs purement analogiques considèrent que ces changements d'état bien qu'étant très soudains se font d'une manière continue. L'emploi de la SED n'est donc pas indispensable. Cette technique est intéressante dans des cas comme les nôtres où l'on s'intéresse essentiellement aux états avant et après les transitions et pas forcement aux transitions elles-mêmes. Nous rappelons que notre objectif est d'obtenir des résultats fiables et rapides sans être exagérément précis. L'emploi de la SED permet ici, comme en électronique numérique, d'avoir des résultats fiables avec une plus grande rapidité que si les événements étaient traités en temps continus. De plus, pour la majorité des composants, tels que les interrupteurs, il est réaliste de considérer que leur changement d'état est réellement discontinu. Notons toutefois que la SED ne constitue pas la totalité de l'analyse événementielle. Son rôle est uniquement de traiter les discontinuités. Le comportement du circuit entre ces discontinuités reste traité à temps continu par des simulations en régime permanent.

Ce qui nous intéresse, c'est la simulation par événement discret comme complément d'un simulateur à temps continu. C'est une fonctionnalité de traitement des événements discrets qui doit compléter la simulation mais n'en constitue pas le cœur. La souplesse et la facilité d'intégration qu'apporte l'emploi du langage de développement de l'application de schématique à laquelle d'intègre le simulateur que nous proposons à été un point déterminant dans ce choix de conserver le langage de programmation classique Delphi **[DEL]**.

4.2 SED en électricité : temps continu et temps discret.

Lorsque l'on traite un système où le temps s'écoule de manière continue, on le qualifie alors de **système continu**. Ainsi un simulateur qui le traitera sera lui aussi qualifié de continu, bien que son noyau de calcul puisse utiliser des méthodes de calcul numériques l'amenant à discrétiser le temps. Les variables du système peuvent prendre un nombre infini de valeurs qui peut changer à n'importe quel instant du temps.
On va considérer qu'un **système est discret** si ses variables prennent des valeurs en un nombre fini de points du temps.

Certains systèmes peuvent être considérés comme discrets ou continus selon le niveau de détail souhaité. En électronique par exemple, on considère communément qu'un transistor fonctionnant en mode bloqué / saturé a un comportement discret, alors que s'il fonctionne en mode bloqué / passant / saturé son comportement est considéré comme continu.
Dans ces systèmes, on peut traiter un sous système comme continu et un autre comme discret.
C'est le cas notamment des circuits électroniques mixtes comportant des composants analogiques (continus) et numériques (discrets).

Nous précisons ici l'application de ces notions à des systèmes électriques en traitant les cas de l'électronique et de l'électrotechnique.

4.2.1 SED en électronique.

Ces concepts de systèmes continus et discrets dans le temps s'appliquent particulièrement bien au domaine de l'électronique. On va considérer ainsi qu'un circuit a un comportement continu si ses signaux sont traités à tout moment du temps. Un circuit sera discret lorsqu'il

traitera les signaux à des instants déterminés. Par abus de langage, on qualifie les circuits continus de circuits analogiques alors que les circuits discrets sont considérés comme numériques (ou logiques). Un circuit qui réunit des composants des deux catégories est dit mixte analogique - numérique.

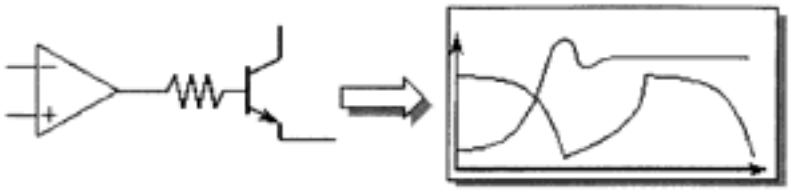


Fig. 13 Circuits et signaux analogiques

Fig. 14 Circuits et signaux numérique

Il convient également de distinguer la discrétisation du temps qui définit si un système est continu ou discret de la discrétisation de l'amplitude des signaux. Ainsi, un composant numérique peut traiter des signaux d'amplitude continue ou discrétisée (le nombre de valeurs possibles est fini).

4.2.2 SED en électrotechnique.

L'analyse en régime permanent exploite des modèles qui décrivent le comportement continu d'un circuit. En introduisant le concept de l'analyse événementiel, nous allons compléter le modèle de certains éléments du circuit pour leur ajouter une composante discrète. En effet, certains éléments du circuit disposent de plusieurs modèles continus possibles. Le modèle utilisé par le simulateur est défini par un paramètre du modèle (en général, l'état logique). C'est le cas notamment du modèle de l'interrupteur qui possède deux modèles continus possibles : un modèle de circuit ouvert pour l'état logique ouvert et un modèle de court circuit si son état logique est fermé.
Le choix du modèle alternatif est lié à l'état logique peut être défini par l'utilisateur. Il peut également être déterminé par des événements précis qui provoquent une discontinuité dans le fonctionnement du circuit. Par exemple, un disjoncteur magnétothermique peut voir son état logique (enclenché / déclenché) modifié s'il est parcouru par un courant dépassant son calibre. La description de ces événements va constituer une définition discrète du comportement. Celle-ci complète la description continue du comportement.

On disposera donc de 2 types de modèles :

- Des modèles purement continus.
- Des modèles continus et discrets. Dans de nombreux cas, il s'agit de composants électromécaniques (relais, interrupteurs, disjoncteur…). L'aspect continu traduit leur comportement électrique alors que l'aspect discret exprime un changement d'état ou de position mécanique.

On ne trouve par contre pas de composants purement discrets comme les composants logiques en électronique. L'aspect continu est définit par les relations entre les tensions et les courants pour chacun de ses nœuds (Analyse Nodale Modifiée). Si aucune de ses relations n'est

exprimée, le composant n'a aucune connection électrique avec le reste du circuit, donc aucune influence sur son comportement. Si on supprime toute liaison électrique avec le circuit, ce composant peut avoir uniquement un lien via son état logique avec celui d'un autre composant. Ce type de lien s'établit lors de la création des références croisées, où différents composants possédant le même nom sont mis en relation. Ainsi un composant maître connaît ses esclaves et peut ainsi leur imposer un changement d'état. C'est par exemple le cas de la bobine de relais (maître) qui peut imposer son état à ses contacts (esclaves).
Ainsi, si un modèle possède exclusivement un comportement séquentiel, sans l'aspect électrique continu, son état n'a pas d'effet sur le comportement électrique du circuit simulé et peut uniquement être transmis à ses éventuels esclaves. De même, son état ne peut être modifié que par une intervention de l'utilisateur ou par son composant maître.
Le seul intérêt de ce type de composant est de permettre une transmission plus ou moins directe de l'état de son maître vers celui de ses esclaves. On peut imaginer qu'il le modifie en le retardant, en transformant un état haut en horloge, en l'inversant, etc.
Dans notre contexte, aucune application de ce type de modèle n'est apparue, donc aucun modèle de ce type n'a été implémenté bien que cela soit théoriquement faisable.

4.3 SED en électronique mixte analogique-numérique.

Comme nous l'avons évoqué plus tôt, les concepts de la simulation pilotées par événements s'appliquent bien à la simulation de circuits électroniques mixtes analogiques-numériques. Ces simulateurs sont généralement constitués d'un noyau de simulation à temps continu qui va traiter les composants analogiques du circuit. Dans leur majorité, ces noyaux analogiques sont dérivés du simulateur Spice. Les éléments logiques seront pris en compte par un second noyaux de calcul à temps discret. L'interface entre les deux sous circuits analogiques et numériques est réalisée par des composants mixtes (convertisseur analogique numérique ou numérique analogique).

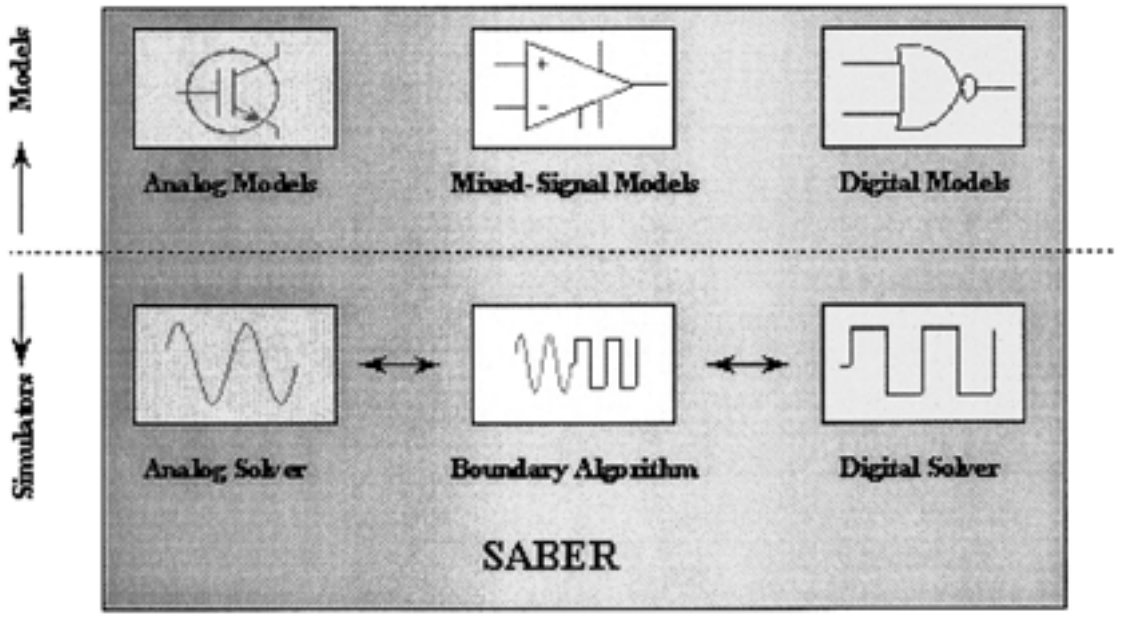
Un des points cruciaux pour l'efficacité du simulateur est la synchronisation des deux noyaux de calcul. La méthode de synchronisation retenue sera toujours un compromis entre la rapidité des calculs et leur précision. Cette problématique générale à la SED a des solutions adaptées au domaine de l'électronique.

4.3.1 Simulation hybride ou mixte analogique numérique.

La réunion de deux simulateurs analogique et numérique est qualifiée de mixte ou encore d'hybride car y on y combine une approche continue et discrète du comportement des composants. Le premier simulateur de ce type employait l'approche orientée événement, comme SPLICE [NEW78] présenté en 1978 et basé sur le simulateur de circuit analogique (continu) Spice2. Désormais, de nombreux simulateurs mixtes adoptent l'approche orientée processus où les 2 tâches concurrentes sont la simulation analogique (continue) et la simulation logique (discrète). C'est notamment le cas de Saber, édité jusque récemment par Analogy [SAB]. Nous disposons alors d'un processus qui réalise la simulation du circuit analogique et un deuxième processus concurrent qui traite le circuit logique. Le simulateur mixte en gère la synchronisation.

4.3.2 Synchronisation des algorithmes.

Les deux algorithmes de calcul analogiques et numériques (qui emploient tous deux des méthodes de résolution numériques) vont évoluer simultanément et de manière concurrente **Fig. 15**. Si on considère le fait qu'un circuit logique est un circuit analogique (comportement continu) pour lequel on fait l'approximation que son comportement est discret. Cette approximation permet des simulations beaucoup plus rapides pour un circuit de même dimension lorsqu'il est traité par le noyau numérique plutôt que par le noyau analogique.
La conséquence immédiate de cette efficacité sur la simulation mixte est que pour un circuit comportant des sous circuits analogiques et numériques de dimensions équivalentes et pour une même durée de calcul, le temps virtuel avancera plus vite dans le noyau numérique. La synchronisation des algorithmes analogique et numérique a donc ici une grande importance. De ce fait, les différents simulateurs possèdent diverses méthodes de synchronisation adaptées à cette simulation, au type d'événements traités et à la nature des signaux échangés.



Source : www.analogy.com

Fig. 15 Répartition de la simulation analogique et numérique dans Saber .

La synchronisation des algorithmes analogique et numérique a donc ici une grande importance. De ce fait, les différents simulateurs possèdent diverses méthodes de synchronisation adaptées à cette simulation, au type d'événements traités et à la nature des signaux échangés. Nous en donnons une description dans le chapitre 1.

4.3.3 Conclusion sur la synchronisation.

Devant la multiplicité des méthodes de synchronisation, on se rend compte que c'est un point délicat dans la mise en œuvre de simulateurs mêlant des comportements continus et discrets. Chaque méthode reste un compromis entre rapidité et précision des résultats, sans négliger l'occupation de la mémoire. Notre approche de la simulation de circuit électrotechniques, nous permet de travailler avec un seul algorithme de simulation. Ainsi, nous évitons ces problèmes de synchronisation. Nous détaillons ce principe en **4.4.3**.

4.4 SED en électrotechnique ou l'analyse événementielle.

En électrotechnique, on va traiter un monde continu et un monde discret est défini par les modèles continus des composants qui traduisent leur comportement électrique. Dans le monde discret, nous allons considérer les événements et les discontinuités liées aux changements d'état logique des composants. Dans le monde continu, les variables d'état sont des tensions et des courants, alors que dans le monde discret on traite des états logiques.

On peut aisément faire le parallèle entre les circuits que traitent les simulateurs électroniques mixtes et Simul'Elec. Dans les deux cas, les simulateurs doivent considérer un circuit analogique et d'autres composants ayant un comportement discret. Dans Simul'Elec par contre, on ne trouvera pas de composants purement logiques. La transmission des variables d'états discrètes se fait directement d'un élément à un autre comme on peut le voir en **Fig. 17** et qui est représenté par le pointillé liant les différents éléments du contacteur KM1. Les signaux d'un simulateur électronique logique sont des niveaux logiques. Selon la norme et la technologie employée, ils expriment la présence d'un certain niveau de tension ou de courant, ou de l'impédance des composants. Il s'agit d'une vue simplifiée de variables d'états qui seraient en réalité continues et que l'on considère comme discrètes. Au contraire, les variables d'état à temps discret qu'un circuit électrotechnique échange sont des états logiques et non pas des niveaux. Ils traduisent des états réellement discrets qui correspondent généralement à une position mécanique du composant simulé. Il ne s'agit pas de variables d'état continues (tensions et courants) que l'on a discrétisé pour diminuer la complexité et le temps de calcul du simulateur. Ces signaux sont donc transmis directement d'un composant à un autre, sans traitement. Il n'existe pas ici d'équivalent du circuit logique. Ce constat va nous permettre de déterminer quelle est la technique la plus adaptée pour l'implémentation de l'analyse événementielle.

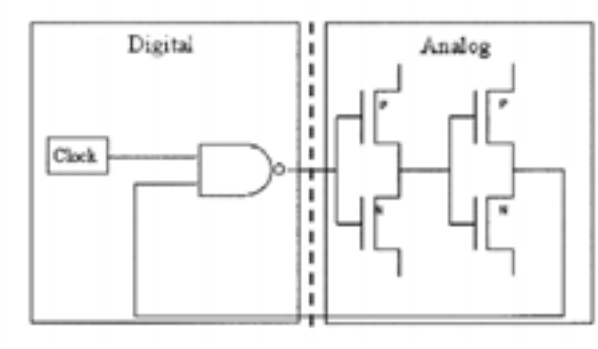


Fig. 16 Circuit électronique mixte

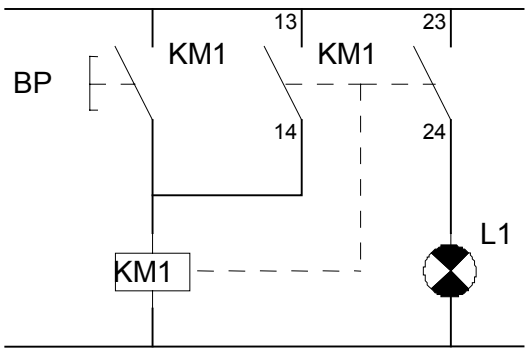


Fig. 17 Circuit électrotechnique de relayage

4.4.1 Variables d'état discrètes d'un circuit électrique.

Les variables d'état discrètes sont ces données exprimant l'état du circuit et dont la valeur change à un instant précis du temps et de manière discontinue. En électronique, ces variables à temps discret sont les niveaux logiques alors qu'en électrotechnique on parlera d'état logique. Il est essentiel de bien définir ces notions afin de comprendre comment elles sont traitées dans d'autres simulateurs et de quelle manière elles doivent être prise en compte pour des circuits électrotechniques.

4.4.1.1 Niveaux logiques d'un circuit électronique numérique.

Dans un simulateur logique, les grandeurs que l'on considère sont les états logiques (correspondant généralement aux niveaux de tension) présents sur les bornes des composants. Pour cela, on distingue :

- Les bornes d'entrée qui observent les états.
- Les bornes de sortie qui fixent les états logiques
- Les bornes d'entrée – sortie qui observent et fixent les états logiques.

Pour chaque composant, le modèle va exprimer les états que l'on doit retrouver en sortie en fonction de ceux que l'on observe en entrée. Ces relations sont généralement établies sous forme d'équation logique ou de table de vérité et prennent en compte les temps de propagation des signaux dans le composant. Un événement va ici correspondre au changement d'état d'une sortie. On considère en général une logique 4 états (seuls les états 0, 1 et Z peuvent être appliqués sur les sorties) décrit dans le **Tab. 2** :

Etat	Nom	Signification courante
0	Etat bas	Borne à un Potentiel de tension bas.
1	Etat haut	Borne à un Potentiel de tension haut.
X	Indéterminé	Etat d'entrée n'influant pas sur l'état de sortie .
Z	Haute Impédance	Etat électrique (et non logique) : Sortie électriquement déconnectée.

Tab. 2 Etats possibles des bornes d'entrées et de sortie en simulation logique.

Dans la plus simple des approximations, on ne considère que les états logiques 0 et 1. Les simulateurs exploitant le langage de description VHDL peuvent prendre en compte jusqu'à 9 états différents [AIR98] lorsqu'ils suivent la norme IEEE 1164.

U	Non-Initialisé	**X**	Inconnu fort	**0**	0 fort
1	1 fort	**Z**	Haute impédance	**W**	Inconnu faible
L	0 faible	**H**	1 fort	**-**	Sans importance

Tab. 3 Etats logiques définis par la norme IEEE 1164.

4.4.1.2 Simulateurs mixtes analogique - numérique.

Un simulateur mixte est composé d'un simulateur numérique qui gère les événements et d'un simulateur analogique qui lui est couplé via des convertisseurs analogique - numérique (CAN) et numérique - analogique (CNA). Pour un simulateur hybride le rôle des événements est notamment de transmettre les évolutions du simulateur numérique au simulateur analogique et inversement. Les événements vont d'abord se traduire par des changements d'états dans le simulateur numérique (comme précédemment). Ils seront ensuite répercutés sur la partie analogique par les CNA par des changements des niveaux de tension. Les échanges d'information et le rythme des événements est étroitement lié à la technique employée pour synchroniser les deux cœurs de simulation.

4.4.1.3 Etats logiques en électrotechnique.

Contrairement au domaine électronique, la notion d'état logique n'est pas rattaché à une grandeur électrique. L'état logique est ici une variable booléenne (vraie ou fausse) qui peut caractériser un élément du circuit et qui peut être transmise à un autre. Dans les cas qui nous concernent, cet état logique va traduire une position mécanique (Actif / repos) de certains

éléments électromécaniques. Ainsi, la transmission de cet état d'un élément à un autre peut traduire une liaison mécanique entre eux.
Notons que rien ne restreint l'application de l'état logique à l'expression d'un état mécanique. La notion de l'état logique n'est pas non plus rattaché à la continuité électrique. Souvent, un état logique exprime la position d'un contact, à savoir s'il est activé ou non. Toutefois, selon si on affaire à un contact à fermeture (NO) ou à ouverture (NF), la position du contact donc la continuité électrique sera différente.

Type de contact	Etat logique	
	0 (Repos)	1 (Actif)
NO	Ouvert	Fermé
NF	Fermé	Ouvert

Tab. 4 Etats logiques de contact NO et NF

Dans le cas de la bobine de relais, par contre, l'état logique actif traduit que celle-ci a reçu une quantité d'énergie suffisante pour déplacer ses contacts. Ce changement d'état n'exprime pas une modification de la continuité électrique du composant. L'état sera toutefois transmis à ses contacts qui sont ses esclaves et dont l'état se calquera sur celui de leur maître.
C'est le modèle de chaque élément du circuit qui définit quels sont les effets de l'état logique sur le comportement du composant ainsi que les changements d'état liés à des événements (variation de tension ou de courant par exemple).

4.4.2 Variables d'état d'un circuit électrotechnique.

Le but de l'analyse événementielle est d'enchaîner des analyses en régime permanent. Les jonctions entre ces analyses constituent les événements où l'on peut voir changer les variables d'état du système, c'est à dire l'ensemble des tensions et courants et des états logiques.
Rappelons que le noyau de résolution analogique du simulateur manipule des valeurs complexes de tension et de courant. Il ne connaît donc que l'amplitude et la phase des signaux pour une fréquence donnée. La résolution ne détermine pas les valeurs instantanées de tension et de courant. C'est uniquement l'interface graphique qui représente les signaux sinusoïdaux. De la même manière, l'analyse événementielle ne connaît pas les valeurs instantanées de tension et de courant bien que l'interface graphique puisse les dessiner. De ce fait, ces valeurs instantanées ne sont pas exploitées par ce mode d'analyse pour déterminer les événements. Les variables d'état du circuit seront donc :

- tension (amplitude, phase et fréquence) en chaque nœud du circuit.
- courant (amplitude, phase et fréquence) en chaque branche du circuit.
- Les états logiques des éléments du circuit.

4.4.3 Séquence de la simulation événementielle.

Les modèles des éléments du circuit peuvent définir le comportement séquentiel à partir des valeurs ou des variations que subissent les variables d'état. Ils devraient donc être sensibles par exemple à un changement de la valeur efficace d'une tension mais pas à la variation de sa valeur instantanée qui ne peut constituer un événement.

4.4.3.1 Succession des étapes de la simulation.

Ainsi lors de l'analyse événementielle, la première résolution analogique en régime permanent peut être suivie immédiatement de l'évaluation de la réaction des composants qui produiront éventuellement des événements futurs. Le simulateur passe alors à la réalisation de l'événement suivant dans l'échéancier puis à la résolution analogique qui en découle. Ce cycle sera renouvelé jusqu'au terme de la simulation. Nous allons donc avoir un répétition de séquences Simulation analogique – Planification – Réalisation de l'événement.
Les figures **Fig. 18** et **Fig. 19** représentent respectivement la répartition des tâches de l'analyse événementielle dans la durée de l'analyse (temps de résolution) et sur le temps virtuel de l'horloge du simulateur qui révèle le comportement du circuit. On notera que les cycles de simulation sont de durées variables car les changements d'états peuvent modifier la dimension du circuit en ouvrant et en fermant certaines branches.

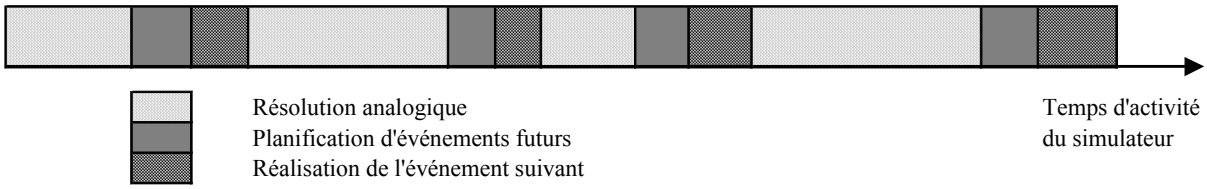


Fig. 18 Répartition des tâches de l'analyse événementielle dans la durée de l'analyse

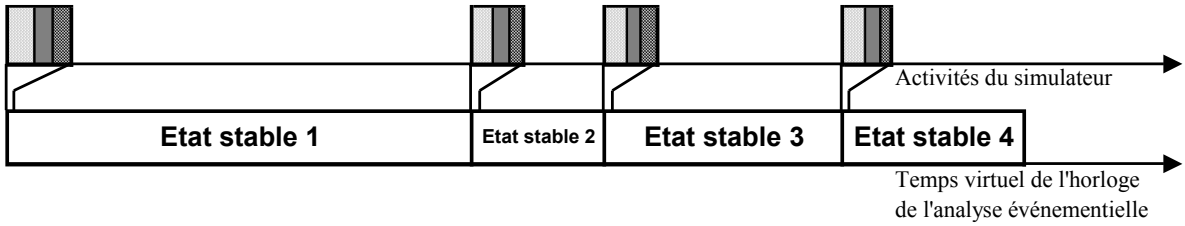


Fig. 19 Répartition des tâches sur le temps virtuel de l'horloge du simulateur

4.4.3.2 Synchronisation des échanges de variables continues et discrètes.

Cette analyse gère comme en simulation mixte analogique – numérique des variables d'état continues et discrètes. En électronique, elle est réalisée par deux noyaux de calcul concurrents. Dans notre cas par contre, la gestion des variables continues est alternée avec celle des variables discrètes. Cela nous rapproche de la technique de synchronisation à pas verrouillé. Or, entre deux synchronisations, on ne réalise aucune simulation bien que l'horloge du simulateur avance, car nous sommes dans un régime permanent. Le traitement des domaines continus et discrets n'est pas concurrent mais alterné.
Maintenant que nous savons quand traiter les variables d'état continues et discrètes, il reste à déterminer comment le faire et de quelle manière nous gérons les échanges entre les domaines continus et discrets. En effet, ici nous avons affaire à un processus unique qui modifie les variables d'état discrètes et continues. Celles ci sont transmises par le calendrier d'événement sans qu'elles soient traitées par un autre processus. En ayant un seul processus, nous n'avons pas d'échange de message ni besoin de synchronisation qui sont les fonctions essentielles de l'analyse orientée processus. Les constats que nous venons de faire nous éloignent d'une solution orientée processus au profit d'une approche orientée événement.

4.4.4 Traitement et transfert de l'information discrète.

L'apport de l'analyse événementielle par rapport à l'analyse en régime permanent est constitué pour l'essentiel par la prise en compte de variables discrètes et de leur interaction avec le comportement continu du circuit.
Contrairement à un circuit logique, dans Simul'Elec, l'information à temps discret (les états logiques) n'est pas traitée. Elle est uniquement transférée du composant à un autre. Aucun composant que nous traitons ne réalise d'opération sur ces valeurs.

De ce fait, faute de traitement à réaliser, il semble superflu de disposer d'un noyau de résolution numérique en plus du noyau analogique comme cela existe en simulation mixte. En rejetant ainsi l'idée d'un deuxième processus de simulation, il nous reste le noyau de résolution analogique que l'on doit piloter au moyen d'un échéancier propre à la simulation par événements discrets. Pour un processus unique (l'algorithme de résolution analogique), il serait injustifié de conserver une approche de simulation orientée processus.
C'est donc assez naturellement que nous avons adopté l'approche orienté événement qui se trouve plus adaptée à notre problématique, bien qu'elle soit, dans un cadre général, moins réaliste et moins performante que l'approche orientée processus.
Ce choix est de plus conforté par l'alternance que nous avons relevée entre le traitement de l'information à temps continu et discret. De cette manière, on écarte les problèmes de synchronisation ainsi que les erreurs de causalité qui peuvent en découler.

4.5 Choix de conception pour l'analyse événementielle.

L'analyse événementielle doit remplir les rôles de tout simulateur piloté par événement soit :

- Génération de nombres et de valeurs aléatoires ou selon des distributions statistiques.
- Gestion de l'avance des temps.
- Détermination du prochain événement.
- Ajout et suppression d'enregistrements à une liste.
- Collecte et analyse statistique des données.
- Report des résultats.
- Détection d'erreur.

Actuellement la génération de nombres ou de valeurs aléatoires ainsi que le traitement statistique des résultats ne présente pas d'intérêt. Ces fonctions n'ont pas été implémentées. Toutefois lors de la conception de l'analyse événementielle, nous avons gardé présent à l'esprit que ces fonctions pourraient devenir nécessaires. L'architecture logicielle retenue peut donc intégrer de nouvelle fonctionnalités.

Pour être adapté au plus près de notre problématique, nous avons redéfini les notions essentielles de ce type de simulation, dans le but d'implémenter les classes d'objets correspondantes :

- L'événement.
- Le calendrier d'événement ou échéancier.
- La notice d'événement (action associée).

4.5.1 Utilisation de Delphi.

L'emploi du langage Delphi pour l'implémentation de ce mode d'analyse fait partie des contraintes techniques définies par Algo'Tech Informatique. De nombreux langages et paquets logiciels pour la simulation par événements discrets existent (voir Chapitre 1). Malheureusement, aucun d'entre eux n'existe pour Delphi, ni le langage Pascal (rappelons que Delphi est un Pascal Objet). Toutefois Law et Kelton [LAW91] présentent plusieurs exemples de tels simulateurs implémentés en Pascal , en langage C et en Fortran. Ceci nous a permis de nous conforter dans la faisabilité de ce mode d'analyse. L'utilisation de Delphi permettra d'ajouter une implémentation orientée objet tout en spécialisant au mieux l'application grâce à la souplesse et à la puissance de ce langage.

4.5.2 Gestion des événements.

On rencontre différentes techniques de gestion des événements. Un événement est généralement défini par un changement d'état du système que l'on simule et qui intervient à un moment bien précis. Par état, on entend l'ensemble des variables qui peuvent caractériser ce système. Pour un circuit électrique, cela peut être aussi bien des tensions, des courants, des états logiques ou les caractéristiques physiques des composants (impédance, puissance, etc.).
Toutes ces méthodes de gestion des événements ont pour rôle d'ordonnancer les événements selon un calendrier qui les regroupe chronologiquement selon la date à laquelle il doivent se produire. Pour permettre le traitement numérique de ces événements, le temps est découpé selon une période d'échantillonnage notée T_e.

4.5.2.1 File Temporelle (Simulateur de circuits logiques).

La file temporelle est le principe le plus couramment utilisé dans les simulateurs de circuits logiques [NEW78]. Cette file est un tableau à une dimension dont chaque case représente un instant de la simulation **Fig. 20**. Le temps séparant deux instants est fixe et correspond à la période d'échantillonnage définie pour cette simulation (automatiquement ou par l'utilisateur).
La première case est le temps 0 où débute la simulation et la dernière représente le dernier instant de la simulation.
Chaque case répertorie tous les événements devant se produire en cet instant. Dans le cas d'un simulateur de circuit logique, chaque événement correspond au changement d'état logique d'une borne. Certaines cases peuvent donc rester vides si aucun événement n'est planifié. Lorsqu'un nouvel événement est prévu, s'il correspond au passage à un état d'une borne identique à l'état courant, l'événement est annulé et n'est pas ajouté au calendrier. Par contre, si l'état prévu est différent de l'état courant, un nouvel événement est ajouté au calendrier. On peut dissocier la file temporelle (le calendrier) qui constitue l'ensemble des données relatives aux événements et le planificateur qui assure le contrôle du calendrier.

4.5.2.2 Extension à la simulation hybride.

SPLICE (à ne pas confondre avec SPICE) décrit par [NEW78] fut l'un des premiers simulateurs de circuits électriques mixtes à réunir un simulateur analogique comme SPICE et un simulateur numérique. Les algorithmes d'analyse de SPLICE sont contrôlés par un planificateur d'événements similaire à ceux employés par la simulation logique à file temporelle. Contrairement à un simulateur logique, les éléments pris en compte sont non seulement des portes logiques mais également des éléments ayant un comportement continu

(non discret) ou encore des groupes de transistors. La nature des événements ne change pas dans le sens où les modifications liées qui y sont liées restent des changements d'état logique (et les changements de niveaux de tension qui en découlent).

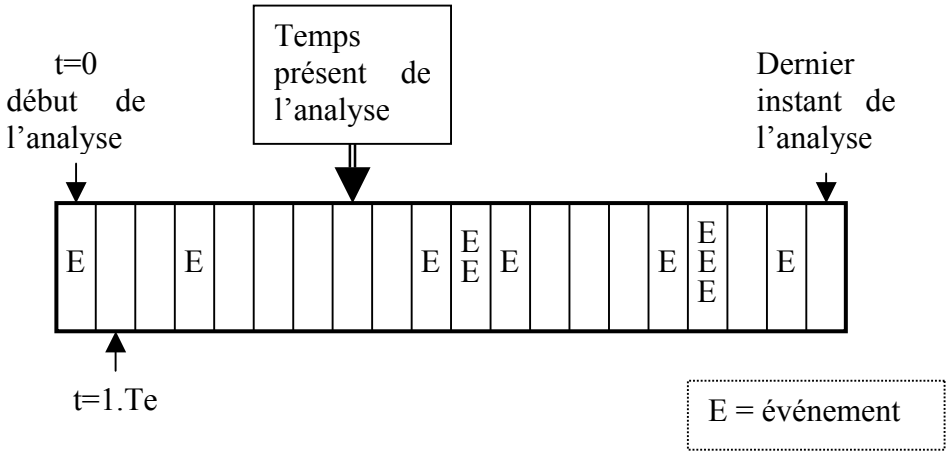


Fig. 20 File temporelle de gestion des événements d'un simulateur de circuit logique.

4.5.2.3 Application à l'électrotechnique.

La technique de file temporelle semble assez bien adapté à notre cas. En effet, nous souhaitons simplement cadencer une succession d'analyse en régime permanent alternée avec les évaluations et les réalisations des événements. Toutefois, le rôle du calendrier d'événement qui gère les événements ne se résume pas uniquement à une planification et réalisation systématique de ces derniers. Il convient notamment de gérer des réalisations conditionnelles ou encore des blocages définis par l'utilisateur.

Dans Simul'Elec, la planification d'un événement traduit la conséquence d'un état stable (régime permanent) sur les éléments du circuit. Or dans certains cas, l'événement planifié ne saurait être réalisé si un événement contradictoire est évalué avant sa réalisation, même s'il sera planifié ultérieurement. Cela peut traduire l'inertie des composants. C'est par exemple le cas d'un relais dont le changement d'état est planifié en fonction du franchissement d'un seuil de tension haut ou bas. Ce changement d'état ne sera effectif que si la condition qui l'a provoqué est maintenue jusqu'à sa réalisation.
On retrouve la même problématique en simulation de circuit logique lorsque l'on doit prendre en compte les temps de propagation des portes logiques par exemple.

Le second point d'importance concerne le forçage de l'état logique. En effet, l'utilisateur peut vouloir forcer un état logique pendant une certaine durée de temps afin de simuler un dysfonctionnement par exemple. Dans ce cas, d'éventuels événements pouvant modifier ces états forcés doivent être ignorés pendant ce laps de temps. Il est donc nécessaire de mettre en place des mécanismes permettant de maintenir ces forçages.

4.6 Implémentation de l'analyse événementielle.

Notre préoccupation est d'offrir un mode d'analyse piloté par événement qui permette de suivre l'évolution naturelle ou forcée de l'installation traitée et pas de simuler des composants électroniques comme peut le faire SPLICE.
La simulation mixte telle qu'on la retrouve dans SPLICE a pour but essentiel d'obtenir les résultats d'une manière plus rapide qu'avec un simulateur exclusivement analogique. Pour cela, on fait l'approximation que les composants dits logiques ont un comportement logique et seront ainsi traités par l'algorithme de simulation logique. En réalité, les portes logiques sont constituées en partie de transistors fonctionnant en mode bloqué - saturé qu'on peut simuler analogiquement avec un temps de calcul bien supérieur. Dans ce cadre, l'intérêt de la simulation pilotée par événement est de modéliser simplement ces changements d'états logiques. La nature des actions événementielles est donc restreinte au changement d'états logiques.

L'objectif de cette analyse est de reproduire virtuellement le fonctionnement de l'installation, ainsi que toutes les manipulations que l'utilisateur pourrait faire d'une manière expérimentale. Ces interventions ne doivent pas être restreintes et on doit permettre de réaliser des actions qui ne seraient pas faisables pratiquement dans la réalité comme des simulations de panne du matériel ou encore de courts-circuits.

4.6.1 Notions d'état et d'événement.

Les besoins d'interventions de l'utilisateur sur le système nous amènent à reconsidérer les notions d'événement et d'état de l'installation. Gardons à l'esprit qu'un événement est représentatif d'un changement d'état du circuit. Un simulateur mixte classique réduit l'état à l'ensemble des états du circuit. Pour couvrir toute la gamme des évolutions possibles, nous devons revenir à une définition plus globale de l'état. en ce qui concerne les résultats de simulation, le changement d'état du circuit se traduit par l'évolution :

- Des tensions des nœuds du circuits (valeurs complexes, pas instantanées).
- Des courants traversant ses branches (valeurs complexes, pas instantanées).
- Des états logiques des composants.

Il est important de noter que l'état logique d'un composant n'est pas ici le reflet d'un niveau de tension ou de courant comme en simulation logique mais l'image d'une configuration physique du composant (position d'un interrupteur, d'un relais, d'un fusible).
Les interventions du simulateur ou de l'utilisateur menant à ces changements d'état peuvent être de natures différentes :

- Changement de l'état logique d'un composant.
- Modification d'un paramètre d'un composant.

Ces actions vont aller modifier la modélisation du composant et ainsi influer sur le résultat de simulation. Elles peuvent être dues au système (déclenchement d'un disjoncteur, fermeture d'un relais) ou résulter d'un forçage provoqué par l'utilisateur (appui sur un bouton, etc.).

4.6.2 Signaux traités.

L'analyse événementielle repose sur l'enchaînement de plusieurs analyses en régime permanent. Cet enchaînement est piloté par un calendrier d'événements. Chaque événement correspond à une modification de l'état du circuit et provoque une nouvelle résolution du système d'équations. Toutes les solutions successivement obtenues décrivent tous les états stables que prend le circuit sur la durée de l'analyse. L'instant où se produit l'événement et

qui sépare 2 états stables est appelé transition. L'analyse ne tient pas compte des phénomènes transitoires, mais on se réserve toutefois la possibilité d'intégrer par la suite une forme de calcul de phénomènes transitoires.

4.6.2.1 Stimuli : variables d'entrée.

L'utilisateur peut planifier un certain nombre de modifications successives ou simultanées à appliquer sur le circuit. On peut ainsi tester de multiples configurations dans le but de vérifier le fonctionnement de l'installation ou de trouver des valeurs optimales de composants par exemple. Il existe 2 types d'interventions :

- Changement de la valeur d'un paramètre d'un composant. Tous les paramètres du modèles sont modifiables. Ces changements peuvent être effectués de manière simple ou l'utilisateur définit les valeurs une par une mais également de manière automatique en prévoyant des balayages de valeurs selon des échelles linéaires, logarithmiques ou en respectant des séries normalisées (E12…).
- Forçage de l'état logique des composants (fermeture ou ouverture interrupteur, déclenchement disjoncteur, etc.). Le terme forçage indique que quoiqu'il arrive , c'est l'état défini par l'utilisateur qui est prioritaire sur l'état naturel du composant. Les 3 états que l'on peut définir sont :
- Aucun forçage : on laisse le système définir l'état du composant.
- Forçage à l'état actif.
- Forçage au repos.

4.6.2.2 Sorties : états et courbes.

Par sorties, on entend tous les résultats de l'analyse que l'on peut exploiter :

- Les tensions et courants mesurés (sous forme de courbes ou de valeurs efficaces).
- Les états logiques résultants. Ils peuvent être différents des états prédéfinis dans les zones sans forçage.

4.6.3 Structure de données retenue.

Le simulateur SPLICE implémenté en Fortran utilise une structure qui comprend :

- un calendrier d'événements qui est une file temporelle qui contient des pointeurs sur les événements à réaliser.
- Un planificateur qui gère la mise en place des événements dans la file.
- Les événements qui contiennent les actions à réaliser.
- Un algorithme qui déroule le temps de simulation et déclenche les événements planifiés.

L'emploi de Delphi qui est un langage orienté objet nous a permis d'avoir une structure de données plus lisible et plus flexible. On y retrouve :

- L'algorithme de simulation événementielle qui gère toute l'analyse et englobe l'algorithme de résolution analogique en régime permanent.
- Le calendrier d'événements qui tient les rôles de file temporelle et de planificateur.
- Les événements qui contiennent les actions de modification et stockent les résultats de simulation qui en découlent.

Les événements sont associés à un objet instance de la classe *TSimEvent*. Ils sont tous contenus dans une table qui fait partie de l'objet unique instance de la classe *TEventSchedule* qui est l'implémentation du calendrier d'événements (**Fig. 21**).

On trouve de plus des objets de moindre importance qui complètent et structure l'information. Il permettent d'identifier plus rapidement certaines informations:

- Classe d'objets *TForcedEvent* qui désignent les événements forcés par l'utilisateur.
- Classe d'objets *TActionWatcher* qui désignent les actions à surveiller.
- Classe d'objets *TEventAction* qui implémente les actions associées aux événements.

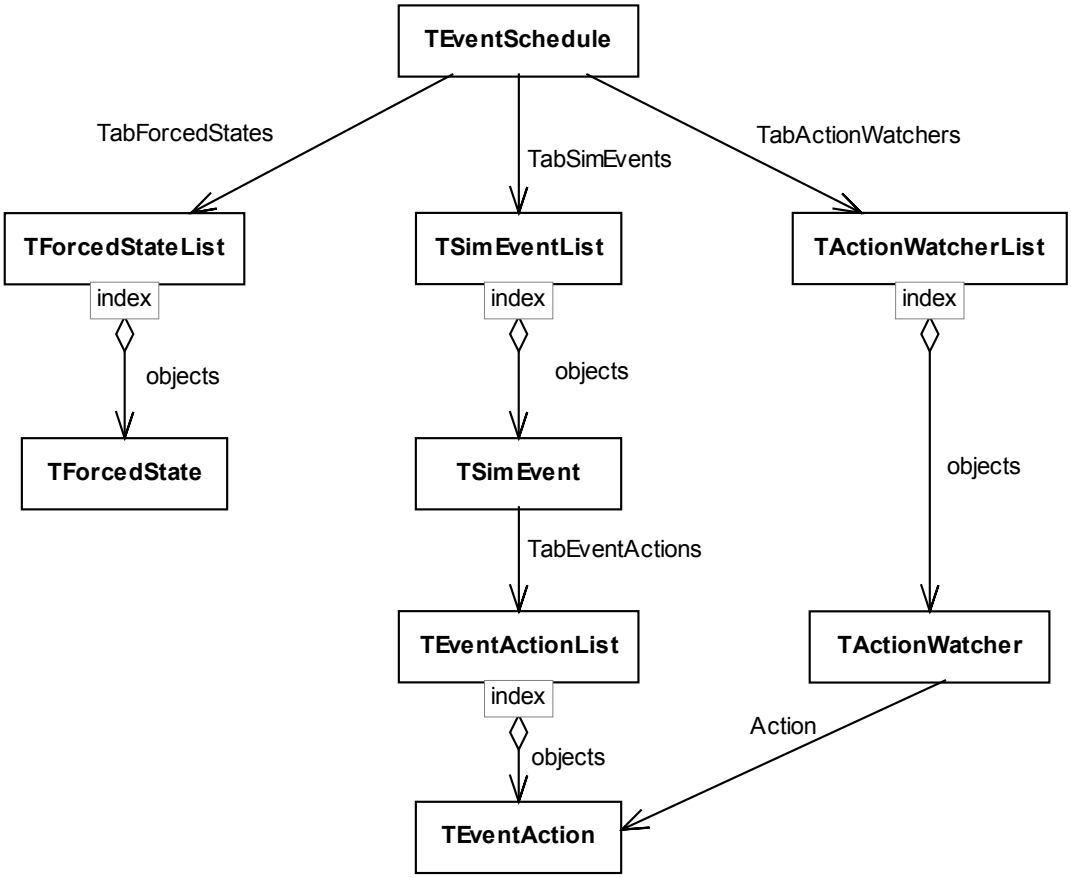


Fig. 21 Relations entre les classes de l'analyse événementielle.

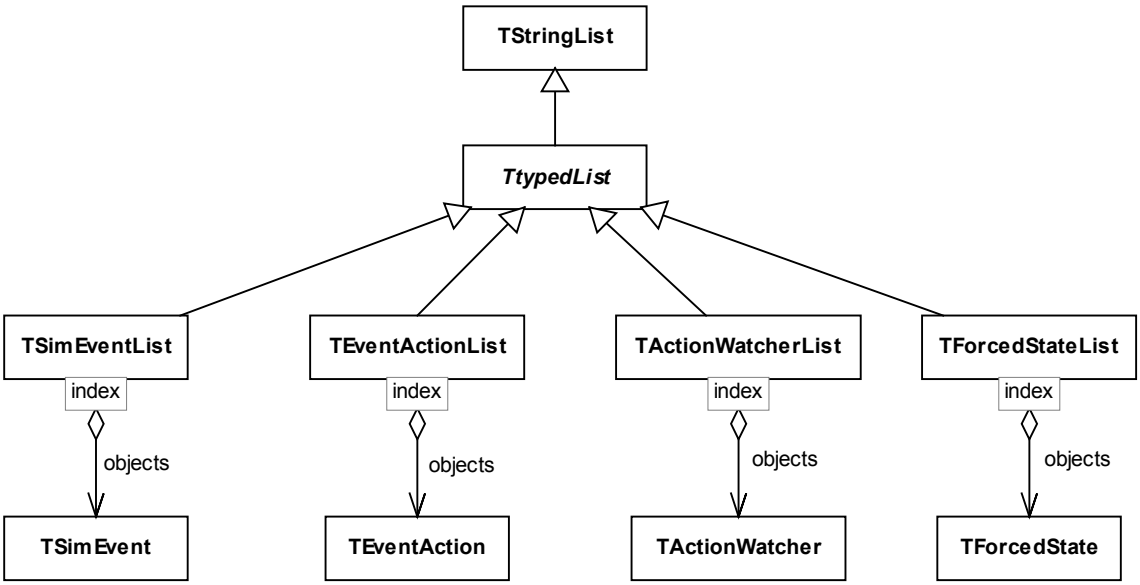


Fig. 22 Héritage des tables de hachage de l'analyse événementielle

4.6.4 Gestion des types.

Les classes de l'analyse événementielle contiennent un certain nombre de tables de hachage permettant de pointer sur un certain nombre d'objets (**Fig. 21**). Chaque table contient toujours le même type d'objet. Il existe en Delphi la classe *TStringlist* définissant une table de hachage générique. Cette table peut contenir tout type d'objet dérivé de la classe *TObject* qui est la classe la plus fondamentale en Delphi. Ce fonctionnement est problématique car chaque table ne doit contenir qu'un type précis d'objet et l'utilisation de la *TStringlist* ne permet pas de le contrôler. De plus Delphi est un langage fortement typé qui nécessite de forcer le type d'un objet lorsqu'on le récupère d'une *TStringlist* après avoir testé son type. Nous avons ainsi défini une classe abstraite *TTypedList* dérivant de *TStringlist* dont une des propriétés est le nom de la classe des objets qu'elle peut contenir. Nous avons ainsi implémenté les quatre tables de hachages *TSimEventList*, *TEventActionList*, *TActionWatcherList* et *TforcedStateList* qui peuvent contenir respectivement des objets *TSimEvent*, *TEventAction* , *TActionWatcher* et *TForcedState* (**Fig. 22**). Ainsi pour chaque table, un contrôle de conformité est automatiquement effectué lors de l'insertion de l'objet dans la liste. L'ajout d'un objet non conforme est impossible et génère une exception. On évite ainsi d'avoir à contrôler le type et à le forcer a posteriori, améliorant ainsi la stabilité du code.

4.6.5 L'algorithme d'analyse événementielle.

Cet algorithme **Fig. 23** va initialiser un nouveau calendrier d'événements à partir des stimuli définis par l'utilisateur. Il va ensuite faire évoluer l'état du circuit en fonction des événements planifiés et pour chaque événement, il déclenche une nouvelle résolution. Après cette résolution, les résultats sont stockés dans l'objet événement associé à cet instant. En fonction de ces résultats, on teste la réaction du circuit à ce nouvel état **Fig. 24**. On va consulter chaque composant du circuit pour déterminer si de nouveaux événements doivent être planifiés. Lorsque l'analyse est terminée, il met à jour l'affichage des signaux.

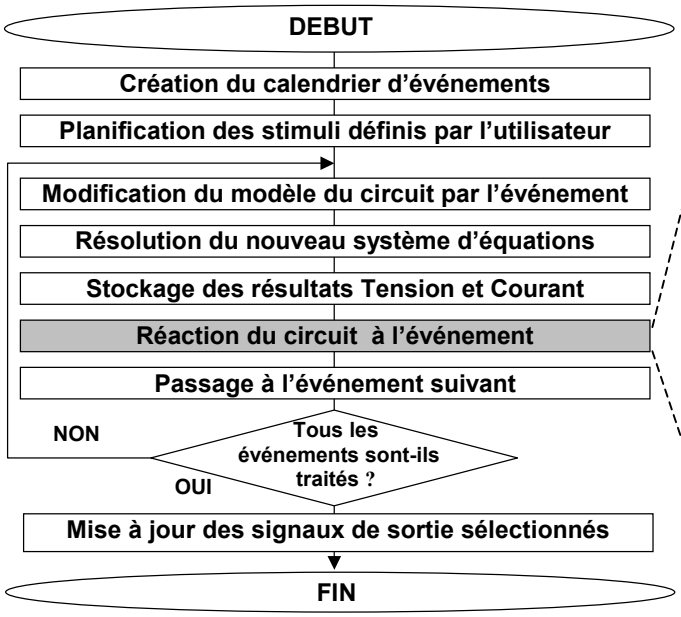


Fig. 23 Algorithme de l'analyse événementielle.

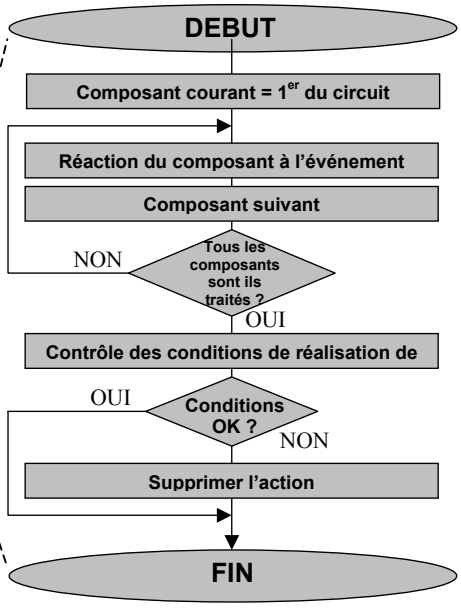


Fig. 24 Détail de la réaction à l'événement.

4.6.6 Description d'un événement : classe TSimEvent.

Toutes les données et opérations directement liées à la notion d'événement sont regroupées dans la classe d'objet *TSimEvent* que l'on désignera plus simplement par événement. Chaque instance de cette classe correspond à un événement, c'est à dire un changement d'état du circuit simulé. Les événements sont stockés dans le calendrier d'événements chronologiquement en fonction de leur date de réalisation. On retrouve ici une notion d'événement légèrement différente de celle précédemment décrite. L'événement contient non seulement une liste de modifications (actions) à effectuer sur le circuit mais également tous les résultats de simulation consécutifs à ces modifications (tensions, courants et états logiques. Il convient de distinguer les actions des événements. Une action est une modification unique d'une caractéristique d'un composant. Aucune notion temporelle n'est attachée à l'action. L'événement intègre par contre cette notion de temps, il regroupe toutes les actions devant se réaliser à un instant précis. Il prend également en charge le stockage de l'état résultant des modifications qu'il a appliqué au circuit soit les courants, tensions et états logiques.

4.6.6.1 Attributs de *TSimEvent*.

Attributs	Type	Dim	Rôle
NTe	Entier		Indique l'instant auquel doit se réaliser l'événement en nombre de périodes d'échantillonnage. A ne pas confondre avec l'instant auquel il a été intégré dans le calendrier.
EventActionList	Tableau de pointeurs	**Na**	Liste des **Na** actions à réaliser. Ces actions de modification du circuit sont implémentées par les objets de la classe TEventAction .
TabBranchCurrent	Tableau de réels	**Nf**x**Nb**	Contient les **Nb** courants parcourant tous les composants (leurs branches) du circuit et pour toutes les **Nf** fréquences simulées. Chaque colonne de la table correspond à une branche et chaque ligne à une fréquence. Le contenu de ce tableau et du suivant est directement issu de la résolution.
TabNodeVoltage	Tableau de réels	**Nf**x**Nn**	Contient les Nn tensions de chaque nœud du circuit et pour les Nf fréquences. Chaque colonne de la table correspond à un nœud et chaque ligne à une fréquence.
TabPulsations	Tableau de réel	**Nf**	Liste des Nf pulsations couvertes par l'analyse. En recoupant avec les 2 tableaux précédents, on peut reconstituer tous les signaux de courants et tensions (série de Fourier).
TabCurrentRMS	Tableau de Réel	**Nb**	A partir des tableaux de courants et de fréquence, on calcule les valeurs efficaces vraies sur une période correspondant à la fréquence la plus faible.
TabVoltageRMS	Tableau de Réel	**Nn**	Même calcul appliqué aux tensions de chaque nœud.
TabCircState	Tableau de Booléens	**Nc**	Contient les états logiques de tous les composants que l'on trouve sur le schéma
TabCompState	Tableau de Booléens	**Ne**	Contient les états logiques de tous les éléments (composants de base) que l'obtient par décomposition hiérarchique des éléments du circuit.

Tab. 5 Liste des attributs de la classe *TSimEvent*.

On peut dissocier ces propriétés en 2 catégories :

- Nte et EventActionList sont les propriétés avant événement qui concernent la modification du circuit.
- Les autres sont les propriétés après événement et contiennent les résultats de la résolution liés à cet événement.

Les dimensions de ces résultats de la résolution peuvent varier d'un événement à l'autre selon les actions à réaliser. Elles sont déterminées par :

- Nn : nombre de nœuds du circuit.
- Nb : nombre de branches du circuit.
- Nf : Nombre de fréquences différentes présentes dans le circuit.
- Nc : Nombre de composants présents sur le schéma (ne varie jamais lors de l'analyse).

Ne : Nombre d'éléments (composants de base) qui forment le circuit (ne varie jamais lors de l'analyse).

4.6.6.2 Méthodes de *TSimEvent*.

La programmation orientée objet permet de demander à notre objet événement d'effectuer des opérations sur lui-même. Opérations qui seraient autrement réalisées par des routines dissociées de la structure de données « événement ». On peut ainsi associer à l'objet *TSimEvent* des méthodes le concernant directement.

Méthodes	Rôle
Create	Création d'un événement. On lui passe en paramètre la première action qu'il devra réaliser et à quel instant.
AddAction	Ajoute l'action passée en paramètre à sa liste.
MakeEvent	Réalise les actions qui se trouvent dans la liste.
StoreSimulation	Stocke les résultats de simulation (courants, tensions et états) dans ses tableaux et effectue les calculs de valeurs efficaces.
ReactToEvent	Après l'événement et en fonction des résultats qu'il a enregistrés, il scrute tous les éléments du circuit pour déterminer si on doivent planifier d'autres événements en réaction.
Destroy	Libère l'objet

Tab. 6 Liste des méthodes de la classe *TSimEvent*.

4.6.7 Gestion des actions : classe TEventAction , TActionWatcher .

Le calendrier d'événements regroupe tous les événements de l'analyse. Chacun contient une ou plusieurs actions décrivant une modification à appliquer sur le circuit. Les actions sont implémentées sous forme d'un objet de la classe *TEventAction* . Une action va modifier une seule propriété d'un élément (*Telement*) qui peut être un de ses paramètres ou son état logique.

Attributs	Type	Rôle
ThisElement	Telement	Composant que l'on doit modifier.
ParamIndex	Entier	Indice du paramètre que l'on doit modifier. S'il vaut –1 on va modifier l'état.
NewValue	Réel	Nouvelle valeur du paramètre à modifier.
NewState	Booléen	Nouvel état à appliquer au composant

Tab. 7 Liste des attributs de la classe TEventAction .

Méthodes	Rôle
Create	Créer une action en suivant les instructions passées en paramètre : élément nouvelle valeur du paramètre, nouvel état.
DoAction	Réalise la modification sur le circuit.
Destroy	Libère l'objet.

Tab. 8 Méthodes de la classe *TEventAction* .

Certaines actions sont particulières dans le sens où entre l'instant où on va les planifier et celui où elles s'exécuteront, il faut que les conditions qui ont produit l'action soient maintenues. Ces actions conditionnelles sont considérées comme des actions « à surveiller » et concerne uniquement les actions produites par le simulateur. C'est par exemple le cas du changement d'état d'un relais suite à sa mise sous tension. A l'instant où la bobine du relais est alimenté sous une tension suffisante pour l'activer, le simulateur planifie une action de fermeture du relais (passage à l'état activé) dans un temps correspondant au retard à la fermeture. Mais cette action ne doit se réaliser que si ces conditions sont maintenues jusqu'à l'instant d'exécution. On attache donc un observateur à l'action (*TActionWatcher*) qui précise le caractère conditionnel de l'action. L'observateur a un rôle totalement passif et ne peut pas gérer la réalisation ou non de l'action qu'il désigne. C'est le planificateur qui par la présence de l'observateur va déterminer le maintien ou le retrait de l'action du calendrier. Tous ces objets *TActionWatcher* sont regroupés dans la liste ActionWatcherList du calendrier d'événement.

Attributs	Type	Rôle
Action	TEventAction	Désigne l'objet Action à surveiller
Event	TSimEvent	Désigne l'événement contenant l'action à surveiller
Méthodes	**Rôle**	
Create	Instancie un observateur sur l'action qu'on lui passe en paramètre.	
Destroy	Libère l'objet.	

Tab. 9 Attributs et méthodes de la classe TActionWatcher .

4.6.8 Calendrier d'événements : classe TEventSchedule.

La classe *TEventSchedule* décrit la planification des événements. Il ne peut exister qu'un seule instance de cette classe qui correspond à l'échéancier des événements de l'analyse événementielle en cours. Cet objet va permettre à l'algorithme de l'analyse événementielle (**Fig. 23** et **Fig. 24**) de piloter les différentes simulations successives. Il assure les fonctions de calendrier d'événements et de planificateur, on le retrouvera sous les désignations de calendrier, échéancier ou planificateur si on parle de ses données ou de ses procédures. Il assure non seulement la mise en place des événements dans son échéancier en fonction de leur date d'exécution mais gère également leur éventuel retrait. Certains seront retirés du calendrier avant leur instant d'exécution car il ne satisfont plus aux conditions nécessaires décrites dans le modèles du composant à modifier (voir le cas de l'inertie du relais).

4.6.8.1 Attributs de *TEventSchedule*.

Parmi ces propriétés, on a d'une part des données liées à l'échéancier, au stockage des événements et d'autre part des informations qui ont pour but d'identifier des actions particulières qui ont une influence sur la planification d'événement futurs.

Attributs	Type	Visibilité	Rôle
SimEventList	Tableau de pointeurs	Privée	Liste des événements à réaliser (objets TSimEvent).
SimEvents[i]	TSimEvent	Publique	Evénement i de la liste précédente.
EventCount	entier	Publique	Dimension de la liste d'événements.
ForcedStateList	Tableau de pointeurs	Privée	Liste de tous les états forcés TForcedEvent résultant de la planification des entrées.
ForcedStates[i]	TforcedState	public	Etat forcé i de la liste précédente
ActionWatcherList	Tableau de pointeurs	Privée	Liste de tous les observateurs d'actions TActionWatcher .
ActionWatchers[i]	TActionWatcher	Publique	Observateurs d'action i de la liste précédente.

Tab. 10 Liste des attributs de la classe *TEventSchedule*.

4.6.8.2 Méthodes de *TEventSchedule*.

La plupart des méthodes du calendrier d'événements réalisent les procédures de planification, d'ordonnancement des événements et des actions. Cela comprends aussi bien l'insertion dans la file d'attente que le retrait de cette file.

Méthodes	Rôle
create	Création et initialisation de l'objet.
PlanInput	A partir des stimuli d'entrées défini par l'utilisateur, il planifie les événements correspondants.
AddEvent	Ajoute à la liste l'événement passé en paramètre.
AddAction	Ajoute l'action passée en paramètre à l'instant passé en paramètre : Si un événement existe en cet instant on ajoute l'action à cet événement sinon on crée un nouvel événement pour cet action.
AddWatchedAction	Ajoute une action comme la méthode précédente et lui accole un observateur (*TActionWatcher*).
EventIndexOf	Renvoie l'indice de l'événement planifié à l'instant reçu en paramètre.
RemoveAction	Retire l'action passée en paramètre. Si c'est la seule action de l'événement concerné, il n'est pas détruit ce qui permet de garder la trace de ce retrait.
Destroy	Vide toutes ses listes, libère l'objet.

Tab. 11 Liste des méthodes de la classe *TEventSchedule*.

4.6.8.3 Planification des entrées.

Au début de l'analyse événementielle, l'algorithme demande au calendrier de planifier tous les signaux d'entrée définis par l'utilisateur. Chaque transition décrite par ces stimuli va produire un objet action (*TEventAction*) qui sera contenu dans un événement lié à l'instant d'exécution de l'action. Les actions peuvent être soit la modification d'un paramètre d'un composant ou de son état logique. Dans le cas du changement d'état, on a affaire à un forçage, où l'utilisateur impose l'état que doit prendre un composant pendant un temps déterminé. Quelque soit l'état naturel calculé par le simulateur, c'est l'état forcé qui sera appliqué. Pour tout forçage, le calendrier va créer un objet Indicateur d'état forcé de la classe *Tforce dState* qui précise quel composant on force ainsi que les dates de début et de fin de cette modification. Tous ces objets sont regroupés dans la liste ForcedStateList du planificateur d'événement. Si le simulateur veut rajouter une action de changement d'état pendant cet intervalle le calendrier la rejettera. Il faut noter que l'on en retrouve pas de notion de forçage à propos des modifications des paramètres. En effet, la valeur d'un paramètre d'un composant ne peut évoluer qu'à la demande de l'utilisateur. Le simulateur ne peut pas faire changer ces valeurs en réaction à un événement comme il le fait avec un état logique. Ces paramètres permettent de renseigner le modèle pour décrire le comportement du composant en toutes circonstances. Changer une de ces valeurs équivaut à remplacer ce composant par un autre et non pas à traduire une évolution de son état comme c'est le cas avec l'état logique.

Chapitre 3

Modélisation.

1 Contexte – démarche de modélisation.

Nous allons d'abord rappeler le contexte lié au développement de notre simulateur et à la modélisation des composants de façon à justifier nos différents choix lors de la démarche de modélisation.
Notre étude est consacrée aux simulateurs de circuits électriques orientés vers l'électrotechnique et l'électricité des courants forts pour un usage « grand public », typiquement bureau d'étude de PME et enseignement. L'outil et les modèles doivent donc être simples d'emploi et minimiser les risques d'obtenir des résultats erronés. Les résultats doivent être obtenus rapidement. Nous privilégierons d'avantage la rapidité de la résolution plutôt que la finesse des résultats. La modélisation s'appuie sur les techniques et le prototype décrits dans le chapitre précédent.

1.1 Domaine traité.

Le simulateur Simul'Elec est orienté vers les circuits électrotechniques. Le rôle essentiel d'un simulateur de circuits électriques est d'estimer la valeur des tensions et des courants présents en tout point du circuit. Ainsi dans un premier temps, nous traiterons des circuits en nous basant sur des relations électriques entre tension et courant. Cependant, le comportement d'un circuit électrique n'est pas uniquement régi par des lois électriques mais également par des phénomènes dépendant de domaines tels que la thermique, la mécanique ou encore l'optique. En outre, les installations à simuler ne sont pas toujours purement électriques. On va trouver fréquemment des composants pneumatiques, hydrauliques, électropneumatiques, etc. Afin de pouvoir simuler ces différents composants dans leur environnement, il est nécessaire de ne pas restreindre le simulateur au domaine électrique, aussi bien par la conception de l'outil que par la méthode de modélisation retenue. Nous nous sommes donc naturellement intéressés par la suite à des principes de modélisation et des langages comme VHDL-AMS [VHD2], qui permettent des descriptions couvrant de multiples domaines.

1.2 Contraintes.

La modélisation est fortement dépendante du simulateur. Ainsi elle est attachée à certaines contraintes inhérentes à tous projets de Recherche et Développement.

1.2.1 Liées à l'existant.

En partant de ce prototype de simulateur, on définit la démarche de modélisation ainsi que la base d'évolution du simulateur et des modèles. La méthode de calcul utilisée par le simulateur influe directement sur le contenu des modèles et le type de comportement qu'ils peuvent décrire.

1.2.2 Liées aux moyens humains.

Ce projet de recherche et développement est à une échelle réaliste qui correspond à une niche technologique assez ciblée. Les moyens humains mis en œuvre pour ces travaux reposent sur l'auteur de cette thèse appuyé par son encadrement scientifique, Hervé Lévi (IXL, ENSEIRB, Bordeaux 1, CNRS) qui intervient sur l'aspect modélisation et modes d'analyse, Nadine Rouillon Couture (LIPSI – ESTIA) qui apporte son expertise à la conception des outils logiciels, avec l'apport ponctuel de Renaud Briand (LIPSI-ESTIA) [LEG03] et l'expertise en développement informatique d'Algo'Tech. Ces moyens, limités, nous imposent de cibler les efforts de recherche et de développement sur le développement de fonctionnalités, de modes d'analyses et de modèles originaux innovants et performants.
Rappelons que SPICE existe depuis plus de 30 ans (première version en 1972 [NAG75]) et que de nombreuses équipes de recherche et développement n'ont eu de cesse d'améliorer l'outil, de développer des modèles ou de créer des outils alternatifs, sans encore parvenir au simulateur idéal réalisant le compromis entre la facilité d'utilisation, la finesse et la rapidité des résultats. Il est donc indispensable d'avoir des objectifs à court terme réalistes (bien qu'ambitieux) aussi bien concernant les capacités de l'outil que la finesse des résultats.

1.2.3 Technologiques.

Les principales contraintes technologiques sont liées à l'environnement de développement et à l'environnement cible. L'application « schématique+simulation » est développée en langage Delphi [DEL] (Pascal objet). Elle est destinée à fonctionner sur des PC standard sous MS Windows. Le simulateur doit être précis et rapide dans ce type de configuration. Notons qu'au début de ces travaux (janvier 2001) un ordinateur équipé d'un processeur de type Pentium 3 à 1Ghz était considéré comme très performant.

1.2.4 Liées aux utilisateurs.

Dans la majorité des cas, les simulateurs analogiques de circuits électriques, Spice et ses dérivés notamment, sont avant tout destinés à un public averti, possédant une bonne perception de l'outil et des modèles (ingénieurs, chercheurs). De ce fait, l'accent a été généralement mis sur la fiabilité des résultats, la finesse de paramétrage des modes d'analyse, au détriment parfois de l'ergonomie et de la simplicité d'utilisation. D'autre part, il existe des outils didactiques tels que Schémaplic [FIT] ou Crocodile [CRO] qui offrent des possibilités basiques de simulation conviviales mais plus approximatives. Simul'Elec et ses modèles sont destinés à des utilisateurs qui peuvent avoir une exigence dans la précision des résultats relativement importante mais qui attendent une grande simplicité d'utilisation. Comme nous

l'avons détaillé au chapitre 2, parmi ces utilisateurs, on peut distinguer deux types de public qui vont déterminer la démarche de modélisation :

- Technicien bureau d'étude : approche métier.
- Enseignement : approche pédagogique.

1.3 Démarche de modélisation.

Le processus de modélisation s'est déroulé en alternance avec le développement de l'outil de simulation, pour ne pas dire en parallèle. Les modèles implémentés doivent être exploitables en totalité par le simulateur. Ainsi, les besoins d'obtenir des résultats plus fins et de nouveaux modèles amènent à modifier le simulateur pour améliorer ses performances ou ajouter de nouveaux modes d'analyses par exemple. D'autres part, les améliorations du simulateur permettent d'affiner la description des modèles. On a donc une interdépendance très forte entre simulation et modélisation.

1.3.1 Modélisation et développement de l'outil.

La **Fig. 1** ci après montre les différentes étapes du processus de développement. L'écriture et la validation des modèles font apparaître les défauts et les lacunes de l'outil et entraînent l'évolution et la correction du simulateur Simul'Elec. Avant la traduction des modèles en VHDL-AMS, les modèles restent inscrits dans le programme de simulation lui-même. Bien que peu flexible et fermé, ce principe permet de maintenir une bonne cohérence entre les modèles et le simulateur. En effet, l'ensemble est compilé et exécuté simultanément. Les outils de déboguage de Delphi permettent en cas de problème de suivre le comportement du simulateur et des modèles, ce qui serait impossible avec des modèles compilés séparément. C'est donc une solution très satisfaisante, durant la phase de stabilisation et de fiabilisation du simulateur. Nous verrons par la suite que pour passer du stade de prototype à celui d'un logiciel ouvert vers l'utilisateur, il est nécessaire de rendre la modélisation accessible à tous via un langage de modélisation externe.

Etape	1	2	3	4	5	6	7	8
Simulateur	Prototype SiCi			Amélioration Outil et interface		Analyse événementielle		
Modélisation		Premières primitives	Ajout de primitives		Modèles structurels plus complexes		Modèles événementiels	Traduction VHDL-AMS

Fig. 1 Séquence des étapes de développement du simulateur et de modélisation

Parmi les grandes étapes du développement du simulateur nous allons dissocier les deux premières qui sont antérieures à cette thèse :

1 / Développement du premier prototype « SiCi ».
2 / Modélisation des premières primitives :

- *Source de tension et de courant sinusoïdales.*
- *Résistance, inductance, condensateur.*
- *Interrupteur, fil conducteur.*

Ensuite, nous retrouvons celles qui font partie du travail de thèse :
3 / Enrichissement de la bibliothèque des primitives et amélioration de celles existantes.
- Sources continues, carrées.
- Quadripôles élémentaires : sources contrôlées, transformateur idéal.

4 / Amélioration de l'outil et de l'interface. Fiabilisation.
5 / Ecriture de macro-modèles (sous-circuits) plus complexes :
- Moteurs divers mono et triphasés.
- Transformateurs réels mono et triphasés.
- …

6 / Développement de l'analyse événementielle.
7 / Modélisation de composant ayant un comportement séquentiel :
- Bobine et contact de relais.
- Dispositif de protection : disjoncteurs fusibles.

8 / Traduction des modèles existants en VHDL-AMS.

1.3.2 Choix des modèles.

A priori, les composants à modéliser sont ceux que les utilisateurs souhaitent pouvoir simuler avec l'outil que l'on propose. Cependant, les modèles à développer dépendent également des possibilités du simulateur. Il serait en effet superflu de modéliser des composants si la totalité de leur description ne peut pas être traitée par l'outil de simulation.

Afin de caractériser les modèles, nous avons rassemblé des informations et des connaissances sur les composants concernés. D'une manière générale, nous avons essayé d'utiliser des sources d'information accessibles aux utilisateurs du simulateur. Ainsi, on peut obtenir des modèles compréhensibles par l'utilisateur et dont il doit savoir renseigner les paramètres à partir de ses connaissances et des documents à sa disposition. En effet, il serait contradictoire de développer des modèles extrêmement fins au paramétrage complexe s'ils ne peuvent pas être utilisés.
Toutefois, un modèle simple à paramétrer n'est pas nécessairement simpliste dans la description de son comportement. C'est par exemple le cas du disjoncteur magnétothermique dont le modèle reçoit en paramètre une simple courbe caractéristique de fonctionnement normal. Le modèle va prévoir des phénomènes complexes que cette courbe ne définit pas mais qui sont nécessaires pour une représentation réaliste de son fonctionnement.

Les modèles les plus simples, comme ceux des primitives sont inspirés des modèles classiques de la Méthode d'Analyse Nodale Modifiée utilisée également par SPICE. Ce sont des modèles largement décrits dans la littérature [LIT97], [LIE98], [NEW78], [NAG75], [VLA81], que nous avons adapté à nos besoins. Ces modèles sont des modèles idéaux et génériques.
Pour la modélisation de matériels plus complexes tels que les transformateurs, moteurs ou appareils de protection, nous avons des sources beaucoup plus diverses. Tout d'abord, nous avons utilisé la connaissance des lois régissant leur comportement (électrique notamment) [MER98], [EDM97], [FOU84], [QUE88], [DEP91] ainsi que des normes et des documents techniques constructeur permettant de définir un paramétrage intelligible à l'utilisateur et correspondant aux règles métier [BOU92], [BLA82], [SCH01], [LEG00], [SOC00], [NFC98], [CEI88], [CIT94], [BOY81].

2 Approche Orientée Objet de la modélisation.

Le simulateur développé en langage Delphi exploite des modèles eux aussi écrits en Delphi. Nous envisageons d'ores et déjà une évolution vers une modélisation externe au simulateur. Chaque modèle sera compilé séparément du simulateur permettant ainsi à l'utilisateur de développer ses propres modèles.
Le langage de modélisation retenu est VHDL-AMS. Les critères nous ayant amené à choisir ce langage est détaillé dans le paragraphe **4** de ce chapitre. Les principaux intérêts de l'utilisation de ce langage sont d'apporter une plus grande lisibilité aux modèles en s'appuyant sur une norme internationale qui permet d'écrire des modèles couvrant différents domaines pouvant faire intervenir différents niveaux de description. Les modèles existants en Delphi doivent être traduits en VHDL-AMS pour devenir accessible. Les principes de la traduction Delphi vers VHDL-AMS sont développés dans le paragraphe **5** de ce chapitre. Notons que cette traduction devra être suivie d'une étape de développement d'un compilateur VHDL-AMS vers Delphi permettant de réaliser le processus inverse et ainsi de rendre compréhensible au simulateur les modèles définis par l'utilisateur. Mais cette partie sort du cadre de ce travail et sera réalisée ultérieurement. Ici, nous détaillerons tout d'abord la démarche de modélisation des composants et leur implémentation en langage Delphi.

2.1 Description d'un modèle « orienté objet ».

L'emploi de Delphi pour la description des modèles permet d'obtenir une description structurée et réutilisable d'un modèle. Il est important de souligner que Delphi est un langage de programmation. Il n'est ni un langage de modélisation ni un langage de description du matériel (HDL : Hardware Description Language). Chaque modèle de composant est décrit par une classe d'objet. Ainsi pour chaque élément du schéma électrique à simuler, le modèle correspondant sera instancié en un objet de la classe associée au modèle. Pour bien saisir les détails de l'implémentation des modèles en Delphi, il est préférable de comprendre quelques termes élémentaires de la programmation orientée objet. Classe, objet , héritage, surcharge, classe abstraite, concrète, polymorphisme, etc. On se reportera au Chapitre 1 où ces concepts sont développés.

2.1.1 Intérêt de la programmation orientée objet en modélisation.

L'application de schématique et le simulateur étant développés en Delphi, l'emploi de la programmation orientée objet pour la description était une contrainte technologique imposée par Algo'Tech Informatique. Elle se justifie notamment par le fait de maintenir une cohérence et une compatibilité maximum entre le simulateur et les modèles. De plus, ce choix permet de tirer parti des avantages d'une programmation orientée objet notamment l'héritage, la surcharge de méthodes.
L'ensemble de ces caractéristiques permet de définir un comportement commun à tout modèle à partir de la classe parente *TElement*. Les modèles obtenus par héritage de TElement sont réutilisables et modifiables par la surcharge de méthodes par exemple.

2.1.2 Eléments, composants et circuits.

Les principes de la modélisation Delphi étant posés, nous allons décrire les grandes familles de modèles. Elles sont définies par 3 classes abstraites fondamentales : *TElement*, *TComposant* et *TCircuit*. La brève description qui suit permet d'éclaircir ces notions d'éléments, de composants et de circuits et d'introduire le vocabulaire.

2.1.2.1 TElement : élément d'un circuit électrique.

La classe *TElement* est la classe abstraite dont dérive tout modèle. Elle représente tout élément d'un circuit électrique qui peut être un composant ou un sous circuit. Ainsi chaque élément du circuit à simuler sera instancié en un objet dont la classe dérive de *TElement*. Il n'y a que deux classes (abstraites également) qui dérivent directement de *TElement*:

- *TComposant*: modèle d'un composant de base de l'électricité (primitive).
- *TCircuit*: modèle d'un sous-circuit (macro-composant).

Pour les familiers des simulateurs de type Spice [VLA81], on peut faire le parallèle entre ces deux classes et respectivement :

- Les primitives Spice.
- Les sous-circuits définit par l'instruction SUBCKT.

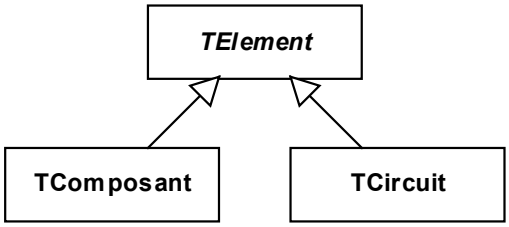


Fig. 2 Héritages des classes fondamentales des modèles.

2.1.2.2 *TComposant* : composant de base du circuit.

La classe *TComposant* décrit les éléments les plus basiques du simulateur. C'est dans cette classe que l'on trouve les méthodes équivalentes aux primitives au sens de SPICE. Elle décrit les relations entre les tensions et courants au sein du composant. Lorsqu'un schéma électrique est analysé, le simulateur crée un réseau d'éléments de type composant (*TComposant*) ou de type circuit (*TCircuit*). Pour pouvoir simuler ce circuit, image du schéma, il est alors nécessaire de décomposer tous les circuits qu'il contient jusqu'au niveau élémentaire, afin d'obtenir un réseau uniquement constitué d'éléments de type composant (*TComposant*).

2.1.2.3 *TCircuit* : sous-circuit, assemblage d'éléments.

La classe *TCircuit* décrit des sous-circuits. Un sous-circuit est un assemblage de composants (*TComposant*) ou de sous-circuits (*TCircuit*). Le schéma que l'on simule est un objet de la classe *TCircuit*, construit à l'analyse du schéma.

2.2 Composition d'un modèle.

Chaque modèle est décrit au sein d'une classe concrète dérivant de la classe *TElement* ou d'une classe abstraite décrivant un méta-modèle, elle même issue de la classe *TElement*. Chaque modèle de comporte 2 aspects distincts :

- **Description** du comportement électrique et séquentiel : c'est le modèle en tant que tel.
- **Intégration** du modèle dans le simulateur.

Rappelons que les modèles Delphi sont des modèles de simulation au sens de [GAR01]. Outre la description du comportement, il doit prendre en considération un certain nombre de contraintes liées notamment à l'environnement de simulation. Nous exploitons ici pleinement le phénomène de l'héritage en décrivant au maximum le comportement commun des modèles

au niveau des classes abstraites les plus près possibles de la classe *TElement* dans l'arbre d'héritage. Ainsi au niveau des classes concrètes seules les parties spécifiques à la description du comportement du modèle sont traitées.

2.2.1 Description du comportement.

La structure du modèle définit son aspect extérieur, son interface avec l'environnement, ses entrées, ses sorties et ses paramètres. Le comportement du modèle définit l'état des sorties et les réactions en fonction de ses entrées et des événements qui le touchent et de l'influence de son environnement(**Fig. 3**). On retrouve d'ailleurs une distinction analogue entre structure et comportement aussi bien dans des langages de programmation orientés objet comme Delphi que dans des langages de modélisation tels que VHDL-AMS (voir **Tab. 1**).

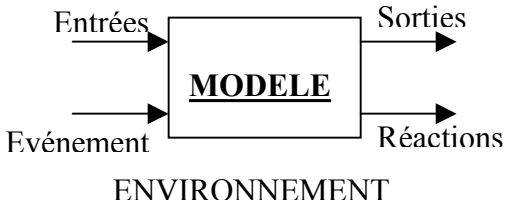


Fig. 3 Structure d'un modèle

Partie du modèle	Delphi	Vhdl-Ams
Structure	Interface	Entity
Comportement	Implémentation	Architecture

Tab. 1 Structure et comportement en Delphi et VHDL-AMS

Ainsi pour chaque modèle, nous devons définir :

- Ses paramètres.
- Son comportement électrique.
  - Entrées.
  - Sorties
  - Les relations Entrées Sorties.
- Son comportement séquentiel
  - Les événements qui le touchent
  - La réaction à ces événements.

Nous rappelons que Simul'Elec est avant tout un simulateur de réseau électrique « analogique ». Chaque modèle est constitué par la description du comportement électrique, nécessaire à tous les modes d'analyse. L'aspect séquentiel du comportement peut y être ajouté comme une couche supplémentaire si l'on souhaite augmenter la précision du modèle. Ce dernier aspect n'est par contre exploité que par le mode d'analyse événementiel.
Ces deux aspects du modèle sont indépendants et sont définis séparément. Ainsi, le comportement séquentiel ne peut pas modifier le comportement électrique et inversement. Ces deux parties du modèle sont toutefois liées par des variables partagées (comme l'état logique du composant) auquel chacun peut accéder, modifier et adapter son fonctionnement en fonction de leurs valeurs. C'est, par exemple, le cas d'un disjoncteur magnétothermique dont le modèle électrique est une résistance en série avec un interrupteur. Cet interrupteur est ouvert ou fermé selon l'état logique du disjoncteur qui peut être défini par l'utilisateur. Mais

cette valeur d'état logique peut être modifiée par la partie séquentielle du modèle qui décrit les conditions de déclenchement du disjoncteur.

Pour résumer, la définition du comportement intervient sur deux plans distincts, électrique et séquentiel. Cette définition est, et doit rester dissociée de l'intégration du modèle dans le simulateur. Toutefois, cette description reste liée à la méthode de modélisation et de simulation. Par exemple, le comportement électrique d'une résistance est défini dans le modèle *TResistance* sous forme d'un système de matrices. Mais la définition du modèle via la classe *TResistance* ne permet pas de savoir de quelle manière ce composant s'intègre dans un circuit. C'est la classe abstraite parent *TDipole* qui décrit comment connecter ce dipôle aux autres éléments du réseau simulé, comme le montre la **Fig. 4**.

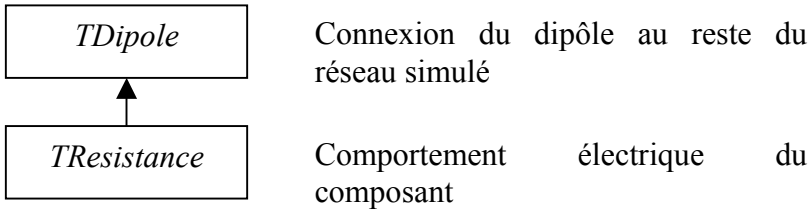


Fig. 4 Répartition de la définition d'un modèle de résistance dans différentes classes

2.2.2 Intégration du modèle dans le simulateur.

Les classes qui décrivent les modèles sont constituées de 2 parties qui ont chacune un rôle bien distinct :

- **Description** du comportement électrique et séquentiel : c'est le modèle en tant que tel.
- **Intégration** du modèle dans le simulateur.

La partie « utile » d'un modèle est constitué par les deux aspects électrique et séquentiel qui décrivent le comportement de l'élément modélisé. Toutefois, nos modèles doivent s'intégrer dans un circuit traité par notre outil de simulation. Il est donc nécessaire de définir les règles qui permettent de mettre en relation la description du comportement avec le simulateur.
Ainsi, la description d'un modèle sous la forme d'une classe dérivant de *TElement* prendra en compte non seulement le comportement mais aussi l'intégration dans le circuit simulé.

Pour mieux comprendre quelles sont ces règles d'intégration, il faut rappeler brièvement le processus de simulation et le type d'information que traite le simulateur. Ce processus est déterminé par les méthodes de modélisation (méthode d'analyse nodale modifiée) et de résolution du système (pivot de Gauss). L'aspect électrique du modèle est ici prépondérant, l'aspect événementiel n'influant que très peu.

2.2.2.1 Décomposition du schéma en primitives.

La méthode de décomposition du schéma est détaillée dans le chapitre 2. Nous en rappelons ici brièvement les principes afin de mieux comprendre comment chaque modèle est intégré dans le simulateur. Lors de la résolution, le noyau de calcul doit disposer d'un réseau de composants interconnectés. Chacun des composants de ce réseau est une primitive du simulateur dont le modèle est défini par un système de matrices élémentaires. Pour résumer, le réseau ainsi défini permet de rassembler toutes ces matrices élémentaires pour constituer un système de matrices globales qui est l'image du réseau électrique. C'est ce système qui fait l'objet de la résolution. D'autre part, l'analyse du schéma représentant ce réseau, nous fournit

une liste d'éléments électriques et de leurs interconnexions. Par contre, ces éléments peuvent aussi bien être des composants (primitives du simulateur) que des sous-circuits, constitués eux-mêmes par un réseau d'éléments électriques. Le simulateur ne peut pas les traiter en tant que tels, il doit disposer uniquement de primitives. Donc, chaque sous-circuit doit pouvoir se décomposer en composants élémentaires et se connecter au reste du réseau.

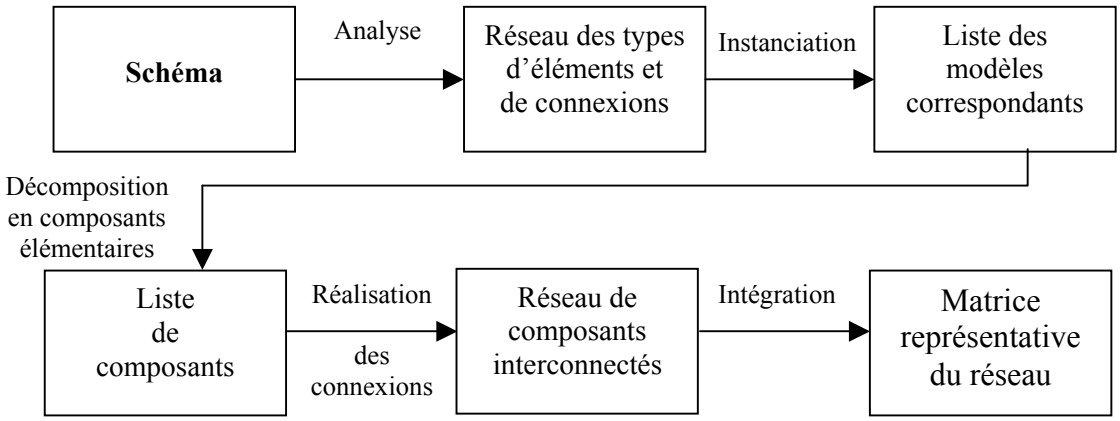


Fig. 5 Cycle de traduction d'un schéma pour la simulation

2.2.2.2 Références croisées.

Pour pouvoir utiliser nos modèles avec l'analyse événementielle, il est nécessaire de compléter le réseau électrique par la création de références croisées. Ces références croisées permettent d'établir une relation maître-esclaves entre plusieurs éléments du schéma via leur état logique. C'est par exemple le cas d'un relais dont l'état des contacts (actif/au repos) sera asservi à l'état logique de la bobine de commande. Ainsi un élément maître impose son état à des composants esclaves. Le terme "références croisées" indique que chaque maître possède une référence sur ses esclaves et que chaque esclave possède une référence à son maître.
Le changement d'état d'un composant maître doit se propager lors de l'analyse événementielle aux composants qui sont ses esclaves et pour transmettre cet état le simulateur doit reconstituer les liens entre composants.

Cette notion est très forte, voir essentielle, pour la simulation des circuits électrotechniques. Fréquemment du matériel modulaire dispose d'une partie centrale qui est généralement le maître et de différents accessoires. Dans certains cas, la relation classique Maître → Esclave est étendue à 3 niveaux hiérarchiques Maître → Maître/Esclave → Esclave. Il existe un niveau intermédiaire « Maître/Esclave » qui définit du matériel qui peut être à la fois maître et esclave. C'est par exemple le cas de certains disjoncteurs qui peuvent être maîtres car leur état de déclenchement définit celui de contacts additifs (pour la signalisation de défaut par exemple). Le disjoncteur peut être lui-même piloté à distance par un moteur qui est alors le maître du disjoncteur.

2.2.2.3 Fonctions d'intégration des modèles.

Les fonctions qui permettent l'intégration des modèles au simulateur sont définies dans les classes abstraites dont dérivent les modèles. Ces classes abstraites sont situées au niveau le plus haut dans l'arbre d'héritage ayant pour racine la classe *TElement*. Le tableau **Tab. 2** suivant résume les différentes fonctions d'intégration des modèles. Il indique à titre d'illustration quelques classes abstraites concernées.

Fonction	Classes concernées
Décomposition du réseau en composants élémentaires	*TCircuit ….*
Connexion des éléments	*TCircuit, TDipole, Tquadripole…*
Mise à jour de la liste des fréquences du simulateur	*TCircuit, TDipoleActif …*
Création des références croisées	*TElement …*

Tab. 2 Fonctions d'intégration des modèles dans le simulateur.

Bien qu'implémentées dans les classes représentant les modèles, on peut considérer que leur définition fait davantage partie de l'outil de simulation que de l'outil de modélisation. En effet, l'aspect purement modélisation dépend principalement de la méthode de modélisation alors que les fonctions d'intégration du modèle sont avant tout liées à l'architecture logicielle du simulateur.

3 Modèles développés.

Nous avons vu en **1.3** que le développement des modèles va de pair avec celui du simulateur. Le niveau de complexité que peut atteindre un modèle est dépendant de celui que peut traiter le simulateur. Dans notre démarche de conception du simulateur, nous avons défini des étapes d'amélioration progressives des outils et des modèles. Nous avons donc développé des modèles de plus en plus complexes au fur et à mesure de l'amélioration du simulateur. Tout d'abord, nous avons décrit le comportement des composants élémentaires qui constituent les primitives du système. Ces primitives nous permettent de représenter notre circuit sous forme matricielle et de le résoudre à l'aide de méthodes d'analyse numérique. Elles sont semblables aux primitives de SPICE [NAG75]. Ensuite nous présenterons des macro-modèles simples, puis des modèles plus complexes intégrant des comportements séquentiels.

3.1 Les Primitives du simulateur.

La méthode d'analyse nodale modifiée, sur laquelle est basé notre noyau de résolution, repose sur l'expression des relations entre les tensions et les courants pour chacun des nœuds du circuit. Le comportement électrique de chaque composant doit définir ces relations en chacun de ses nœuds. Afin de ne pas décrire à nouveau toutes ces relations, on définit un ensemble de primitives qui regroupe les différentes relations possibles. Elles sont associées à des composants élémentaires. Tout composant d'un circuit peut être décrit par la combinaison de ces primitives en réalisant des connections entre composants élémentaires. De ce fait, avant de pouvoir réaliser des modèles complexes (macro-modèles ou sous-circuits), il est indispensable d'implémenter toutes les primitives qui peuvent s'avérer nécessaires. Par primitive, on désignera aussi bien la fonction réalisée que le composant électrique qui l'implémente.

3.1.1 Primitives du prototype SiCi.

Les premières primitives dans la phase de développement du prototype SiCi (**Fig. 1– Etape 2**) sont les composants les plus élémentaires de l'électricité.

Composants	Relation électrique
Source de tension sinusoïdale	$u(t)=U\sin(\omega t+\varphi)$
Source de courant sinusoïdale	$i(t)=I\sin(\omega t+\varphi)$
Résistance idéale	$u=R.i$
Inductance idéale	$u=jL\omega.i$
Condensateur idéal	$u=\frac{-j}{C\omega}.i$
Court-circuit	Identité de 2 noeuds
Interrupteur	Selon état circuit ouvert ou court-circuit

Tab. 3 Primitives du prototype SiCi

Litovski et M. Zwolinski [LIT97] décrivent la méthode de description de ces primitives existant pour la plupart dans SPICE. Il a toutefois été nécessaire d'adapter ces descriptions au régime sinusoïdal de SiCi ainsi qu'à la structure logicielle du simulateur. On notera qu'à ce stade de son développement, le simulateur SiCi ne pouvait traiter que des dipôles.
A partir de ces primitives, il est possible de simuler des circuits simples en régime permanent et fréquentiel. Cela peut s'appliquer à l'étude de circuits simples de chauffage, d'éclairage ou contenant des filtres passifs. La première lacune qui apparaît est le manque de composant

« amplificateurs de signal ». D'autre part, on ne dispose que de sources sinusoïdales ce qui limite le champ d'étude. Ces modèles valables en régime sinusoïdal uniquement ont été adaptés et améliorés notamment pour les régimes continus.

3.1.2 Définition matricielle des modèles de primitives.

La méthode d'analyse nodale modifiée permet de traduire un système d'équations sous forme matricielle [LIT97] en exprimant les relations entre les tensions et les courants en chacun des nœuds**.** La description plus détaillée de cette méthode est donnée dans le Chapitre 1.

$$[G]x[V]=[I]$$

Fig. 6 Système de matrices obtenu par l'analyse nodale modifiée

Chaque ligne du système de matrices ou colonne de la matrice des admittances (G) correspond à un des nœuds du circuit. Pour chaque primitive, le modèle définit la participation du composant en décrivant quelles valeurs complexes notre modèle insère dans les matrices pour les cellules correspondantes à ses nœuds. Dans notre système de matrices, la matrice des tensions V n'est pas renseignée par nos modèles car c'est la matrice des inconnues. Ainsi chaque modèle doit décrire son comportement au travers de sa participation en terme d'impédance ou de courant d'excitation. Toute participation sous forme de tension sera transformée pour être inscrite dans les matrices G et I. Pour certains composants, on introduit donc de nouvelles équations (méthode du giratoire [LIT97]). On ajoute de ce fait une dimension au système de matrices en créant une nouvelle branche. On adopte ici une représentation sous forme de tampon (stamp) résumant le modèle sous forme tabulaire, illustré par l'exemple en **Fig. 7** et **Fig. 8**.

A, B et C sont les coordonnées de la matrice correspondantes aux nœuds du composant, br_1 à br_n sont les nouvelles branches créées pour ajouter des équations. VSM regroupe les valeurs que l'on ajoute au Vecteur Second Membre des excitations. Notons que les primitives ont un comportement purement électrique. Elles n'ont pas de comportement événementiel particulier.

Une case laissée vide n'apporte aucune contribution. Nous rappelons que pour tous les modes d'analyses les valeurs contenues dans le système de matrices sont des valeurs complexes.

		A B C	$br_1..br_n$	VSM
	A B C			
	br_1 br_n			

Fig. 7 Représentation tabulaire d'un modèle

Fig. 8 Composant modélisé

3.1.3 Dipôles passifs.

On va considérer qu'un dipôle est passif s'il n'introduit aucune excitation dans le système. Parmi les modèles que nous avons développés, les dipôles passifs regroupent :

- Résistance.
- Condensateur.
- Inductance.
- Admittance.

Pour tous ces dipôles passifs, l'insertion dans les matrices sera identique, seule la manière de calculer l'admittance Y des composants diffère.

	A	B	VSM
A	Y	-Y	
B	-Y	Y	

Fig. 9 Tampon d'un dipôle passif

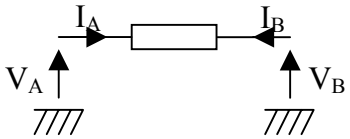


Fig. 10 Dipôle passif

Ici la valeur Y représente l'admittance complexe du composant qui sera calculée différemment pour chacun. Nous traitons des régimes sinusoïdaux ce qui a pour effet de simplifier la détermination de ces impédances. Nous utilisons la forme sinusoïdale réduite.

	Forme sinusoïdale réduite	Forme différentielle	Paramètre
Résistance	$\underline{Y}=\frac{1}{R}$	$Y=\frac{1}{R}$	Résistance R
Condensateur	$\underline{Y}=jC\omega$	$Y=C\frac{du(t)}{dt}$	Capacité C
Inductance	$\underline{Y}=\frac{-j}{L\omega}$	$Y=L\int i(t)dt$	Inductance L
Admittance complexe	$\underline{Y}=\frac{1}{\mathrm{Re}+j\mathrm{Im}}$	Ne s'applique pas	Partie réelle Re, partie imaginaire Im

Tab. 4 Différentes expressions de l'impédance pour les composants passifs

Comme pour SPICE, l'insertion de valeurs nulles dans les matrices risque d'empêcher la résolution. On peut à force de placer des zéros (complexes) dans les matrices disposer de moins d'équations que l'on a d'inconnues, on peut également avoir des divisions par zéros insolubles. Pour éviter ce type de problème, lorsque le composant reçoit un paramètre nul (résistance, capacité, inductance selon le cas). On lui substitue une valeur Z_{low} (valeur = 10^{-12}) permettant d'insérer des valeurs différentes de zéro.

De même lorsque le composant reçoit une excitation continue, on substitue la pulsation nulle par une pulsation ω_{low} (valeur = 10^{-12}), pour la détermination de l'admittance.

3.1.4 Sources

Simul'Elec traite des régimes sinusoïdaux établis. Il considère le régime continu comme un cas particulier de régime sinusoïdal à fréquence nulle. Pour la résolution du système comportant plusieurs fréquences d'excitation, il applique le principe de superposition en réalisant une simulation pour chaque fréquence en éteignant à chaque fois les sources dont la fréquence est différente (voir chapitre 2, paragraphe. 2-2-3). Ces sources sont idéales ne subissent pas de limitation et n'ont pas d'impédance interne.

3.1.4.1 Source de courant.

Un composant unique permet de modéliser une source de courant sinusoïdale ou continue. Si la source est continue, la fréquence est considérée comme nulle par le simulateur et sera traitée comme telle. La source de courant est caractérisée par :

- L'amplitude du signal Is.
- Sa fréquence f_s.
- Son déphasage φ_s.(représente le décalage du signal par rapport au temps t=0 de la simulation)

Lorsque le simulateur traite la fréquence de la source, il va introduire les données décrites en **Fig. 11** dans le système de matrices.

	A	B	VSM
A			$-(I_s, \varphi_s)$
B			(I_s, φ_s)

Fig. 11 Tampon d'une source de courant

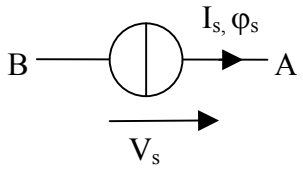


Fig. 12 Source de courant

3.1.4.2 Source de tension.

Sur le même principe que la source de courant, cette source de tension peut être sinusoïdale ou continue. Elle est caractérisée par :

- L'amplitude du signal V_s.
- Sa fréquence f_s.
- Son déphasage φ_s.

Avec la méthode d'analyse nodale modifiée, toutes les excitations doivent être exprimées comme des courants. On utilise le principe du giratoire [LIT97] pour transformer la source de tension en source de courant. Cette méthode à pour effet d'ajouter une branche supplémentaire dans le circuit (**Fig. 13**)

	A	B	Br	VSM
A			-1	
B			1	
Br	-1	1		(V_s, φ_s)

Fig. 13 Tampon d'une source de tension

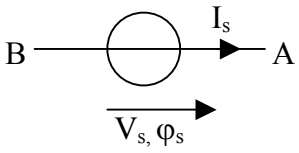


Fig. 14 Source de Tension

3.1.4.3 Source de tension multifréquences.

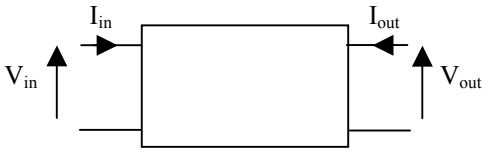
Cette source de tension multifréquences est une superposition de N sources de tension sinusoïdales dont chacune est caractérisée par :

- L'amplitude du signal.
- Sa fréquence.
- Son déphasage.

Pour réaliser ce type de source, on pourrait réaliser un circuit avec N sources de tension en série. On obtiendrait le même résultat qu'avec cette source multifréquences mais ce principe est peu efficace car pour chaque fréquence, le simulateur doit court-circuiter les sources de fréquences différentes. Cela a pour effet de modifier le réseau et nécessite une reconstruction du système de matrice ce qui est relativement coûteux en temps. En utilisant notre source multifréquence, nous avons en réalité une seule source dont les caractéristiques (V_s, φ_s) sont différentes pour chaque fréquence. Ainsi lorsque le simulateur passe d'une fréquence à une autre, la structure du circuit n'est pas modifiée, c'est uniquement les valeurs (V_s, φ_s) qui sont mises à jour, permettant une efficacité optimale.

3.1.5 Quadripôles.

Les premières primitives modélisées sont des dipôles. Elles permettent de décrire des relations électriques entre les tensions et courants des 2 nœuds, les bornes du composant. Cette bibliothèque de modèle a besoin d'être complétée notamment par des quadripôles qui vont décrire des relations plus complexes. En **Fig. 15**, le schéma d'un quadripôle est représenté.



$$I_{OUT} = f(I_{IN}, V_{IN})$$
et
$$V_{OUT} = f(I_{IN}, V_{IN})$$

Fig. 15 Quadripôle élémentaire de Simul'Elec.

Ces quadripôles sont des quadripôles idéaux qui ne prennent pas en compte les impédances d'entrée ou de sortie. Nous avons modélisé cinq quadripôles qui nous permettent d'exprimer différents types de relation entre les tensions et courants, d'entrée et de sortie.

- Source de tension contrôlée en tension : STCT.
- Source de courant contrôlée en tension : SCCT.
- Source de tension contrôlée en courant : STCC.
- Source de courant contrôlée en courant : SCCC.
- Transformateur idéal.

Nous avons défini différents composants qui décrivent les relations entre les signaux d'entrée (V_{IN} et I_{IN}) et les signaux de sortie (V_{OUT} et I_{OUT}). Pour cela nous avons modélisé plusieurs sources contrôlées ainsi qu'un transformateur idéal. Ils sont caractérisés par un rapport de proportionnalité G s'appliquant de diverses manières entre les signaux d'entrée et de sortie (**Tab. 5**).

Composant	Relations		
Transformateur idéal	$V_{OUT}=G.V_{IN}$	et	$I_{OUT}=I_{IN}/G$
Source de tension contrôlée en tension : STCT	$V_{OUT}=G.V_{IN}$	et	$I_{IN}=0$
Source de courant contrôlée en tension : SCCT	$I_{OUT}=G.V_{IN}$	et	$I_{IN}=0$
Source de tension contrôlée en courant : STCC	$V_{OUT}=G.I_{IN}$	et	$V_{IN}=0$
Source de courant contrôlée en courant : SCCC	$I_{OUT}=G.I_{IN}$	et	$V_{IN}=0$

Tab. 5 Fonction des quadripôles élémentaires de Simul'Elec

On considère ici que l'on a deux bornes d'entrées et deux bornes de sorties. Les impédances d'entrée et de sortie ne sont jamais définies. Nous disposons ici de composants idéaux Dont le rapport entre les signaux d'entrée et de sortie (tension ou courant) G est constant et complexe.

3.1.5.1 Source de tension contrôlée en tension.

L'entrée de la STCT est un circuit ouvert et sa sortie une source de tension proportionnelle à la tension présente entre les bornes d'entrée. La nouvelle branche est introduite par l'ajout de la source de tension (méthode du giratoire).

	A	B	C	D	Br	VSM
A						
B						
C					-1	
D					1	
BrS	G	-G	-1	1		

Fig. 16 Tampon d'une STCT

Fig. 17 STCT

3.1.5.2 Source de courant contrôlée en tension.

L'entrée de la SCCT est un circuit ouvert et sa sortie une source de courant proportionnelle à la tension présente entre les bornes d'entrée. Dans ce cas, comme on introduit une source de courant, on n'ajoute pas de branche supplémentaire

	A	B	C	D	VSM
A					
B					
C	G	-G			
D	-G	G			

Fig. 18 Tampon d'une SCCT

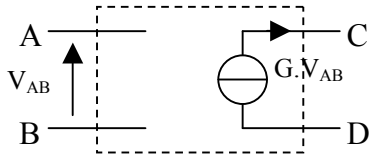


Fig. 19 SCCT

3.1.5.3 Source de tension contrôlée en courant.

L'entrée de la source est un court-circuit et sa sortie une source de tension proportionnelle au courant traversant la branche d'entrée. On ajoute une branche d'entrée qui permet d'exprimer l'égalité des potentiels V_A et V_B sont toutefois confondre les nœuds pour pouvoir mesure le courant qui traverse la branche. Comme pour la STCT, la branche de sortie correspond à la source de tension

	A	B	C	D	BrE	BrS	VSM
A					-1		
B					1		
C						-1	
D						1	
BrE	-1	1					
BrS			-1	1	G		

Fig. 20 Tampon d'une STCC

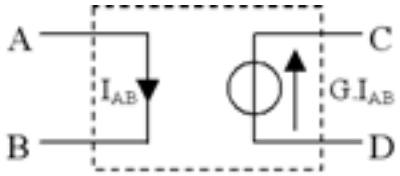


Fig. 21 STCC

3.1.5.4 Source de courant contrôlée en courant.

L'entrée de la SCCC est un court-circuit et sa sortie une source de courant proportionnelle au courant traversant la branche d'entrée. Contrairement au modèle de la STCC seul la branche d'entrée est ajoutée car la source est une source de courant.

	A	B	C	D	BrE	VSM
A					-1	
B					1	
C					-G	
D					G	
BrE	-1	1				

Fig. 22 Tampon d'une source de tension

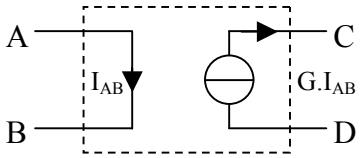


Fig. 23 Source de Tension

3.1.5.5 Transformateur idéal.

Le transformateur peut être modélisé de plusieurs façons : paire d'inductances mutuelles couplées, transformateur avec un seul courant de sortie ou avec deux courants de sortie [LIT97]. Nous avons retenu cette dernière approche où le courant de chaque enroulement est une nouvelle variable de notre système d'équations. Pour cela on introduit deux nouvelles branches correspondant aux courants du primaire et du secondaire du transformateur. Ainsi nous pouvons exprimer les rapports de proportionnalité qui existent entre les entrées et sorties :

$$\mathbf{V_{AB} = G.V_{CD}} \qquad \text{et} \qquad \mathbf{I_{CD} = G.I_{AB}}$$

Ce modèle simplifié traduit uniquement les relations de proportionnalité entre les tensions et courants. Il ne prend pas en considération les inductances de chaque enroulement ni les pertes fer ou par effet Joule.

	A	B	C	D	BrE	BrS	VSM
A					-1		
B					1	-1	
C						1	
D							
BrE	-1	1	G	-G			
BrS			-		G	1	

Fig. 24 Tampon d'un transformateur idéal

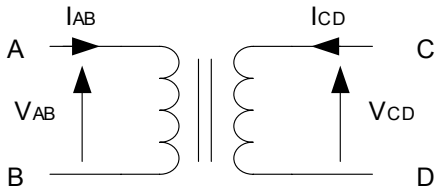


Fig. 25 Transformateur idéal

3.2 Macro modèles statiques

Disposant des briques de bases élémentaires grâce aux primitives définies auparavant, nous avons pu entamer la modélisation des composants plus complexes de l'électrotechnique (Fig. 1 - Etape 5).
A partir des besoins que nos utilisateurs potentiels (techniciens et enseignants) ont exprimés et de l'analyse de schémas standards, nous avons défini un ensemble d'éléments modélisables :

- Inductance réelle.
- Transformateur réel, monophasé ou triphasé.
- Ballast (de tube d'éclairage).
- Tube néon.
- Potentiomètre.
- Fil et câble électrique résistants.
- Contacts multipolaires (pour interrupteurs, sectionneurs, contacteurs…).
- Moteur à courant continu.
- Moteur et génératrice :
    - Synchrone et asynchrone.
    - Monophasé et triphasé.

Tous les modèles de ces composants sont des sous-circuits (ou macro-modèles). Ils sont réalisés à partir d'un assemblage de primitives ou d'autres macro-modèles. Contrairement aux primitives, on ne définit pas directement les relations entre les tensions et les courants mais on combine des relations élémentaires via l'assemblage des primitives (modèle structurel).
Notons que les modèles définis ici sont des modèles statiques valables en régime permanent. Les modèles de moteurs par exemple n'intègrent pas les phases transitoires de démarrage ou de variation de charge. Ces modèles bien qu'incomplets se révèlent suffisants pour un certain nombre d'applications qui nécessitent des résultats rapides, tel que le pré-dimensionnement, l'étude des consommations ou l'équilibrage de l'énergie réactive par compensation du cos φ.

On retrouvera en **Annexe 1** les détails de macro-modèles simples :

- Potentiomètre.
- Résistance triphasée.
- Transformateurs réels monophasés et triphasés.
- Moteurs et génératrices à courants continus, alternatifs monophasés et triphasés.

3.3 Macro modèles à comportement séquentiel.

Le développement des modèles statiques décrits précédemment a été suivi par la réalisation du mode d'analyse événementielle. Ce mode a permis d'une part de simuler des composants ayant un comportement séquentiel, tels que les contacteurs (bobines et contacts, instantanés ou temporisés) ou des appareils de protection (fusible, disjoncteur magnétothermiques). Il a permis d'autre part d'affiner les modèles existants en leur ajoutant ces aspects événementiels.

Les nouveaux composants modélisés vont permettre de décrire l'évolution de l'état du circuit. Les évolutions séquentielles des circuits électrotechniques sont dues soit à l'intervention de l'environnement sur le circuit (opération de l'utilisateur, changement d'un paramètre d'un composant) ou bien à la réaction naturelle d'un de ses éléments. Ces réactions naturelles vont

permettre de simuler la séquence des états du circuit qui correspondent à la fonction que doit remplir l'installation.
Nous distinguons deux grandes catégories de fonctions :

- Contrôle - commande.
- Protection des circuits et des personnes.

Nous avons donc centré nos efforts sur le développement de modèles réalistes des principaux composants réalisant ces fonctions soit respectivement :

- Bobines et contacts de relais.
- Fusibles et disjoncteurs magnétothermiques.

Grâce à l'analyse événementielle, nous avons pu affiner certains modèles existants en leur ajoutant un comportement pseudo transitoire lors de changements d'états du circuit. C'est par exemple le cas du transformateur réel qui peut simuler un appel de courant élevé lors de sa mise sous tension.

3.4 Relais.

Le relais est un des composants clé de la fonction de contrôle-commande de l'électrotechnique. Relais est un terme générique pour une famille de composants que l'on retrouve parfois sous les dénominations de contacteur, relais, micro relais, interrupteur piloté, etc. On rencontre également des relais en électronique qui peuvent être simulés par des logiciels tels que SPICE. Hormis le facteur d'échelles et la différence des puissances mises en jeu, les relais que l'on trouve en électronique ou électrotechnique ont un fonctionnement assez similaire mais une constitution différente. Les relais de l'électronique que l'on désigne plutôt par micro-relais sont généralement destinés à être implantés sur une carte. Au contraire, le relais de l'électrotechnique, d'un coût plus élevé, met en œuvre une modularité importante qui permet d'équiper un relais de nombreux accessoires tels que divers types de contacts additifs.
Au niveau de la représentation sur le schéma, les différentes composantes du relais seront réparties en différents endroits du circuit en fonction de leur rôle (bobine, contacts principaux, contacts auxiliaires…) sans former un symbole unique. Cette modularité importante nous a amené à adopter une approche de la modélisation de ces relais assez différente de celle rencontrée en électronique.

La difficulté de modélisation réside dans le fait que c'est un dispositif électromécanique dont on doit traduire les comportements électrique, magnétique et mécanique sous une forme exploitable par un simulateur de circuit électrique.

3.4.1 Constitution du relais.

En électrotechnique, un relais regroupe plusieurs composants distincts qui vont être représentés par différents symboles:

- Un électroaimant (bobine).
- Un groupe de contacts (de puissance, de commande, instantanés ou temporisés).

Ces composants sont liés mécaniquement et peuvent se trouver dans un même boîtier ou non si l'on a affaire à des blocs additionnels. Lorsque la bobine est suffisamment excitée, le champ magnétique qu'elle produit est suffisamment intense pour déplacer les contacts et ainsi

fermer (ou ouvrir) un circuit électrique. Si le champ n'est pas assez intense, le ressort de rappel maintient les contacts en position repos. Le relais est conçu pour fonctionner en mode Tout Ou Rien. Il peut avoir 2 positions, actif ou repos, auxquelles ont peut aisément associer un état logique.

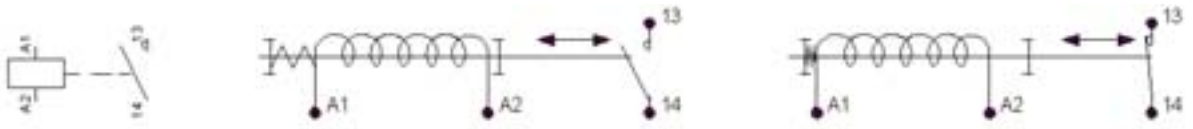


Fig. 26 Symbole d'un relais et structures simplifiées d'un relais ouvert puis fermé .

Pour décrire ce type de composant, on peut adopter différentes approches dont :

- Une approche « tout électrique », comme celle de Spice, où le relais est considéré comme un système électrique unique défini par une fonction de transfert et par des équations non linéaires.
- Notre approche qui modélise séparément chaque composante du relais tout en propageant l'état logique de la bobine vers les contacts qui représentent le lien mécanique qui existe entre eux.

3.4.2 Modélisation du relais en électronique.

Les modèles que l'on rencontre dans les simulateurs électroniques considèrent en général qu'il s'agit d'un seul système à base de composants électriques dont l'entrée est la bobine et la sortie est le contact. Dans la bibliothèque du simulateur de Mentor Graphics « Accusim» [MEN] on trouve plusieurs modèles qui ont des niveaux de complexité différents dont notamment :

- S : Voltage controlled Switch et W : Current controlled Switch.
- VCAS : Voltage Controlled Analog Switch Function.
- DK1A9V : DK1a-9V Matsushita relay model, conforme à la documentation du constructeur.

3.4.2.1 Interrupteurs pilotés par la tension S ou le courant W.

On considère ici un relais comme un interrupteur piloté par une tension ou par un courant. Ces modèles font partie des primitives de SPICE. On utilise plus fréquemment le modèle piloté par la tension car les relais sont généralement alimentés sous une tension fixe correspondant à sa tension nominale. Le principe est identique dans les 2 cas sauf que la grandeur d'entrée considérée est différente.

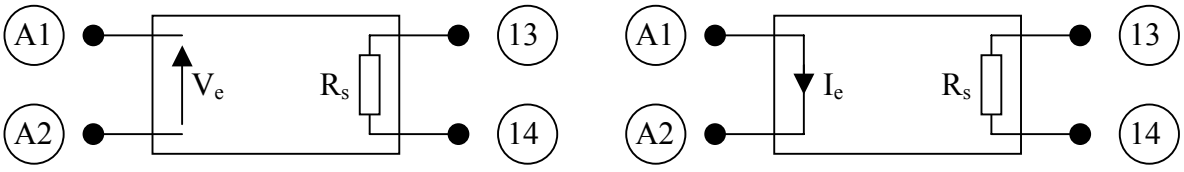


Fig. 27 Modèles d'interrupteur pilotés par la tension et par le courant (bobine + contact).

Le contact est modélisé par une résistance R_s. Sa valeur vaut R_{on} (par défaut 1Ω : représente la résistance du contact) lorsqu'il est fermé et Roff (par défaut 1MΩ : représente la résistance de l'air) lorsqu'il est ouvert. L'ouverture et la fermeture de l'interrupteur sont déterminées par 2 seuils sur la tension d'entrée (tension d'alimentation de la bobine).

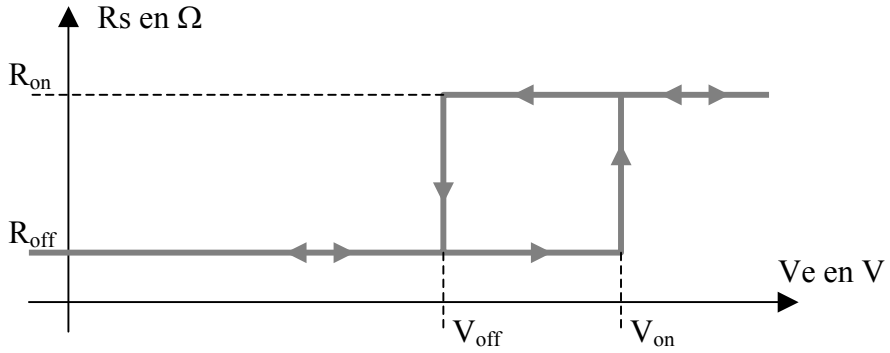


Fig. 28 Cycle d'hystérésis d'ouverture fermeture d'un interrupteur piloté en tension

Ce modèle reste un modèle idéal d'interrupteur piloté par la tension. Il permet d'en modéliser le comportement d'une manière fonctionnelle. L'entrée de ce composant reste idéale et ne prend pas en compte l'inductance de la bobine dans le cas d'un relais. Toutefois, il respecte le principe de l'isolation galvanique car la sortie n'est pas électriquement liée à l'entrée. On obtient une assez bonne traduction des 2 états logiques que peut prendre l'interrupteur. Par contre, il ne permet de modéliser qu'un seul contact à la fois. Si on souhaite modéliser un relais à plusieurs contacts, il est nécessaire de disposer d'autant de composants que de contacts.

3.4.2.2 VCAS : Voltage Controlled Analog Switch Function.

Cet autre modèle présent dans la bibliothèque de « Accusim » (en **Fig. 29**) n'est pas une primitive du simulateur mais un macro modèle complexe. Il utilise une description relativement compliquée pour définir un comportement simple. Le changement d'état sera continu lors de l'analyse transitoire alors que l'on a affaire à un comportement tout ou rien.

```
* Copyright (c) Mentor Graphics Corporation, 1988-2001 All Rights Reserved.
*                  UNPUBLISHED, LICENSED SOFTWARE.
*     CONFIDENTIAL AND PROPRIETARY INFORMATION WHICH IS THE
*   PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS.
*
*** VOLTAGE-CONTROLLED ANALOG SWITCH FUNCTION
******        <+> <-> <SIDE1> <SIDE2>
.SUBCKT VCAS 1    2      3     4 {RON=1} {ROFF=1E6} {VTH=0.5}
II1 1 0 0
II2 2 0 0
UBUF 0 7 1 2 INV VTH=VTH ISOURCE=(1/ROFF) ISINK=(1/RON) RO=1
RSW 3 4 1T
GSW 3 4 POLY(2) 3 4 7 0 0 0 0 0 1
.ENDS VCAS
```

Fig. 29 Modèle du VCAS « Accusim »

3.4.2.3 DK1A9V : micro-relais Matsushita.

Comme pour le modèle précédent, son comportement est notamment défini par une fonction de transfert décrite par un polynôme du 2ème ordre. Il s'agit dans ce cas (voir **Fig. 30**) d'un modèle qui correspond à un matériel précis, devant être conforme à la documentation du constructeur.

```
* Copyright (c) Mentor Graphics Corporation, 1988-2001 All Rights Reserved.
*                  UNPUBLISHED, LICENSED SOFTWARE.
*     CONFIDENTIAL AND PROPRIETARY INFORMATION WHICH IS THE
*   PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS.
*
* J1.0
** DK1a-9V RELAY MACROMODEL
** MATSUSHITA DATA SHEET
**
**           <PIN6> <PIN1> <PIN3> <PIN4>
* USAGE XNAME <CP>   <CN>   <SW>   <SW>  DK1A9V
.SUBCKT DK1A9V 6 1 3 4
L 6 7 140.625M
RC 7 1 405
G 3 4 POLY(2) (3,4) (8,0) 0 0 0 0 1
R 3 4 500G
IOFF 9 0 42.60001M
U1 0 9 7 1 BUF ISOURCE=57.0642M ISINK=1P RO=100MEG
+          VTH=3.6 TDF=2U TDR=2U
C 9 0 100U
D1 9 0 DMOD
D2 0 9 DMOD
U2 0 8 9 0 BUF ISOURCE=370.37037M ISINK=2F RO=1K
+          VTH=0 TDF=2U TDR=2U
.MODEL DMOD D modtype=accusim
.ENDS DK1A9V
```

Fig. 30 Modèle du micro-relais Matsushita DK1A9V.

3.4.2.4 Défauts de l'approche électronique de la modélisation des relais.

Les différents modèles rencontrés permettent d'intégrer dans une simulation le comportement de ces relais. On a affaire à des modèles simplistes (S et W) ou relativement complexes (VCAS et DK1A9V). Sans être un composant atypique, le relais n'est pas un composant clé des circuits électroniques comme c'est le cas en électrotechnique. Cela peut justifier le fait que l'outil de simulation et les modèles ne soient pas vraiment bien adaptés à ce type de composant. Les modèles les plus complexes introduisent des non-linéarités ce qui augmente la complexité et le temps de calcul. Parmi les autres inconvénients de ces modèles, nous avons également relevé le fait qu'un relais est considéré comme un système complet (bobine et contacts) auquel un symbole unique est associé dans la schématique. Cela n'apporte pas la flexibilité quant au nombre de contacts, primordial en électrotechnique du fait de la modularité du matériel. De plus, des caractéristiques essentielles du comportement du relais tels que les retards à l'ouverture et à la fermeture n'apparaissent pas explicitement.

3.4.3 Approche Simul'Elec.

Sur un schéma électrotechnique, les différentes parties du relais, bobine et contacts sont généralement éclatées et considérées comme des symboles distincts. Pour décrire un relais, on le dissocie alors en plusieurs modèles correspondant à chacun des symboles :

- Modèle de la bobine.
- Modèles des contacts.

Chacun de ces modèles décrit son comportement électrique (en terme d'admittance) et son comportement séquentiel (traduit par l'état logique).

3.4.3.1 Modèle de la bobine.

D'un point de vue strictement électrique, on considère la bobine du relais comme une inductance et elle sera intégrée comme telle dans les matrices de description du circuit. Mais c'est également un électroaimant qui doit déplacer une charge (le contact et le ressort de rappel). On lui associe un état logique indiquant si le relais est actif (charge déplacée) ou au repos (charge en position initiale). La valeur de cet état est fonction de la tension d'alimentation. Les caractéristiques essentielles d'un relais sont les deux tensions de seuil Von et Voff qui déterminent les basculements d'un état à l'autre (voir Fig. 31). Typiquement, si U_n est la tension nominale du relais, on a dans le cas d'un relais courant « télémécanique LC1-D09 » [SCH01] :

$$Von = 0{,}85.U_n \qquad \textbf{et} \qquad Voff = 0{,}6.U_n$$

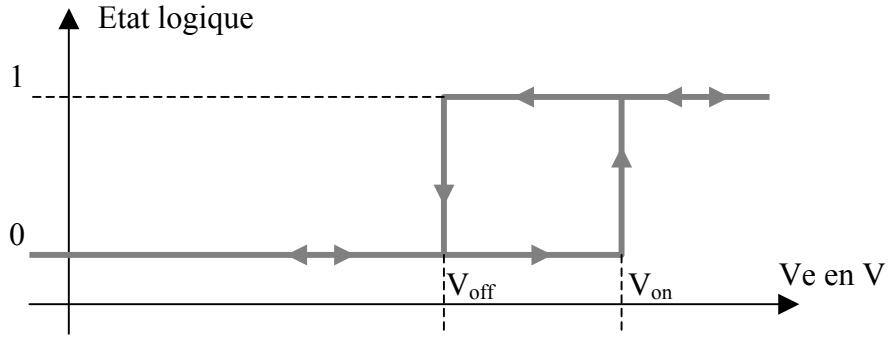


Fig. 31 Changement d'état d'un relais en fonction des niveaux de tension.

Ce modèle décrit un comportement pour lequel la présence d'un niveau de tension active le relais. La désactivation est lié à l'absence de ce signal de commande et est réalisée par le ressort de rappel. Il n'est pas nécessaire d'envoyer une consigne inverse pour désactiver le relais.
Le traitement de ces changements d'états est géré par l'analyse événementielle car il correspond au passage entre 2 états stables.

3.4.3.2 Lien entre bobine et contacts.

Les normes de représentation des schémas d'installations électrotechniques définissent que deux symboles ayant le même nom représentent deux parties d'un même composant. C'est de cette manière que l'on réalise le lien logique entre une bobine de relais et ses contacts. Pour différencier le composant, on utilise les numéros de bornes qui doivent être différents. Par exemple pour un relais nommé KM1, on pourra trouver sur le schéma :

- Une bobine KM1 (bornes A1-A2).
- Un contact à fermeture KM1(bornes 13-14).
- Un contact à fermeture KM1(bornes 23-24).
- Un contact à ouverture KM1(bornes 15-16).

Il ne faut pas oublier que selon le type d'élément, l'état logique n'aura pas le même sens .

Etat Logique	Bobine du relais	Contact NO	Contact NF
0 (Repos)	Ouvert	Ouvert	Fermé
1 (Activé)	Fermé	Fermé	Ouvert

Tab. 6 Signification des états logiques pour les éléments d'un relais.

3.4.3.3 Prise en compte des constantes de temps : retard ou inertie.

Dans le cas de composants idéaux, on a des temps de réaction instantanés qui ne prennent pas en compte les temps caractéristiques des composants (retard, temps d'inertie). Cette inertie peut se traduire par un retard entre l'instant où les conditions sont réunies pour produire un changement d'état et l'instant où il se produit effectivement. Dans une application électrotechnique, cela s'applique notamment à la modélisation du relais.

3.4.3.4 Comportement séquentiel de la bobine du relais.

La bobine d'un relais est un électro-aimant qui sert à déplacer les contacts. Lorsqu'on alimente suffisamment la bobine, le champ magnétique résultant déplace les contacts. Toutefois la magnétisation du relais n'est pas immédiate et nécessite un temps charge (à la fermeture) T_{on} qui est de l'ordre de quelques dizaines de ms. De la même manière, il existe un temps de retard à l'ouverture T_{off} lié à la décharge de la self. Ces retards représentent les temps mesurés entre le moment ou la tension atteint les seuils de déclenchement V_{on} et V_{off} pour provoquer un changement d'état et l'instant auquel il se produit effectivement. En simulation de circuits logiques, on trouve un comportement similaire lors de la propagation d'état des portes logiques.

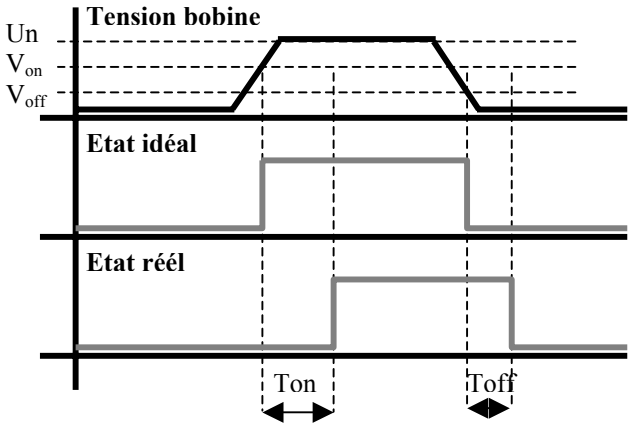


Fig. 32 Retard à l'ouverture et à la fermeture

3.4.3.5 Modèle séquentiel de la bobine.

Pour l'aspect électrique du modèle, nous rappelons que la bobine du relais est considérée comme une inductance réelle (inductance pure et résistance). L'algorithme de la **Fig. 33** modélise la réaction du relais à un nouvel événement : il est appelé à chaque nouvel événement de la simulation pour planifier un éventuel changement d'état. NTe est l'instant (en terme de période d'échantillonnage) où se produit cet événement.

Ce modèle simplifié exploite 4 paramètres essentiels qui sont :

- V_{on} : Tension de fermeture.
- V_{off} : Tension d'ouverture.
- T_{on} : retard à la fermeture.
- T_{off} : retard à l'ouverture.

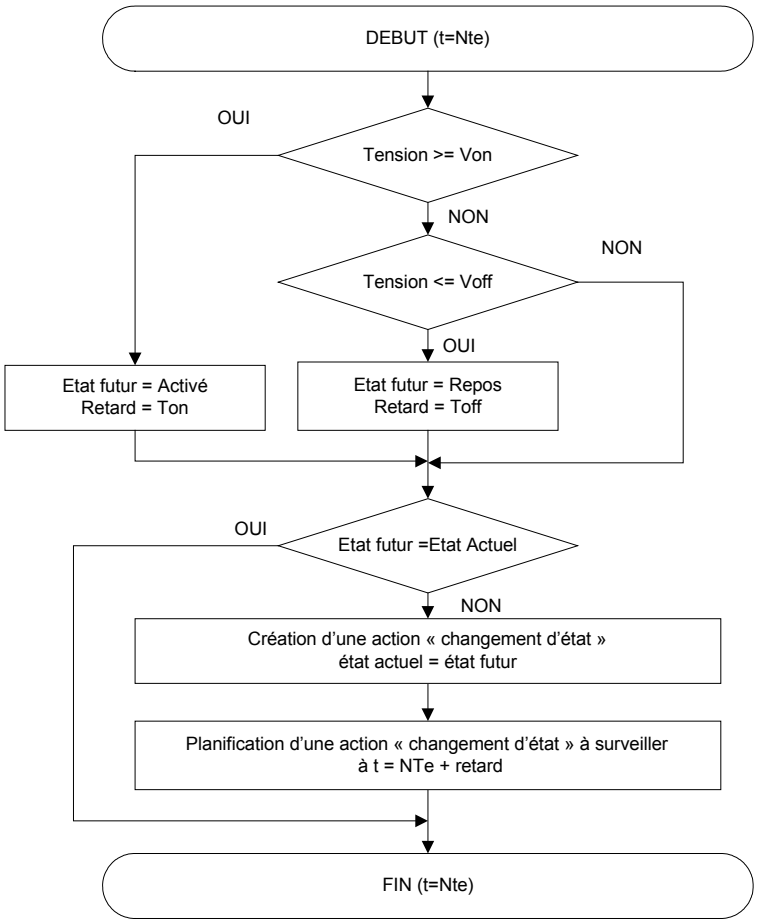


Fig. 33 comportement séquentiel de la bobine

3.4.3.6 Cas particulier : retard inertiel de la bobine .

Le modèle doit prendre en considération le cas particulier des impulsions de commande de durée faible devant les constantes de temps. Ce phénomène est généralement pris en compte par les simulateurs de circuits logiques et connu sous le nom de retard inertiel. Ce retard correspond à un temps minimum pendant lequel les conditions d'un changement d'état doivent être maintenues pour qu'il ait effectivement lieu [NEW78]. Ce temps est nécessaire pour que le niveau d'énergie reçu soit suffisamment élevé pour que la porte logique commute. Dans notre application, si un relais reçoit une impulsion de tension d'une durée inférieure à Ton, il ne sera pas suffisamment magnétisé pour provoquer un changement d'état. Ici le retard inertiel est lié aux retards à l'ouverture et à la fermeture.
La difficulté que l'on rencontre pour traiter ce phénomène est liée au fait que notre simulateur n'observe pas tous les instants d'échantillonnage pour déterminer les changements d'états futurs, mais seulement les transitions. Il ne peut donc pas déterminer si la bobine à été suffisamment alimentée pour obtenir un niveau d'énergie suffisant.

A t = t0, on trouve un passage à une tension supérieure à V_{on}. Il planifie donc un passage à l'état 1 pour t = t2. Par contre, ce changement ne doit être effectif que si la tension reste supérieure à V_{on} jusqu'à t = t2.

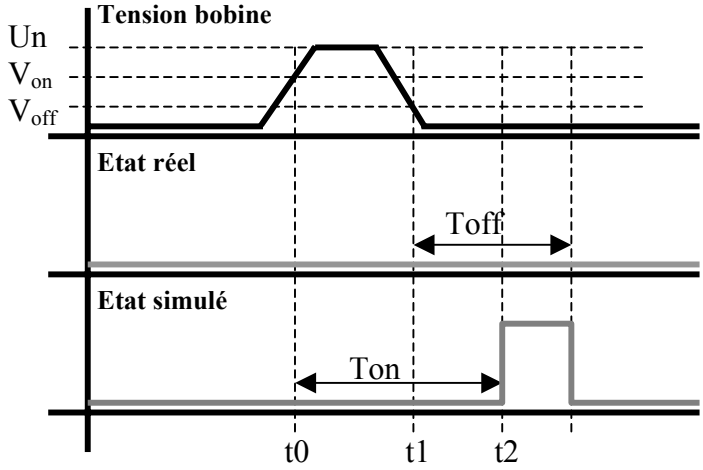


Fig. 34 Etats parasites

Pour notre exemple (voir **Fig. 34**), le comportement attendu est le suivant :
- à t = t0 : la tension est suffisante pour que l'on planifie une fermeture à t2 (t0 + Ton).
- à t = t1 : la tension n'est plus suffisante, la fermeture planifiée précédemment n'a plus lieu d'être et doit être supprimée.

Pour résumer, si entre t0 (l'instant de planification) et t2 (l'instant d'exécution) on planifie une action contradictoire (même si elle est planifiée pour t > t2), la première action n'à plus lieu d'être et doit être retirée du calendrier d'événements. Pour être plus rapide, le simulateur observe uniquement les transitions qui pourront provoquer une variation de la tension et donc un éventuel changement d'état. En fonction de ces tensions, il va planifier les changements d'états mais sans contrôler au moment de les réaliser si les conditions sont toujours correctes.

La première conclusion est que l'on peut distinguer 2 types d'actions :
- Les actions « normales » qui auront lieu quoiqu'il advienne (forçage par exemple).
- Les actions « à surveiller » qui sont planifiées mais dont la réalisation dépend de l'évolution du circuit entre l'instant de planification et son exécution (cas du relais).

En outre, les actions peuvent être de deux natures différentes :
- Modification de l'état logique d'un élément due à un forçage de l'utilisateur ou en réponse à action antérieure.
- Modification d'un paramètre d'un élément due à un forçage ou résultant d'un phénomène transitoire.

Les actions « à surveiller » sont des actions qui sont des réponses à une action antérieure (évolution d'une tension ou d'un état par exemple) donc ne peuvent être que des modifications d'état logique. Ces actions « à surveiller » différent des autres par le fait qu'elles sont suivies par un « observateur d'action ». Tous les observateurs d'action (objets *TActionWatcher*) sont connus par le calendrier d'événements ce qui permet de surveiller facilement les actions en question. La description de ces différents types d'actions et leur implémentation a été développée dans le chapitre 2.
On notera que sur l'algorithme **Fig. 33** décrivant le comportement séquentiel de la bobine, on ne prend pas en compte le problème d'inertie, hormis le fait de planifier l'action de changement d'état comme action « à surveiller ». La solution à ce problème est reportée au niveau du simulateur pour être généralisée à plusieurs composants, au lieu d'être intégrée au modèle de chacun.

3.4.3.7 Gestion de l'inertie par le simulateur.

Une des possibilités est de gérer la suppression des actions à l'intérieur du modèle en fonction de chaque événement. L'avantage est que la gestion de ce comportement est visible dans le modèle. Par contre pour chaque composant ayant des actions à surveiller, il sera nécessaire d'implémenter à nouveau la gestion des actions.
La solution retenue modifie peu le modèle. Au lieu de créer une action de changement d'état, on va créer une action « à surveiller ». Cette action sera créée même si l'état futur est identique à l'état actuel.

Afin d'avoir une gestion commune à tous les composants de ce type d'action, ce n'est pas le modèle qui va gérer leur suppression éventuelle mais c'est le calendrier d'événements qui le fait lorsqu'on lui ajoute une nouvelle action (planification, voir **Fig. 35**).

3.4.4 Contacts.

On peut associer un ou plusieurs contacts à chaque bobine. Etant donné que c'est la bobine qui pilote la position des contacts, on la considère comme étant le maître des contacts. Les contacts sont donc esclaves, ce qui se traduit par l'imposition de leurs états logiques par la bobine. Il existe deux types de contacts :

- Normalement Ouvert (NO ou à fermeture).
- Normalement Fermé (NF ou à ouverture).

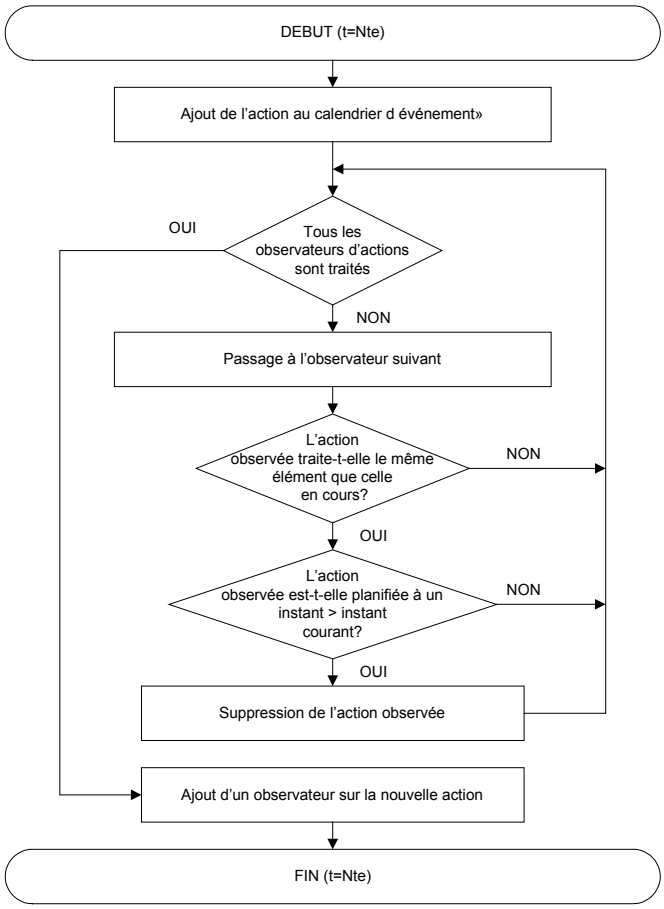


Fig. 35 Ajout d'une action à surveiller au calendrier d'événements

3.4.4.1 Contacts à fermeture, à ouverture

Le contact à fermeture (ou NO pour Normalement Ouvert) est celui que l'on rencontre le plus couramment. Lorsque le contact est ouvert, son modèle électrique est celui d'un circuit ouvert et lorsqu'il est fermé, il est remplacé par une résistance représentative de la résistance du contact (de l'ordre de 1 mΩ) comme le montre la **Fig. 36**.

Fig. 36 Modèle d'un contact à fermeture au repos et activé.

Le fonctionnement du contact à ouverture (ou NF pour Normalement Fermé) est l'inverse du précédent. Au repos, il est fermé et il s'ouvre lorsqu'il est activé.

Fig. 37 Modèle d'un contact à ouverture au repos et activé.

Rappelons que ces modèles de contact sont complètement dissociés des modèles de bobines et que les contacts peuvent être utilisés seuls ou pilotés par d'autres composants maîtres.

3.4.4.2 Contacts temporisés.

Dans le cas général, les contacts d'un relais sont liés mécaniquement à la bobine et ils sont donc actionnés simultanément (contacts instantanés). On rencontre toutefois quelques cas où un retard existe entre l'activation des relais :

- Retard à l'activation et/ou au repos.
- Fonctions de temporisation plus complexes.

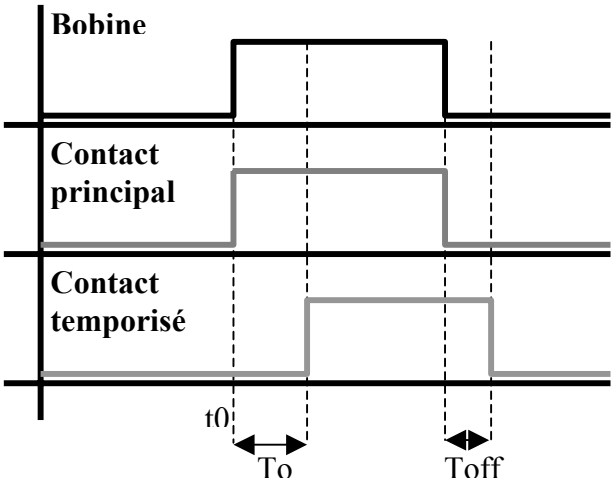


Fig. 38 Transmission des états logiques avec des contacts temporisés.

On peut introduire volontairement un retard à l'ouverture ou à la fermeture des contacts. On considère en fait le retard par rapport au changement d'état (activation ou repos). Par exemple, on peut utiliser des blocs additifs que l'on fixe sur le relais : lorsque le relais change

d'état (ainsi que ses contacts principaux), il active une minuterie mécanique qui déclenchera le basculement des contacts temporisés après un temps T_{on} ou T_{off} préréglé.
Comme c'est la bobine qui est maître du contact, c'est elle qui va déterminer l'instant ou celui ci va commuter. Lorsque le simulateur va planifier la fermeture de la bobine à t=t0, celle des contacts principaux sera prévue à t=t0 et celle des contacts temporisés à t=t0 +T_{on}.

3.4.4.3 Fonctions de temporisation plus complexes.

On trouve de nombreuses autres fonctions de temporisations plus complexes mais qui sont généralement réalisées par des relais temporisés dédiés :

- Clignotement.
- Passage à l'enclenchement.
- Multifonctions.
- Interrupteurs horaires.
- Minuteurs.
- Télérupteurs.
- Temporisateur.
- Temporisé à la mise sous tension.
- etc.

Dans la majorité de ces cas, c'est un appareil dédié qui réalise la fonction. Ces composants moins fréquents n'ont pas été modélisés à ce jour.

3.4.4.4 Pertinence du modèle de contact.

Par rapport au modèle SPICE d'un interrupteur piloté (voir paragraphe **3.4.2.1**), ce modèle est plus réaliste. Pour l'interrupteur fermé le modèle est identique. Pour l'interrupteur ouvert, on retrouve ici un circuit ouvert plutôt qu'une résistance de grande valeur. Le circuit ouvert est une représentation idéale, car la résistance d'un contact ouvert n'est pas réellement nulle du fait de la conductivité de l'air. Mais cette représentation semble plus adaptée car en premier lieu, on ne trouvera aucun courant circulant dans le contact ouvert. Ensuite, l'introduction d'une résistance élevée pour modéliser un circuit ouvert se traduit par l'insertion d'une admittance très faible dans les matrices de calcul. Lors de la résolution et à cause des diverses manipulations et arrondis cette valeur peut devenir un 0. On peut alors se retrouver avec un système d'équations insoluble (plus d'inconnues que d'équations). De plus, ce modèle respecte davantage la logique de fonctionnement du contact car c'est uniquement son état qui va déterminer s'il est ouvert ou fermé et pas un seuil de tension qui ne le concerne pas directement.

3.5 Disjoncteurs et fusibles.

La modélisation de ces équipements de protection est exploitée essentiellement par l'analyse événementielle. En simulant le comportement de ces équipements, on va pouvoir contrôler :

- Le bon fonctionnement individuel de chaque dispositif de protection ;
- Le dimensionnement de ces équipements (type, calibre) en fonction du circuit qu'ils doivent protéger ;
- Le respect des règles de sélectivités horizontales ou verticales entre les différents équipements d'une installation ;
- Etc. ;

Ces équipements peuvent être unipolaires ou multipolaires. Les équipements multipolaires sont constitués d'équipements unipolaires réunis dans un même boîtier. L'objet de la modélisation est de décrire l'élément de base qu'est le "pôle".

3.5.1 Généralités.

Dans le cas d'installation Haute Tension ou Basse Tension, il existe différents équipements permettant d'assurer une protection contre les surintensités de divers matériels.

	Fusibles	Relais Thermique	Relais Magnétique	Relais Magnéto-Thermique
Surcharges 2 à 3 In		OUI		OUI
Courts-circuits > 10 In	OUI		OUI	OUI

Tab. 7 Domaine de fonctionnement des différents dispositifs de protection.

Dans les surintensités, il faut discerner :

- Les surcharges qui correspondent à un courant de 2 à 3 fois de courant nominal.
- Les courts-circuits avec un courant supérieur à 10 fois le courant nominal.

Tous ces équipements de protections ont une caractéristique de fonctionnement Td =f(I) qui donne son délai de déclenchement Td en fonction du courant I qui le traverse. Pour chaque type d'équipement et chaque classe de matériel, on va trouver une courbe caractéristique qui va refléter son comportement particulier.

Dans un premier temps, nous avons choisi de modéliser les équipements dédiés aux installations de distribution et de protection HT/BT : fusibles et disjoncteurs magnétothermiques.
On peut distinguer, pour ces équipements, deux classes de caractéristiques:

- Caractéristiques principales : expriment le comportement idéal du point de vue fonctionnel.
- Autres caractéristiques : traduisent les limitations physiques des équipements.

3.5.2 Fusible.

C'est une cartouche contenant un fil métallique qui est calibré pour fondre lorsqu'il est soumis à un courant trop important et ainsi couper un circuit. Une cartouche fondue est détruite et doit être remplacée.

3.5.2.1 Caractéristiques principales :

- **Calibre (ou courant nominal)** : c'est l'intensité normale d'utilisation qui peut traverser le fusible sans provoquer sa fusion.
- **Type :** Il existe 2 catégories de fusibles adaptés aux différents cas d'emploi.

Type	Protection assurée.	Emploi
gI	Faible et fortes surcharges, courts-circuits	Usage Industriel : pour circuit sans courant d'appel élevé.
AM	Fortes surcharges et courts-circuits	Accompagnement Moteur : doit résister aux surcharges dues au démarrage d'un moteur.

Tab. 8 Emploi des types de fusible

- **Courbe caractéristique : courbe de fusion.**

Les constructeurs fournissent des séries de courbes T = f(I) qui donne le temps de fusion en fonction du courant qui le traverse (voir **Annexe 2**). En se reportant à la **Fig. 39**, Il suffit de connaître le type et le calibre du fusible pour trouver la courbe correspondante. A partir de cette courbe, on peut déterminer pour chaque valeur du courant le temps de fusion associé.

- **Consommation :**

La cartouche fusible se comporte comme une résistance et consomme donc de l'énergie. Selon le calibre cette valeur peut varier entre 1Ω et 1 mΩ.

Cartouches	Calibres (A)																	
	1	2	4	6	8	10	12	16	20	25	32	40	50	63	80	100	125
8,5 x 31,5	0,4	0,6	0,7	1	1,2	1,2	1,2	1,9									
10 x 38	0,27	0,50	0,90	1,05	1,30	1,35	1,45	2,20	2,50	3,50							
14 x 51		0,80	0,90	1,40		2		2,60	3,10	3,50	3,50	3,70	4,60				
22 x 58			1,50	1,60		1,90		3	2,90	3,90	3,60	3,90	5,30	6,30	8	8	11

Tab. 9 Consommation d'un fusible selon le calibre

3.5.2.2 Autres Caractéristiques.

Les fusibles possèdent d'autres caractéristiques que nous n'avons pas encore exploitées essentiellement pour préserver la simplicité du paramétrage du modèle, dont notamment :

- Tension Nominale : Tension maximum d'utilisation normale.
- Pouvoir de coupure : Courant maximum que le fusible peut couper.
- Courbe de limitation.
- Contraintes thermiques.
- Etc…

Elles ne sont pas utilisées car elles définissent pour l'essentiel les limites fonctionnelles de l'utilisation des fusibles ainsi que le domaine de validité des caractéristiques plus essentielles.

Les modèles que nous avons développés permettent de simuler le fonctionnement d'un fusible dans des conditions d'utilisation normales.

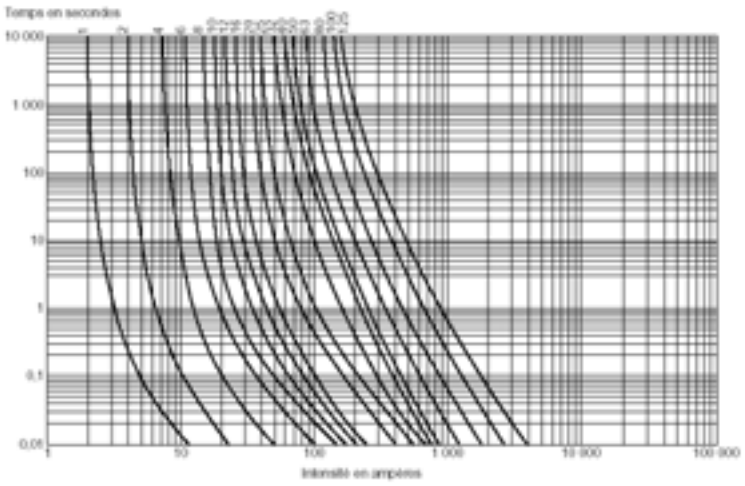


Fig. 39 Courbes de fusion de fusibles

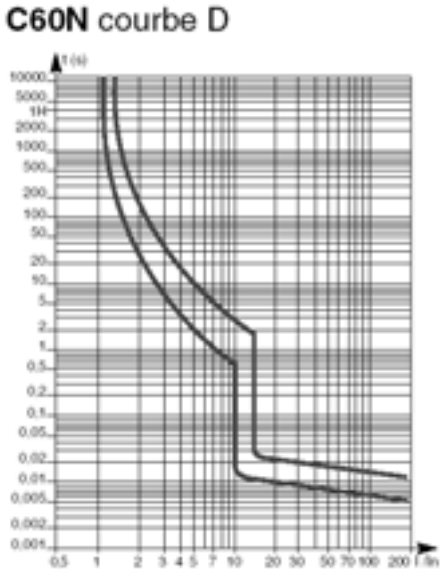


Fig. 40 Courbe de déclenchement de disjoncteurs

3.5.3 *Disjoncteur magnétothermique.*

Le Disjoncteur MagnétoThermique (DMT) combine les effets de deux types de disjoncteurs :

- Disjoncteur Magnétique pour les courts-circuits.
- Disjoncteur Thermiques pour les surcharges.

Comme un fusible, c'est un dispositif qui coupe un circuit au passage d'un courant trop élevé pendant un temps suffisant. Contrairement à la fusion du fusible, le déclenchement du disjoncteur n'est pas destructif. Il peut être réarmé presque indéfiniment.

3.5.3.1 Caractéristiques principales.

- **Calibre (ou courant nominal)** :c'est l'intensité normale d'utilisation qui peut traverser le fusible sans provoquer son déclenchement.
- **Type** : il existe 4 grandes classes de DMT, détaillés dans le **Tab. 10**.

Le disjoncteur permet de protéger les circuits et les personnes contre les surcharges mais pas contre les fuites de courant. Pour cela, il doit être complété par un dispositif de déclenchement différentiel, sensible à la différence de la valeur efficace des courants des différents pôles. Ce type de dispositif n'est pas modélisé ici.

Comme pour les fusibles, on dispose d'une caractéristique qui détermine l'instant où la protection ouvre le circuit en fonction du courant qui le traverse Td=f(I). Toutefois, ici on dispose de deux courbes minimum et maximum qui délimitent la plage de fonctionnement nominal du disjoncteur (voir **Fig. 40**). Le simulateur utilise la moyenne de ces deux courbes pour déterminer le temps de déclenchement. On retrouvera en **Annexe 3** les différents types de courbes

Type	Déclenchement	Application
B	Entre 3 et 5 In Seuil Bas	Source à faible puissance de court-circuit. Grandes longueurs de câbles
C	Entre 5 et 10 In Seuil Standard	Protection des circuits dans le cas général
D	Entre 10 et 20 In Seuil haut	Circuits à fort appel de courant (transformateurs ou moteurs)
MA	à 12 In	Protection des moteurs en association avec un discontacteur.

Tab. 10 Types de courbes de déclenchement de disjoncteur

Il est essentiel de noter que cette caractéristique est donnée par des essais en courant continu. Le comportement en régime sinusoïdal peut être en fonction de la zone de fonctionnement sensiblement différent comme nous le présentons dans le chapitre 4 (validation).

3.5.3.2 Autres caractéristiques.

Comme dans le cas du fusible, nous n'avons pas pris en compte ces caractéristiques moins essentielles, mais qu'il serait intéressant d'utiliser pour affiner la précision du modèle.

- Tension Nominale : Tension maximum d'utilisation normale.
- Pouvoir de coupure : Courant maximum que le fusible peut couper.
- Limitation en courant.
- Limitation en contraintes thermiques.

3.5.4 Cas des équipements multipolaires.

Pour un équipement isolé, le fonctionnement d'un fusible et d'un Disjoncteur MT est sensiblement le même, sauf que l'allure des courbes caractéristiques est différente.
Lorsque l'on dispose d'équipements multipolaires, l'interaction entre les pôles est différente.

- Ouverture ou fermeture Manuelle du circuit (déclenchement manuel du disjoncteur ou ouverture/fermeture du porte-fusible) : Dans les 2 cas tous les pôles sont ouverts/fermés en même temps.
- Défaut sur l'une des phases :
    - Fusible : Le fusible fond et coupe la phase concernée sans que les autres pôles ne soient affectés.
    - Disjoncteur MT : Tous les pôles sont liés mécaniquement, donc si un défaut fait déclencher l'un des pôles, tous les pôles sont déclenchés.

3.5.5 Modèles d'équipement de protection contre les surintensités :

Les fusibles et les disjoncteurs ont des comportements très similaires, aussi bien concernant l'aspect électrique que séquentiel. Les principales différences tiennent à leurs paramètres qui sont dépendants du type de matériel et la forme de leur courbe de déclenchement. Nous avons donc développé un modèle générique d'un équipement de protection contre les surintensités dont nous avons pu faire dériver les deux modèles de fusible et de DMT.

3.5.5.1 Complexité du modèle.

Ces modèles n'exploitent que les caractéristiques principales des équipements. La prise en compte des limitations physiques de ce type de matériel (Tension nominale, pouvoir de coupure, etc) n'est pas réellement pertinente. En effet, l'objectif est dans un premier temps d'obtenir un modèle simple qui permettent de simuler fidèlement le fonctionnement des équipements. Ces choix permettent de caractériser ces matériels en utilisant seulement trois paramètres : **Le type de matériel, le calibre et l'impédance des pôles.**

3.5.5.2 Différents aspects des modèles.

Le modèle doit nous permettre de définir le composant en terme :

- D'impédance et d'excitation, afin d'intégrer sa participation dans le circuit et obtenir ainsi ses tensions et courants : c'est l'aspect électrique du modèle.
- De fonctionnement, pour exprimer sa réaction aux diverses sollicitations extérieures (exploité par l'analyse événementielle) : c'est l'aspect événementiel du modèle.
- D'interaction entre le composant et le circuit auquel il est intégré (protections multipolaires) : propagation des états logiques.

3.5.6 Implémentation du modèle de protection.

Cette implémentation concerne la description d'un matériel unipolaire qui comporte deux bornes. L'implémentation de matériels multipolaires sera obtenue par la création de circuits à base de matériels unipolaires.
Pour chaque nouveau type de composant, il est nécessaire d'implémenter un nouvel objet qui décrira son comportement. L'un des intérêts de la programmation orientée objet réside dans l'exploitation de l'héritage : Un objet qui hérite de son « ancêtre » a les mêmes propriétés que son ancêtre. Cela permet d'avoir pour deux objets héritant d'un ancêtre commun d'avoir un comportement identique bien qu'ils aient de nombreux paramètres qui les différencient.
Nous avons alors défini une classe abstraite *TProtectionCC* qui regroupe les propriétés communes des fusibles et des disjoncteurs qui eux sont implémentés par les classes enfants *TFusible* et *TDisjoncteurMagnTher*.

3.5.6.1 Nouvelles classes d'objets.

Dans notre cas, nous avons deux composants qui ont une caractéristique commune (courbe Td = f(I)) qui régit leur comportement. Par contre, il existe quelques divergences qui nécessitent d'avoir deux objets distincts ayant des propriétés différentes.
Dans ce cas de figure, le choix le plus adapté est de définir deux classes *TFusible* et *TDisjoncteurMagnTher* qui décrivent le comportement de nos équipements. Par soucis d'efficacité, on définit une classe parente abstraite qui décrit leur comportement commun par rapport au déclenchement.

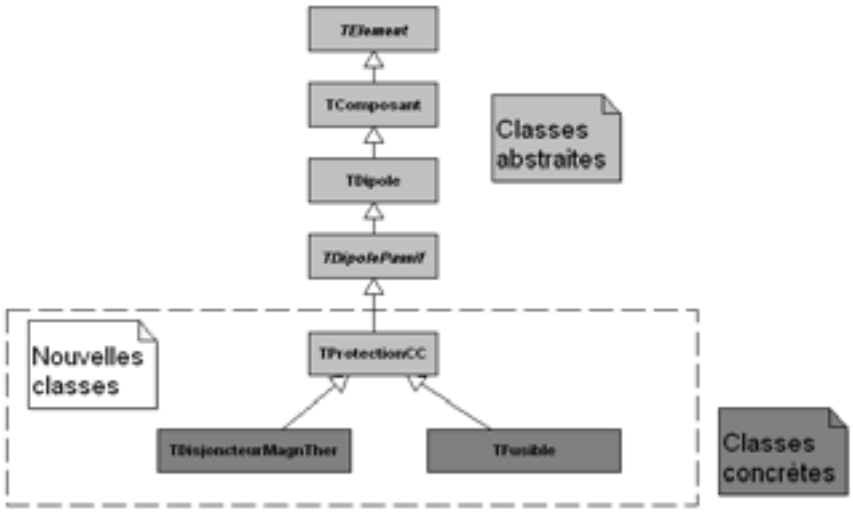


Fig. 41 Diagramme d'héritage des classes descendantes de TProtectionCC

Classes	Super-Classes	Rôle
TElement	*Tobject*	Décrit tous les éléments simulables (composants de base et circuits).
TComposant	*TElement*	Composants de base (par opposition aux circuits)
TDipole	*TComposant*	Composant possédant deux bornes et une branche
TProtectionCC	*TDipole*	Décrit le comportement des protections contre les surintensités (déclenchement)
TFusible	*TProtectionCC*	Implémentation d'un fusible
TDisjoncteurMagnTher	*TProtectionCC*	Implémentation d'un disjoncteur Magnétothermique

Tab. 11 Différentes classes intervenant dans les modèles d'équipement de protection

3.5.6.2 Classe Abstraite et parente TProtectionCC :

Ce composant sera modélisé comme une résistance lorsqu'il est fermé (*State = True*) et comme un circuit ouvert s'il est ouvert (*State = False*).
Cette procédure de description du fonctionnement *TProtectionCC .OnNewEvent* est appelée après chaque nouvel événement pour répercuter l'évolution d'installation sur ces équipements de protection.

Champs	Type	Rôle
Inominal	Réel	Calibre de l'équipement (en A) : courant nominal de fonctionnement.
OldState	Booléen	Etat de l'équipement lors de l'événement précédent.
Résistance	réel	Valeur de la résistance du contact (Disjoncteur) ou du fil (Fusible).
MeltingCurve	Tableau de Points	Contient les points de la courbe de déclenchement
CurrentHeat	Enregistrement	Etat de progression du déclenchement.

Tab. 12 Propriétés de la classe *TProtectionCC*

Méthodes	Type	Rôle
Create	Constructeur	Création et initialisation de l'objet.
Destroy	Destructeur	Destruction de l'objet.
Assign(Source)	procédure	Attribue aux champs ceux de l'objet « source ».
InsertionMatrice	procédure	En fonction de l'état (State) insert une résistance dans les matrices.
OnNewEvent	procédure	Description du comportement événementiel (déclenchement de l'équipement).

Tab. 13 Méthodes de la classe *TProtectionCC*

3.5.6.3 Première étape de la modélisation.

Cette première implémentation répond aux premières attentes qui sont de reproduire fidèlement le fonctionnement des équipements à partir de la seule courbe de déclenchement :

- La mesure du courant efficace permet de planifier un déclenchement dans un temps correspondant à ce courant.
- Il faut toutefois distinguer la planification de cet événement, qui correspond à sa mise en file d'attente, de l'exécution de cet événement qui ne sera réalisée que si toutes les conditions de déclenchement sont réunies.
- Il faut qu'entre l'instant où l'on a planifié et le déclenchement le courant n'ait pas changé. Dans le cas contraire, l'événement prévu est annulé et est remplacé par un événement correspondant au nouveau courant.
- L'annulation d'un évènement prévu est réalisée automatiquement par le calendrier d'événement car ces déclenchements sont des événements classés comme événements à surveiller.
- **Problème rencontré** : à chaque variation du courant, le déclenchement précédent est annulé et on repart à 0, on ne garde pas de trace de l'action qu'a pu avoir le courant d'où un modèle plus complexe. Nous avons donc complété ce modèle afin de résoudre ce défaut.

3.5.6.4 Deuxième étape de la modélisation.

La courbe caractéristique donne T= f(I). Pour un courant constant il est simple de connaître le temps qu'il faudra pour déclencher. Lorsque la valeur efficace du courant va évoluer (tout en restant supérieur au courant nominal), on peut difficilement prévoir le comportement de la protection. Le but de cette nouvelle modélisation du comportement est d'enrichir le modèle précédent pour prendre en compte des variations du courant. La **Fig. 42** décrit le comportement séquentiel des protections. L'apport de cette deuxième étape y figure en gris.

La modélisation retenue consiste à pondérer le temps de déclenchement en prenant en compte le passé du composant, de manière à traduire l'état d'échauffement des dispositifs thermiques par exemple. Pour cela, on associe de nouvelles variables sous forme de l'enregistrement (ou structure) CurrentHeat qui sauvegarde des informations concernant l'échauffement.

Champs de CurrentHeat	type	Rôle
Start	Entier	Début de l'échauffement (instant de planification du déclenchement).
Duration	Entier	Temps de déclenchement planifié.
DoneRate	Réel	Taux de progression du déclenchement (de 0 à 1).

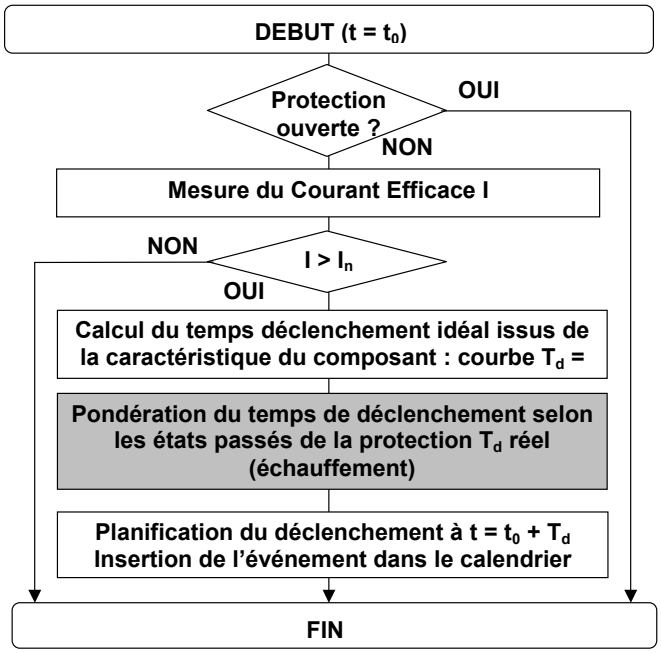


Fig. 42 Comportement séquentiel des équipements de protection

Illustration par un cas pratique :

Nous avons ici un cas où le disjoncteur subit une variation de la valeur efficace du courant. Cela peut correspondre à une surcharge passagère ou à un phénomène transitoire.

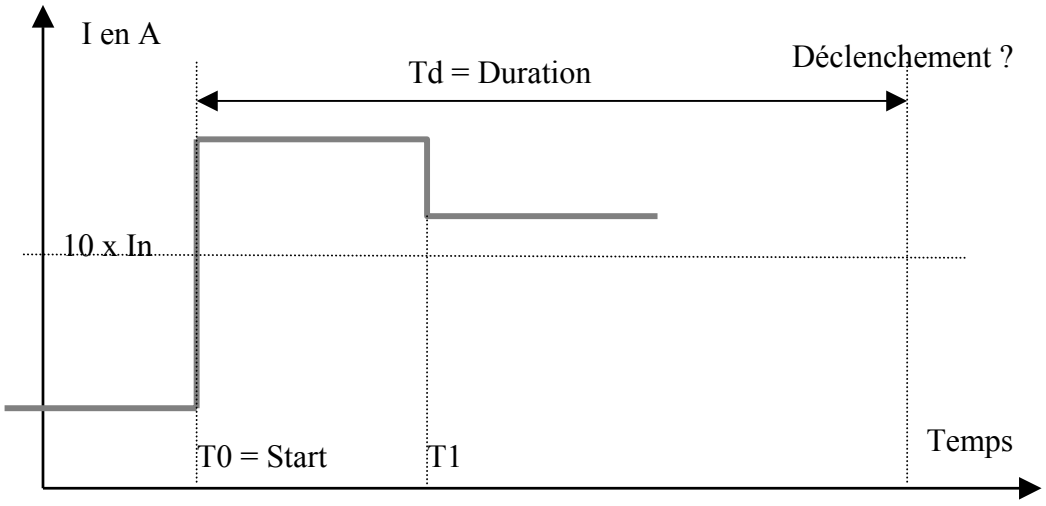


Fig. 43 Surcharge de courant avec variation de la valeur efficace

- A t = T0, le courant atteint un niveau qui peut provoquer un déclenchement après un temps Td.
- On enregistre CurrentHeat.Start = T0 et CurrentHeat.Duration = Td
- On planifie un déclenchement à T0+ Td (à surveiller).
- A t = T1, le courant change.
- On calcule le taux de réalisation du déclenchement soit :
  CurrentHeat.DoneRate = CurrentHeat.DoneRate + (T1 – T0) / CurrentHeat.duration.
  Avec T1 = Nte (instant présent) et T0 = CurrentHeat.Start.
- Calcul du nouveau Td correspondant au nouveau courant.

- On enregistre CurrentHeat.Start = T1 et CurrentHeat.Duration = Td
- On calcule le Nouveau CurrentHeat.DoneRate.
- On pondère le temps de déclenchement Td = Td (1- CurrentHeat.DoneRate.)
- On planifie ce nouvel événement ce qui a pour effet de retirer le précédent.

3.5.7 Modèle du fusible : Classe concrète TFusible :

Cet objet modélise une cartouche fusible. Le principal apport de cette classe *TFusible* par rapport à son ancêtre *TProtectionCC* est de permettre l'affectation d'un tableau de point représentant la courbe de déclenchement. A partir du calibre et du type de matériel, il choisit la courbe prédéfinie qui lui correspond. Elle décrit la relation Td=f(I). Cette classe n'a pas d'influence sur la définition du comportement du composant qui est totalement décrite au niveau de *TProtectionCC* , mais uniquement sur le paramétrage du composant.

Attributs	**Type**	**Rôle**
TypeAm	Booléen	Type de cartouche fusible Vrai si Type Am Faux si type gI.
Méthodes	**Type**	**Rôle**
Create	Constructeur	Création et initialisation de l'objet. Paramètres : type, calibre, puissance
Destroy	Destructeur	Destruction de l'objet.
Assign(Source)	procédure	Attribue aux champs ceux de l'objet « source ».
Clone	Clone	Renvoie un *TFusible* qui est une copie de cet objet

Tab. 14 Attributs et méthode de la classe *TFusible*.

3.5.8 Modèle du disjoncteur : Classe TDisjoncteurMagnTher :

Cet objet modélise un disjoncteur Magnétothermique. Comme pour le fusible (*TFusible*), le modèle de disjoncteur *TDisjoncteurMagnTher* ne modifie pas le comportement du composant, il va seulement permettre d'attribuer une courbe de déclenchement au composant. A partir du type de seuil, on désigne une courbe normalisée Td=f(I/In) qui à partir de la valeur In du calibre choisi est ramenée à une forme Td=f(I) utilisable par le modèle.

Attributs	**Type**	**Rôle**
Threshold	chaîne	Type de seuil B, C, D, MA
MultiParent	pointeur	Si ce composant fait partie d'un multipôle, ce pointeur pointe dessus.
State	Propriété(booléen)	Redéfinit l'accès en lecture écriture à l'état.
Méthodes	**Type**	**Rôle**
Create	Constructeur	Création et initialisation de l'objet. Paramètres : seuil, calibre.
Destroy	Destructeur	Destruction de l'objet.
Assign(Source)	procédure	Attribue aux champs ceux de l'objet « source ».
Clone	Clone	Renvoie un TFusible qui est une copie de cet objet
SetState	procédure	Si multiparent existe copie son état sur celui de l'objet en cours

Tab. 15 Attributs et Méthodes de la classe *TDisjoncteurMagnTher*

4 Nécessité d'un langage standard pour la modélisation.

Les différents modèles de composants ont été modélisés en utilisant le langage Delphi, employé également pour l'implémentation du simulateur. L'emploi de ce langage offre une grande souplesse d'utilisation pour la modélisation du matériel. Il permet d'utiliser avantageusement les principes de la programmation orientée objet du fait que le simulateur et les modèles sont écrits en Delphi, compilés ensemble et intégrés dans une seule application cela assure une plus grande cohérence entre l'outil et les modèles. Toutefois, cette solution, plus simple pour une utilisation standard, amène un certain nombre de contraintes dans une utilisation plus avancée du simulateur :

- Delphi est un langage de programmation et non de modélisation. De ce fait il ne constitue pas un standard en terme de modélisation. Il est alors nécessaire de faire preuve d'une grande rigueur à la définition d'un modèle afin de conserver une cohérence dans la modélisation ceux-ci pouvant être implémentés de multiples manières pour un même résultat.
- Les modèles ainsi décrits sont « en dur ». Ils sont inscrits dans l'application et ne sont nullement lisibles ou modifiables par l'utilisateur.
- De la même manière que les modèles existants sont inaccessibles, l'utilisateur n'a pas la possibilité de créer ses propres modèles et de les simuler. Il devrait pour cela les écrire en Delphi et disposer du code source de toute l'application (schématique et simulation) pour recompiler l'ensemble.

De ces différents constats, il apparaît que la modélisation via Delphi peut être relativement satisfaisante pour l'utilisation courante et simple du simulateur. Mais, dans l'optique d'obtenir un outil ouvert, convivial et validé scientifiquement, il est indispensable d'utiliser un langage de description matérielle ou HDL (Hardware Description Language).
Un langage de description matérielle contrairement à un langage informatique n'est pas seulement utilisé pour l'exécution qui dans notre cas est la simulation. Cette description de matériel peut servir les différentes étapes du cycle de conception, spécifications, modélisation, simulation, documentation, synthèse…[HER02]

4.1 Choix du langage.

Dès lors que la nécessité d'utiliser un langage de modélisation externe au simulateur est apparue indispensable, il convient de déterminer quel sera ce langage. Différentes solutions s'offrent à nous. Pour faciliter l'intégration des modèles et répondre aux contraintes techniques imposées par Algo'Tech, les modèles décrits dans le langage retenu seront compilés pour produire des modèles Delphi du même type que ceux utilisés à l'origine. Ces modèles « Delphi » seront eux même recompilés sous forme d'un fichier binaire exploitable par le simulateur. Afin de préserver la simplicité d'utilisation du simulateur, il serait même préférable de n'utiliser cet HDL que comme un support standard à la description des modèles. Dans le cas idéal, les modèles seraient entièrement construits de manière graphique pour être traduit dans le langage choisi (voir **Fig. 44**). Ainsi, tout utilisateur pourra définir simplement des modèles sans l'apprentissage d'un langage.

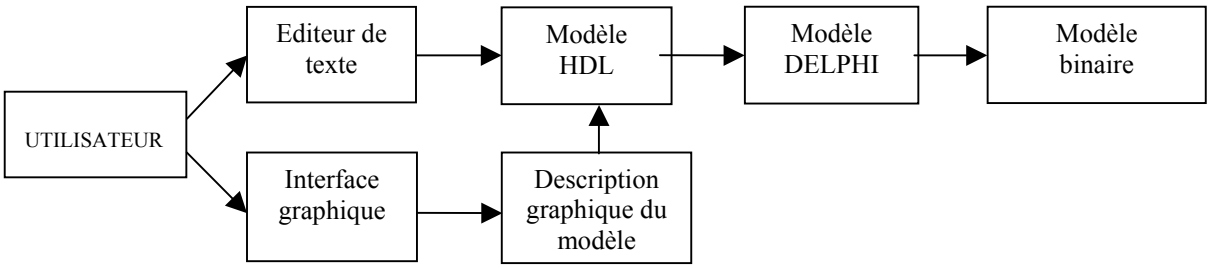


Fig. 44 Différentes étapes de la construction et compilation des modèles VHDL-AMS.

4.2 Langage dédié au simulateur.

Un langage propriétaire est défini par le concepteur de l'application qui l'exploite. Dans notre cas, il s'agirait de mettre en place un langage propre à notre simulateur permettant à chacun d'écrire et de compiler ses propres modèles dans le but de les simuler. Le langage ainsi défini présenterait l'avantage d'être adapté au plus près à l'outil de simulation. Pour cela, il peut reprendre la démarche de modélisation utilisée dans les modèles « Delphi » et intégrer des instructions permettant de piloter le simulateur. On pourrait même envisager d'utiliser un langage qui soit un sous-ensemble (ne supporte pas tous les lexemes) de Delphi, réduisant ainsi les besoins de développement de compilateur. Ainsi chaque modèle serait un script Delphi que l'on compilerait séparément pour qu'il soit pris en compte par le simulateur.
Pour résumer l'ensemble des avantages que peut présenter cette solution, on disposerait donc d'un langage mieux adapté, plus souple, plus facile à mettre en œuvre (pour le concepteur du simulateur). En fait ces avantages sont essentiellement favorables au concepteur de l'outil plutôt qu'à son utilisateur.
Rappelons qu'à chaque étape de la conception de l'outil de simulation, un soin particulier est apporté pour préserver la **simplicité d'emploi vis à vis de l'utilisateur**. Un langage propriétaire n'étant pas défini par une norme, il nécessitera un apprentissage. Les modèles ainsi décrits ne seront utilisables dans aucun autre outil. De plus, bien qu'accessibles, les modèles ne seront compréhensibles qu'aux seuls initiés qui auront fait l'effort d'intégrer ce langage. Enfin, la mise en place nécessite une réflexion approfondie nécessaire pour définir ses domaines d'application, sa grammaire, sa syntaxe afin de pouvoir tous les besoins présents et à venir du simulateur en fonction de son potentiel et des futures évolutions.

4.3 Langages standards.

La solution d'un langage propriétaire n'est bien entendu pas entièrement satisfaisante. Nous nous sommes naturellement intéressés à des langages existants, considérés comme des standards. Les deux voies qui nous paraissent les plus prometteuses sont les langages Xml [W3C], de plus en plus répandus notamment grâce au développement d'Internet et VHDL-AMS [VHD1] un langage de description du matériel multi-domaine et multi-technologique également issu de l'industrie électronique. Nous avons ainsi étudié les possibilités d'exploiter un langage existant en l'utilisant complètement, partiellement ou en l'adaptant à nos besoins à travers une extension.

4.4 XML.

XML (eXtensible Markup Language) est un format de fichier texte dérivé de SGML (ISO8879). Il a été conçu à l'origine pour répondre au problème de la publication électronique à grande échelle. Il joue également un rôle croissant dans l'échange de données diverses sur le Web en particulier. C'est un langage standard, approuvé en 1998 (XML 1.0) par le W3C (World Wide Web Consortium) [W3C]. A l'instar du HTML (Hyper Text Markup Language), XML utilise un système de balise. Contrairement à HTML, XML n'est pas figé car il permet à chacun de définir ses propres balises [COG00].
Un document XML peut être créé depuis un simple éditeur de texte, via des éditeurs dédiés, des interfaces utilisateurs personnalisées ou par programmation. De nombreux composants logiciels sont disponibles pour traiter ce genre de document. On trouve des parseurs (analyseurs syntaxiques), vérificateurs et éditeurs adaptés à différents langages et plateformes [WOO01]. Un document XML peut être associé à une DTD (Document Type Definitions) qui définit un modèle de document XML valide. Elle décrit le type d'éléments et de données qui peuvent être contenu dans le document ainsi que sa structure. On peut définir un langage XML à l'expression de données particulières. Ainsi on retrouve par exemple MathML [MATH] dédié à la spécification et au traitement de contenu mathématique et scientifique ou encore SVG (Scalable Vector Graphics) permet de décrire des graphiques à deux dimensions [SVG].

L'essor de XML, notamment grâce à Internet, nous a naturellement amenés à rechercher des langages qui nous permettraient de définir nos modèles de composants. Parmi les XML existants nous avons distingué un XML utilisé par Schneider Electric que nous avons baptisé SchneiderML et un format d'échanges pour la conception de circuit électronique EDA-XML.

4.4.1 Delphi et XML.

Grâce à La structure arborescente des éléments dans les documents XML, on peut retrouver très rapidement des données d'après une analyse lexicale du fichier texte. Dans son environnement de développement, Delphi propose différents moyens de traiter les documents XML.

- Classe *TXMLDocument* : cette classe d'objet définit un conteneur de documents XML qui facilite la recherche de données. En l'associant à fichier XML, on pourra lire, écrire ou créer ce fichier. *TXMLDocument* est un analyseur DOM (Document Object Model) qui permet de manipuler les différents éléments (via la classe *TXMLNode*) du document [BOR02].
- Expert liaison de données XML : cet assistant va à partir d'un document XML ou de sa DTD construire des interfaces qui exposent les différents éléments du document. On dispose ainsi avec cet ensemble d'interface de toutes les méthodes permettant de manipuler un type de document XML précis [JAM]. L'expert XML construit immédiatement le code d'un parseur (analyseur syntaxique) adapté au type de document XML que l'on veut traiter et que l'on peut intégrer dans notre application.

Cette simplicité d'utilisation du format XML permet une mise en œuvre très rapide économisant le temps de développement de l'analyseur syntaxique (parseur) au profit de travaux plus fondamentaux. Dans notre contexte particulier, il est intéressant de noter qu'Algo'Tech Informatique a déjà mis en œuvre avec succès ces technologies pour l'exploitation d'un format d'échange XML avec l'application de calcul Ecodial de Schneider Electric.

4.4.2 Schneider-ML.

Au cours de développements indépendants de la simulation, Algo'Tech Informatique a mis en œuvre des technologies XML. Il s'agissait d'exploiter un format de fichier XML issu du logiciel de calcul d'installation électrique Ecodial de Schneider Electric. Ce format de fichier que nous appellerons SchneiderML décrit un réseau électrique (voir la DTD en **Annexe 4**). On peut voir sur la **Fig. 45** que ce document est constitué d'un réseau électrique « NETWORK » qui se compose de deux listes :

- Liste des composants du réseau « `LISTECOMPONENTS` ».
- Liste des connections électrique « `LISTECONNECTIONS` ».

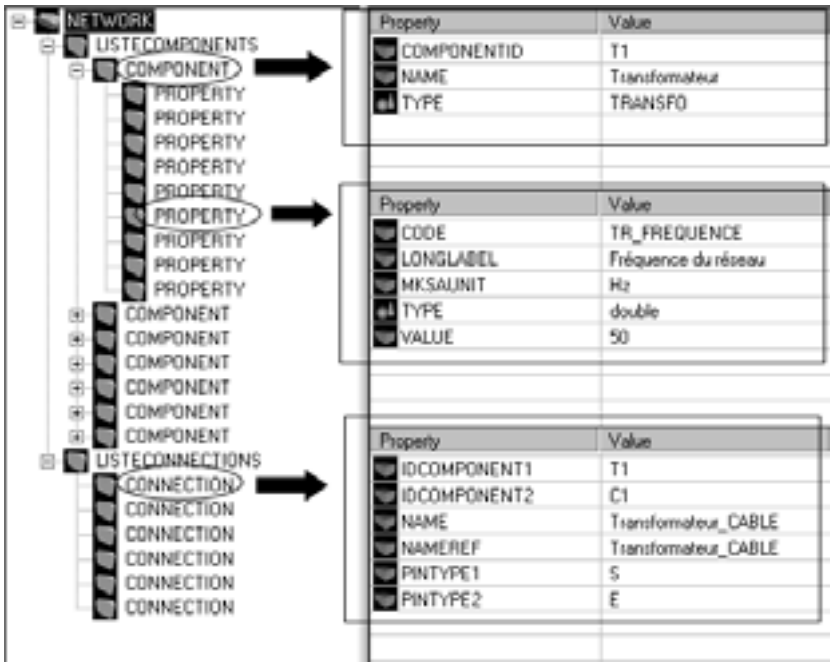


Fig. 45 Exemple de fichier au format « SchneiderML »

Chaque composant « `COMPONENT` » possède un certain nombre d'attributs « `PROPERTY` » dont les valeurs « `VALUE` » le caractérisent. Les connections « `CONNECTION` » met en relation deux composants via leurs bornes.
A partir de ces informations, il est possible de reconstituer un réseau électrique et de connaître les caractéristiques des composants. Ainsi, on peut dessiner le schéma et si les caractéristiques sont adaptées et un modèle est associé à chaque composant, il serait possible de simuler ce réseau.

4.4.3 EdaXML.

EDA-XML est un langage XML d'échange pour l'EDA (Electronic Design Automation). EDA-XML est basé sur un modèle d'information compréhensible semblable à celui utilisé pour la conception de EDIF V4.0.0. Sa DTD de 1675 lignes (autant d'éléments) est disponible en ligne [ETO]. EDA-XML couvre la schématique, la connectique, le PCB (Printed Circuit Board) et le MCM (MultiChip Module). Des outils de visualisation et d'analyse syntaxique sont disponibles. Des outils de traduction bidirectionnels pour différentes applications d'EDA ont été développés par e-tools [ETO] notamment pour :

- Mentor Graphics' DesignArchitect and Boardstation
- Viewlogic's Powerview/Workview
- Cadence OrCAD Capture 9.10
- PADS Powerlogic
- P-CAD
- Cadence Concept Schematic Capture

D'autres outils permettent l'intégration d'EDA-XML dans des applications clients ou des liaisons avec des bases de données de matériel [COV].

Comme le langage précédent, EDA-XML est intéressant car on va y retrouver le réseau électrique correspondant à un circuit électronique. Il va offrir de plus de nombreuses possibilités concernant des informations propres à l'EDA telles que le routage, les masques de PCB qui sont superflues dans notre contexte électrotechnique. Le modèle d'information proposé est par contre insuffisant en ce qui concerne la modélisation comportementale des composants et nécessiterait une extension.

4.4.4 De la pertinence du choix d'un langage XML.

Le premier avantage à utiliser un XML comme support de la modélisation est la simplicité de sa mise en œuvre et d'intégration dans une application. Toutefois, aucun des langages existants que nous avons étudiés ne permet de couvrir la totalité de nos besoins en terme de représentation des circuits. Notre approche consisterait donc à intégrer ces travaux pour proposer un langage XML permettant à la fois de décrire la topologie du circuit et son comportement.

4.5 Langage VHDL-AMS.

Dans notre recherche d'un langage de modélisation, nous nous sommes intéressés tout particulièrement au langage VHDL-AMS qui permet d'utiliser des descriptions du matériel multi-niveaux, multi-technologique, multi-domaines.

4.5.1 Généralités.

VHDL-AMS est un langage de description de matériel (HDL) définit par la norme IEEE 1076.1 de 1999 [VHD1]. Il est basé sur le langage VHDL (norme IEEE 1076 de 1993) dont il constitue l'extension analogique (AMS : Analog and Mixed Signals). L'acronyme VHDL rappelle que le langage est issu de l'électronique. VHDL signifie **V**HSIC (Very High Scale Integrated Circuit) **H**arware **D**escription **L**anguage – *et non pas Very Hard Description Langage, comme certains se plaisent à l'appeler soulignant ainsi sa puissance et sa complexité*. VHDL permet la description de circuits logiques et à l'aide des outils adaptés de la simuler et même de les synthétiser. L'extension analogique de VHDL rajoute de nouvelles instructions et objets, les modèles écrits en VHDL restant exploitables en VHDL-AMS. Ainsi, en étendant VHDL à l'analogique, VHDL-AMS couvre un bien plus large domaine que l'électronique numérique et analogique [HER02]. VHDL-AMS permet de décrire des modèles selon différents niveaux d'abstractions (fonctionnel, comportemental, structurel, physique**),** permettant de considérer différents domaines (électromagnétisme, mécanique, thermique…). Les modèles peuvent être décrits de manière discrète (numérique) ou continue (analogique) qui peuvent être exprimés via des équations différentielles, des transformées de Laplace ou en z notamment.

4.5.2 Utilisation.

Il est essentiel de rappeler que VHDL-AMS est un langage de description et pas un langage de simulation. Un langage de simulation doit permettre de définir aussi bien les modèles que l'outil de simulation. VHDL-AMS est destiné à la description des circuits et pas nécessairement à leur simulation. Il est de ce fait indépendant de l'outil auquel on souhaite l'associer, bien que certaines instructions (comme BREAK) soient destinées aux noyaux de simulation. A ce jour, aucun des outils du marché ne traite la totalité de la norme, donc certains modèles écrit en respectant la norme pourront être compilés mais pas simulés par les simulateurs ou traités seulement par certains. Ainsi chaque logiciel de simulation exploitant VHDL-AMS définit les instructions et déclarations qu'il supporte, comme par exemple les logiciels AdvanceMS [AMS] ou Hamster [HAM] (voir la liste en **Annexe 5**).

4.5.3 Particularités.

Il est intéressant de souligner que VHDL-AMS, ainsi que VHDL, possède dans sa sémantique des instructions propres à un HDL permettant le traitement du temps mais également des variables et procédures comme dans un langage informatique. Avec VHDL-AMS, on dispose à la fois d'un HDL et d'un langage classique. Cela permet d'avoir une grande souplesse dans la description des modèles ainsi que de multiples façons de les écrire tout en bénéficiant de la rigueur imposée par la norme.

Toutes ces caractéristiques font de VHDL-AMS un langage relativement puissant mais complexe ce qui peut rendre son apprentissage assez laborieux. Toutefois il peut être utilisé pour faciliter la génération de modèle via une interface graphique.

4.5.4 Structure d'un modèle VHDL-AMS.

Pour l'essentiel, un modèle est constitué de deux parties :

La spécification de l'entité (mot clé ENTITY) qui définit la vue extérieure du composant, son interface avec le monde extérieur grâce aux mots clés GENERIC et PORT.

- GENERIC : définit les paramètres génériques du composant (constantes à valeurs différées).
- PORT : interface de connexion avec l'extérieur, permet d'échanger des informations.

La description de la fonction du modèle dans la partie Architecture (mot clé ARCHITECTURE), où l'on définit l'implémentation du composant. Cette description peut se faire sous forme d'instanciation de composants, d'instructions concurrentes ou simultanées en manipulant les valeurs disponibles via les ports ou reçues en paramètres.

Il est intéressant de noter que pour un modèle, la norme permet d'associer plusieurs architecture à une déclaration d'entité. Elles peuvent correspondre à la description du composant selon différents niveaux d'abstraction par exemple. C'est à l'instanciation du composant que l'on choisira l'architecture adaptée.

La structure générale d'un modèle [VHD1] est :

Ouverture de bibliothèque (LIBRARY).
Déclaration d'utilisation du contenu des bibliothèques ouvertes (USE).
Spécification d'entité (ENTITY).
    Définition des paramètres génériques (GENERIC).
    Définition des ports de connexion possibles (PORT).
        SIGNAL (in/out) support des informations à événements discrets.
        QUANTITY (in/out) support des informations signal-flow.
        TERMINAL support de connexions « Kirchhoff ».
    Corps de l'entité.
        Ensemble d'instructions passives.
Architecture de l'entité.
    Zone de déclaration.
    Corps de l'architecture.
        Instanciations de composants [support de la hiérarchie].
        Instructions concurrentes (dont les process) [traitement à l'événement discret].
        Instructions simultanées [traitement temps continu].

4.5.5 Exemple de modèle VHDL-AMS : résistance idéale.

Ce modèle de résistance relativement classique reçoit en paramètre la valeur de la résistance. Ses ports sont ses deux bornes. Dans l'architecture, on décrit la loi d'Ohm au sein du composant.

```
------------------------------------------------------------
-- résistance idéale
--
-- p o----/\/\/\----o m
-- R
------------------------------------------------------------
--déclaration des bibliothèques
LIBRARY DISCIPLINES;
LIBRARY IEEE;
--utilisation des bibliothèques
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;
-- declaration d'entité
ENTITY fl_resistance IS
      GENERIC (r :real:= 1.00); --Paramètres générique
      PORT ( TERMINAL RIN, ROUT: ELECTRICAL); -- port du composants
END fl_resistance;
-- declaration de l'architecture
ARCHITECTURE resistance_BODY OF fl_resistance IS
 quantity vr across ir through RIN to ROUT;
BEGIN
      ir == vr / r;
END resistance_BODY;
```

Fig. 46 Modèle VHDL-AMS d'une résistance idéale.

4.5.6 Faisabilité de la traduction.

Un point important dans le choix du langage est la faisabilité de la traduction. L'utilisateur familier des langages informatiques, ne connaissant pas la norme VHDL-AMS, établira assez facilement l'analogie avec un langage de programmation structurée ou orientée objet. En effet, VHDL à été développé en s'inspirant assez ouvertement du langage ADA. On y retrouve une partie de déclarations (mot clé INTERFACE en Delphi) qui regroupe les appels des bibliothèques, les déclarations des différentes classes d'objets, des types, des variables, des constantes. De même, il existe aussi une partie implémentation (mot clé IMPLEMENTATION en Delphi) où les corps des classes d'objets et des fonctions sont décrits.

D'autre part, est-il possible d'utiliser un langage de description comportementale pour un simulateur à primitives comme le nôtre ? Des travaux dans ce sens ont été menés avec succès. Notamment les travaux de thèse d'O. ALALI [ALA98] et de S. JEMMALI [JEM03] qui ont abouti à la réalisation d'un traducteur VHDL-AMS vers SPICE. Notre simulateur comporte des primitives comme SPICE et possède de plus des possibilités de description du comportement séquentiel. La faisabilité de cette traduction semble donc raisonnable.

4.6 Solution retenue : le langage VHDL-AMS.

Les langages XML, EDA_XML et SchneiderML existants et se rapprochant le plus de nos préoccupations sont intéressants à bien des égards par la facilité de leur mise en œuvre, mais ne sont pas utilisables en l'état. Ils doivent être enrichis pour supporter notre besoin de modélisation plus complexe.
A court terme, VHDL-AMS apparaît donc comme la plus adaptée à notre problématique. Bien qu'issue de l'électronique, cette norme récente (1999) n'est pas restreinte à ce domaine et permet de décrire des composants avec de multiples possibilités en prenant en considération des comportements continus et discrets. On peut donc de modéliser l'évolution séquentielle d'un système tel que nous l'avons fait grâce à des modèles Delphi. De plus, l'introduction de la modélisation multi-domaines de VHDL-AMS est intéressante pour la prise en compte de phénomènes thermiques ou encore la modélisation de composants hydrauliques, pneumatiques ou encore électropneumatique. Le principal inconvénient lié à VHDL-AMS est sa complexité qui vient en opposition avec les principes de simplicité et de fonctionnement intuitif qui ont été mis en place dans le simulateur. Pour cela, nous envisageons de développer dans une phase ultérieure des outils graphiques permettant de produire un modèle VHDL-AMS utilisable par le simulateur.

5 Traduction des modèles Delphi vers VHDL-AMS.

Rappelons tout d'abord que la finalité de l'utilisation de VHDL-AMS comme langage de modélisation n'est pas d'étendre le potentiel du simulateur en exploitant toutes les possibilités du langage mais de disposer d'un moyen de formaliser la description des composants en se basant sur une norme internationale et reconnue. Nous utiliserons donc ce langage pour décrire les modèles de nos composants plutôt que l'emploi actuel du langage Delphi. La démarche présentée ici consiste à définir les principes de la traduction Delphi vers VHDL-AMS.

La finalité de cette traduction Delphi vers VHDL-AMS est d'une part de rendre plus accessible les modèles et d'autre part de valider la faisabilité de la traduction inverse VHDL-AMS vers Delphi. Disposant d'une bibliothèque de modèles VHDL-AMS et de leur équivalent Delphi, il sera plus facile de développer, dans une phase ultérieure, les outils réalisant la traduction inverse. Ainsi, les modèles VHDL-AMS définis par l'utilisateur seront rendus compréhensibles au simulateur.

Simultanément, nous définissons le sous-ensemble de la norme qui sera exploité par le simulateur.

Note : pour faciliter la lecture des différents modèles présentés (Delphi ou VHDL-AMS) nous représentons les mots clés du langage en gras et les commentaires en italique.

5.1 Démarche de traduction.

Dans le simulateur, on distingue deux grandes classes d'éléments électriques :

- Les composants qui sont les briques de base constituant tout circuit. Parmi ces composants, on va trouver :
  - Les primitives qui sont les composants les plus élémentaires du simulateur.
  - Les composants dérivés de ces primitives qui reprennent le comportement de leur ancêtre en leur ajoutant des propriétés particulières. C'est par exemple le cas de la source de tension carré qui dérive de la primitive « source de tension ».
- Les circuits qui sont réalisés à partir d'un assemblage de ces composants, auxquels on peut ajouter un comportement séquentiel.

5.2 Test et validation des modèles VHDL-AMS.

Afin de pouvoir vérifier que chaque modèle traduit est correct, il est nécessaire de pouvoir le tester. Etant donné que notre outil n'accepte pas encore le langage VHDL-AMS, nous devons disposer d'un autre simulateur pour valider ces modèles. Pour cela nous disposons d'une version d'évaluation du logiciel hAMSter (version 2.0) [HAM] fonctionnant sous Windows. Pour qu'un modèle puisse être validé, il est nécessaire que les primitives qu'il instancie soient définies et que des bancs de test (test bench) utilisent ces modèles dans des conditions analogues à celles que l'on peut rencontrer dans notre simulateur.
Notons bien que la validation concerne la traduction du modèle et non pas la modélisation elle-même. Le modèle traduit doit exprimer le même comportement que celui décrit initialement en Delphi. La validation du comportement est traitée dans le chapitre 4 (validation des modèles).

5.3 Modélisation Delphi.

Dans un modèle Delphi, la description du comportement peut prendre trois formes différentes.

Type de description	Composition	Méthode des objets	Concerne	Note
Implémentation du composant dans les matrices	Traduction des lois de Kirchhoff sous forme matricielle	Méthode SetMatrice	Uniquement les primitives	Comportement purement analogique
Structure d'un circuit	Assemblage de plusieurs composants ou circuits, défini par leurs interconnexions (netlist)	Constructeur des objets (méthode create)	Uniquement les circuits	
Comportement événementiel de l'élément	Décrit la réaction d'un élément à un nouvel événement, généralement changement d'état logique	Méthode OnNewEvent	Les circuits ou les composants dérivés des primitives mais pas les primitives	Utilisé uniquement par l'analyse événementielle

Tab. 16 Formes de la description du comportement

Seuls les deux derniers types de descriptions doivent être traduits. L'utilisateur n'aura pas la possibilité de modifier les primitives ni d'en ajouter de nouvelles. Il pourra par contre définir de nouveau circuits et de nouveau comportements séquentiels.

5.4 Organisation de la traduction.

Nous avons vu précédemment que pour notre simulateur, il était nécessaire de traduire les modèles de tous les éléments électriques sauf les primitives. Toutefois, pour valider ces modèles, il faut traduire ces primitives et écrire des bancs de test pour les valider avec le simulateur hAMSter. La traduction s'est faite en suivant ces étapes.

- Traduction de toutes les primitives.
- Traduction des modèles les plus complexes.
- Pour chaque modèle traduit, création d'un *testbench*, simulation et validation.

5.4.1 Traduction des primitives.

Avant de démarrer la traduction des modèles plus complexes, il est essentiel de pouvoir disposer des briques de base que constituent les primitives. Celles-ci, font définitivement partie intégrante du simulateur et leur traduction n'a d'autre but que la validation des modèles plus complexes dans un environnement extérieur à notre simulateur.
Les primitives ainsi traduites sont classées dans deux catégories, les dipôles et les quadripôles.

Les dipôles traduits sont :

- Dipôles passifs élémentaires idéaux :
  - Résistance.
  - Inductance.
  - Condensateur.

Dipôles actifs :

  - Source de tension sinusoïdale.
  - Source de courant sinusoïdale.
  - Source de tension rectangle.

Les quadripôles traduits sont :

- Sources contrôlées ;
  - STCT : Source de tension contrôlée en tension.
  - STCC : Source de tension contrôlée en courant.
  - SCCC : Source de courant contrôlée en courant.
  - SCCT : Source de courant contrôlée en tension.
- Transformateur idéal.

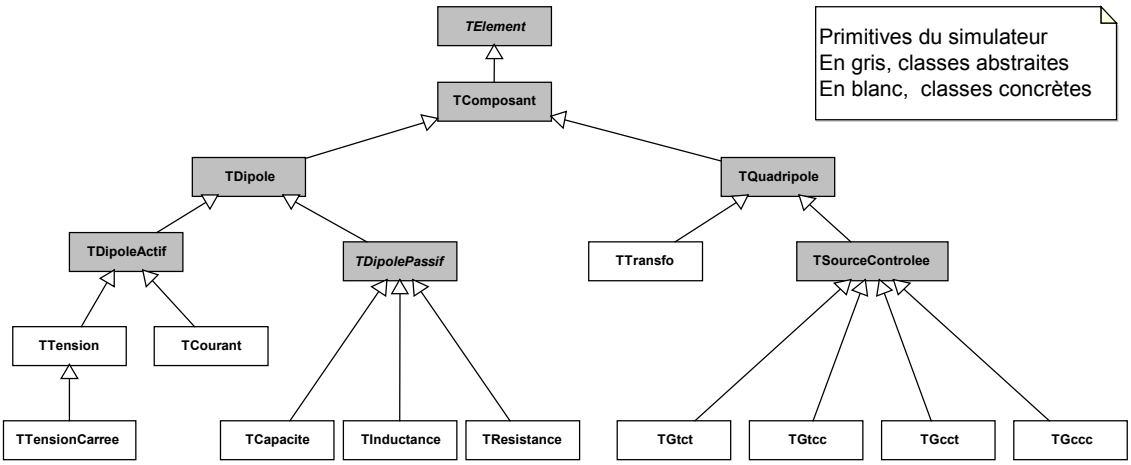


Fig. 47 Héritage des classes d'objets correspondant aux primitives du simulateur

Notons que ces composants restent implémentés dans le simulateur sous forme d'objets héritant de la classe Delphi TComposant (voir les diagrammes des classes ci dessus).
A priori, comme les modèles VHDL-AMS des primitives ne seront pas intégrées au simulateur, nous n'avons pas de contrainte particulière concernant leur écriture qui permettrait d'assurer leur traduction en Delphi. Nous avons alors obtenu des modèles qui pour certains sont assez proches de ceux que l'on trouve dans la littérature. Pour exemple, ci dessous un modèle de résistance idéale issu des exemples proposés avec le logiciel « Vamspice Designer » **[JEM03]**. Dans la déclaration d'entité, on trouve comme paramètre générique (GENERIC) la valeur de la résistance et dans les ports (PORT), 2 terminaux (TERMINAL) de nature électrique (Electrical) qui sont les bornes du composant. Enfin, en architecture, les relations tensions – courant au sein du composant sont décrites par sa loi d'Ohm.

```
-- VamSpiceDesigner Copyright (c) 2002 GET
-- Created by ENST / COMELEC Department -- All rights reserved
-- VHDL-AMS automatically generated from facet resistance{ic}
-- and new facet name is AMS_resistance_1036007246{ic}
-- Modif : S.JEMMALI
-- Date : Mon Nov 04 16:18:51 2002
ENTITY ams_resistance_1036007246 IS
GENERIC (r :real:= 1.00);
PORT (TERMINAL RIN, ROUT: Electrical);
END ams_resistance_1036007246;
ARCHITECTURE ams_resistance_1036007246_BODY OF ams_resistance_1036007246 IS
quantity vr across ir through RIN to ROUT;
BEGIN
ir == vr / r;
END ams_resistance_1036007246_BODY;
```

Fig. 48 Modèle VHDL-AMS d'un condensateur dans VamspiceDesigner

Ce modèle est relativement satisfaisant pour les cas les plus courants mais s'est révélé contraignant par la suite. En effet, en renseignant la valeur de la résistance on obtient un modèle relativement clair mais ce paramètre ne peut recevoir dans l'environnement hAMSter qu'une valeur constante. Or dans certains cas d'utilisation, comme le modèle du potentiomètre décrit plus loin, on peut vouloir affecter une valeur à ce paramètre qui soit le résultat d'un calcul stocké dans une QUANTITY. Ceci justifie le fait que nous ayons finalement retenu un modèle ou la valeur de la résistance n'apparaît plus en GENERIC mais comme une QUANTITY de PORT.

```
-------------------------------------------------------------
-- résistance idéale
--
-- p o----/\/\/\----o m
--              R
-------------------------------------------------------------------------
LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ENTITY fl_resistance IS
      PORT ( QUANTITY r :real:= 1.00;
             TERMINAL RIN, ROUT: ELECTRICAL);
END fl_resistance;
ARCHITECTURE resistance_BODY OF fl_resistance IS
      Quantity vr across ir through RIN to ROUT;
BEGIN
      ir == vr / r;
END resistance_BODY;
```

Fig. 49 Modèle de résistance avec sa valeur en QUANTITY de PORT

Ce principe peu habituel permet d'obtenir une plus grande souplesse dans le paramétrage des modèles et de leur imbrication, bien que cela les rende moins lisibles et plus complexes à instancier. Mais cette notion de souplesse est essentielle pour notre simulateur. C'est pourquoi nous l'avons ensuite généralisé à l'ensemble de nos modèles.

5.4.2 Ecriture des bancs de test.

Quel que soit le simulateur ou le langage employé, l'écriture d'un modèle doit être accompagnée de l'écriture d'un circuit de test (testbench) qui intègre le composant modélisé. Ce banc de test doit permettre à chacun de vérifier la validité du modèle dans des conditions de fonctionnement données. Dans notre cas, il s'agira de la description VHDL-AMS d'un circuit intégrant le modèle à tester et généralement une excitation et d'autres composants.
L'association Beams [BEA] dont un des buts est de promouvoir la modélisation comportementale, notamment via VHDL-AMS, propose une démarche pour la validation et la publication des modèles On en retrouve le détail en **Annexe 6**. Cette méthode demande une description assez exhaustive du modèle validé par différents testbench commentés. Cette rigueur est nécessaire dans une optique de diffusion et de validation de modèle. Dans notre cas, nous souhaitons vérifier la traductibilité des modèles et leur bon fonctionnement. Nous nous sommes donc limités à un testbench pour chaque modèle reproduisant des conditions d'essai équivalent à celle de Simul'Elec.
Ici encore, comme pour la modélisation des primitives, nous n'avons pas de contraintes concernant la modélisation des bancs de test. En effet, ils ne sont utilisés que dans l'outil hAMSter pour valider nos modèles. Toutefois, les circuits qu'ils décrivent doivent pouvoir être reproduits dans notre simulateur afin d'obtenir des résultats comparables et ainsi valider le modèle traduit. En considérant le fait que pour Simul'Elec, le régime établi par défaut est sinusoïdal, le testbench comportera généralement une excitation de ce type. Notons d'ailleurs que les résultats d'une simulation en régime établi sinusoïdal de Simul'Elec seront comparés à des résultats de l'analyse transitoire (TR) de l'outil hAMSter, car ce dernier ne dispose pas de mode d'analyse AC.

5.4.3 Modélisation d'éléments complexes.

Nous avons vu précédemment que les éléments concernés par la traduction sont :

- Les composants dérivés des primitives.
- Les circuits.

Dans la modélisation Delphi, les relations entre ces éléments et les primitives sont respectivement :

- Héritage pour les dérivés des primitives.
- Composition pour les circuits.

On peut traduire les liens de composition par la définition structurelle du modèle, via une netlist (liste d'éléments interconnectés). Par contre, on ne peut pas traduire la relation d'héritage car c'est une notion propre aux langages de programmation orientés objet que l'on ne trouve pas dans VHDL-AMS. Toutefois, on peut considérer qu'un élément dérivé d'une primitive est un circuit contenant un unique composant qui est cette primitive, auquel on va ajouter certaines propriétés. Pour prendre un exemple simple, on peut considérer le cas du modèle d'un fil électrique qui dérive d'une résistance lorsqu'il est modélisé en Delphi et est un circuit lorsqu'il est modélisé en VHDL-AMS.

```
-----------------------------------------------
-- Modèle de fil électrique en cuivre dont on prend
-- en compte la résistance
-----------------------------------------------
LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL ;
USE IEEE.MATH_REAL.ALL;
entity FL_Fil_elec is
PORT( quantity Longueur : real ;    -- longueur du fil en m
      quantity section : real ;     -- section du fil en m²
      terminal p,m : electrical) ;
end;
architecture FL_Fil_elec_body of FL_Fil_elec is
      quantity res : real := 10.0e-6 ;
      constant ro : real := 1.72 * 10.0e-8; -- resistivité du cuivre Ohm.m
begin
      if (longueur /= 0.0) and (section /= 0.0) use
            res == ro*Longueur/section ;
      else
            res == 10.0e-6 ; --évite les résistances nulles, les div par 0
      end use;
      Rfil : ENTITY Fl_resistance PORT MAP(res, p,m) ;
      -- instanciation d'une résistance idéale
end;
```

Fig. 50 Modèle VHDL-AMS d'un fil électrique

5.4.4 Composition d'un modèle Simul'Elec.

En suivant les principes définis précédemment, chaque modèle à décrire sera une *netlist* d'éléments électriques (composants ou circuits) à laquelle sont ajoutées certaines propriétés. Parmi ces propriétés, on distingue :

- Une liste de paramètres « externes » et leur correspondance avec les paramètres « internes » des éléments qui constituent le modèle.
- La description du comportement séquentiel.

Le tableau **Tab. 17** suivant reprend l'ensemble des différents points composant un modèle Delphi et leur traduction possible en VHDL-AMS.

Partie du modèle	Description Delphi	Description VHDL-AMS
Déclaration des paramètres externes.	Section interface : variables privées.	ENTITY : PORT QUANTITY ou GENERIC
Règles de correspondance entre les paramètres externes et ceux des éléments.	Méthode ChangeParamètres.	ARCHITECTURE, avant ou à l'instanciation des composants.
Définition de la netlist d'éléments.	Constructeur du circuit. Méthode create	ARCHITECTURE à l'instanciation des composants.
Définition des ports de connexion vers l'extérieur.	Constructeur du circuit. Méthode create	ENTITY PORT TERMINAL.
Définition des nœuds internes.	Constructeur du circuit. Méthode create	ARCHITECTURE TERMINAL
Description du comportement séquentiel.	Méthode OnNewEvent	Process par exemple.

Tab. 17 Composition d'un modèle Delphi

Il faut noter que pour un modèle Delphi, les ports de connections externes ou internes ne sont pas déclarés explicitement comme dans un modèle VHDL-AMS. C'est en réalisant la netlist qu'ils sont définis. Une borne nommée INTn (INT1, INT2…) est considérée comme une borne interne alors qu'une borne nommée EXTn (EXT1, EXT2…) est un nœud externe du circuit, traduit en VHDL-AMS par un port Terminal.
De plus, la définition du comportement séquentiel (en Delphi : méthode OnNewEvent) n'intervient que pour un événement. Il semble donc cohérent de la traduire par un « process » dont la clause « wait » sera l'arrivée d'un nouvel événement. Nous avons toutefois à notre disposition de nombreuses possibilités pour exprimer ces relations.

5.5 Correspondance entre VHDL-AMS et Delphi.

Ce tableau **Tab. 18** définit des règles de correspondance entre VHDL-AMS et les modèles Delphi de Simul'Elec. Il servira de base à l'élaboration du compilateur VHDL-AMS → Simul'Elec. Cette liste ne reprend pas les 108 mots clés de VHDL-AMS et ne traite pas les 285 règles de la grammaire. Nous avons utilisé pour ce début de traduction la grammaire interactive proposé en ligne [HER].
On retrouve ici les termes, instructions, types standards ou déclarations qui pourront être traduits pour la génération de modèle Delphi. Cela constitue une première étape dans la définition du sous-ensemble de la norme VHDL-AMS que Simul'Elec pourra traiter. Nous disposons avec VHDL-AMS et Delphi de deux langages fortement typés. Ce typage fort est bien plus important en VHDL-AMS notamment grâce à la présence des 6 classes d'objets qui permettent le transport de l'information (CONSTANT, VARIABLE, SIGNAL, TERMINAL,QUANTITY, FILE). Il serait possible de définir ces classes d'objet en Delphi pour que les modèles VHDL-AMS traduits en Delphi restent aussi rigoureux. Rappelons toutefois qu'à partir du moment ou l'on dispose de la possibilité de décrire un modèle en VHDL-AMS, le modèle Delphi n'est plus qu'un intermédiaire de compilation compréhensible par le simulateur. Ainsi l'utilisateur n'écrit pas ses modèles en Delphi, la rigueur de la sémantique des modèles Delphi n'est alors pas essentielle car ils seront toujours générés à partir de modèles VHDL-AMS plus stricts dans leur expression.
On peut voir également que certaines instructions n'ont pas trouvé d'équivalence car pour cela le simulateur à besoin d'être modifié pour les supporter. Il conviendra par la suite de justifier de la nécessité de cette extension du simulateur. Rappelons que l'objectif premier de l'utilisation de VHDL-AMS est l'emploi d'un sous-ensemble de ce langage pour décrire les modèles existant en Delphi et les modèles futurs en suivant une trame commune.

VHDL AMS			**DELPHI**
Type			Type
Subtype			Type
Signal			Variable
	Bit		Boolean
	BitVector		Array of boolean
Quantity			Variable
	Libre		
		Real	Double
		Integer	Integer
		Time	À définir
	Liée		
		Across	Pas d'équivalence
		Throught	Pas d'équivalence
	De port		
		IN	Variables traitées comme quantités libres

	OUT	Variables traitées comme quantités libres
Constant		
	Real	Constante non typée
	Integer	Constante non typée
	Time	Constante non typée
	String	Constante non typée
Terminal		
	Port déclaré en ENTITY	Nœud de connexion externe nom EXTn (EXT1, EXT2)
	Déclaré en ARCHITECTURE	Nœud de connexion interne nom INTn (INT1, INT2)
Attribut		
	Quantité‘Integ	Pas d’équivalence
	Quantité ‘Dot	Pas d’équivalence
	Quantité ‘Above(tension)	Function above(Quantité, tension)
Function( )		Function( )
Procedure( )		Procedure( )
Variable		
	De procedure	Variable
	De procedural	Variable
	De process	Variable
If…use…else…end use ;		If…then…else;
If…then…else…end if;		If…then…else;
Case…is…When…When…Other…End case;		Case…Of…else…end;
For…generate		For…to…do
For…Loop		For…to…do
Assert		Fonction assert à implémenter
Process		
ENTITY		
	Déclaration du modèle	Interface
	En architecture	Instanciation du composant
Array(…to…) of…		Array[…,…] of…
Range		Set of
Record		Record
Access		Pointer
New		Déclaration de pointeur
Break		?
Alias		?
Attribute		?
Noise (bruit sur une quantité)		Pas d’équivalence
Tolérance (précision des quantités)		Sur réel : type simple, double, extended
‘High		Function High( )
‘Low		Function Low( )
‘succ(i), ‘prec(i)		?
‘Leftof(i), ‘Rightof(i)		?
’Delayed		?
‘Reference		?
Surcharge d’opérateur ex : function « + »		Impossible en Delphi : définition de function ADD.
Package		?
Procedural		Procedure
Procedure		Procedure
Wait		?
= =		Pas d’équivalence
:=		:=
<=		:=
/=		<>
UNITS		Type

Tab. 18 Correspondance sémantique Delphi et VHDL-AMS

5.6 Modèles traduits.

La démarche de traduction s'est déroulée en deux phases. Tout d'abord, une traduction « brute » du comportement de l'élément à partir de la connaissance du modèle. Pour cette étape, on ne se préoccupe pas de l'implémentation du modèle en Delphi. Cette première étape permet d'une part d'acquérir les bases du langage VHDL-AMS et d'autre part de vérifier la faisabilité de la traduction. Ainsi, nous pouvons écarter des solutions ne fonctionnant pas ou peu satisfaisantes et en dégager d'autres correspondant mieux à nos besoins. Disposant de modèles ainsi traduits, on peut les modifier pour qu'ils respectent la trame de modélisation définie dans le paragraphe précédent et rendre ainsi la traduction VHDL-AMS → Delphi plus facile. Notons que pour faire fonctionner l'ensemble des modèles traduits sous hAMSter, il est nécessaire d'inclure certaines bibliothèques (voir **Fig. 51**).

```
LIBRARY DISCIPLINES;  -- définit les domaines technologiques
LIBRARY IEEE;                          -- définit les constantes, fonctions
standard de la norme
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.
```

Fig. 51 Code d'appel des bibliothèques à placer en début de description sous hAMSter.

5.6.1 Potentiomètre.

Ce premier modèle relativement trivial permet de faire facilement le parallèle entre le modèle Delphi et sa traduction VHDL-AMS. Ce potentiomètre est modélisé par deux résistances en série. On ne connaît pas la valeur de chaque résistance, mais la résistance totale du composant, ainsi que la position du curseur en %. On notera que ce modèle ne comporte pas de description séquentielle de son comportement mais uniquement une description structurelle. Le modèle Delphi donné ici à été modifié pour rendre la parallèle plus évident. Ainsi ce modèle Delphi peut être aussi bien le modèle d'origine que le résultat de la traduction VHDL-AMS → Delphi. Il faut noter que la présence de la fonction « éditParametres » n'est pas utile à la modélisation mais est nécessaire à l'interface graphique du simulateur pour voir et modifier les paramètres.

Pour valider l'écriture du modèle, nous utilisons un banc de test relativement simple qui applique une tension sinusoïdale au potentiomètre. On observe la tension de la source et celle présente au curseur, et sa variation en fonction des valeurs des paramètres.

```
ENTITY TestbenchPot IS
END TestbenchPot;
ARCHITECTURE TestbenchPot_Body OF
TestbenchPot IS
TERMINAL n1,n2  ELECTRICAL;
BEGIN
Mon_Pot : ENTITY
FL_POTENTIOMETRE_RtCur  (Pot_BODY_RR)
PORT MAP (1.0, 20.0, n1, n2,
electrical_ground) ;
Source  : ENTITY fl_sineSource
PORT MAP ( n1, electrical_ground) ;
END TestbenchPot_Body;
```

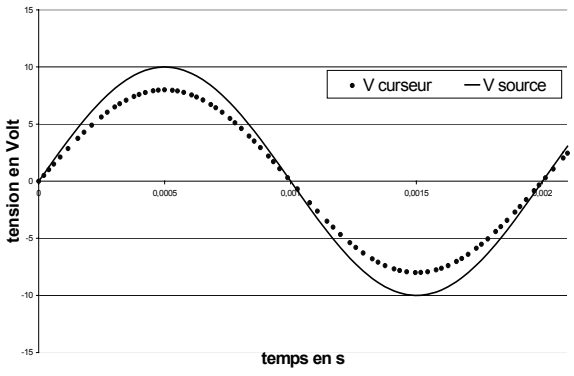


Fig. 52 Testbench du potentiomètre

Fig. 53 Courbes relevées avec hAMSter

```
                unit s_potentiometre;
interface
TPotentiometre= class(TCircuit) // Modéle d'un potentiometre .
private
        Rtotal, PosCur, // déclaration des paramètres à stocker
        ValR1, ValR2 :extended; // des variables internes
        Pot_R1: TResistance; // et des éléments à instancier
        Pot_R2: TResistance;
public
        constructor create(name,ref,familly: string; Resist, PosCurseur : extended);
        procedure editParametres; override;
        procedure changeParametres(index : integer ; niouVal: extended); override;
end;

implementation

constructor Tpotentiometre.create( name,ref,□amily: string; Resist, PosCurseur : extended);
begin
        inherited create(name,ref,familly, 3 ) ; // déclarations interne au simulateur
        Rtotal := Resist ; // Stockage de la valeur de la résistance
        PosCur := PosCurseur ; // position du curseur
        // instanciation des éléments (valeur temporaire)
        Pot_R1:= TResistance.create('R1','Potentiometre',famille,resist);
        Pot_R2:= TResistance.create('R2','Potentiometre',famille,resist);
        // connexions des éléments
        ajouTDipole(Pot_R1, 'EXT1', 'EXT2') ;
        ajouTDipole(Pot_R2, 'EXT2', 'EXT3') ;
        // initialisation des paramètres (pour interface graphique
        editParametres ;
        // calcul des valeurs des résistances
        changeParametres(0, resist);
end; // Tpotentiometre.createPotentiometre

procedure Tpotentiometre.editParametres;
begin // on définit le nom, l'unité et la valeur des paramètres pour l'interface graphique
        nbrParam:= 2;
        SetLength(TabParam,nbrParam);
        TabParam[0].nom := 'Résistance' ;
        TabParam[0].unite := 'Ohm';
        TabParam[0].valeur := Res ;
        TabParam[1].nom := 'Position Curseur' ;
        TabParam[1].unite := '%';
        TabParam[1].valeur := Curseur ;
end; // Tpotentiometre.editParametres

procedure Tpotentiometre.changeParametres(index : integer; niouVal: extended);
begin
        inherited ; // permet d'ajuster les valeurs des éléments quand un
        case index of // des paramètres est modifié
                0 : Res := niouVal ;
                1 : curseur := niouVal ;
        end ; // Calcul des valeurs des Resistances
        ValR1 := Rtotal * PosCur / 100 ;
        ValR2 := Rtotal * (1 – PosCur / 100 ) ;
        Pot_r1.valeur := ValR1 ; // Affectation des valeurs aux résistances
        Pot_r2.valeur := ValR2 ;
end; // Tpotentiometre.changeParametres

end.
```

Fig. 54 Modèle Delphi du potentiomètre

```
ENTITY FL_POTENTIOMETRE_RtCur IS
PORT ( QUANTITY Rtotal : real := 1.0 ;                    -- résistance totale
       QUANTITY PosCur : Real := 0.0 ;        -- position du curseur en %
       TERMINAL Borne1, Curseur, Borne2 : ELECTRICAL); -- les 3 bornes du pot.
-- Note : les paramètres des composants sont des port (quantity) et pas des generic
-- pour pouvoir leur transmettre une quantity : exemple ici du potentiomètre dont les
-- résistances sont calculée à partir de la résistance totale et de la position du curseur
END FL_POTENTIOMETRE_RtCur;

ARCHITECTURE Pot_BODY_RR OF FL_POTENTIOMETRE_RtCur IS
-- potentiometre constitué de deux resistances en serie
       quantity ValR1 , ValR2 : real ;
BEGIN
       -- Calcul des valeurs des résistances composant le potentiomètre
       ValR1 == Rtotal * PosCur / 100.0 ;
       ValR2 == Rtotal * (1.0 - (PosCur / 100.0)) ;
       -- Instanciation des composants
       Pot_R1 : ENTITY fl_resistance PORT MAP (ValR1, Borne1, Curseur) ;
       Pot_R2 : ENTITY fl_resistance PORT MAP (ValR2, Curseur, Borne2) ;
END Pot_BODY_RR;
```

Fig. 55 Modèle VHDL-AMS du potentiomètre

5.6.2 Autres modèles structurels simples.

De la même manière, nous avons obtenu d'autres modèles structurels relativement simples. Nous n'avons pas jugé nécessaire de représenter pour chacun le modèle d'origine en Delphi. La méthode de traduction étant semblable à celle utilisée pour le potentiomètre, on se reportera en **Annexe 7** où ces modèles sont détaillés. Ces modèles sont :

- Transformateur Réel.
- Source de tension alternative triphasée.
- Transformateur Réel.

5.6.3 Modèles de relais et contacts.

Dans Simul'Elec, nous avons modélisé séparément les composantes maîtres et esclaves des relais. Ainsi, on dispose d'un modèle de bobine (maître) et de modèles de contacts correspondants aux différents types (NO, NF ou temporisés). L'état logique de la bobine se propage aux contacts et fixe leur état. Pour la simulation de circuits électrotechnique ; il est essentiel, comme on l'a développé en **3.4**, de préserver cette séparation des différentes composantes du relais.

5.6.3.1 Banc de test des modèles.

La relation maître-esclave que définit les références croisées est traduite en VHDL-AMS par le signal « State » qui représente l'état logique. C'est une entrée pour les contacts et une sortie pour le modèle de bobine. On retrouve la propagation de l'état dans le test bench par la transmission du signal booléen représentant l'état.

```
ENTITY network IS
END

ARCHITECTURE behav OF network IS
   TERMINAL n1,n2,n3,N4: ELECTRICAL;
            SIGNAL km1state : bit := '0' ;
            SIGNAL Km1Tp    : time := 0 ms ;
            SIGNAL newstate : bit := '0';
BEGIN
    Sour : ENTITY fl_VoltSineSource
       (Veff_Freq_Deg)
            PORT MAP ( 20.0, 50.0, 0.0, N1, Electrical_ground) ;
    KM1Bob  : ENTITY fl_relais
            PORT MAP ( 0.1, 10.0 ,8.0, 1 ms, 0.5 ms

,km1state,Km1Tp, N1, Electrical_ground);
    KM1Cont : ENTITY fl_Contact(Type_NF)
                        PORT MAP ( 0.001, 5 ms, 3 ms,

km1state,Km1Tp,newstate,N1,N2);
    RC : ENTITY fl_resistance
             PORT MAP (1000.0, N2, Electrical_ground);

END;
```

Fig. 56 TestBench du relais et des contacts

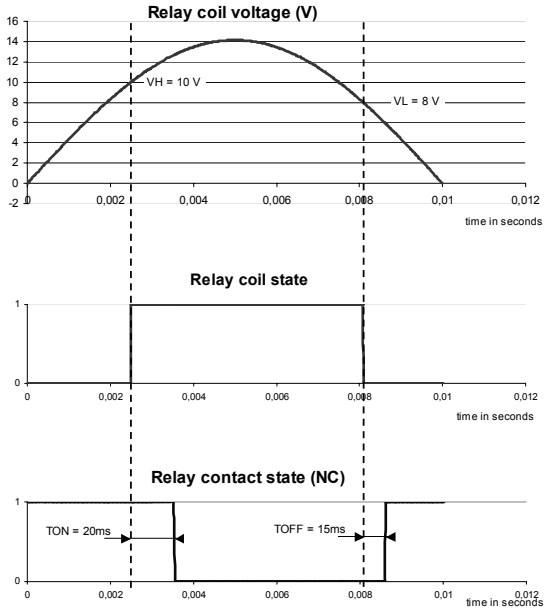


Fig. 57 Résultat de simulation avec hAMSter

Le principe de propagation de l'état logique tel qu'il est défini dans les modèles de la bobine et des contacts nous amène à définir deux signaux, Km1State et Km1Tp correspondant respectivement à l'état logique du relais et au temps de propagation de l'état entre la bobine et les contacts. On applique une demi-sinusoïde à notre relais qui fonctionne en continu afin d'observer les changements d'état.

5.6.3.2 Modèle de la bobine de relais

Comme pour les modèles précédents, les paramètres du modèle (**Fig. 58**) apparaissent sous forme de quantité de port et non pas de paramètres génériques.

Il possède deux signaux de sortie :

- State : état logique du relais. Il est conditionné par le passage du seuil haut ou bas de tension.
- TrTime : temps de propagation de l'état. Il indique le retard entre le changement d'état de la bobine du relais et celui de ses contacts.

On notera que ce modèle est valable uniquement si la tension est continue.
L'état logique est synchronisé avec l'évolution de la tension. Ce sont uniquement les contacts qui subissent le temps de propagation. Le modèle Delphi, par contre, impose ce retard à la bobine auxquels sont synchronisés les contacts.

```
ENTITY FL_relais is
        PORT(  QUANTITY L    : real := 1.0 ;        -- impédance de la bobine
               QUANTITY VH   : real := 1.0 ;        -- tension d'activation
               QUANTITY VL   : real := 0.0 ;        -- tension de désactivation
               SIGNAL TON    : time := 20 ms;       -- retard à l'activation
               SIGNAL TOFF   : time := 15 ms;       -- retard au repos
               SIGNAL  State   : out bit  := '0' ;  -- Etat du relais
               SIGNAL  trTime  : out time := 0 ms ;  -- retard à la transition
               TERMINAL A1,A2 : electrical );       -- bornes de la bobine
END;
```

```
ARCHITECTURE FL_relais_body OF fl_relais IS
        QUANTITY Vb ACROSS Ib THROUGH A1 TO A2 ;
BEGIN
   -- Comportement inductif du relais
   Vb == L * Ib'dot ;
   -- Détermination de l'état logique
   BREAK State => '1'    WHEN Vb'above(vh);
   BREAK State => '0'    WHEN NOT Vb'above(vl);
   -- temps de propagation de l'état logique
   BREAK TrTime => Ton   WHEN Vb'above(vh);
   BREAK TrTime => Toff  WHEN nOt Vb'above(vl);
END;
```

Fig. 58 Modèle VHDL-AMS de la bobine de relais

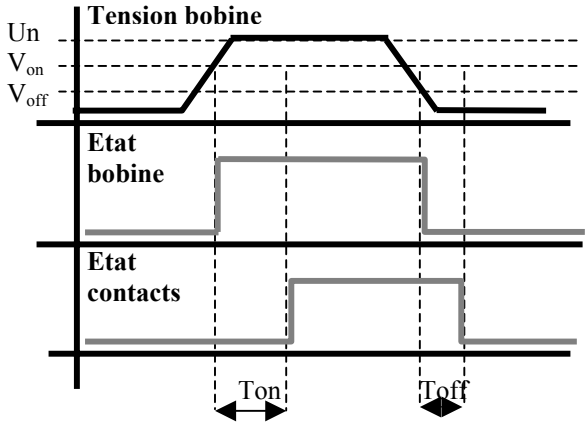


Fig. 59 Changement d'état du relais avec le modèle VHDL-AMS

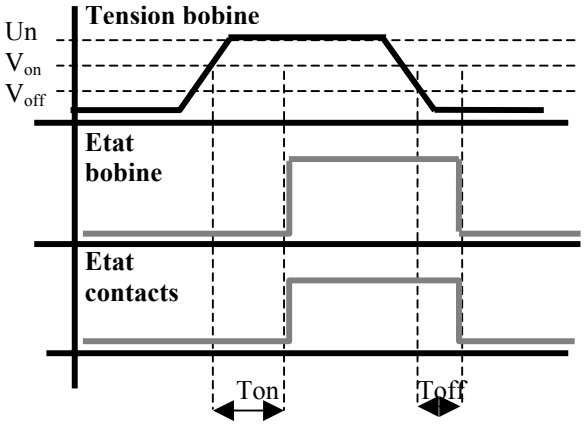


Fig. 60 Changement d'état du relais avec le modèle Delphi

Cette différence entre les deux modèles n'est pas réellement problématique étant donné que la notion d'état de la bobine n'as pas d'impact sur le comportement électrique. C'est uniquement l'état des contacts que l'on peut observer et qui aura des répercussions sur le circuit simulé.

5.6.3.3 Modèle des contacts.

Les contacts modélisés dans Simul'Elec peuvent être :

- Instantanés ou temporisés (à l'activation et/ou au repos).
- A fermeture (NO) ou à ouverture (NF).

Nous avons fait le choix de développer un modèle VHDL-AMS (**Fig. 61**) permettant de couvrir les différents cas possibles. En fait, on a toujours affaire à un contact temporisé à l'activation et au repos. Afin d'avoir un contact instantané, il suffit de fixer la valeur de Ton et Toff à zéro. Pour déterminer si on utilise un contact NO ou NF, on dispose de deux architectures « Type_NO » et « Type_NF ». On instancie donc le composant en choisissant l'architecture adaptée. On remarque que les deux architectures sont identiques dans la détermination de l'état logique et diffèrent seulement dans l'expression de l'effet électrique de cet état.

Une autre solution permettant d'alléger ce modèle consisterait à ajouter une quantité de port précisant si le contact est à ouverture ou à fermeture. Mais dans ce cas, on ne peut pas avoir le même état initial pour la bobine et le contact en raison du temps de propagation.

```
ENTITY FL_contact is
    PORT( QUANTITY R   : real := 1.0 ;       -- résistance du contact
        SIGNAL   TON  , TOFF : time ;      -- tempo à la fermeture
                                -- et à l'ouverture
                SIGNAL   masterstate    : IN bit ; -- état du composant maître
                SIGNAL   tp             : IN time; -- temps de propagation
                SIGNAL   nioustate   : inout bit ; -- état du contact
                TERMINAL n1,n2 : electrical );     -- bornes du contact
END;

ARCHITECTURE Type_NO OF FL_contact IS
-- architecture d'un contact à fermerture ( NO normalement ouvert)
   QUANTITY Vc ACROSS Ic THROUGH n1 TO n2 ;
   SIGNAL clk   :BIT :='0' ;       -- horloge de contrôle de l'état
   SIGNAL delai : time := 0 ms;    -- retard au changement d'état
BEGIN
        clk<=NOT clk AFTER 100 us ;
        PROCESS (clk)
        BEGIN
 --le delai cumule le temp de propagation et la temporisation du contact
                IF masterState = '1' THEN
                        delai <= tp + ton;
                ELSE
                        delai <= tp + toff;
                END IF;
                -- modification de l'etat après le delai
        nioustate <= TRANSPORT masterstate AFTER delai ;
        END PROCESS;
        -- consequence électrique de changement d'état
        IF nioustate = '1' USE
                Vc == R * IC ;
        ELSE
     Ic == 0.0 ;
        END USE ;
END Type_NO;

ARCHITECTURE Type_NF OF FL_contact IS
-- architecture d'un contact à fermerture ( NO normalement ouvert)
   QUANTITY Vc across Ic through n1 to n2 ;
   SIGNAL clk   : bit  :='0' ;
   SIGNAL delai : TIME := 0 MS;
BEGIN
        clk<=not clk after 100 us ;
        PROCESS (clk)
        BEGIN
                if masterState = '1' THEN
                        delai <= tp + ton;
                ELSE
                        delai <= tp + toff;
                END IF;
        nioustate <= TRANSPORT NOT masterstate AFTER tp ;
        END PROCESS;
        IF nioustate = '1' use
                Vc == R * IC ;
        ELSE
     Ic == 0.0 ;
        END USE ;
END TYPE_NF;
```

Fig. 61 Modèle VHDL-AMS des contacts de relais

5.7 Bilan de la traduction.

Cette étape de traduction de Delphi vers VHDL-AMS nous permet d'établir les bases de la traduction inverse de VHDL-AMS vers Delphi qui sera réalisée ultérieurement. Nous avons pu vérifier la faisabilité de la traduction des modèles et d'un certain nombre d'instructions du langage. Une des difficultés de cette traduction est qu'on doit la valider avec des outils comme hAMSter [HAM] qui ne traitent pas la totalité de la norme, limitant ainsi le champ des possibilités des modèles traduit. Par contre dans le sens inverse les modèles que l'on écrira en VHDL-AMS ne seront limités que par le respect de la norme et les possibilités de Simul'Elec. Nous avons constaté que l'on obtient des modèles plus clairs, plus lisible et plus facilement compréhensibles. Par contre, le fait d'utiliser un langage normalisé rend la traduction du comportement des composants plus complexe, notamment pour les aspects événementiels. En effet la description du comportement doit être plus générale que celle faite en Delphi qui est plus proche du simulateur.

Chapitre 4

Validation.

Ce chapitre traite de la validation des modèles et des modes d'analyses développés. Après avoir conçu et réalisé le simulateur de circuits électrotechniques Simul'Elec et sa bibliothèque de modèle, il est indispensable d'estimer la précision des résultats qu'il donne et de définir les limites de sa validité.

1 Objectifs de la validation.

La validation est essentiellement ciblée sur les parties innovantes du simulateur (modèles de composants et modes d'analyse). Son objectif est de déterminer si le simulateur répond aux critères d'exigence des utilisateurs. Cette précision est mesurée de deux manières :

- Par comparaison avec des résultats connus et validés (issus d'autres simulateurs par exemple) ;
- Par confrontation à des mesures expérimentales sur une installation réelle au travers de bancs d'essais.

Nous devrons vérifier la justesse du modèle mais également contrôler le fonctionnement global d'une installation. Afin de tester différents types d'installations en modifiant leurs compositions et d'essayer différentes configurations et situations de défaut, un banc d'essai évolutif représentant la maquette d'une installation électrique de distribution et de protection a été utilisée.

N'ayant pas la possibilité technique et financière de valider tous les composants du simulateur nous avons centré nos efforts sur les parties les plus intéressantes, qui posent le plus d'interrogation ou dont le fonctionnement est critique. Dans ce but, nous nous sommes tout particulièrement intéressés au comportement des disjoncteurs étant donné leur rôle crucial dans la protection des circuits et des personnes.

1.1 Déroulement de la validation.

Pour mener à bien cette validation, un projet tripartite à été mené en parallèle du déroulement de la thèse. Ce projet est soutenu par le pôle Technologique EITICA (Electronique, Informatique et TIC en Aquitaine) [EIT]. Il réuni Algo'Tech Informatique, le laboratoire IXL et le laboratoire LIPSI. Ce projet financé par la région Aquitaine via un fond Feder a permis de mettre en place une équipe de travail constitué de l'auteur de cette thèse, de deux élèves ingénieurs et d'une partie du personnel de chacun des trois partenaires. Nous avons également

pu réaliser un banc d'essai évolutif, nous permettant de faire des mesures adaptées à nos besoins et aux caractéristiques à valider. La réalisation du banc à été confié à la section MAI (Mécanique et Automatisme Industriels) du lycée Louis de Foix de Bayonne qui nous a fait également bénéficier de ses locaux, de ses formations (habilitation électrique) et de son matériel d'essai comme le banc de sélectivité en Basse Tension.

Les essais, les comparatifs et de l'analyse des résultats sont en majeure partie le fruit du travail des deux élèves ingénieurs :

- Henri Abitbol [ABI03] élève ingénieur ESTIA 2ème année.
- Nicolas Depail [DEP1] élève ingénieur ENSEIRB 2ème année.

Ce chapitre est le résultat de leur analyse des résultats des différents essais comparatifs.

1.2 Validation du comportement des disjoncteurs.

Dans l'optique des tests sur banc d'essais, nous traitons en particulier la validation des modèles des équipements de protection et de distribution des installations électriques (fusibles, disjoncteurs, …). Nous devons vérifier la justesse du modèle mais également contrôler le fonctionnement global d'une installation. Ce dernier doit respecter les règles de sélectivité de ses protections. Ici, la précision exigée de chaque modèle n'est pas primordiale quand on considère individuellement les composants. La fiabilité des résultats doit toutefois être suffisante pour que malgré le cumul des erreurs, le simulateur puisse reproduire le comportement de toute l'installation.
La validation porte sur plusieurs aspects :

- Fonctionnel : le modèle du composant doit être performant dans un fonctionnement simple (respect du temps de déclenchement).
- Dimensionnement : le composant doit pouvoir être bien dimensionné dans un certain mode de fonctionnement du circuit dans lequel il est utilisé (protection d'un circuit).
- Respect de la sélectivité : On doit pouvoir vérifier que le dimensionnement du composant ne perturbe pas d'autres appareils de protection en amont ou en aval.

Pour effectuer ces vérifications nous n'avons recourt qu'à l'expérimentation. En effet, nous n'avons pas trouvé de simulateur permettant de modéliser des disjoncteurs.
Nous disposons de deux équipements distincts :

- Le banc de sélectivité en basse tension : ce banc didactique permet de tester de multiples configurations mettant en jeu différentes notions de sélectivité. Il nous donnera une vue qualitative des résultats attendus.
- Le banc d'essai évolutif : ce banc d'essai de notre conception peut être adapté à plusieurs types de manipulation dont le but est de déterminer plus finement le comportement des disjoncteurs. On peut réaliser de multiples cas de figure comme des essais en surcharge, en court-circuit, en courant continu, alternatif, à courant constant ou des échelons de courant.

1.3 Validation des modes d'analyse.

Simul'Elec utilise des modes d'analyse différents de ceux des autres simulateurs connus. Il nous permet avec moins de calculs d'obtenir des résultats proches des résultats théoriques. L'ergonomie du logiciel a été adaptée aux besoins de nos utilisateurs pour privilégier la simplicité et la clarté de l'utilisation.
Nous avons réalisé des essais comparatifs entre nos modes d'analyses et ceux d'autres logiciels. Nous souhaitons tout d'abord vérifier la pertinence de nos résultats et ensuite estimer les performances de notre outil par rapport à d'autres logiciels du marché.

2 Comparaison logicielle.

Afin de valider le fonctionnement de Simul'Elec, nous devons comparer ses résultats à des références fiables. Deux solutions complémentaires s'offrent à nous, la comparaison avec des mesures sur le matériel et la comparaison avec d'autres simulateurs. Nous traitons ici cette seconde solution, moins coûteuse et plus facile à mettre en œuvre qui doit nous permettre de valider les résultats de nos modes d'analyse et une partie des modèles.
Certains de nos modèles n'ont pas d'équivalent dans les simulateurs de référence comme les disjoncteurs. D'autres tels que les relais ne sont pas décrits d'une manière adaptée au domaine de l'électrotechnique. Nous complèterons donc ces comparaisons par des mesures sur des installations réelles qui sont traitées dans les paragraphes **3** et **4**.

2.1 Choix des logiciels de référence.

De nombreux logiciels de simulation de circuit électrique sont disponibles gratuitement dans des versions libres ou des versions d'évaluation limitées. Nous présentons un rapide comparatif de ces logiciels en chapitre 1.

Après cette série de tests, trois logiciels ressortent plus particulièrement. Il s'agit de **PSpice**, très répandu et probablement le plus utilisé que ce soit dans l'industrie ou dans le monde éducatif. Celui-ci a fait ses preuves aussi bien dans son efficacité que dans la pertinence des résultats offerts. Il est disponible intégré à la suite Orcad 9 [ORC] dans une version d'essai qui limite entre autre la taille des circuits simulables.

Le deuxième logiciel sélectionné pour la comparaison avec Simul'Elec est **Simplorer** [SIM] qui offre une solution alternative aux nombreux logiciels construits autour du noyau Spice. De plus, il est utilisé en électronique de puissance, domaine qui se rapproche de celui ciblé par Simul'Elec.

Enfin, il reste le cas de Schemaplic [FIT] qui est le seul logiciel permettant de tester facilement des disjoncteurs et des circuits à relais. Bien qu'étant un bon outil pédagogique, son utilisation est très limitée. Il n'apporte aucune information comparable aux résultats obtenus grâce à un logiciel de simulation analogique comme Simul'Elec ou Pspice. Il affiche uniquement des notions d'états logiques et de continuité électrique sans calculer de valeurs de tension ou de courant. L'apport réduit que présente ses résultats pour nous, ne justifie pas son utilité pour la validation.

Etant donné que nous utilisons des versions d'utilisation limitées, nous devons simuler des schémas de dimensions réduites afin qu'ils puissent être traités par ces outils de référence.

Pour l'étude des circuits dans le domaine temporel, nous comparerons les résultats des modes d'analyses en régime permanent de Simul'Elec avec les modes d'analyse transitoire de SPICE ou de Simplorer. Concernant l'étude harmonique, on fera la comparaison entre le mode d'analyse fréquentiel de Simul'Elec avec le mode AC de SPICE ou de Simplorer.

2.2 Schémas de tests.

Nous avons sélectionné trois schémas simples mais représentatifs du type de circuits que l'on peut avoir besoin de simuler. La comparaison traitera uniquement des résultats de simulations analogiques (tensions et courants) qui sont les seuls comparables avec Simul'Elec.

Les comparaisons des simulateurs pour les circuits testés ne se feront pas en point par point mais plutôt en fonction des grandeurs caractéristiques, méthode plus révélatrice. Pour les analyses fréquentielles, seules les courbes de gains seront analysées, même si les courbes de phase seront utilisées par exemple pour aider à déterminer la fréquence de coupure.

2.2.1 Circuit RLC série.

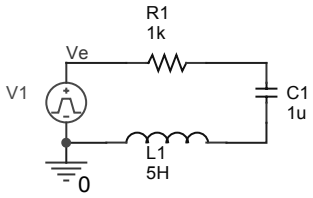


Fig. 1 Circuit RLC série

Ce circuit classique permet de faire apparaître des oscillations transitoires en régime temporel. Il sera donc possible de comparer les résultats obtenus lors des simulations au niveau du temps de montée, du dépassement, de la pseudo période, …
Il est également possible d'observer la fréquence de résonance et l'amortissement de ce circuit en analyse fréquentielle.
Les comparaisons s'effectuent sur les formes des courbes en régimes temporel et fréquentiel ainsi que sur les valeurs caractérisant ce genre de montages.

La source est un générateur de tension carré d'amplitude 100V et de fréquence 7Hz. Ici nous ne considérerons que le premier front qui apparaît au bout de 10ms. La résistance vaut 1kΩ, le condensateur a une capacité de 1µF et la bobine une inductance de 5H.

2.2.1.1 Etude théorique.

Ce circuit RLC des plus classiques peut être décrit par l'équation différentielle [DEP91] :

$$Ve(t)=R\cdot i(t)+L.\frac{di}{dt}+\frac{1}{C}\int_0^t i\,dt$$

Nous ne détaillerons pas les étapes de calcul qui nous amènent à la solution générale qui nous donne la tension aux bornes du condensateur :

$$Uc = Ve_{(t=0^+)} + \frac{A}{C}\cdot e^{-\alpha\cdot t}\left[\cos(\omega t+\varphi)\right]$$

En appliquant ce résultat aux valeurs de nos composants, nous obtenons :

$$\boxed{Uc=100-102{,}6.e^{-100\cdot t}\left[\cos(436.t-0.225)\right]}$$

Dans ce cas nous avons affaire à un circuit faiblement amorti. La tension aux bornes du condensateur a une allure de sinusoïde amortie. Ses valeurs caractéristiques sont :

Pulsation propre : 436 rad/s Pseudo période : 14,14 ms

Fréquence de résonance : 69.4 kHz
Fréquence de coupure : 71,18 kHz

2.2.1.2 Manipulations dans Simul'Elec.

Dans Simul'Elec, nous utilisons la source de tension carrée obtenue par décomposition en série de Fourier. Nous pouvons jouer sur la finesse du résultat en modifiant le nombre d'harmoniques de la source. Ici, nous voulons surtout mesurer la différence entre nos résultats et ceux des autres simulateurs. Nous avons choisi d'utiliser 1000 harmoniques qui nous donnent des courbes presque parfaitement lissées.
En effet, pour avoir des carrés propres, il faut définir un nombre d'harmoniques élevé. Ainsi, pour 620 points, au lieu de prendre 1,272s, le calcul prend 1,4s avec 100 harmoniques (déformation importante des créneaux) et 15,6s pour 1000 harmoniques (faible déformation des créneaux). Le graphique ci-dessous donne une idée de l'évolution du temps de calcul en fonction du nombre d'harmoniques.

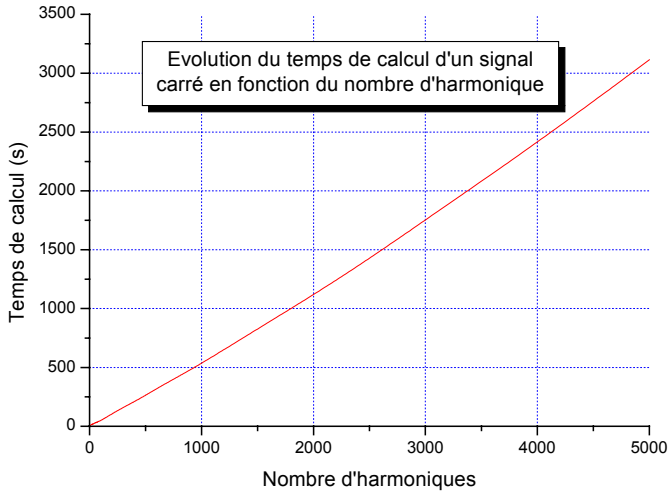


Fig. 2 Evolution du temps de calcul en fonction du nombre d'harmonique sous Simul'Elec

Pour l'étude fréquentielle c'est le simulateur qui se charge de remplacer automatiquement la source carrée par une source sinusoïdale pour le balayage en fréquence.

2.2.1.3 Analyse temporelle.

Pour les trois simulateurs, Simul'Elec, Pspice et Simplorer, nous obtenons des résultats quasiment identiques. Nous pouvons constater que les différentes valeurs sont très peu dispersées notamment entre les différents simulateurs. C'est encore plus flagrant lorsque l'on observe en même temps tous les signaux en **Fig. 3**. Nous notons d'ailleurs que tous les courbes issues des simulateurs sont quasiment confondues contrairement à la courbe théorique. Cela peut s'expliquer par le fait que la méthode de résolution n'est pas la même dans les deux cas, les approximations ne sont font pas de la même manière selon si on utilise une résolution analytique ou numérique.

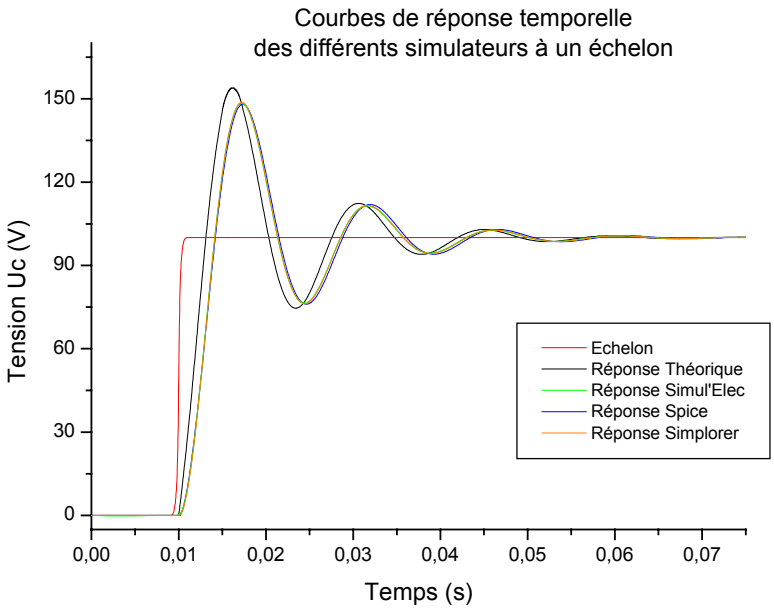


Fig. 3 Comparaison des réponses indicielles du circuit RLC

2.2.1.4 Analyse fréquentielle.

En comparant les résultats obtenus dans le domaine fréquentiel, l'équivalence des résultats des simulateurs est encore plus flagrante **Fig. 4** que lors de l'étude temporelle.

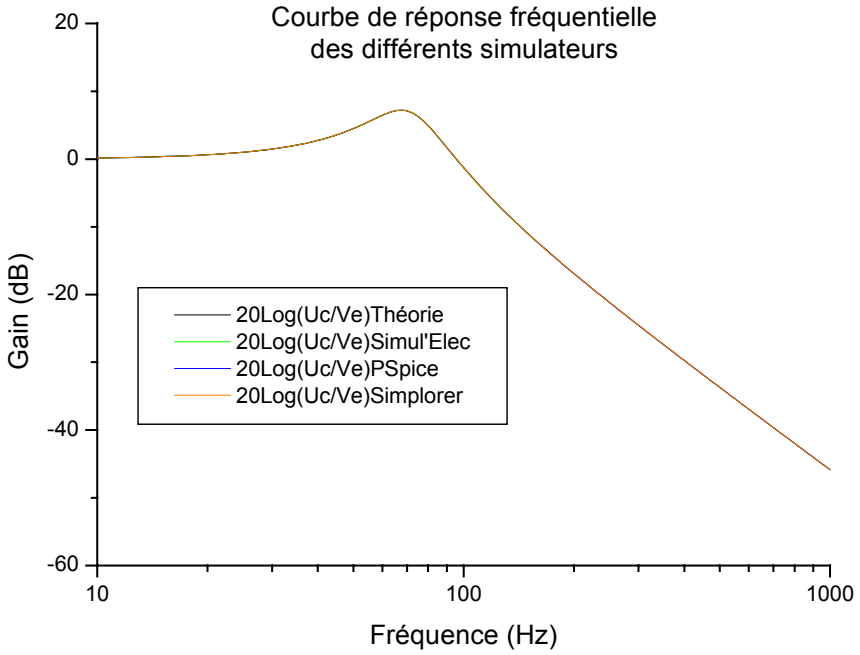


Fig. 4 Comparaison des réponses fréquentielles du circuit RLC

2.2.2 Filtre passif passe-bas ordre 1.

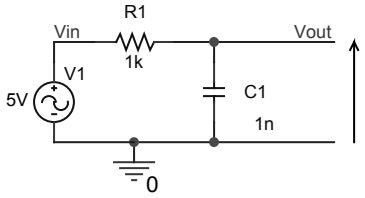


Fig. 5 Filtre RC

Ce montage sert de filtre passe-bas. En plus de la forme des courbes et des valeurs caractéristiques comme la fréquence de coupure à -3dB ou la pente, ce montage permettra de comparer les temps mis par les simulateurs pour afficher les courbes en analyse fréquentielle. Pour ce cas un peu plus simple que le précédent, nous avons relevé le comportement fréquentiel du circuit pour chacun des simulateurs.

De la même manière que pour le cas précédent, nous obtenons des courbes identiques pour tous (voir **Fig. 6**). Pour affiner notre analyse, nous avons de plus relevé les temps de calcul des simulateurs en fonction des indications que donne chacun. Nous avons fait ces essais pour des précisions différentes de l'analyse se traduisant par un nombre plus ou moins élevé de point dans le balayage en fréquence.

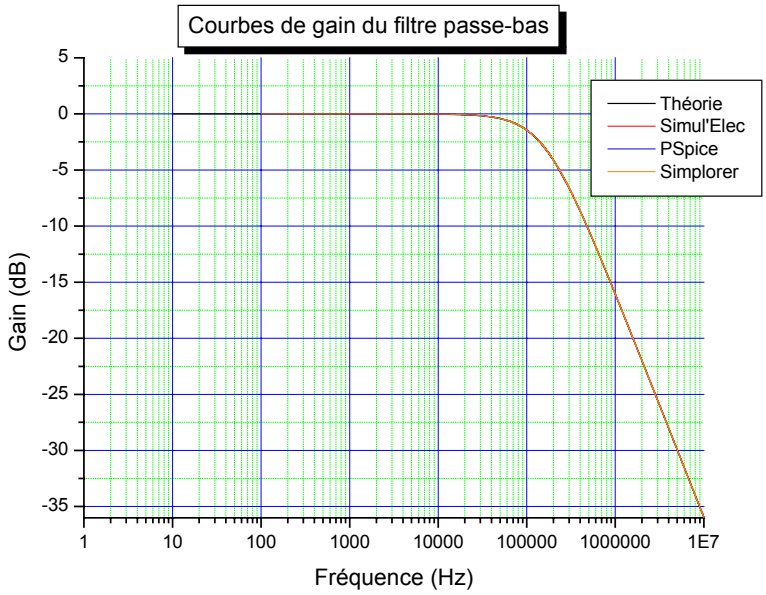


Fig. 6 Réponses fréquentielles d'un circuit RC (gain)

Nombre de points	Simul'Elec (ms)	PSpice (ms)	Simplorer (ms)
65	140	70	9603
650	1141	143	10183
6500	12341	1153	11500

Tab. 1 Comparaison des temps de calcul en analyse fréquentielle

De la synthèse présentée en **Tab. 1**, il ressort que c'est PSPICE qui est le plus efficace dans ce type d'analyse. Ensuite, c'est Simul'Elec qui est le plus performant tant que l'on reste avec une précision raisonnable. On constate sur la figure ci contre qu'au-delà de 6000 points de fréquence, Simplorer rejoigne Simul'Elec. Nous pouvons toutefois constater que pour ce type de circuit, 6000 points constitue une précision excessive.

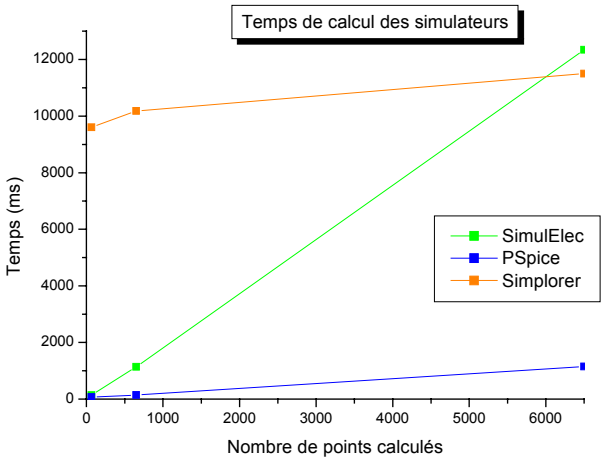


Fig. 7 Temps de calcul des simulateurs en fonction du nombre de points.

2.2.3 Filtrage d'une source bruitée.

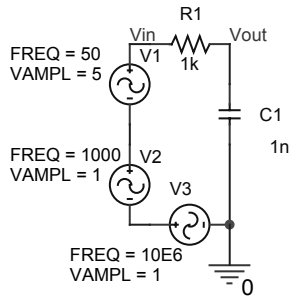


Fig. 8 Source bruitée

Cette analyse a pour but de comparer les vitesses de calcul des logiciels en analyse temporelle ainsi que la finesse des résultats observés. Aux bornes du condensateur, seuls les signaux V1 et V2 oscillant respectivement à 50Hz et 1kHz doivent apparaître. En effet, le signal à 10MHz est filtré par le circuit RC placé en sortie ayant une fréquence de coupure de 160kHz. Ce cas est intéressant surtout pour sa simulation dans le domaine temporel. Il comporte plusieurs sources de fréquences très différentes. Pspice et Simplorer travaillent en analyse transitoire et déterminent le tracé de la courbe de réponse point après point. Pour avoir une précision suffisante, ils doivent baser leur pas de calcul sur le signal dont la période est la plus faible (10MHz). Par contre, la durée de la simulation est déterminée par la période du signal fondamental à 50Hz. Pour avoir un ordre d'idée du nombre de points de calcul on peut faire le rapport simple ces deux périodes extrêmes : 10MHz/50Hz = 200 000 points.

Simul'Elec utilise par contre le principe de superposition. Il réalise une résolution pour chaque source de fréquence différente puis fait la somme des résultats. On aura donc dans ce cas seulement 3 résolutions pour Simul'Elec.
Etant donné les différences de méthodes, nous pouvons donc attendre une grande disparité dans les performances des logiciels.

2.2.3.1 Simulation dans Simul'Elec.

Nous affichons le résultat sur deux périodes en **Fig. 9**. Au premier essai, il a fallu 1,2 secondes pour simuler et tracer la courbe avec la résolution par défaut de 620 points.

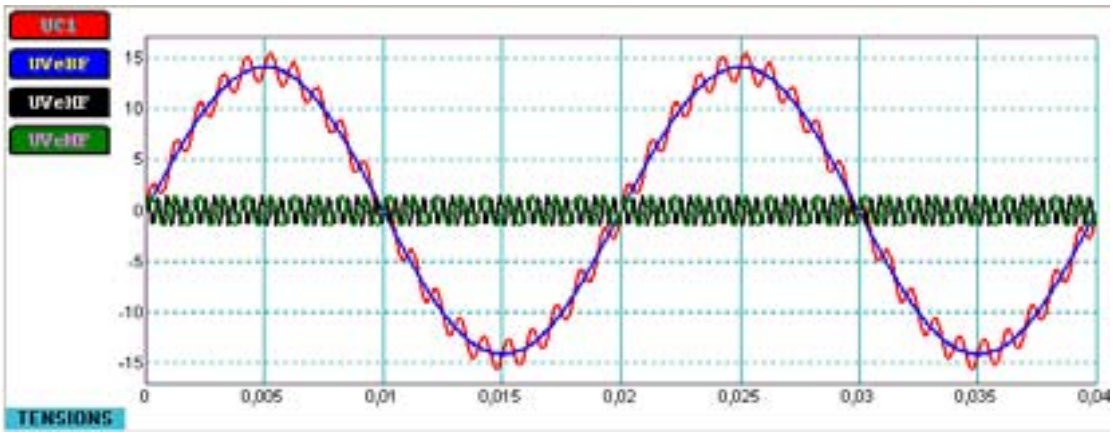


Fig. 9 Circuit avec source bruité sous Simul'Elec

On remarquera que Simul'Elec peut affiner le niveau de détail sans avoir à relancer de simulation et de résolution du système d'équations représentant le circuit.

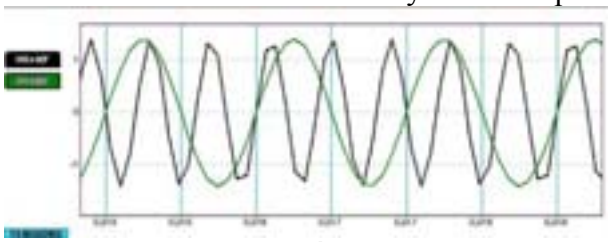


Fig. 10 Loupe fenêtre sur le signal HF

Fig. 11 Loupe classique sur le signal HF

Deux modes permettent d'agrandir les courbes dessinées :

- Un mode « loupe fenêtre » qui permet d'agrandir la courbe tracée à partir de la sélection d'une zone (**Fig. 10**). Ce mode ne recalcule pas les points intermédiaires. On peut alors trouver des résultats incorrects du fait d'un sous échantillonnage.
- Un mode « loupe classique » **Fig. 11** qui agrandi la courbe en recalculant tous les points nécessaires. Ces nouveaux points sont calculés par l'interface graphique sans avoir recours à aucune simulation.

2.2.3.2 Simulation dans PSPICE.

La simulation de ce même circuit dans PSPICE donne des résultats équivalents à ceux de Simul'Elec. Le temps de calcul est beaucoup plus important. En effet, avec PSPICE, il n'est pas possible de fixer le TSTEP qui détermine l'intervalle minimum entre deux points, cette valeur est déterminée automatiquement par le logiciel. Nous pouvons par contre définir la valeur TMAX qui permet d'augmenter le nombre de points à calculer. Avec le réglage par défaut, le calcul de PSPICE a duré plus de 10 min (indication du fichier de sortie). Ce temps n'inclut pas le temps d'affichage des courbes mais seulement la durée de la résolution numérique. Cette mesure est le résultat d'un troisième essai car les deux premiers ont provoqué un blocage du système d'exploitation et nous ont obligé à redémarrer la machine (Processeur Celeron 1GHz, 192Mo de RAM). En examinant la consommation de la mémoire au cours de la simulation, nous avons constaté que cette simulation est très gourmande en espace mémoire et a certainement provoqué la défaillance.

2.2.3.3 Simulation dans Simplorer.

Pour ce circuit, Simplorer semble plus efficace que PSPICE, bien qu'il reste encore assez lent. Il a au moins le mérite de mieux gérer la mémoire et de ne pas se bloquer. Il présente de plus des défauts d'affichage importants lorsque le nombre de points calculés est insuffisant.

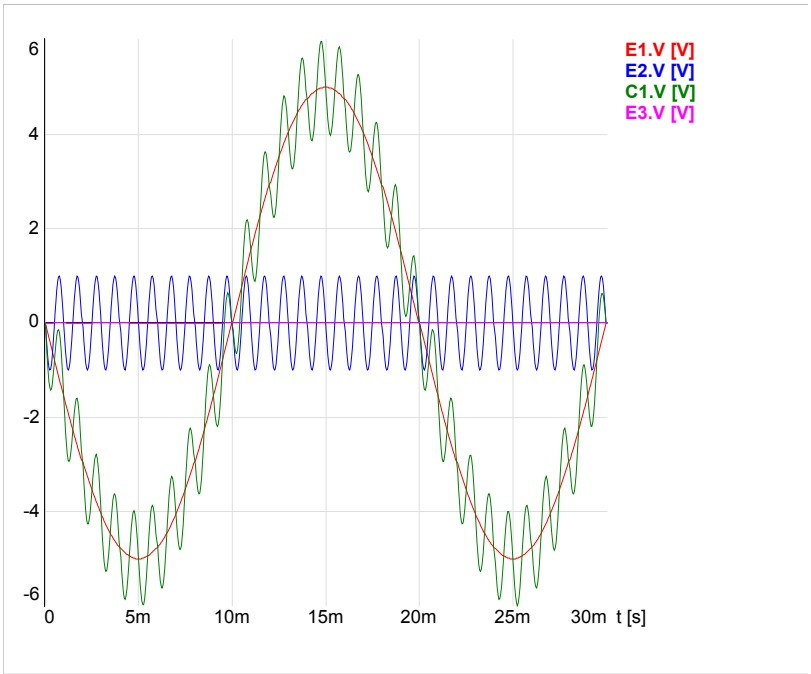


Fig. 12 Filtrage d'une source bruité simulé par Simplorer

Les courbes obtenues montrent bien le fonctionnement du filtre. Cependant, si l'on observe le signal de la source E3 en **Fig. 12**, il apparaît que cette tension est toujours à 0V au lieu d'osciller entre +1V et -1V à 10MHz. Il n'y a donc pas une précision suffisante pour pouvoir observer tous les signaux différents. L'affichage de ces 620 points a pris 6 secondes. En augmentant le nombre de points, on obtient le résultat en 13.8 secondes pour 6200 points et en 80 secondes pour 62000points. Il faut un minimum de 62000 points pour obtenir un signal non nul sur E3. A ce moment là, le bruit apparaît sur le signal de sortie et il est peu atténué par rapport au signal d'entrée comme le montre la **Fig. 13** ci-contre. Cela ne correspond donc pas aux résultats attendus. Si l'on augmente encore la précision, les résultats deviennent aléatoires puisque avec un pas de 100ns, le signal E3 repasse à 0V.

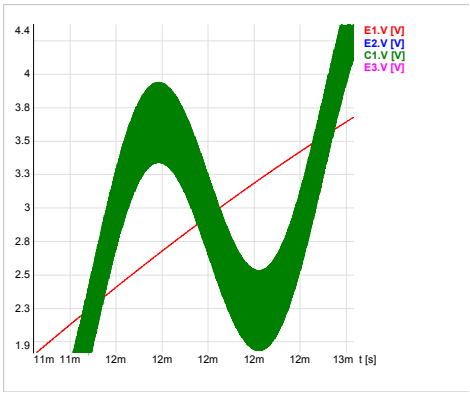


Fig. 13 Zoom du filtrage du signal bruité avec 62 000 points

Afin d'avoir un résultat correct, il faut énormément augmenter la précision et donc le temps de calcul comme le montre la **Fig. 14**. Il faut plus de 30 minutes pour obtenir les 30 millisecondes de simulation. Sur ce dernier tracé, on observe que le signal d'entrée à 10MHz a été très fortement atténué.

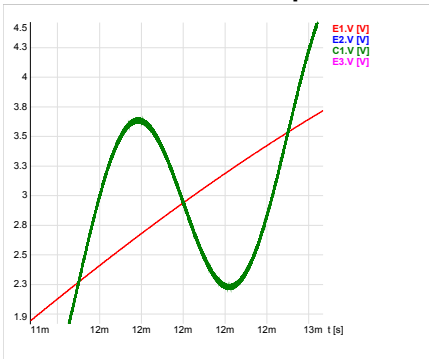


Fig. 14 Zoom du filtrage du signal bruité avec 620 000 points

2.2.3.4 Comparaison des performances.

Contrairement à l'étude fréquentielle, les trois logiciels fonctionnent de façons différentes en analyse temporelle. Ainsi, les résultats auront une précision plus ou moins grande en relation avec le temps de calcul qui peut varier de moins de deux secondes à plus de 30 minutes.
Le logiciel le plus rapide est Simul'Elec qui donne des résultats satisfaisants dès 620 points en moins de 2 secondes. PSpice donne des courbes de grande précision mais il n'est pas possible de réduire le nombre de points et du coup, les temps de calcul et d'affichage sont très longs, de l'ordre de 10 minutes. Simplorer a un temps de calcul intermédiaire mais affiche des résultats trompeurs comme la source de 10MHz nulle ou non selon le pas minimum choisi ou encore la sortie plus ou moins filtrée selon la précision demandée.
Le tableau ci-dessous résume les temps de calcul extraits des tests.

	Temps de la simulation (s)		
	Simul'Elec	Ppice	Simplorer
Temporel 620 pts	1,3	Impossible de préciser	6
Temporel 6200 pts	2		13,8
Temporel 62000 pts	20		80
Précision suffisante pour affichage correct	20	608	1 800

Tab. 2 Comparaison des temps de calcul du circuit des sources bruitées

3 Banc de sélectivité des protections.

Dans le cadre de la validation du module de simulation de circuits électrotechniques Simul'Elec, une série de tests a été effectuée sur un banc didactique de sélectivité des protections en TBT (Très Basse Tension). Ce banc va nous permettre de réaliser des mesures sur les disjoncteurs en reproduisant des courts-circuits dans des conditions sécurisées avec des courants pouvant atteindre 200A. Nous l'avons exploité notamment pour valider la simulation de la sélectivité des disjoncteurs.
La sélectivité correspond à l'aptitude d'un système de protection à détecter un défaut dans une zone déterminée d'un réseau électrique et à provoquer le déclenchement des protections appropriées pour éliminer ce défaut, avec le minimum de perturbation pour la partie saine du réseau.
Le but de ces manipulations n'est pas de vérifier l'exactitude parfaite des résultats mais plutôt le comportement général des disjoncteurs en fonction du type de montage et du type de sélectivité. On observe également la cohérence des temps de déclenchement des disjoncteurs.

3.1 Description du banc.

Le "Banc de sélectivité des protections en TBT"a été conçu dans un but didactique [BAN] par Merlin-Gérin du groupe Schneider-Electric. Cinq disjoncteurs sont utilisables simultanément sur trois niveaux différents :

- D1 est un disjoncteur 20A (1er niveau). Les courbes B, C et D sont disponibles.
- D2 est un disjoncteur 10A (2ème niveau). Les courbes B, C et D sont disponibles.
- D3 est un disjoncteur 10A (2ème niveau). Les courbes B, C et D sont disponibles.
- D4 est un disjoncteur 6A (3ème niveau). Les courbes B, C et D sont disponibles.
- D5 est un disjoncteur 4A (3ème niveau). Les courbes B, C et D sont disponibles.

Le banc dispose également d'un bouton poussoir (BP) qui permet de créer le défaut dans le circuit et d'ampoules témoins qui indiquent l'état des disjoncteurs amont.
On trouvera en **Annexe 8** des images des faces du banc.

Chaque disjoncteur peut être relié à une bobine ayant une faible impédance mais permettant de limiter le courant lors des courts-circuits. Chaque bobine dispose de trois bornes de branchements utilisables selon la valeur de l'impédance désirée.
Ce banc didactique a été créé dans un but pédagogique afin d'illustrer le fonctionnement d'un système électrique, ses protections et les éventuels problèmes pouvant survenir en cas de défaut. Il permet de sensibiliser les personnes sur l'importance des protections et de leur dimensionnement.

Le banc est alimenté en 48V via un transformateur 230V/48V. C'est la tension de sécurité en local sec. Les disjoncteurs disponibles sont des disjoncteurs Merlin-Gérin C60N de calibres 20A, 10A, 6A et 4A. Ces disjoncteurs sont généralement utilisés en 230V mais ils peuvent également fonctionner sur les circuits alimentés en triphasé. Leur tension nominale est de 400V. Pour chaque disjoncteur, les courbes B, C et D sont disponibles. Le temps de coupure d'un disjoncteur par rapport au courant est exprimé sur les courbes constructeur fournies en **Annexe 3**.

La gamme C60N est la plus répandue de Merlin-Gérin et fait partie des matériels de protection les plus installés en France, tous fabricants confondus. Les différentes courbes disponibles sur ce banc correspondent à différents types d'utilisation :

- Courbe B : protection des générateurs, des personnes et des grandes longueurs de câbles
- Courbe C : protection de câbles alimentant des récepteurs classiques
- Courbe D : protection des câbles alimentant les récepteurs à fort courant d'appel

Les valeurs en ohms des impédances des bobines, transformateur, générateur et disjoncteurs qui nous ont servi pour les calculs sont résumées dans les tableaux en **Annexe 9**.

3.2 Schémas testés.

Le banc didactique est fourni avec un guide de manipulation à l'usage des enseignants et des élèves. En se basant sur ce support et les expériences proposées, nous avons défini 3 schémas de base d'une complexité progressive permettant de contrôler la sélectivité.

3.2.1 Un Disjoncteur.

Ce montage représente une installation simple qui n'est protégée que par un seul disjoncteur. En cas de défaut, toute l'installation est privée de courant. Si certains équipements ne doivent pas être stoppés, il est préférable de les séparer du reste de l'installation électrique.

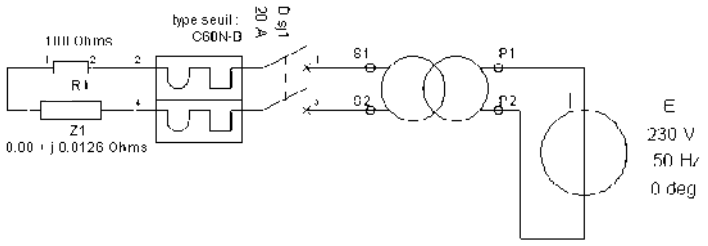


Fig. 15 Schéma du circuit de test à 1 disjoncteur

Les mesures des courants de court-circuit et des temps de déclenchements ont été effectuées successivement sur les différents disjoncteurs disponibles avec un temps de repos entre chaque mesure sur un même disjoncteur d'au moins 5 minutes. En effet, lorsqu'un courant important traverse le disjoncteur, le déclencheur thermique chauffe. Le temps de repos de 5mn est nécessaire afin de le laisser refroidir et de pouvoir refaire les tests dans les mêmes conditions initiales. Le disjoncteur est connecté en série à une impédance Z1 permettant de limiter le courant de court-circuit et à une résistance de charge (ampoule) qui est court-circuitée pour créer le défaut.

3.2.2 Deux Disjoncteurs.

Ce montage plus complexe que le précédent permet de tester le principe de sélectivité. Il est également plus répandu car il permet d'isoler certains équipements. Un défaut sur une partie de l'installation ne doit déclencher que le système de protection directement amont sans perturber le reste du système. Ce type de montage fait apparaître le principe de disjoncteur général en amont et de disjoncteur divisionnaire en aval.

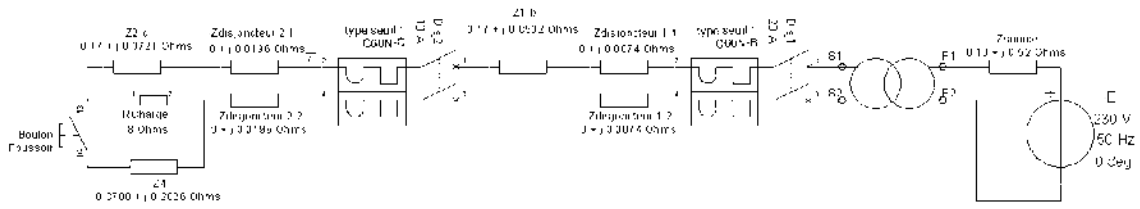


Fig. 16 Schéma du circuit de test à 2 disjoncteurs

Comme pour les circuits n'ayant qu'un disjoncteur, les mesures ont été effectuées successivement sur les différents disjoncteurs avec un temps de repos entre chaque mesure sur un même disjoncteur d'au moins 5 minutes. Les disjoncteurs sont connectés en série à une impédance Z1 et/ou Z2 et/ou Z3 et/ou Z4 permettant de limiter le courant de court-circuit et à une résistance de charge (ampoule) qui est court-circuitée pour créer le défaut.
L'avantage de tester des montages à plusieurs disjoncteurs est de permettre de tester non seulement le disjoncteur dans son fonctionnement mais également la sélectivité d'un disjoncteur par rapport à l'autre.

3.2.3 Trois Disjoncteurs.

Les domaines d'utilisation de ce type de montage sont les mêmes que pour celui à deux disjoncteurs. Les tests de sélectivité sont donc toujours réalisables et il est possible de vérifier de plus nombreuses configurations.

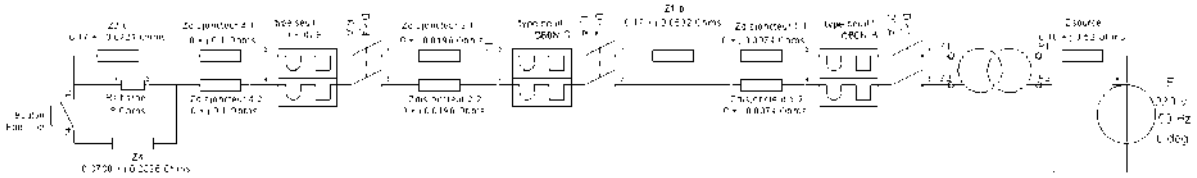


Fig. 17 Schéma du circuit de test à 3 disjoncteurs

Les mesures se font comme pour les circuits à deux disjoncteurs.

3.3 Analyse théorique.

Lors de la création du court-circuit, on ferme le circuit avec une impédance d'une valeur très faible. Cette impédance s'ajoute à celle des autres composants en série avec elle, notamment celle des appareils de protection. Leurs valeurs généralement faibles ont donc une grande influence sur la valeur du courant de court-circuit Icc et donc du fonctionnement des protections.

3.3.1 Définition du courant de court circuit.

Un courant de court-circuit (Icc) est un courant provoqué par un défaut d'impédance négligeable entre des points d'installation présentant normalement une différence de potentiel [SOC00]. On distingue 3 niveaux de courant de court-circuit. :

- Le court-circuit crête (Icc crête) correspond à la valeur extrême de l'onde générant des forces électrodynamiques élevées notamment au niveau des jeux de barres et des contacts ou connexions d'appareillage.
- Le courant de court-circuit efficace (Icc eff) : valeur efficace du courant de défaut qui provoque des échauffements dans les appareils et conducteurs et peut porter les masses des matériels à des potentiels dangereux.
- Le courant de court-circuit minimum (Icc min) : valeur efficace du courant de défaut s'établissant dans les circuits d'impédance élevée (conducteur à section réduite et canalisation de grande longueur…) et donc cette impédance a été en plus augmentée par l'échauffement de la canalisation en défaut. Il est nécessaire d'éliminer rapidement ce type de défaut par les moyens appropriés.

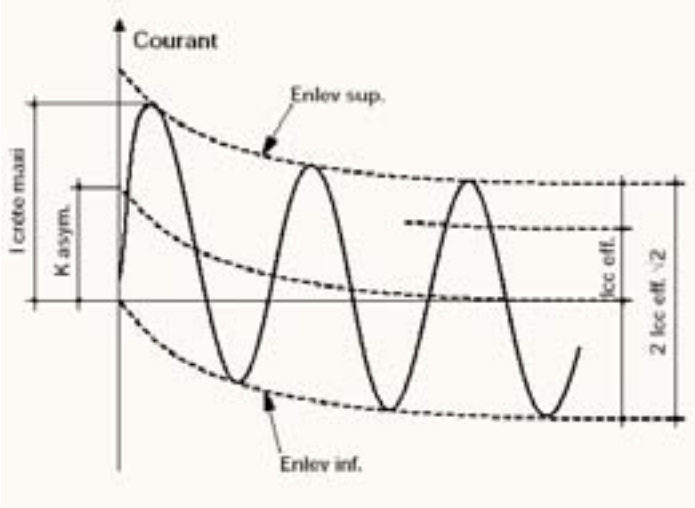


Fig. 18 Allure du courant de court-circuit à l'apparition du défaut

Notons que le calcul théorique qui suit porte sur la détermination de la valeur efficace du courant de court-circuit. De même, le simulateur ne prend pas en compte le courant Icc crête mais seulement la valeur efficace.

3.3.2 Calcul du courant de court-circuit.

Le courant de court-circuit Icc se calcule en fonction des impédances de court-circuit. On utilise la méthode des impédances décrite par la norme CEI 60-909 [CEI88]. Au cours du calcul, les composants du circuit sont modélisés par leur impédance afin de calculer simplement la valeur de l'Icc.
Il est nécessaire de prendre en compte l'impédance interne du générateur ramenée au secondaire, l'impédance interne du transformateur au secondaire, les impédances de "charge" Z et les impédances internes des disjoncteurs.

Exemple de calcul :
On considère le montage de la **Fig. 16** comprenant une alimentation 50Hz, 230V de puissance 10kVA, un transformateur 230V/48V, un disjoncteur D1, 20A, un disjoncteur D2, 10A, une charge Z1c, une charge Z2c et une ampoule mise en court circuit pour créer le courant de défaut.

a) Impédance du générateur (Za)

$Za = \frac{U^2}{Scc} = \frac{230^2}{100.10^3} = 0{,}53\Omega$ où U est la tension composée du réseau non chargé.

En BT, $Xa = 0{,}98 \times Za$ soit $\boxed{Xa = 0{,}52\Omega}$ et $Ra = 0{,}2 \times Za$ pour une puissance inférieure à 150kVA donc $\boxed{Ra = 0{,}10\Omega}$.

b) Impédance du générateur ramené au secondaire (Za')

$Za' = k^2 \times Za$ k étant le rapport de transformation du transformateur donc $Za' = \left(\frac{48}{230}\right)^2 \times 0{,}53$ soit $Za' = 21{,}2\Omega$ et donc $\boxed{Xa' = 20{,}7m\Omega}$

et $\boxed{Ra' = 4{,}2m\Omega}$.

c) Impédance du transformateur

Les données constructeur pour le transformateur sont $Zs = 86{,}4m\Omega$, $\boxed{Xs = 83{,}8m\Omega}$, $\boxed{Rs = 20{,}8m\Omega}$.

d) Impédance des disjoncteurs

Le constructeur annonce une impédance de 7,4mΩ pour le disjoncteur de 20A et de 19,6mΩ pour le disjoncteur de 10A. **Ces valeurs s'entendent par pôle**. Il apparaît que les disjoncteurs utilisés ont 2 pôles. On obtient donc une impédance totale des disjoncteurs $Zd = 7{,}4 \times 2 + 19{,}6 \times 2 = 54m\Omega$. On considère que cette impédance est essentiellement inductive donc que $Xd \approx 54m\Omega$.

e) Impédance des charges

On utilise ici l'impédance $Z1c = 270m\Omega$ où $\boxed{R1c = 260m\Omega}$ et $\boxed{X1c = 75m\Omega}$ et l'impédance $Z2c = 185m\Omega$ où $\boxed{R2c = 170m\Omega}$ et $\boxed{X2c = 72m\Omega}$.

f) Impédance totale

On obtient donc une impédance totale $Zcc^2 = (Ra' + Rs + Rd + R1c + R2c)^2 + (Xa' + Xs + Xd + X1c + X2c)^2$ soit

AN : $Zcc = \sqrt{(4{,}2 + 20{,}8 + 0 + 260 + 170)^2 + (20{,}7 + 83{,}8 + 54 + 75 + 72)^2}$

$\boxed{Zcc = 548m\Omega}$.

g) Courant de court-circuit

Connaissant la valeur de l'impédance du circuit et la tension à la sortie du transformateur, il est possible de calculer le courant de défaut. On obtient : $Icc = \frac{Us}{Zcc}$ donc

$\boxed{Icc = \frac{48}{548} \times 10^3 = 87{,}6A}$.

3.3.3 Valeur théorique des Icc pour les montages étudiés.

Nous avons déterminé pour chacun des trois montages les valeurs des Icc attendues par chaque configuration possible. On voit dans le **Tab. 3** le résultat de calcul pour le montage à un disjoncteur. Le détail des calculs pour les autres montages est donné en **Annexe 11** et **Annexe 12**.

			Valeurs totales des impédances utilisées (mΩ)		Impédance totale de défaut (mΩ)	**Courant de défaut (A)**
Charge	**Disj.**	**Courbe**	Réelle	Imaginaire	Zcc	**Icc**
Z2c	**D1**	**B**	170	72,1	273	**176**
		C	170	72,1	273	**176**
		D	170	72,1	273	**176**
	D2	**B**	170	72,1	273	**165**
		C	170	72,1	273	**165**
		D	170	72,1	273	**165**
	D3	**B**	170	72,1	273	**165**
		C	170	72,1	273	**165**
		D	170	72,1	273	**165**
Z4	**D4**	**B**	378	282,6	569	**67**
		C	378	282,6	569	**67**
	D5	**B**	378	282,6	569	**60**
		C	378	282,6	569	**60**

Tab. 3 Calcul de Icc pour le montage à un disjoncteur

3.3.4 Détermination des temps de coupure et de la sélectivité.

Comme nous l'avons vu dans le chapitre 3, le temps de coupure des disjoncteurs dépend de la valeur du courant de défaut et du type de disjoncteur utilisé. Ils se lisent sur les courbes constructeur (voir **Fig. 19** ci-contre). Elles définissent une enveloppe à l'intérieure de laquelle on peut situer le déclenchement du disjoncteur. Ces courbes sont fournies de manière plus précise en **Annexe 3**. On peut ainsi définir une plage de temps de déclenchement à partit de la valeur de l'Icc

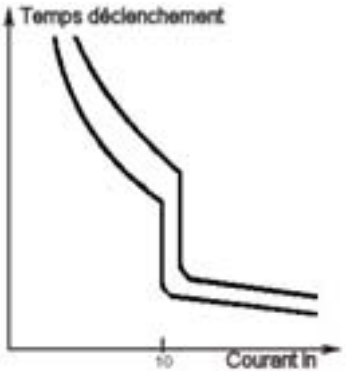


Fig. 19 courbe de déclenchement d'un disjoncteur

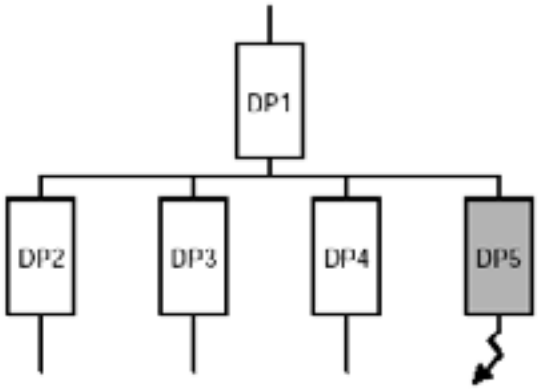


Fig. 20 Création d'un défaut en aval du réseau.

On rappelle que la sélectivité est l'aptitude d'un système de protection à détecter un défaut dans une zone déterminée d'un réseau électrique et à provoquer le déclenchement des disjoncteurs appropriés pour éliminer ce défaut, avec le minimum de perturbation pour la partie saine du réseau [BLA82].

Cette notion de sélectivité est valable entre deux (ou plusieurs) dispositifs de protection tels que fusibles, disjoncteur, discontacteur.

La sélectivité totale est assurée lorsque les zones temps/ courants des deux dispositifs de protection ne se recouvrent pas (**Fig. 21**). La sélectivité partielle intervient lorsque les courbes se chevauchent en partie. Le fonctionnement sera correct si le courant de défaut est inférieur au point de croisement des courbes **[SOC00].** On a une non-sélectivité dans les zones où les courbes se chevauchent c

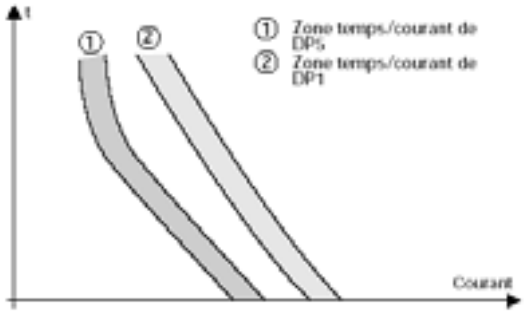


Fig. 21 Sélectivité totale

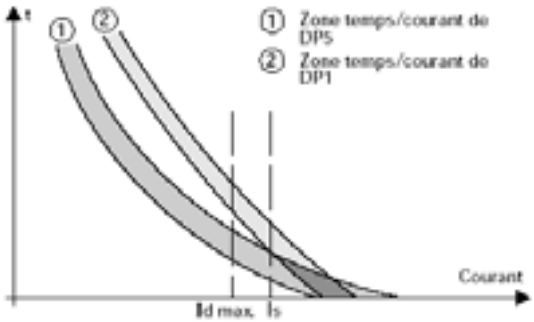


Fig. 22 Non-sélectivité

A partir des valeurs de Icc précédemment calculées, nous avons pu determiner le temps de déclenchement de chaque disjoncteur. Ainsi on peut déterminer lequel déclenche en premier et définir la sélectivité correspondante. Nous avons ainsi déterminé les sélectivités théoriques pour les montages à deux disjoncteurs et à trois disjoncteurs (voir tableaux en **Annexe 11** et **Annexe 12**).

3.4 Simulation des schémas.

Pour chaque cas de circuit (un, deux et trois disjoncteurs), on dispose d'un schéma (**Fig. 23**) type pour lequel on modifie les valeurs des composants afin de correspondre au matériel que l'on utilise dans chaque essai (disjoncteurs et charges).

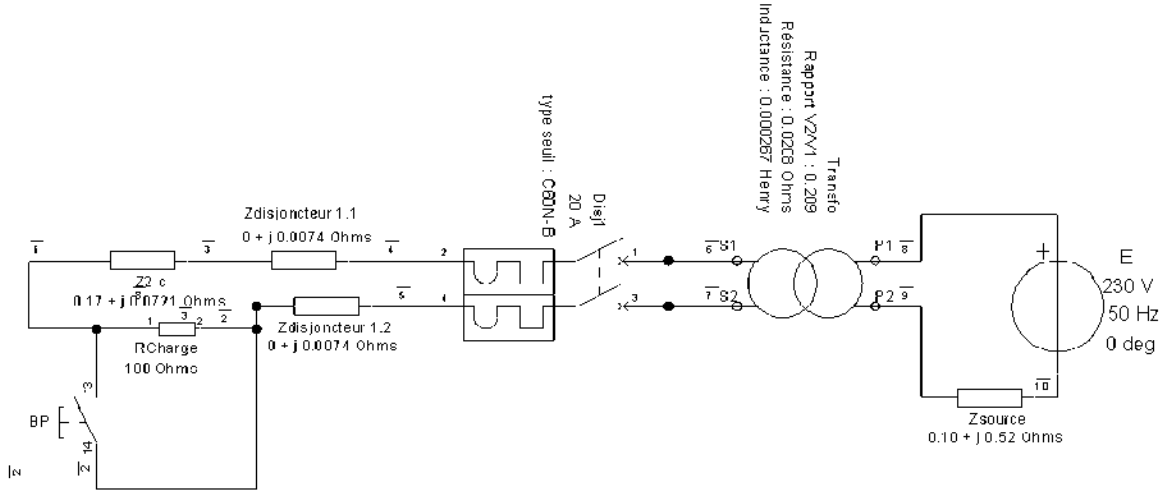


Fig. 23 Schéma simulé du circuit à un disjoncteur

Au moment de la validation, les modèles de composants du simulateur ne prennent pas en compte les résistances internes des générateurs et des disjoncteurs. Les impédances des différents composants comme Zsource et Zdisjoncteur ont été rajoutés lors de la saisie des schémas. Le choix du transformateur s'est porté sur un transformateur dont le modèle intègre l'impédance interne ramenée au secondaire (modèle de Kapp) et qui correspond à celui retenu par la méthode de calcul des Icc. La charge (Ampoule de charge) a été modélisée par une Résistance de 100 Ω.
Le défaut est crée en court-circuitant la charge à l'aide d'un bouton poussoir. Celui-ci permet de déclencher le disjoncteur. Ce court-circuit peut être franc ou réalisé via une impédance.

Pour prendre l'exemple du schéma à un disjoncteur, nous suivrons la démarche suivante :
Apres avoir lancé le module de simulation, et choisi le mode de résolution en régime événementiel, nous allons procéder à la modification des états logiques et à la visualisation des sorties pour pouvoir afficher les résultats.

- Le BP (bouton poussoir) est sélectionné dans "modifier les états logiques" puis il est rajouté dans les entrées logiques. On l'active pour créer le court-circuit.
- Le Disj1 est sélectionné dans "Etat logique" puis il est rajouté dans les sorties logiques. On visualise ainsi l'état logique du disjoncteur (ouvert ou fermé)
- IZ2c est sélectionné dans "Sorties" puis elle est rajoutée dans "Courant", cela nous permet de visualiser la valeur efficace du courant.
- Z2c est sélectionné dans "Visualisation sortie", "Sélection des courbes" puis rajouté dans "Courant". Cela nous permet de visualiser la courbe du courant de court-circuit.

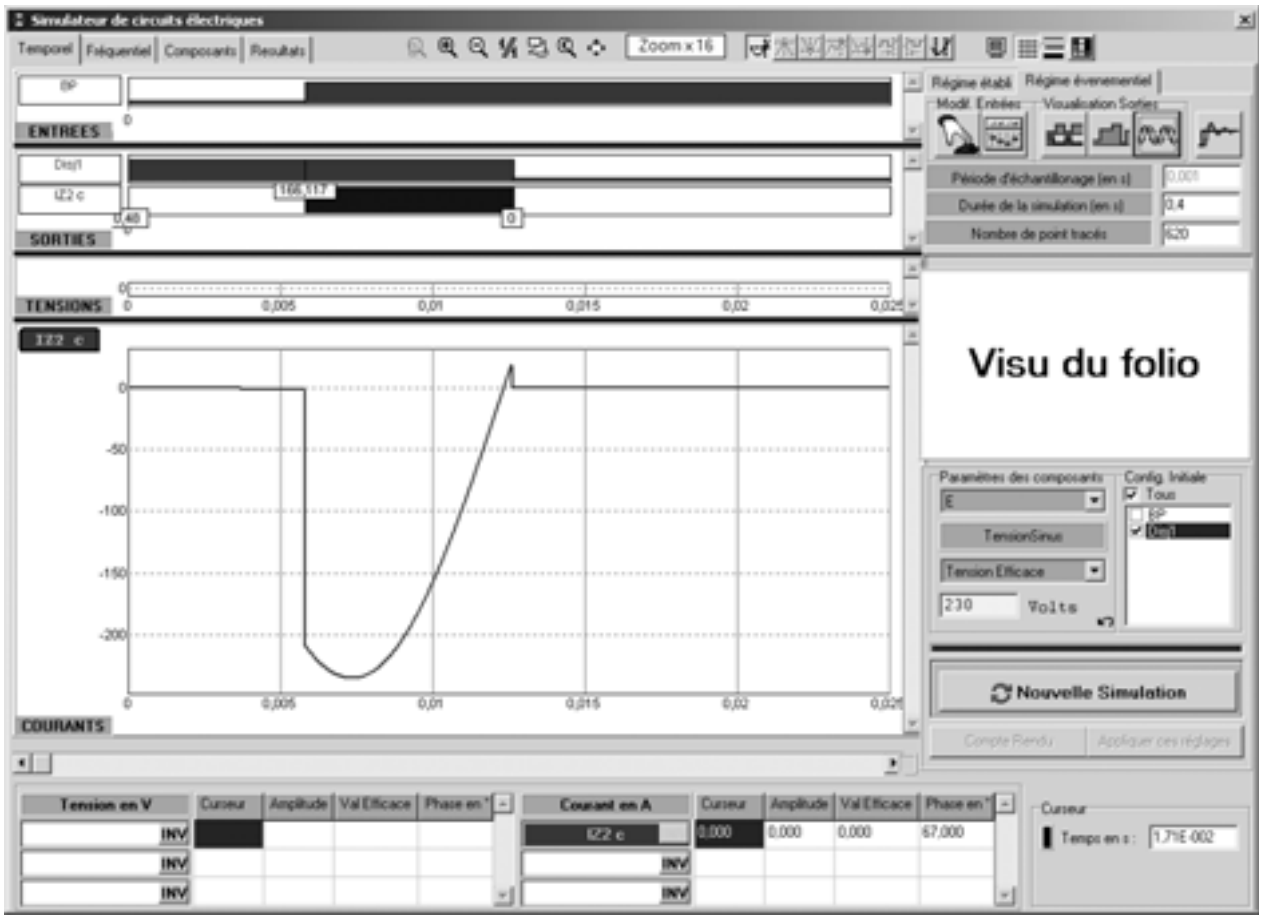


Fig. 24 Affichage du résultat d'une simulation

La valeur courant de court-circuit est indiquée sur la sortie IZ2.
Pour mesurer le temps de déclenchement du disjoncteur, on utilise les marqueurs du logiciel pour relever le temps écoulé entre la création du court-circuit par l'appui sur le bouton poussoir et l'ouverture du disjoncteur.
Ici, on relève Icc = 166.117A et Td = 7ms.

3.5 Mesures.

Pour chaque cas d'essai, on provoque un court-circuit. Nous relevons la courbe du courant traversant les disjoncteurs. A partir de cette courbe, Nous mesurons la valeur du courant de court-circuit et le temps de déclenchement du disjoncteur.

3.5.1 Méthode de prises de mesures.

Les mesures se sont effectuées à l'aide d'un oscilloscope numérique, d'une sonde isolée de courant et d'un ordinateur portable de type PC.

L'oscilloscope à mémoire numérique est un TDS2002 (Tektronix). Il dispose d'une mémoire de 2500 points permettant d'afficher les courbes provenant de deux voies différentes sur l'écran couleur. Un module d'extension de communication TDS2CMA a été ajouté afin de permettre une connexion à un ordinateur via un câble Null Modem.

La sonde de courant utilisée est une MN60 BNC (Chauvin Arnoux). Cette sonde à effet Hall se connecte directement sur l'une des voies d'un oscilloscope. Elle dispose de deux calibres permettant de mesurer des courants compris entre 100mA et 20A ou entre 500mA et 200A. Son rapport de conversion est de 1A/10mV pour le plus grand calibre et 1A/100mV pour le plus petit. Sa plage de fonctionnement optimale est comprise entre 48Hz et 65Hz et son facteur de distorsion est inférieur à 1%.

L'ordinateur portable est un PC Packard Bell disposant d'un connecteur RS232 permettant de connecter l'oscilloscope numérique. Le logiciel WaveStar Version 2.6 (Tektronix) y a été installé afin de traiter les informations provenant de l'oscilloscope.

3.5.2 Relevé de courant.

La sonde de courant est connectée à l'oscilloscope et placée sur l'un des fils du montage. Le calibre sélectionné est le plus grand car les courants de courts-circuits peuvent dépasser les 200A. Afin de mesurer le temps de déclenchement des disjoncteurs ainsi que les courants de court-circuit, l'oscilloscope a été utilisé dans son mode SEQ UNIQUE qui permet de capturer une séquence unique du signal et de la conserver à l'écran. Le déclenchement se fait sur front montant lorsque le signal dépasse un seuil prédéfini. La partie antérieure au déclenchement est sauvegardée n'est donc pas perdue. Par contre, il peut arriver que sur un signal court ou un front descendant, l'acquisition ne se fasse pas. Il faut alors recommencer la manipulation.

Une fois le signal affiché sur l'écran de l'oscilloscope, celui-ci est transféré sur l'ordinateur qui permet aisément de mesurer le temps de déclenchement des disjoncteurs et les valeurs de courants. Toutes les courbes sont sauvegardées afin d'être éventuellement réutilisées ultérieurement.

3.5.3 Problèmes de mesures.

Il est apparu dès les premiers tests que les résultats obtenus pour deux essais identiques peuvent varier. Ces variations peuvent provenir des conditions de température ambiante différentes, d'échauffement des disjoncteurs et des bobines et de l'instant de déclenchement du court-circuit par rapport à la valeur instantanée du signal sinusoïdal de fréquence 50Hz comme le montre la **Fig. 25**.

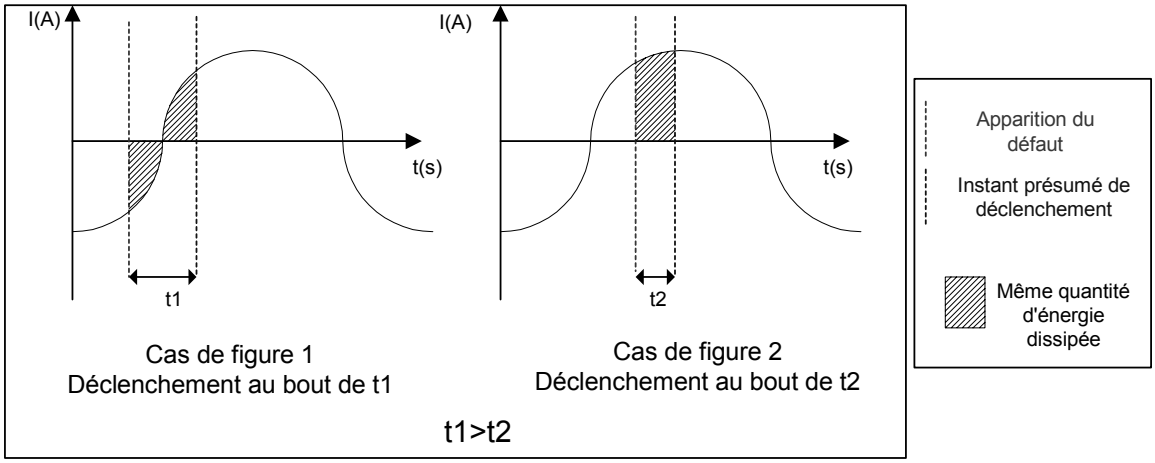


Fig. 25 Variation du temps de déclenchement selon l'instant d'apparition du défaut

En effet, si le défaut apparaît alors que le signal est dans sa partie haute (cas 2 de la **Fig. 25**), la quantité d'énergie qui traverse le disjoncteur est importante. A l'inverse, si le défaut apparaît alors que le signal est près de l'origine, le temps nécessaire pour accumuler la même quantité d'énergie pour déclencher le disjoncteur est plus grand (cas 1 de la **Fig. 25**).

Les 5 courbes suivantes représentent le courant de court-circuit à partir du moment où celui-ci commence jusqu'au moment où le disjoncteur se déclenche. Les conditions d'essai sont identiques : même disjoncteur, même charge. Au moins 5 minutes se sont écoulées entre chaque déclenchement afin que le système, et en particulier le déclencheur thermique, ait le temps de refroidir.

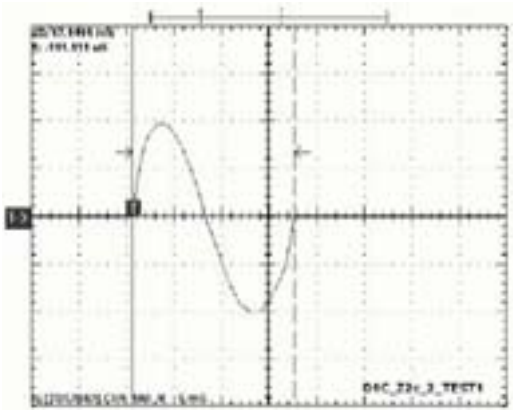


Fig. 26 D1C Z2c TEST 1 Front montant

Courant Efficace : 142A
Temps de coupure : 17.1ms

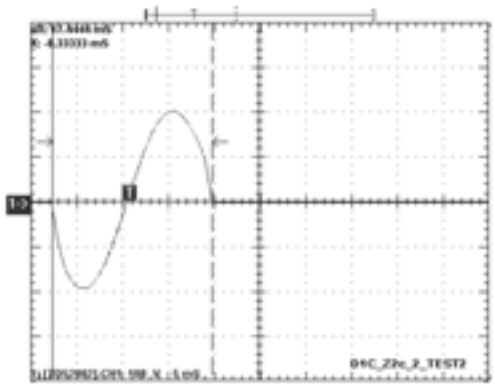


Fig. 27 D1C Z2c TEST 2 Front montant

Courant Efficace : 142A
Temps de coupure : 17.4ms

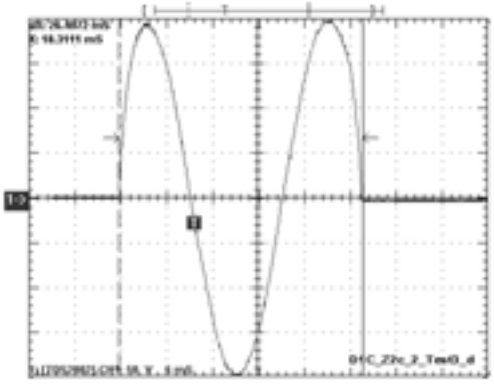


Fig. 28 D1C Z2c TEST 3 Front descendant

Courant Efficace : 141A
Temps de coupure : 26.5ms

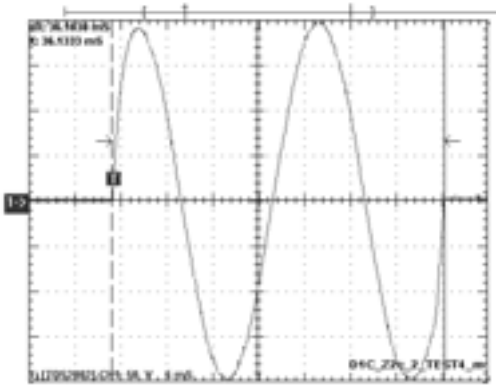


Fig. 29 D1C Z2c TEST 4 Front montant

Courant Efficace : 141A
Temps de coupure : 36.2ms

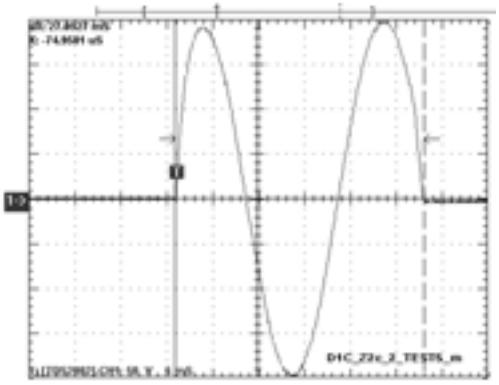


Fig. 30 D1C Z2c TEST 5 Front montant

Courant Efficace : 141A
Temps de coupure : 27.1ms

En raison de la dispersion des résultats pour un même essai, nous avons décidé d'effectuer chaque mesure trois fois et de garder la moyenne des résultats.
Lors des simulations, le résultat obtenu sera toujours le même car le simulateur considère que la forme du signal est toujours la même avec le déphasage de la tension toujours nul au temps t=0. De plus la simulation ne prend pas en compte les phénomènes transitoires liés à la présence d'inductances dans le circuit et qui peuvent avoir un effet sur la forme des signaux.

3.5.4 Mesures effectuées.

Les résultats relevés pour les trois schémas sont regroupés dans les tableaux présentés en **Annexes 13, 14** et **15** . A chaque fois les trois mesures sont données mais lors de l'analyse des résultats, nous nous intéresserons plus particulièrement aux valeurs moyennes.

Le banc est composé de deux surfaces de travail à peu près identiques. Les essais sur les circuits à un disjoncteur ont été effectués sur les deux cotés. Il est apparu que les disjoncteurs installés sur le coté 1 ne donnaient pas toujours des temps de déclenchement cohérents comme D1 courbe C ou D3 courbe D qui ont des temps de déclenchement beaucoup trop longs. C'est en effet la surface la plus utilisée et par conséquent les composants sont plus sollicités. Nous avons donc décidé de n'utiliser que les disjoncteurs situés sur le coté 2 du banc pour la suite des tests. Les disjoncteurs disponibles sont D1, D3 et D4.
Les résultats obtenus grâce au simulateur ne sont valables que pour des appareils neufs ou utilisés dans des conditions normales. Les modèles ne prennent pas en compte l'usure des composants.

3.6 Comparaison des résultats.

Les tableaux suivants (**Tab. 4**, **Tab. 5**, **Tab. 6**) présentent les valeurs théoriques, simulées et mesurées des courants de courts-circuits et des temps de déclenchement des disjoncteurs. Il est à remarquer que les valeurs des courants sont en valeurs efficaces pour les valeurs théoriques et simulées et en valeur max pour celles mesurées. En effet, lors des manipulations, il n'était pas toujours possible de relever la valeur efficace du courant car le temps de déclenchement du disjoncteur était le plus souvent inférieur à une période de sinusoïde (50Hz donc 20ms).

Nous disposons de 55 configurations différentes :

- 8 cas pour le circuit à un disjoncteur.
- 24 cas pour le circuit à deux disjoncteurs.
- 23 cas pour le circuit à trois disjoncteurs.

Pour chacune, nous avons comparé les différences entre le comportement simulé et le comportement réel. Dans le cas du circuit à un disjoncteur, Nous nous sommes plus particulièrement intéressés à la précision des valeurs de Icc et du temps de déclenchement. Les cas des circuits à deux et trois disjoncteurs nous permettent plutôt de vérifier la sélectivité avec leurs 47 cas de test.

										Ecarts										
Charge	Disjoncteur	Courbe	(1) Valeurs Théoriques			(2) Valeurs Simulées		(3) Valeurs Mesurées		|(1)-(2)|		|(1)-(2)|/1 **(%)**		|(1)-(3)|	|(1)-(3)|/1 **(%)**	|(2)-(3)|	|(2)-(3)|/2 **(%)**	|(1)-(3)|-|(2)-(3)|	|(1)-(3)|-|(2)-(3)| %
			Icc eff (A)	Temps de coupure		**Icc eff (A)**	Temps (ms)	Banc côté 2		**Icc eff (A)**	Tps (ms)	**Icc eff**	Tps	Tps (ms)	Tps	Tps (ms)	Tps	Tps(ms)	Tps
				min (ms)	max (ms)			**Icc max (A)**	Temps de coupure (ms)										
Z2c	D1	B	**176**	10	17	**166**	7	**183**	8,0	**9,6**	3,0	**5,4**	30,0	2,0	20,0	1,0	14,3	1,0	5,6
		C	**176**	10	1800	**166**	9	**200**	22,9	**9,6**	1,0	**5,4**	10,0	12,9	129,3	13,9	154,7	-1,0	-25,5
		D	**176**	700	1800	**166**	848	**196**	1200,0	**9,6**	148,0	**5,4**	21,1	500,0	71,4	352,0	41,5	148,0	29,9
	D3	B	**165**	8	20	**157**	6	**163**	6,0	**8,1**	2,0	**4,9**	25,0	2,0	25,0	0,0	0,1	2,0	25,0
		C	**165**	8	20	**157**	7	**169**	7,0	**8,1**	1,0	**4,9**	12,5	1,0	11,9	0,0	0,7	0,9	11,3
		D	**165**	8	20	**157**	10	**171**	13,5	**8,1**	2,0	**4,9**	25,0	5,5	68,4	3,5	34,7	2,0	33,7
Z4	D4	B	**67**	9	19	**66**	7	**96**	6,1	**1,1**	2,0	**1,7**	22,2	2,9	32,1	0,9	12,7	2,0	19,4
		C	**67**	9	19	**66**	8	**93**	9,7	**1,1**	1,0	**1,7**	11,1	0,7	8,1	1,7	21,6	-1,0	-13,5

Tab. 4 Calcul des écarts pour le circuit à un disjoncteur.

			Théorie						Simulation			Mesures		
D1	D2	Impédances utilisées	Icc (A)	Sélect	Temps de déclenchement (ms)				Icc (A)	Disj.	Tps (ms)	Icc (A)	Disj	Tps (ms)
					min D1	max D1	min D2	max D2						
B	B	Z1b, Z2c, Z4	51	T	8000	28000	**10,5**	**29**	51	2	**7**	67	2	**19**
		Z1c, Z2c	88	P	13	6000	**9**	**23**	92	2	**7**	93	2	**8**
		Z1c, Z2a	116	A	13	19	**8,5**	**20,5**	107	2	**7**	107	2	**7**
		Z1a, Z2b	167	A	11	18	**8**	**20**	151	2	**6**	154	2	**6**
	C	Z1b, Z2c, Z4	51	T	8000	28000	**3100**	**8000**	51	2	**26**	67	2	**5736**
		Z1c, Z2b	103	A	12	20	**9**	**21**	96	1	**8**	108	2	**27**
		Z1a, Z2c	131	A	11	19	**8**	**20**	120	1	**7**	129	2	**7**
		Z2a, Z3a	221	A	**9**	**14**	**7**	**18**	189	**2**	**6**	171	**1&2 et 2**	**4**
	D	Z1c, Z2c	88	A	13	6000	**1400**	**2900**	82	1	**8**	95	2	**1814**
		Z1b, Z2b	126	A	**11**	**19**	**7,5**	**2000**	116	1	**7**	129	1	**30**
		Z1a, Z2a	202	A	**10**	**15**	**6,5**	**19,5**	180	1	**7**	173	1&2	**10**
C	B	Z1c, Z2c, Z4	47	T	9000	30000	**10,5**	**1000**	47	2	**8**	58	2	**23**
		Z1b, Z2c	104	T	2000	5000	**8,9**	**21**	97	2	**7**	111	2	**8**
		Z1a, Z2a	202	A	10	18	**7,5**	**19,5**	180	2	**6**	158	2	**4**
		Z1a	239	A	9	17	**7**	**18,5**	213	2	**6**	207	2	**9**
	D	Z1b, Z2a	146	P	11	2200	**7**	**20**	133	1	**10**	137	2	**824**
		Z1a, Z2a	202	A	10	18	**6,5**	**19,5**	180	1	**8**	171	2	**12**
		Z1a	239	A	9	17	**6**	**18**	202	1	**8**	223	2	**4**
D	B	Z1b, Z2a, Z4	60	T	5500	18000	**10**	**19**	59	2	**7**	73	2	**11**
		Z1a	239	P	9	1100	**7**	**18,5**	171	2	**6**	197	2	**4**
	C	Z1c, Z2a	116	T	1800	3900	**8,5**	**20,5**	107	2	**8**	115	2	**17**
		Z1a	239	P	9	1100	**7**	**18**	213	2	**6**	225	2	**4**
	D	Z1b, Z2c	104	P	2000	5000	**9**	**2300**	97	2	**631**	109	2	**1317**
		Z1a, Z2a	202	P	10	1300	**6,5**	**19,5**	180	2	**10**	173	2	**6**

Tab. 5 Comparatif entre calcul théorique, résultats de simulation et comportement mesuré pour le circuit à deux disjoncteurs.

T	Sélectivité Totale	P	Sélectivité Partielle	A	Aucune Sélectivité	Courant de Court-circuit	
	Différence entre simulation et mesure		Sélectivité simulée mais non respectée			**Temps à comparer**	

				Théorie								Simulation			Mesures		
D 1	D 3	D4	Impédances utilisées	Icc (A)	Sélect	Temps de déclenchement (ms)						Icc (A)	Disjoncteur qui coupe	**Tps (ms)**	Icc (A)	Disjoncteur qui coupe	**Tps (ms)**
						min D1	max D1	min D3	max D3	min D4	max D4						
B	B	B	Z1c, Z4, Z5	33	Part. (3/4)	30000	130000	11	19000	**12**	**22**	33	4	**7**	46	4	**16,9**
			Z1c, Z2c, Z4	42	Part. (3/4)	13000	50000	10,5	10000	**11**	**21**	40	4	**7**	54	4	**10,8**
			Z1c, Z2c	71	Part. (1/3/4)	13	9000	**10**	**27**	**9**	**20**	66	**4**	**7**	77	**3&4**	**6,8**
			Z2b, Z3b	101	NS	12	20	**9**	**22**	**8**	**19**	82	**4**	**7**	100	**3&4**	**4,4**
		C	Z1c, Z2c, Z4	42	NS (3/4)	13000	50000	**10,5**	**10000**	11	6500	40	3	**8**	55	3	**78,8**
			Z2a, Z4	61	NS (3/4)	6300	12000	10	29	**9,5**	**20**	58	3	**7**	77	4 et 3&4	**7,0**
			Z1b, Z2a	99	NS (1/3/4)	12	20	**9**	**22**	**7**	**17**	93	3	**7**	103	3&4	**7,9**
			Z1a, Z2a	117	NS (1/3/4)	11	19	**7**	**21**	**6**	**16**	110	3&4	**7**	122	3&4	**5,6**
	C	B	Z1c, Z2a, Z4	47	Totale	10000	30000	3100	8000	10,5	**20,5**	45	4	**7**	57	4	**6,8**
			Z2c, Z3c	77	Part. (1/3/4)	13	8000	9,5	3500	9	**19,5**	73	4	**7**	83	4	**7,7**
			Z1b, Z2a	99	NS (1/3/4)	12	20	9	29	8	**19**	93	4	**6**	105	4	**4,9**
			Z1a, Z2a	117	NS (1/3/4)	11	19	7,5	21	7	**14**	110	4	**6**	125	4	**3,2**
		C	Z1a, Z4	59	Part. (3/4)	6000	20000	2200	5500	10	**2400**	56	4	**8**	75	4	**11,0**
			Z1b, Z2b	91	Part. (1/3/4)	12	5000	9	2800	8	**18**	85	4	**7**	91	4	**5,5**
			Z2b, Z3b	101	NS (1/3/4)	12	20	9	29	7	**17**	94	4	**7**	104	4	**4,4**
			Z1a, Z2a	117	NS (1/3/4)	11	19	7,5	21	6	**16**	110	4	**7**	121	4	**5,9**
	D	B	Z2b, Z4	57	Totale	6000	20000	2700	5800	10	**20**	55	4	**7**	69	4	**5,9**
			Z1b, Z2a	99	NS (1/3) Part. (1/4)	12	20	29	2500	8	**19**	93	4	**6**	102	4	**6,8**
			Z1a, Z2a	117	NS (1/3/4)	11	19	8	2000	7	**14**	110	4	**6**	115	4	**4,3**
			Z2a, Z3a	121	NS (1/3/4)	11	19	8	2000	7	**14**	114	4	**6**	118	4	**3,3**
		C	Z2b, Z4	57	Part. (1/3)	6000	20000	2800	7000	10	**2400**	55	4	**8**	74	4	**9,2**
			Z1b, Z2a	99	NS (1/3) Part. (1/4)	12	20	29	2500	7	**17**	93	4	**7**	105	4	**4,7**
			Z1a	124	NS (1/3/4)	11	19	8	2000	6	**16**	118	4	**7**	142	4	**3,2**

Tab. 6 Comparatif entre calcul théorique, résultats de simulation et comportement mesuré pour le circuit à deux disjoncteurs.

T	Sélectivité Totale	P	Sélectivité Partielle	A	Aucune Sélectivité	Courant de Court-circuit	
	Différence entre simulation et mesure		Sélectivité simulée mais non respectée			**Temps à comparer**	

3.7 Analyse des différences.

Comme le montrent les tableaux précédents, des écarts peuvent survenir entre les valeurs théoriques, les simulations et les mesures effectuées aux cours des essais. Ces écarts se situent à trois niveaux : au niveau des courants de défauts, des temps de déclenchement des disjoncteurs et des sélectivités des montages. Ces grandeurs étant interdépendantes, un écart sur le courant de court-circuit va impliquer un écart sur le temps de déclenchement et peu aboutir à une différence de sélectivité. Rappelons que l'appréciation de la qualité de résultats ne peut être que qualitative et portera sur la cohérence du comportement. Les méthodes de calcul normalisées et les données constructeur ne donnent qu'une estimation assez large du comportement attendu.

3.7.1 Différence des courants de défaut.

A l'analyse du **Tab. 4**, Il paraît évident que les résultats obtenus ne peuvent pas être rigoureusement les mêmes lors des manipulations et des calculs théoriques et simulés. Cependant, les valeurs lors des simulations devraient approcher de manière assez précise les valeurs théoriques. On relève par exemple un courant de défaut de 176A théorique pour un courant de défaut de 166A simulé pour le montage à un disjoncteur 20A. Cet écart diminue ensuite pour les disjoncteurs à plus faible calibre. Il est possible que les méthodes de calcul utilisées par le simulateur ne soient pas les mêmes que celles utilisées dans ce document. Cette différence va donc également entraîner un écart sur le temps de déclenchement du disjoncteur.

Si on analyse finement les résultats, on constate qu'entre la simulation et la mesure, les différences de temps de déclenchement (écart **|(2)-(3)|**) sont généralement minimes (de l'ordre de la milliseconde) et correspondent aux écarts attendus. On trouve toutefois des écarts important pour certaines valeurs. Ils sont explicables en partie par une différence importante du courant de court circuit mesuré qui se répercute sur la valeur du temps de déclenchement. Ces différences sont d'autant plus flagrantes si on se trouve à la jonction entre les deux déclencheurs où la variation de la courbe de déclenchement est quasi verticale. Ici une variation faible du courant Icc peut se traduire par un écart très important en terme de temps de déclenchement. Il est important de relever que dans la majorité des cas la simulation donne une meilleure estimation du résultat que la méthode de calcul (écart **|(1)-(3)|-|(2)-(3)|)**.

De plus les temps de déclenchement sont donnés par les constructeurs à partir de mesures en courant continu. Pour des courants très élevés, on a des temps de déclenchement très rapides inférieurs à une période du signal (20 ms) comme on l'a décrit en **Fig. 25**. Or le simulateur détermine les temps de déclenchement par rapport à la valeur efficace du courant de court-circuit. La notion de valeur efficace est valable pour un signal sur un nombre fini de périodes mais pas sur une portion de cette période. Il conviendrait dans ces cas de résonner davantage par rapport à l'énergie apporté qu'en fonction de la valeur efficace du courant.

Il n'est pas forcement alarmant d'avoir un écart fort entre les valeurs de Icc issues de la simulation et celles mesurées. En effet, on peut difficilement prévoir la valeur de l'impédance qui provoque le court-circuit et donc la valeur de l'Icc. Le point le plus critique est que le disjoncteur ait bien le même comportement en simulation que dans la réalité.
C'est donc la sélectivité qui sera le plus important de respecter lors de la simulation.

3.7.2 Cohérence des résultats au niveau de la sélectivité.

Lors des prises de mesure, il est apparu que les disjoncteurs se comportaient comme prévu par le simulateur lorsque l'on se trouve en sélectivité totale. Cependant, dans certains cas où le choix du matériel ne permet pas de respecter la sélectivité (non-sélectivité ou sélectivité partielle) le simulateur annonce parfois des résultats différents de ceux mesurés, où ce n'est pas le disjoncteur attendu qui déclenche. Nous attendons du simulateur qu'il nous aide à prédire des cas de non-sélectivité. On distinguera donc les cas où il prévoit une non-sélectivité qui ne se vérifie pas dans la réalité des autres cas plus rares (3 sur 47 cas de test) où le simulateur présente un fonctionnement d'apparente sélectivité alors que dans la réalité on a une non-sélectivité.

3.7.2.1 Cas de non-sélectivité observés.

Les cas de non-sélectivité que nous avons relevés correspondent à des cas relativement réalistes. Ils peuvent apparaître quand un appareil aval et amont ont des calibres proches et des courbes de déclenchement mal adaptées. Dans ce cas, leurs caractéristiques Td=f(Icc) sont en partie confondues notamment sur la partie thermique de la courbe.
On peut constater que les cas de non-sélectivité interviennent pour des temps de déclenchements très faibles de l'ordre de la milliseconde. Dans tous les cas de non-sélectivité que nous avons relevés, les deux disjoncteurs déclenchent simultanément. On n'a jamais de cas ou seul l'appareil amont déclenche. On trouve également des cas de non-sélectivité flagrante quand on compare les courbes mais ou le fonctionnement réel semble correct. Dans ce cas, le simulateur doit faire apparaître le risque de non-sélectivité.

3.7.2.2 Déclenchement amont et aval simultané.

Nous avons relevé 5 cas sur 47 où dans la réalité les disjoncteurs amont et aval déclenchent simultanément alors que la simulation ne prévoit que le déclenchement du disjoncteur amont. Dans la simulation et en mesure, on constate dans les deux cas que l'on a un cas de non-sélectivité, le simulateur remplit donc correctement sa fonction. Toutefois Les modèles de disjoncteur utilisés lors des simulations ne permettent pas de déclenchement simultané de 2 disjoncteurs.
Dans le cas D1(B),D2(C,Z2a,Z3a) on peut constater que les temps minimum et maximum des différents appareils sont quasiment identiques. Le simulateur va donc déterminer un temps de déclenchement pour chacun des disjoncteurs qui sera assez proche mais pas confondu. Les deux déclenchements seront planifiés à des instants très proches (écart de l'ordre de 1ms).
Dès que le premier disjoncteur a déclenché, dans la simulation, le circuit est ouvert donc le deuxième disjoncteur n'est plus parcouru par le courant de défaut. Sans défaut, le déclenchement prévu est annulé et n'aura pas lieu. Or dans la réalité, il déclenche tout de même. Pour ces valeurs de Icc (Icc >> 10.In) c'est le déclencheur thermique qui opère. En raison de l'inertie thermique, même après l'ouverture du circuit, ce déclencheur continue à s'échauffer pendant un temps suffisant pour ouvrir le disjoncteur.

Il sera nécessaire de corriger le modèle afin d'intégrer ce phénomène. Cela pose surtout un problème dans les cas inverses ou la sélectivité semble respecté en simulation mais ne l'est pas en réalité ni en théorie.

3.7.2.3 Non-sélectivité mesurée mais non simulée.

Dans de rares cas, on mesure une non sélectivité avec déclenchement des composants amont et aval, alors que la simulation affiche un comportement ou seul le disjoncteur aval disjoncte.
Ce cas est plus problématique que le précédent car il laisse passer un risque et une non-conformité de l'installation. Cela représente 2 cas sur 47. Les causes en sont les mêmes que pour le cas précédent sauf qu'ici c'est le disjoncteur aval qui déclenche en premier suivi par le disjoncteur amont, un très court instant plus tard.

3.8 Conclusion.

Le simulateur a démontré un bon fonctionnement global. Les ordres de grandeur correspondent dans l'ensemble à ceux observées lors des simulations. La sélectivité est en général bien respectée, même si des différences ont pu être observées. En cas de résultat différent, la plupart du temps, le simulateur est plus sélectif que la réalité. Il fait apparaître des défauts potentiels ce qui apporte une bonne sécurité d'utilisation.
Le comportement global du simulateur est satisfaisant puisqu'en sélectivité totale, les résultats obtenus après simulation sont toujours ceux observés sur le banc. En sélectivité partielle ou non-sélectivitée, seuls quelques rares résultats diffèrent entre les simulations et la réalité.
Dans l'ensemble des cas, nous obtenons une meilleure précision que celle du calcul théorique qui est utilisé pour le dimensionnement des protections.

Il reste à prendre en considération l'inertie thermique des déclencheurs pour la simulation de déclenchement quasi instantanés. Il serait également utile de prendre en considération de valeur crête du courant de court-circuit pour les déclenchements rapides. De plus, le modèle devra prendre en compte plus précisément le comportement du disjoncteur dans la zone de jonction des deux déclencheurs ou une augmentation de 5% du court de court-circuit peut se traduire par une diminution de 99 % du temps de déclenchement.

Ces essais nous ont permis de vérifier le comportement des disjoncteurs en cas de court-circuit. Cela ne couvre qu'une partie de leur rôle de protection. Le banc d'essai évolutif (voir paragraphe **4**) a été réalisé afin de compléter cette validation.

4 Banc d'essai évolutif.

Suite à la première série de tests effectués sur un banc de sélectivité "Très Basse Tension", un deuxième banc de mesure a été utilisé dans le cadre de la validation du module de simulation de circuits électrotechniques. Nous avons conçu ce banc d'essai évolutif pour valider le comportement de différents types de matériel tels que les disjoncteurs, contacteur, câbles. Les essais réalisés ici sont complémentaires de ceux faits sur le banc de sélectivité car il couvre d'autres plages de fonctionnement des disjoncteurs et il inclut d'autres types de matériel.
Contrairement aux tests effectués sur le premier banc, ici la sélectivité n'est pas traitée. C'est plutôt le temps de déclenchement dans le cadre d'un fonctionnement simple, aussi bien en courant alternatif qu'en continu, qui sera observé. Les nombreuses possibilités de tests offertes par ce banc permettront également d'étudier l'influence d'une variation en forme d'échelon du courant sur le temps de déclenchement d'un disjoncteur, le temps de réaction d'un relais ou encore le comportement d'un câble de grande longueur et la chute de tension qui apparaît à ses bornes en fonction de son mode de pose.

4.1 Description du banc.

Ce banc à été dimensionné pour être complémentaire du banc d'essai de sélectivité. Ce dernier permettait d'étudier des courts-circuits et de travailler avec des courants allant jusqu'à 200A mais pas de travailler sur les zones de surcharges des disjoncteurs (In > Icc > 10.In) ni de traiter d'autres appareils. Le banc d'essai évolutif permet de travailler avec des courants allant jusqu'à 40A sous 230V ou 240 V alternatif ou en courant continu.
Nous utilisons des disjoncteurs C60N Merlin-Gérin de calibres très faibles, entre 0,5 et 2 A afin de pouvoir couvrir suffisamment la courbe de déclenchement en travaillant avec des courants inférieurs à 30 A. Ainsi, nous avons pu dimensionner notre banc de manière relativement réaliste (section de câbles, puissance des sources et du transformateur, etc) et adaptée à nos possibilités.

Ce banc est divisé en 2 maquettes séparées : un "mini banc mobile" et un "banc d'essai principal".

4.1.1 Mini banc mobile.

Le mini banc ne sert que de support à un disjoncteur. Il ne comporte donc que 2 bornes d'entrée, 3 bornes de sortie, un disjoncteur et une résistance de charge variable branchée en série. Les 3 bornes de sortie correspondent à la masse, à la valeur maximale de la résistance et à la valeur intermédiaire, variable, de la résistance. Cette dernière borne est appelée "curseur".
Ce mini banc sert à caractériser les disjoncteurs en tension continue afin de comparer les résultats avec les données constructeurs et éventuellement d'affiner les courbes utilisées dans le simulateur.

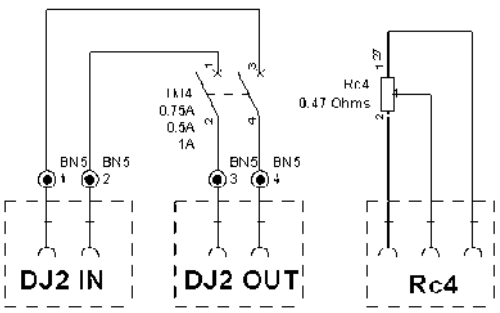
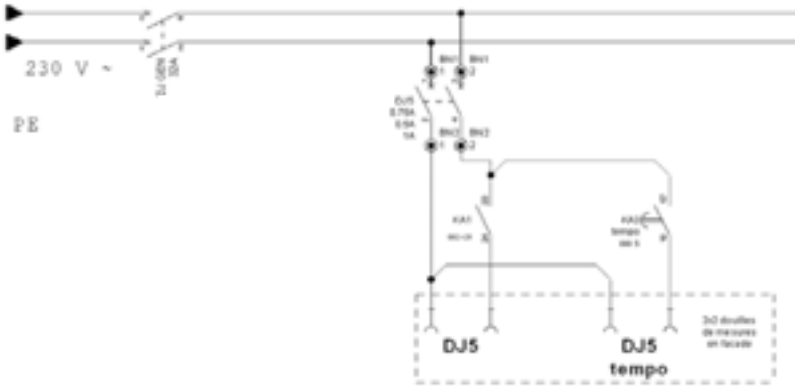


Fig. 31 Schéma du"mini banc mobile"

Ses dimensions permettent de le transporter facilement et il est très simple d'utilisation. La conception extrêmement simple du mini banc permet des manipulations basiques. L'objectif est de caractériser les disjoncteurs en faisant varier le courant circulant dans le circuit. Ces tests s'effectuent en courant continu grâce à une source de courant programmable. Ils vont permettre de vérifier les courbes de temps de déclenchement fournies par le constructeur. Une fois ces courbes retracées, il est possible de comparer les résultats avec le simulateur et éventuellement de modifier les données saisies pour le configurer.

4.1.2 Banc d'essai principal.

Cette deuxième maquette, plus complexe, sert à vérifier le bon fonctionnement des disjoncteurs en régime sinusoïdal, que ce soit en 24V ou en 230V. Il est donc divisé en 2 parties, une première qui est alimentée en 230V et une seconde en 24V.



Comme le montre la **Fig. 32** ci-contre, la partie alimentée en 230V est contrôlée par deux relais, *KA1* et *KA3*. Ces relais sont commandés par le même interrupteur mais *KA3* dispose d'une temporisation réglable qui permet d'ajouter un retard à la fermeture.

Fig. 32 Banc d'essai principal, partie Basse Tension 230V

Les sorties de banc sont reliées à un boîtier dans lequel des ampoules de 100W ont été branchées en parallèle. Ces ampoules sont réparties en 2 groupes de 3 ampoules chacun, ce qui permet de brancher une partie sur la sortie notée *DJ5* et l'autre sur la sortie *DJ5 temporisation* qui, comme son nom l'indique est la sortie retardée par *KA3*.
Grâce à ces ampoules, il sera possible d'observer le comportement des disjoncteurs suivant le nombre d'ampoules connectées et donc du courant circulant dans le circuit.

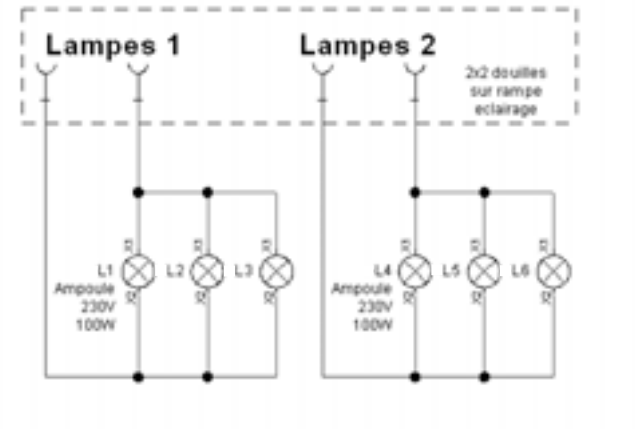


Fig. 33 Boîtier contenant les ampoules

La partie basse tension est alimentée en 230V et est destinée à être reliée à des ampoules branchées en parallèle. Deux sorties sont disponibles qui sont commandées par deux relais dont un dispose d'un retard programmable. Il est donc possible d'observer le comportement des disjoncteurs en fonction du nombre d'ampoules connectées, donc du courant circulant dans le circuit.
Le retardateur peut permettre d'effectuer des tests sur l'influence d'une seconde surcharge apparaissant brusquement alors que le circuit subissait déjà une première surcharge.

La deuxième partie du banc principal (**Fig. 34**) est alimentée en 24V via un transformateur 230V/24V de 1000 kVA. Il dispose de 4 douilles de mesures en façade. Il est possible d'y connecter, sur la sortie DJ1, le câble de 100m afin de relever la chute de tension se produisant selon la manière dont est disposé le câble. Sur la sortie DJ2, il est possible de connecter un rhéostat afin de faire varier la résistance du circuit et donc de modifier la valeur du courant traversant le disjoncteur. On dispose également de résistances variables montées sur la maquette qui s'ajoutent au rhéostat. On peut sélectionner selon le besoin une résistance de 22Ω supportant une puissance de 100W ou deux résistances parallèles de 4,7Ω chacune et supportant cette fois une puissance de 280W chacune soit une résistance globale de 2.35Ω pour 560W (courant pouvant monter jusqu'à 15.4A).

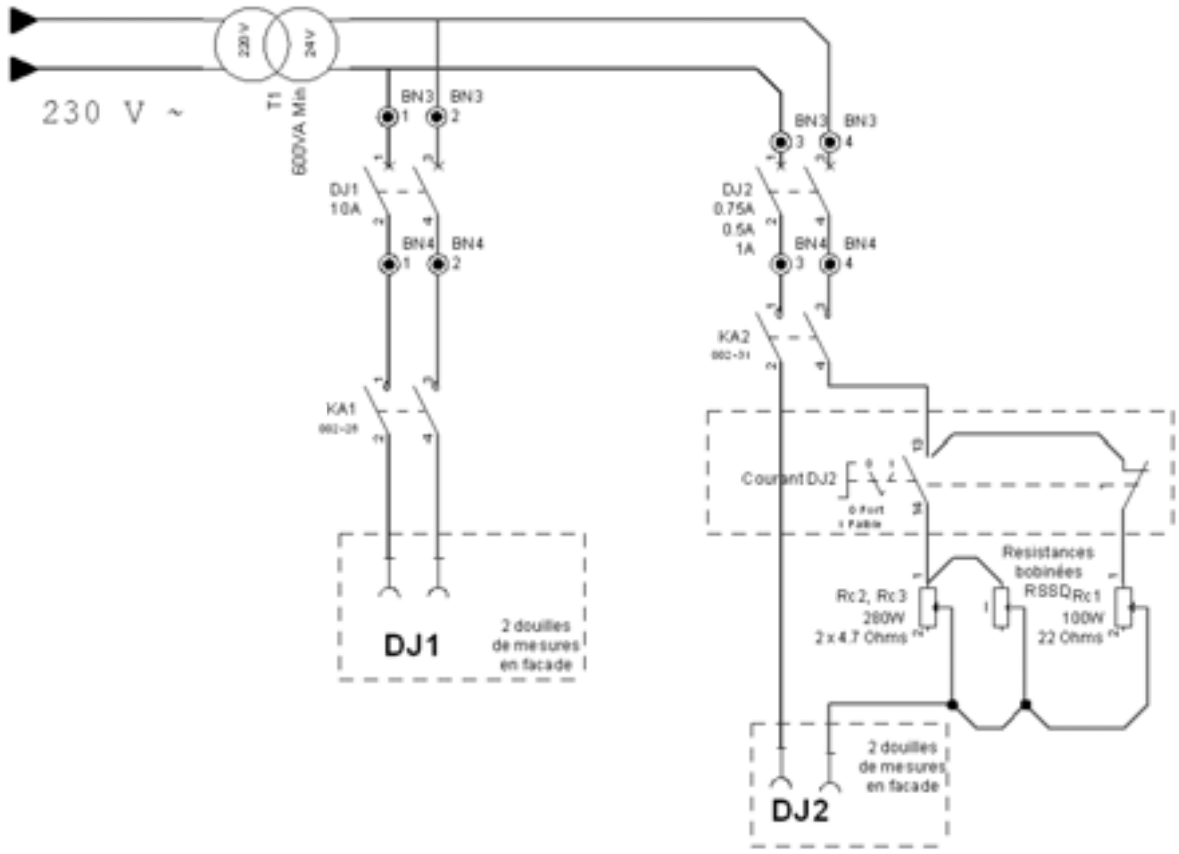


Fig. 34 Banc principal, partie Très Basse Tension 24V

La partie Très Basse Tension du banc principal est alimentée en 24V sinusoïdal 50Hz via un transformateur 230V/24V. La présence des résistances variables connectées à un rhéostat permet de régler précisément la valeur du courant désiré dans le circuit. Il est donc possible de vérifier les résultats obtenus précédemment en continu avec le mini banc. Ces tests feront apparaître les différences pouvant exister entre les courbes constructeur fournies pour une tension continue et les courbes obtenues en courant alternatif (domaine le plus fréquent d'utilisation des disjoncteurs).
Cette partie TBT possède également une sortie protégée par un disjoncteur 10A qui peut être utilisée pour connecter un câble de longue distance. Il est ainsi possible de mesurer les effets de déploiement du câble sur la chute de tension apparaissant à ses bornes.

4.2 Caractéristiques du banc.

Afin de pouvoir renseigner au mieux les modèles des composants, il est indispensable de relever certaines caractéristiques de ces composants. Il s'agit notamment leur impédance, nécessaire pour l'évaluation des valeurs des courants de défaut.

4.2.1 Impédance des disjoncteurs.

Dans les premiers essais, nous avons noté l'importance de la prise en compte des impédances des disjoncteurs. Afin de les vérifier, ceux-ci ont été montés en série avec une résistance de charge purement résistive de 5Ω et un GBF fonctionnant à 50Hz comme le montre le schéma **Fig. 35** ci-contre.
Le résultat obtenu est présenté sur la **Fig. 36** ci-dessous. La courbe la plus grande représente la tension aux bornes du disjoncteur et correspond à la voie 2 (CH2). L'autre est la tension fournie par le GBF (voie 1-CH1). Les deux signaux sont parfaitement en phase, ce qui implique que les disjoncteurs sont parfaitement résistifs.

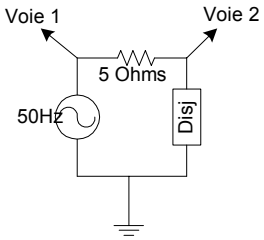


Fig. 35 Mesure d'impédance des disjoncteurs

Ce montage est celui d'un pont diviseur de tension donc la tension de sortie vaut :

$$Vout = Vin \cdot \frac{Z_{Disj}}{5 + Z_{Disj}} \qquad (1.2.1.1)$$

soit pour l'impédance du disjoncteur :

$$Z_{Dis} = 5 \cdot \frac{Vout}{Vin - Vout} \qquad (1.2.1.2)$$

Pour le disjoncteur 0,5A courbe C dont le relevé à l'oscilloscope numérique est montré ci-dessous, on mesure une valeur Vin de 4.04V et Vout de 2.56V.

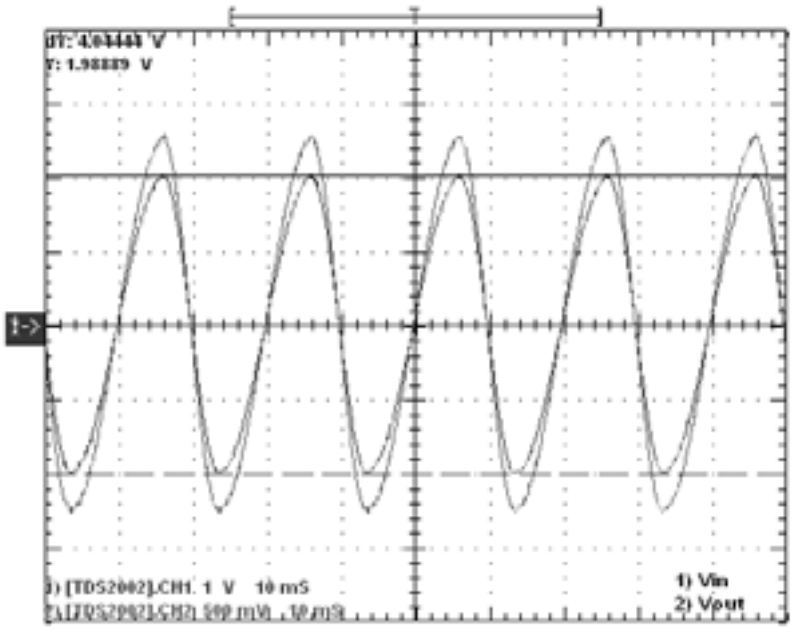


Fig. 36 Tension aux bornes du disjoncteur 0,5A courbe C en fonction de la tension d'alimentation

D'après l'équation (1.2.1.2), on obtient :

$$Z_{Disj} = 5 \cdot \frac{2.56}{4.04 - 2.56} = 8.64\Omega \qquad (1.2.1.3)$$

La même méthode est utilisée pour l'ensemble des disjoncteurs, ce qui permet d'obtenir le tableau ci-dessous :

Disjoncteur (Courbe Calibre)	C0,5	D0,5	C0,75	C1	D1	B10
Impédance par pôle (Ω)	8,64	8,42	4,9	2,2	2,2	0,04

Tab. 7 Impédance des disjoncteurs en fonction du calibre

Dans le simulateur, l'impédance des pôles est négligée, ce qui peut parfois poser des problèmes si le calibre est faible. Afin de modéliser cette impédance, il est nécessaire de trouver une relation liant le calibre à son impédance. Le graphe ci-dessous montre l'évolution de l'impédance en fonction du calibre.

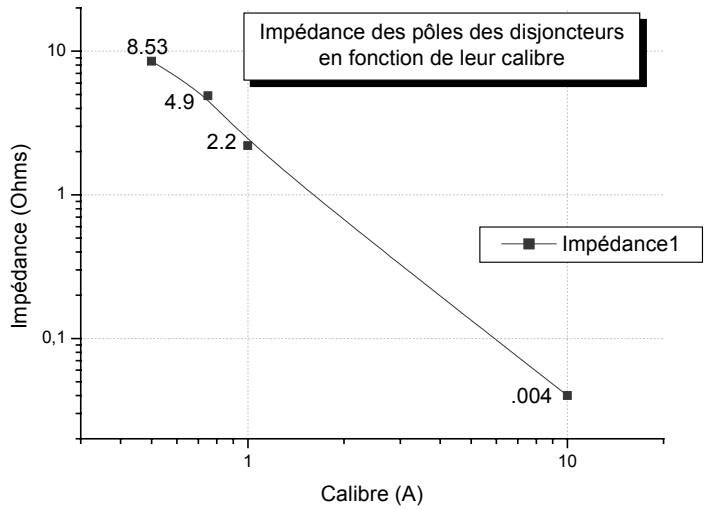


Fig. 37 Evolution de l'impédance des pôles d'un disjoncteur en fonction de leur calibre

Il apparaît une certaine linéarité lorsque les échelles sont logarithmiques. Cela nous permet alors d'en déduire l'équation de la courbe de tendance :

$$Y = -1.8026 \cdot X + 0.4001 \qquad (1.2.1.3)$$

avec Y représentant le logarithme base 10 de l'impédance et X le logarithme base 10 du calibre.

L'équation globale est alors :

$$LOG(y) = -1.8026 \cdot LOG(x) + 0.4001 \qquad (1.2.1.4)$$

et donc :

$$\boxed{y = 10^{(-1.8026 \cdot LOG(x) + 0.4001)}} \qquad (1.2.1.5)$$

4.2.2 Impédance des ampoules.

Les ampoules utilisées dans le boîtier sont des ampoules à baïonnette de puissance 100W. Il est intéressant de remarquer qu'un appel de courant se produit au moment de la mise sous tension comme le montre la figure suivante.

Afin de mesurer leur impédance, une ampoule a été branchée directement sur le 230V sans disjoncteur ni relais afin que rien ne vienne perturber la mesure. Il apparaît un courant efficace de 451mA. L'impédance de l'ampoule est donc de $R = \frac{U}{I} = \frac{230}{451 \cdot 10^{-3}} = 510\Omega$. On peut vérifier ce résultat par le calcul de la puissance nominale P_N=100W. En effet, le calcul $P = \frac{U^2}{I}$ donne P=103,7W. La valeur de l'impédance trouvée est donc correcte.

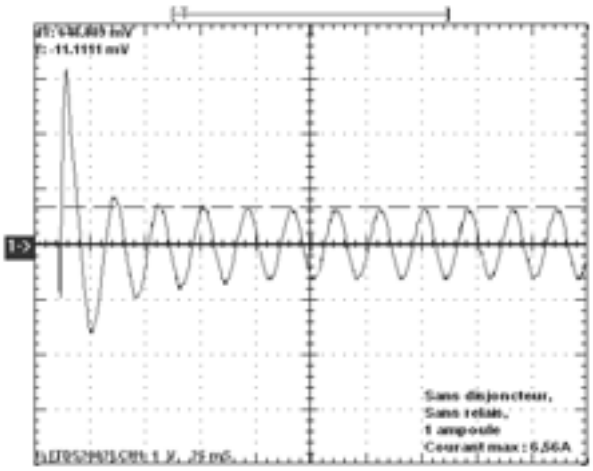


Fig. 38 Courant dans une ampoule de 100W alimentée en 230V

4.2.3 Impédance des résistances variables.

On dispose de 2 résistances variables de 4,7Ω montées en parallèle, d'une résistance variable de 22Ω et d'un rhéostat de 5,6Ω. Afin de vérifier ces mesures, un multimètre a été utilisé en mode ohmmètre. Les résultats obtenus sont résumés dans les tableaux suivants.

	Résistance Rhéostat (Ω)	Résistance RC1 (Ω)	Résistance RC2//RC3 (Ω)
Curseur à 0 %	0	0,1	0,02
Curseur à 50 %	2,77	11,47	1,18
Curseur à 100 %	5,5	22,82	2,35

Tab. 8 Impédances des résistances variables

4.3 Mesure du déclenchement des disjoncteurs.

Tout d'abord nous allons vérifier la conformité du déclenchement des disjoncteurs par rapport aux courbes constructeur et aux résultats de la simulation. Contrairement aux essais réalisé sur le banc de sélectivité, nous allons mesurer pour les différents disjoncteurs disponibles différents points sur la courbe de déclenchement et pas uniquement une valeur de courant de court-circuit. Les surintensités que nous provoquons peuvent être des surcharges entre 3 et 10 fois le courant nominal In, prises en compte par le déclencheur magnétique. Cela peut être des court-circuit pour des valeurs supérieures à 10.In qui seront traitées par le déclencheur thermique

4.3.1 Temps de déclenchement en Basse Tension (230 V).

Dans un premier temps, nous mesurons le comportement des disjoncteurs dans des conditions nominales d'utilisation, sous 230 V alternatif. La charge du circuit est constitué par un ensemble de 6 ampoules de 100W consommant environ 0,43 A chacune. Nous faisons varier la charge en modulant le nombre d'ampoules.

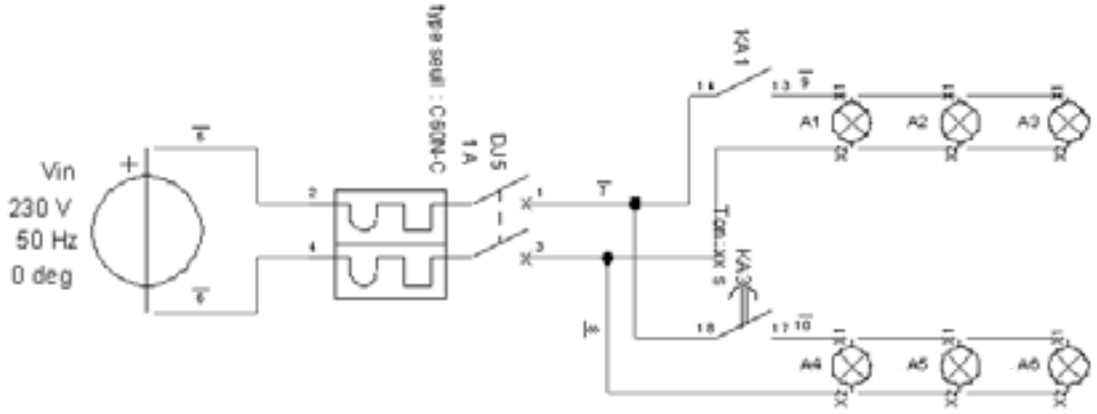


Fig. 39 Déclenchement sous 230 V chargé par des lampes

4.3.1.1 Utilisation d'un retard.

La première mesure effectuée a fait apparaître un bref appel de courant à la mise sous tension. Cet appel n'est pas négligeable. Pour 6 ampoules en sortie, il va provoquer le déclenchement du disjoncteur dans la plupart des cas.

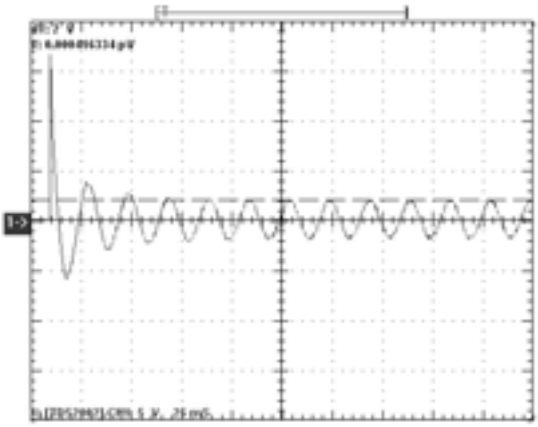


Fig. 40 Appel de courant à la mise sous tension des ampoules (4 ampoules)

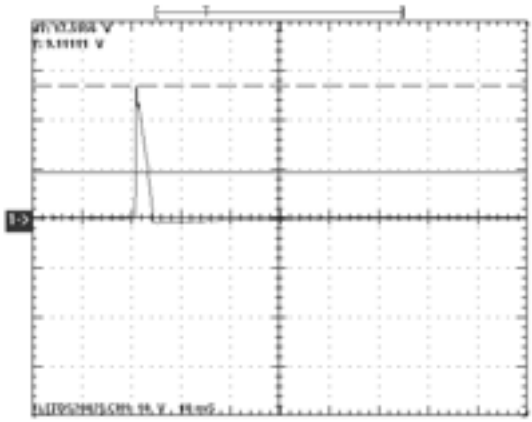


Fig. 41 Déclenchement immédiat du disjoncteur à cause de l'appel de courant (6 ampoules)

Afin de limiter cet appel, les ampoules vont être réparties sur les deux sorties disponibles et le deuxième groupe ne sera branché que quelques millisecondes après le premier comme le montre la **Fig. 42**.
L'appel de courant est alors réparti en 2 fois et évite d'atteindre une valeur de courant sollicitant le déclencheur magnétique du disjoncteur.

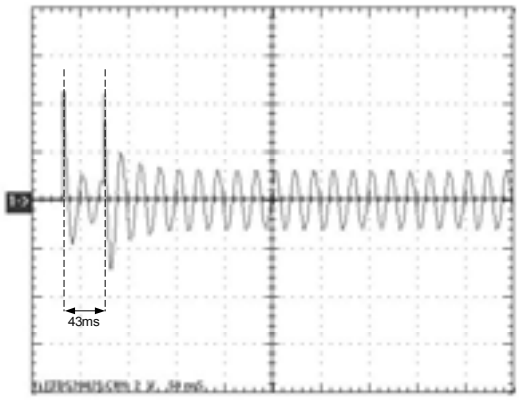


Fig. 42 Utilisation du retard pour limiter l'amplitude de l'appel de courant

4.3.1.2 Résultats obtenus.

Les mesures ont donc été effectuées selon la méthode décrite précédemment avec le retard de 43ms. Etant donné la faible valeur de ce retard, on peut considérer qu'il est négligeable par rapport à la seconde. Or, étant donné les valeurs de courant qui doivent circuler dans le circuit, nous sommes toujours en surcharge et jamais en court-circuit, ce qui implique des temps de déclenchement supérieurs à la seconde.

Les résultats obtenus par la mesure sont détaillés sur un tableau en **Annexe 16**. Ils sont repris plus visuellement dans les **Fig. 43** à **Fig. 45**.

Les courbes ci-contre montrent les différences entre les valeurs mesurées et les valeurs simulées. A part pour le disjoncteur C60N-C 0,5A qui se déclenche à cause de l'appel de courant important à la mise sous tension, les valeurs simulées sont proches des valeurs mesurées.

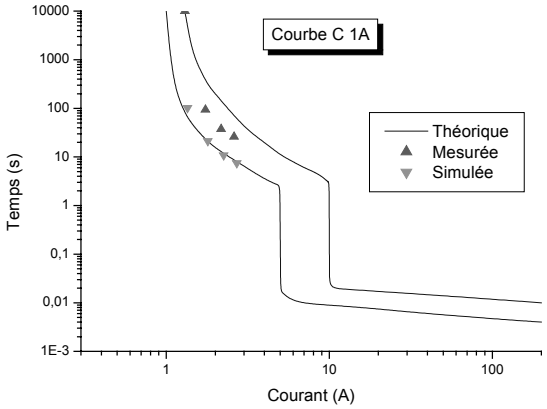


Fig. 43 Comparaison des temps de déclenchement mesurés et simulés d'un disjoncteur C1 en 230V

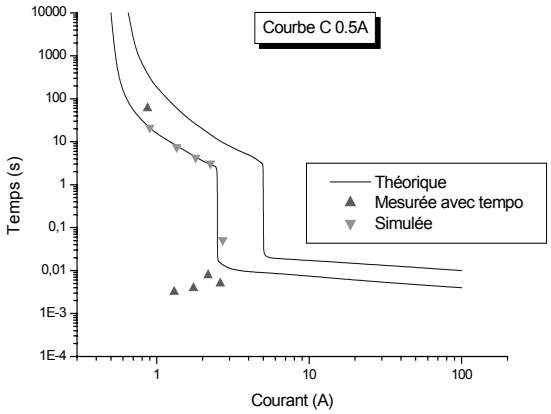


Fig. 44 Comparaison des temps de déclenchement mesurés et simulés d'un disjoncteur C0,5 en 230V

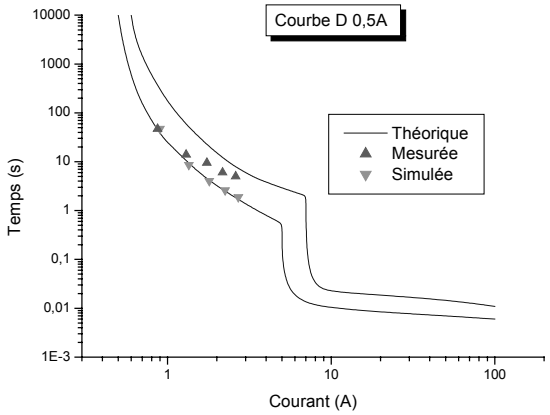


Fig. 45 Comparaison des temps de déclenchement mesurés et simulés d'un disjoncteur D0,5 en 230V

4.3.1.3 Corrections à apporter au simulateur.

Il serait sans doute intéressant de modifier les courbes prises en compte pour le déclenchement du disjoncteur. Au cours des tests effectués sur le banc de sélectivité TBT, il était apparu que les temps de déclenchement se situaient plus vers la limite basse. Cependant, les disjoncteurs avaient beaucoup été utilisés contrairement à ceux installés sur le deuxième banc qui sont neufs.
Les modifications à apporter sur les courbes simulateurs seraient donc :

- Faire une moyenne des temps de déclenchement haut et bas sur la partie thermique du déclencheur.
- Prendre en compte l'appel de courant qui se fait au moment de la mise sous tension afin de prévoir le déclenchement des disjoncteurs.

4.3.2 Temps de déclenchement en Très Basse Tension (24V).

Avec l'essai précédent, nous avons pu vérifier dans des conditions nominales le fonctionnement du disjoncteur. Ces conditions de fonctionnement ne nous ont pas permis de couvrir la totalité de la courbe. Nous allons donc compléter cet essai par des mesures à tension réduite.
Cette partie du banc, relié à un rhéostat permet de mesurer les temps de déclenchement des disjoncteurs en fonction de la valeur des impédances, donc de la valeur du courant.

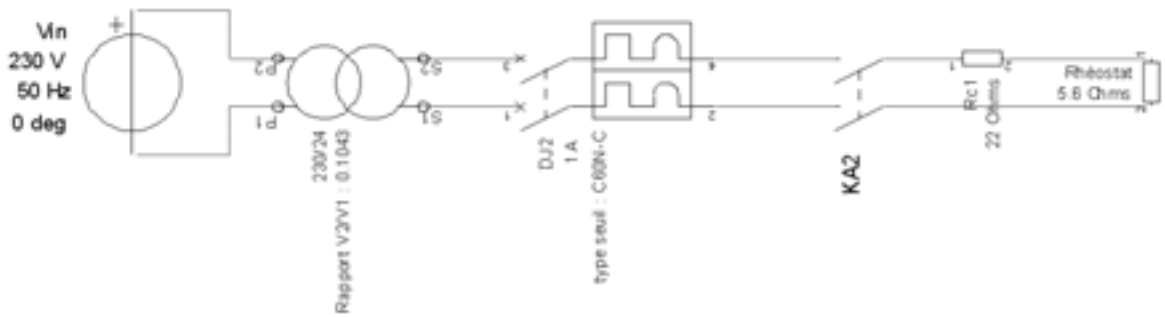


Fig. 46 Schéma de la mesure du temps de déclenchement en TBT

Les différentes configurations de résistances possibles sont soit de connecter la résistance variable 100W de 22,8Ω avec le rhéostat de 5,5Ω soit de connecter les résistances variables 280W branchées en parallèle et ayant une valeur maximale de 2,35Ω avec le rhéostat. La valeur maximale de la résistance dans le circuit que l'on peut obtenir est donc 22,8+5,5=28.3Ω. Cela implique une valeur minimale de courant de 848mA et une valeur maximale de courant très élevée pouvant atteindre 20A si l'on considère que le disjoncteur a une résistance négligeable.

Cependant, il est apparu que l'impédance interne des disjoncteurs n'est pas négligeable en raison de leur calibre faible. En effet, elle est d'environ 2Ω par pôle pour les disjoncteurs à tension de seuil 1A et 8,5Ω par pôle pour les disjoncteurs à tension de seuil 0,5A. Le courant maximal est donc $I_{\max} = \frac{U}{R_{\min}} = \frac{24}{2 \cdot 2.2} = 5.45A$ pour un disjoncteur 1A. Cette valeur est insuffisante pour pouvoir balayer toute la courbe de fonctionnement.
L'idée de n'utiliser qu'un seul pôle a été avancée, ce qui permet de monter jusqu'à 10,9A. Cette valeur n'est toujours pas suffisante pour balayer toute la courbe de fonctionnement.
La solution retenue a donc été de modifier le banc d'essai pour pouvoir remplacer les 24V alimentant cette partie par les 230V d'origine. Il est alors possible d'obtenir des courants plus importants.
Si l'on suppose l'utilisation de la résistance variable de 22Ω avec le rhéostat réglé au maximum, on obtient :

$$I_{\min} = \frac{U}{Rc1 + R_{rhéostat} + R_{Disjonteur}} = \frac{230}{22.8 + 5.5 + 2*2.2} = 7.03A \qquad (4.2.1.1)$$

La puissance dissipée par la résistance de 22Ω est donc :

$$P_{Rc1} = Rc1 * I_{\min}{}^2 = 22.8 * 7.03^2 = 1128W \qquad (4.2.1.2)$$

La puissance dissipée par cette résistance est donc bien trop élevée par rapport aux 100W pour laquelle elle est prévue et il apparaît qu'elle n'est pas utilisable dans cette configuration.
Si maintenant on utilise les deux résistances de 280W montées en parallèle, le courant minimum devient :

$$I_{\min} = \frac{U}{Rc2 // Rc3 + R_{rhéostat} + R_{Disjonteur}} = \frac{230}{2.35 + 5.5 + 2*2.2} = 18.75A \qquad (4.2.1.3)$$

La puissance dissipée par ces résistances est donc :

$$P_{Rc1} = Rc2 // Rc3 * I_{\min}{}^2 = 2.35 * 18.75^2 = 826W \qquad (4.2.1.4)$$

Même si la puissance dissipée est toujours supérieure à celle prévue qui est de 560W, celle-ci est acceptable à condition que le déclenchement se fasse rapidement. A 20A, le disjoncteur réagit en quelques millisecondes.

Nous avons décidé de relever une première série de mesures en 24V afin d'obtenir le temps de déclenchement pour les valeurs basses du courant (0 à 4,45A) puis de relever une deuxième série de mesure en 230V pour les valeurs hautes de courant (18,75A à 40A).
Il est regrettable de ne pouvoir relever la plage de valeur pour laquelle le disjoncteur passe du déclencheur thermique au déclencheur magnétique mais aucune solution n'a été trouvée avec le matériel disponible.

4.3.3 Résultats obtenus.

Le tableau suivant représente les temps de déclenchement relevés en fonction du courant circulant dans le circuit. Ce courant a été réglé en modifiant la valeur des résistances variables et en particulier du rhéostat.

	24V							230V		
Courant (A)	1,51	2,02	3,02	4,05	4,55	5	5,2	18,42	30	40
Temps de déclenchement (s)	455	56,45	20,02	11,66	9,66	8,47	8,3	0,00775	0,00462	0,00373

Les résultats obtenus sont parfaitement dans l'intervalle spécifié par le constructeur pour les faibles courants. Par contre, il apparaît que le disjoncteur déclenche plus vite pour les forts courants.

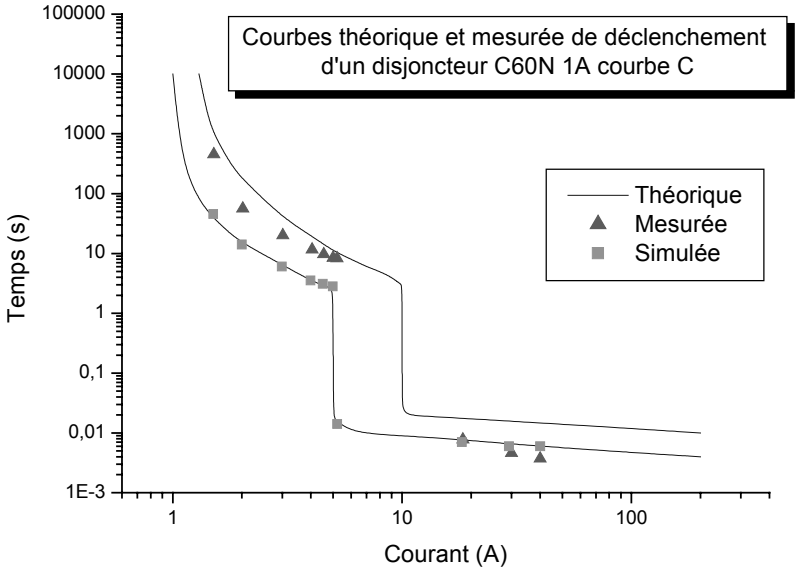


Fig. 47 Courbe de déclenchement mesurée et simulée d'un disjoncteur C60N-C 1A

4.3.4 Corrections à apporter aux modèles.

Les résultats précédents confirment ceux obtenus aux étapes précédentes et réaffirment donc qu'il serait sans doute intéressant de modifier les temps de déclenchement pris en compte dans le simulateur dans la partie déclencheur magnétique.
Les modifications à apporter sur les courbes simulateurs seraient donc :

- Faire une moyenne des temps de déclenchement haut et bas sur la partie "déclencheur magnétique" (1 à 10.In).
- Garder la courbe basse pour la partie "déclencheur thermique".

4.3.5 Influence d'un échelon de courant.

Ce section a pour but d'étudier l'influence d'un échelon de courant sur le temps de déclenchement des disjoncteurs. Ainsi, le circuit est laissé en fonctionnement pendant une durée pouvant aller jusqu'à 40s avant que le courant y circulant ne soit doublé.
La partie principale basse tension du banc est utilisée.

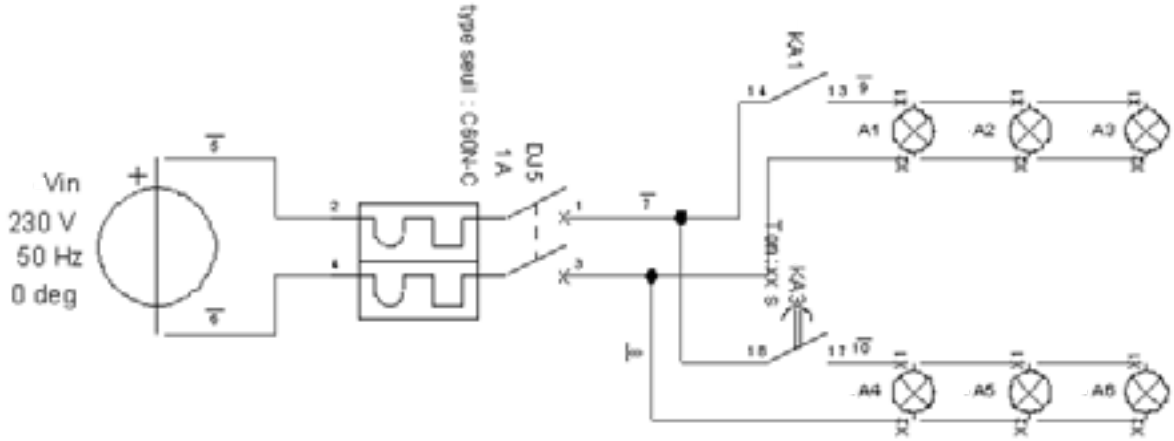


Fig. 48 Montage pour la mesure de l'effet d'un échelon de courant

Dans un premier temps, le réglage de la durée de la temporisation est réglé sur le relais KA3. Ensuite, les disjoncteurs C60N 1A courbe C et 0,5A courbe D sont testés alternativement. Les sortie DJ1 et DJ3 commandées respectivement par les relais KA1 et KA3 sont reliées en sortie à 2 ampoules chacune. Le temps de déclenchement est mesuré.
Après ces tests, une ampoule est rajoutée à chaque sortie.

4.3.5.1 Résultats obtenus

Le tableau suivant résume les résultats obtenus lors des mesures :

4 ampoules C60N 0,5A Courbe D		4 ampoules C60N 01A Courbe C		6 ampoules C60N 0,5A Courbe D		6 ampoules C60N 1A Courbe C	
Retard (s)	Déclenchement (s)	Retard (s)	Déclenchement (s)	Retard (s)	Déclenchement (s)	Retard (s)	Déclenchement (s)
0	9	0	109,45	0	5	0	26
5,2	12,4	5	113,28	1,1	5,15	3,38	30,05
10,77	16,6	10,5	117,16	2,5	5,95	11,16	36,38
20,78	24,78	19,94	128,3	5,1	8,06	20,4	42,72
30,44	33,11	31,6	139,4	7,55	9,84	30,95	50,23
41,5	43,16	42,1	146,6	10,77	12,31	39,17	58,66
47	47			14	14		

L'évolution du temps de déclenchement semble linéaire, ce qui est confirmé par les courbes présentées ci-après.

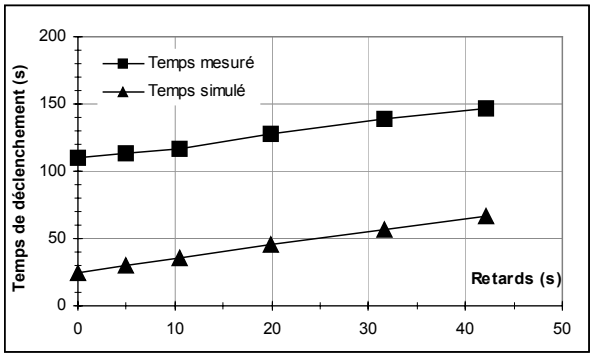


Fig. 49 Déclenchement D0,5 après temporisation (4 ampoules)

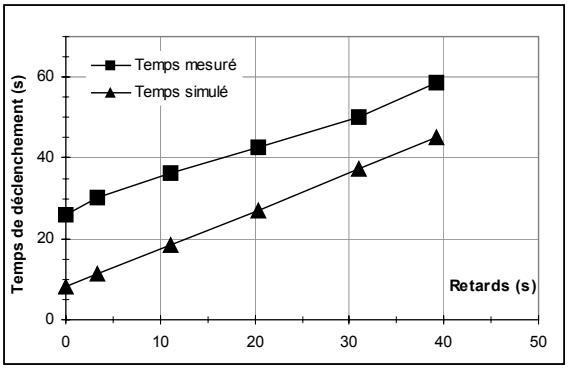


Fig. 50 Déclenchement C1 après temporisation (4 ampoules)

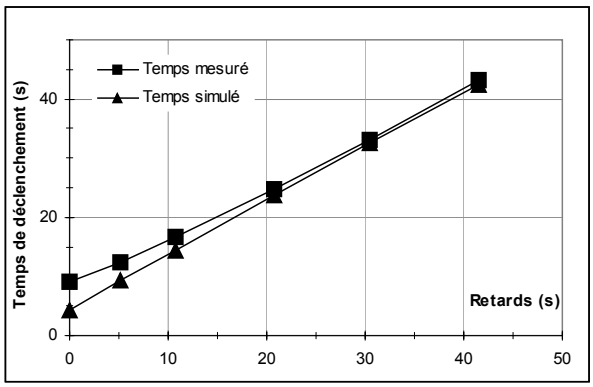


Fig. 51 Déclenchement D0,5 après temporisation (6 ampoules)

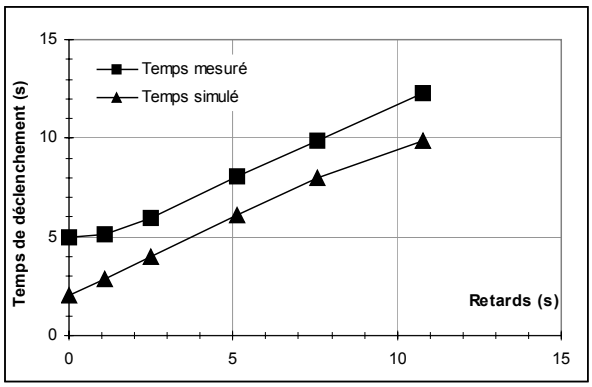


Fig. 52 Déclenchement C1 après temporisation (6 ampoules)

Nous obtenons dans tous les cas des caractéristiques quasi-linéaires dont les pentes sont très proches pour la simulation et la mesure. La prise en compte des échelons de tension par les modèles semble correcte. Pour certaines caractéristiques, la courbe simulée est décalée vers le bas. Cet écart correspond au fait que le simulateur utilise la courbe minimum donné par les constructeur au lieu d'utiliser une moyenne des courbes minimum et maximum. Les temps de déclenchement qui sont en jeu ici nous confirment que c'est le déclencheur magnétique qui opère, pour lequel ce problème est connu.

4.3.6 Mesure du temps de déclenchement en courant continu.

Nous avons à notre disposition un générateur programmable TTi TSX1820P de forte puissance (18V, 20A). Il alimente les disjoncteurs directement en courant. Il permet de fixer plus facilement la valeur de courant de défaut.
Nous souhaitons ici valider le fait que les caractéristiques sont équivalentes en courant continu et alternatif.

Il est aisé de mesurer l'impédance des pôles des disjoncteurs en réglant le générateur en mode générateur de courant et en fournissant 1A par exemple. Pour le disjoncteur C60N de tension de seuil 1A, courbe C, la tension à ses bornes est de 2.24V. Sa résistance est donc de 2,24Ω. On mesure de même la résistance des disjoncteurs de tension de seuil 0,75A et 0,5A. On obtient respectivement 5.08Ω et 8,86Ω. Ces valeurs correspondent à peu près à celles

trouvées au cours de la mesure en alternatif et présentées dans la partie 1.2.1, ce qui confirme bien l'impédance purement résistive des disjoncteurs.

La valeur de résistance de charge prévue est de 0,47Ω, ce qui implique une tension de 14,1V à 30A. Si le circuit n'avait pas eu de résistance interne, il aurait été possible de balayer toute la courbe. Malheureusement, la valeur importante de la résistance interne des disjoncteurs va nous empêcher d'effectuer les tests sur toute sa plage de fonctionnement. En effet, si l'on prend l'exemple du disjoncteur C60N-1A courbe C, sa résistance est d'environ 2,2Ω par pôle et le courant de sortie va donc être très limité comme le montre le calcul suivant :

$$I_{MAX} = \frac{V_{MAXgenerateur}}{R_{Min}} = \frac{18}{2 \cdot 2.2} = 4.09A \qquad (2.2.1)$$

Les valeurs de courant sont encore plus faibles pour les disjoncteurs à tension de seuil inférieure. Il y a donc peu d'intérêt à mesurer les temps de déclenchement de ces disjoncteurs. Il est éventuellement possible de n'utiliser qu'un seul pôle du disjoncteur, ce qui nous permet alors de doubler la plage de tests jusqu'à 8,18A. Cette valeur est toujours insuffisante mais certains résultats intéressants peuvent en ressortir.

4.3.6.1 Mesures effectuées

Les mesures n'ont été faites que sur le disjoncteur C60N-C de seuil 1A car c'est le disjoncteur disponible qui nous permet de relever le maximum de points intéressants. La valeur de courant maximale est de 7,3A. Le tableau ci-dessous présente les résultats obtenus.

Courant (A)	1,5	2	3	4	5	6	7	7,3
Temps de déclenchement (ms)	460	52,1	18,7	11,2	8,03	6,27	5,2	5,25

4.3.6.2 Comparaisons par rapport aux mesures effectuées en alternatif

Afin de comparer ces mesures à celles relevées en alternatif et aux valeurs théoriques, toutes les données ont été insérées dans la **Fig. 53** de la page suivante.

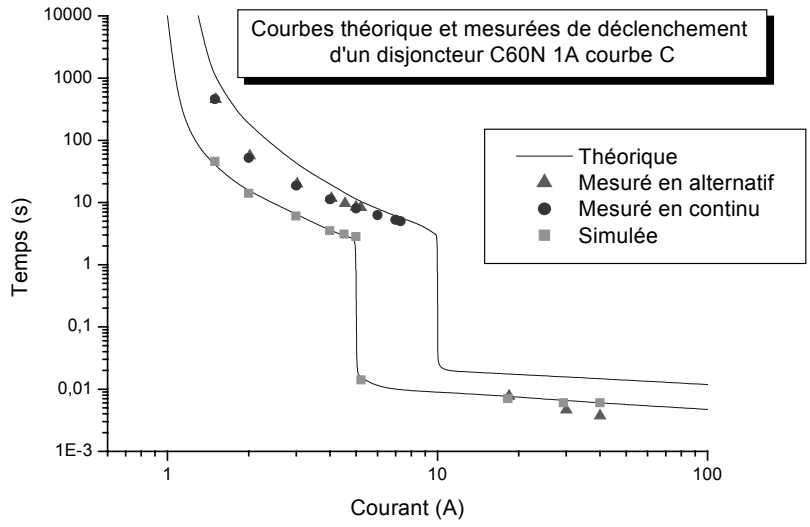


Fig. 53 Courbes théorique et mesurées de déclenchement d'un disjoncteur C60N-C 1A

Les valeurs relevées en continu coïncident parfaitement avec celles relevées en alternatif. Ce résultat est prévisible étant donné que la valeur prise en compte en alternatif est la valeur efficace du courant.

Il aurait été intéressant de pouvoir observer le comportement du disjoncteur lorsqu'il est alimenté en continu et que c'est le déclencheur thermique qui est sollicité. Malheureusement, l'impédance élevée du disjoncteur de seuil 1A nous empêche de monter assez haut en courant avec le matériel disponible. En choisissant un calibre plus faible, avec un courant plus faible nous pourrions couvrir une plus grande partie de la courbe. Mais l'impédance du disjoncteur serait plus forte limitant ainsi le courant. Aucune solution n'a été trouvée pour augmenter le courant circulant dans le disjoncteur. Pour un régime sinusoïdal, les temps de déclenchement sont inférieurs à la période du signal. Nous avons vu précédemment que cela posait des problèmes d'incertitude de mesure et des écarts avec la simulation. On peut s'attendre à observer dans ce cas une différence de résultat entre le continu et l'alternatif.

4.4 Mesure du temps de réaction d'un relais

Nous disposons de relais (contacteurs) qui servent à faciliter la commande des bancs. Il est donc possible de caractériser le temps mis par un relais pour s'enclencher lorsqu'un courant circule dans la partie commande.

4.4.1 Caractéristiques du relais

Le relais est composé de deux parties distinctes. La première est la partie de commande qui fonctionne à faible puissance. C'est elle qui contrôle la deuxième partie du relais, c'est-à-dire les contacts qui permettent ou non le passage du courant.
Entre l'instant où une tension apparaît aux bornes de la commande et l'instant où le relais laisse circuler le courant, il existe une certaine inertie. C'est ce retard que est caractérisé dans cette partie.

Les relais utilisés sur le banc de test sont des contacteurs LC1-D12 de chez Télémécanique. Le constructeur donne une consommation moyenne de 7VA à 24V soit une impédance de $Z = \frac{U^2}{P} = \frac{24^2}{7} = 82.3\Omega$. Le courant efficace est donc de $I_{eff} = \frac{U_{eff}}{Z} = 0{,}29A$. Le cosφ est de 0,3 donc la partie résistive de l'impédance est de $R = Z \cdot \cos\varphi = 24.69\Omega$. La partie inductive est de $X = 7 \cdot \sin\varphi = 78.5\Omega$. La bobine fait donc $L = \frac{X}{2 \cdot \pi \cdot f} = \frac{78.5}{2 \cdot \pi \cdot 50} = 250mH$.

4.4.2 Temps de réaction du relais

Afin de vérifier le temps de réaction de ces relais, on utilise la partie TBT du banc, d'un oscilloscope numérique et deux pinces de courant. Le déclenchement se fait sur le premier front du *signal 1* sur lequel est branché la sonde mesurant le courant de commande circulant dans la bobine. Le temps de réaction est mesurable en faisant la différence entre le premier front du *signal 1* et le premier front du *signal 2*.

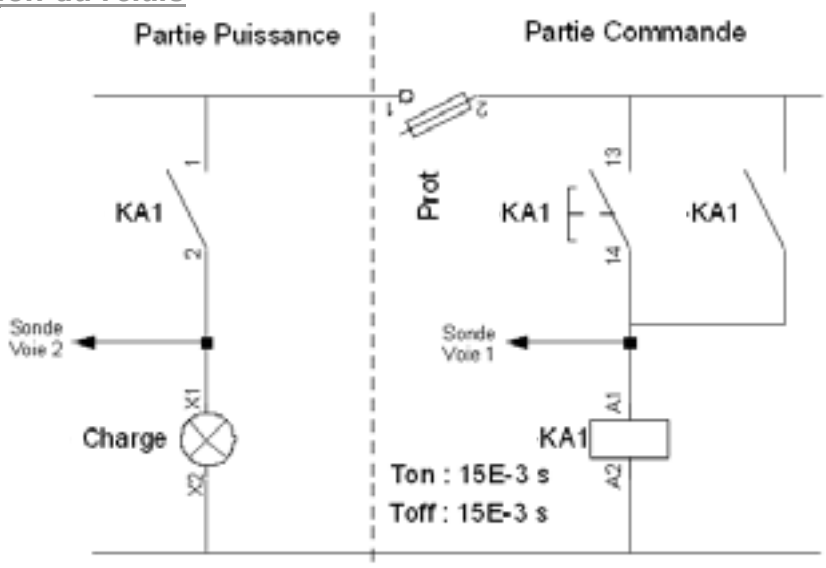


Fig. 54 Temps de fermeture ou d'ouverture des contacts d'un relais, schéma du montage

Dans un premier temps, on mesure le délai lors de la fermeture du contact, puis, dans un deuxième temps, on mesure le délai lors de l'ouverture du contact. Ces deux situations sont montrées sur les deux figures ci-dessous.

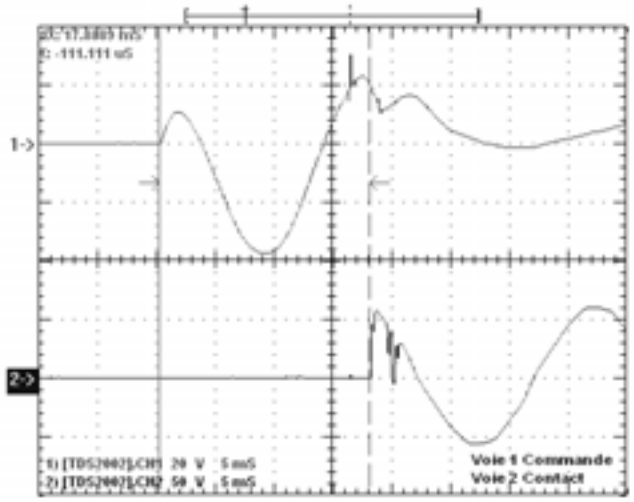


Fig. 55 Délai à la fermeture du contact

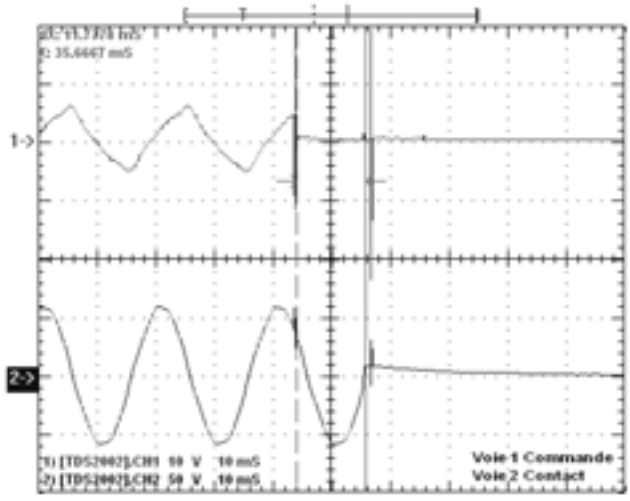


Fig. 56 Délai à l'ouverture du contact

On observe sur la **Fig. 55** que le signal de commande n'est pas sinusoïdal. La présence de ce transitoire est du à l'appel de courant de la bobine du relais.
Afin de minimiser les erreurs, 5 mesures ont été faites et la moyenne a été gardée. 2 relais ont été utilisés pour ces mesures. Les résultats sont résumés dans le tableau ci-dessous.

	Relais n°1		Relais n°2	
	fermeture (ms)	ouverture (ms)	fermeture (ms)	ouverture (ms)
test1	17,22	18,88	15,44	17,11
test2	17,22	15,77	16,66	22,66
test3	15,44	11,78	15,33	19,55
test4	17,89	18	18,55	17,78
test5	15,55	9,77	17,55	5,55
Moyenne	**16,664**	**14,84**	**16,706**	**16,53**

Tab. 9 Relevé des temps d'ouverture et de fermeture des relais

La durée mise par le contact à se fermer est assez stable par contre, la durée d'ouverture du contact est très variable.

4.4.3 Impédance des câbles de grandes longueur.

Nous avons vu précédemment lors du calcul des courants de court-circuit que chaque impédance doit être prise en compte pour prévoir au mieux la valeur de l'Icc. La valeur de l'impédance des câbles n'est plus négligeable lorsqu'ils atteignent des longueurs importantes et que leurs sections sont faibles. Nous rappelons la relation $R=\rho.L/S$ ou R est la résistance du câble, L et S sont sa longueur et sa section et ρ est la résistivité du matériau conducteur. Cette impédance provoque un chute de tension qui doit être estimée finement car elle peut rendre les protections moins efficaces et nuit à la qualité du courant.

4.4.3.1 Intensité admissible et mode de pose.

L'intensité admissible Iz d'un câble est une valeur de courant que le câble peut supporter pendant une courte durée sans risque de destruction. Cette intensité Iz est supérieure à l'intensité l'utilisation nominale In du câble. Le câble doit pouvoir supporter un courant Iz correspondant à une surcharge.
La norme NFC-15 100 [NFC98] définit des règles de calcul de cet Iz en fonction des caractéristiques de câbles et des modes de poses. Les modes de poses sont définis essentiellement par le type de canalisation dans laquelle on place le câble, le nombre et l'agencement des câbles dans cette canalisation et le milieu ambiant.
Elle utilise des coefficients de pondération qui permettent de déterminer la valeur de Iz en fonction de toutes ces caractéristiques.

Ces modes de pose ne sont pas intégrés dans nos modèles. Nous avons voulu mesurer quelle peut être leur influence pour des types et longueur de câbles pouvant se trouver dans des installations domestiques.
L'intensité admissible est à un courant correspondant à un surchage de l'installation pendant laquelle la valeur du courant peut provoquer une chute de tension dans le cable et degrader la qualité de la tension du réseau.

4.4.3.2 Méthode de mesure de la chute de tension.

Pour tester la chute de tension dans un câble de grande longueur, nous disposons d'un câble de 100m, de section 1,5mm^2. Son isolant est en PVC. Celui-ci est branché sur la partie TBT du banc principal et est donc alimenté en 24V. Ce type de câble supporte un courant d'environ 18A. Il est donc nécessaire d'ajouter une résistance de charge. Nous nous servons du rhéostat de 5,5Ω. Nous faisons circuler un courant de 5A dans le câble afin de comparer les résultats.

4.4.3.3 Evaluation des résultats de la chute de tension

Afin d'évaluer les résultats, un logiciel de calcul en ligne a été utilisé. Celui-ci est proposé en ligne par Elec-Planet [ELE]. Il propose un calcul de la chute de tension dans le câble.
La simulation de la configuration précédente donne une impédance de 1,48Ω par phase donc de 3Ω pour le câble. La limite en courant est donc de 8A car la chute de tension serait supérieure à 100% avec un courant plus important.
Le simulateur en ligne donne donc une chute de tension de 61.7% soit 14,8V avec un courant de 5A.

4.4.3.4 Mesures effectuées

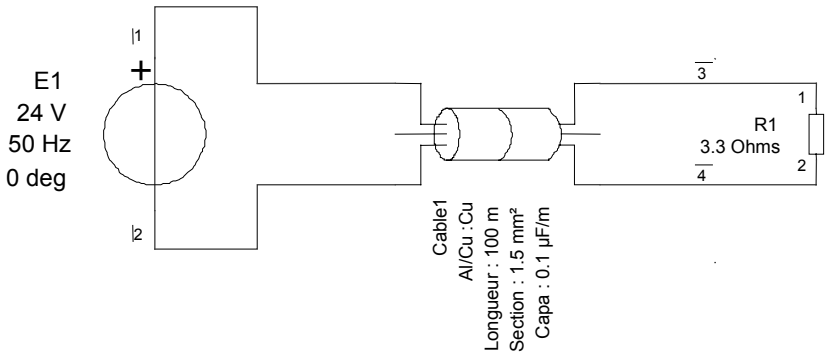
Trois modes de pose différents ont été testés. Le premier correspond à un câble déroulé en ligne droite, le deuxième à un câble disposé en accordéon et enfin, le dernier à un câble enroulé en spires larges, dont on retrouvera des illustrations en **Annexe 17**.
Afin d'obtenir le courant de 5A, la résistance variable a été réglée à 1,93Ω.

Câble droit
Le câble est déployé de manière à effectuer une large courbe. La tension au début du câble est de 24,14V. A la fin du câble, il reste 10,41V. La chute de tension est donc de 13,73V soit 56.88%.L'impédance du câble est de 2,74Ω donc de 1.37Ω par phase.

Câble en accordéon
Le câble est déployé de manière à effectuer des "allers-retours" successifs. Chaque partie droite fait environ un mètre de long. La tension au début du câble est de 24,4V. A la fin du câble, il reste 10,33V. La chute de tension est donc de 14.07V soit 57.6%.L'impédance du câble est de 2,8Ω donc de 1.4Ω par phase.

Câble en spires larges
Le câble est déployé de manière à effectuer des spires d'environ 1m de diamètre. La tension au début du câble est de 24,2V. A la fin du câble, il reste 10,36V. La chute de tension est donc de 13,84V soit 57.2%.
L'impédance du câble est de 2,77Ω donc de 1.38Ω par phase.

4.4.3.5 Schéma du circuit simulé



Chaque phase du câble est modélisée que par une seule résistance de valeur R :

$$R = \rho_{Cu} \cdot \frac{L}{S} = 1.72 \cdot 10^{-8} \cdot \frac{100}{1.5 \cdot 10^{-6}} = 1.15\Omega .$$

Un modèle plus complexe prenant en compte deux résistances en parallèle avec un condensateur est en cours d'implantation. Ce nouveau modèle permettra de prendre en compte les effets capacitifs des câbles.

Afin de régler le courant de 5A, la résistance de charge est fixée à 2,5Ω. La simulation donne une tension de 24V au début du câble et de 12.48V en sortie. Cela implique une chute de tension de 11.52V soit 48%.

4.4.3.6 Comparaison des résultats

Afin de comparer la chute de tension dans les câbles, toutes les valeurs simulées et mesurées ont été résumées dans le tableau ci-dessous :

	Disposition du câble	Tension au début du câble (V)	Tension à la fin du câble (V)	Pourcentage de perte	Chute de tension (V)	Impédance du câble (ohms)
Mesure	Droit	24,14	10,41	56,88%	13,73	2,746
	Accordéon	24,4	10,33	57,66%	14,07	2,814
	Spires larges	24,2	10,36	57,19%	13,84	2,768
elec-planet.com		24	9,2	61,70%	14,8	2,96
Simul'Elec		24	12,48	48,00%	11,52	2,304

Tab. 10 Mesures des chutes de tension pour différents modes de poses

Les mesures confirment les résultats donnés par elec-planet [ELE]. Simul'Elec est un peu plus éloigné mais les résultats restent assez proches de ceux mesurés. Nous pouvons expliquer cette différence par le fait que le modèle de câble retenu n'est pas un modèle de câble mais plutôt un ensemble de modèle de fil nu (sans isolant) qui déterminent l'impédance de l'âme de chaque conducteur. En effet, ce modèle ne prend pas en compte les effets inductifs ou capacitif entre les différents fils du câbles ni la nature de l'isolant.

Conclusion générale et perspectives

Une majorité des simulateurs de circuits électriques sont basés sur SPICE [NAG75] ou intègrent son noyau de calcul. Ce simulateur à été développé à l'université de Berkeley [SPI1] pour répondre au besoin de simulation de l'industrie électronique et en particulier des composants à semi-conducteurs comme les transistors. Cet outil et ses déclinaisons se révèlent mal adaptés à la simulation de circuits électrotechniques qui ont entre autre la particularité d'avoir des comportements séquentiels qui font intervenir des durées très longues devant les constantes de temps des équipements électriques.

Sur le marché français, il n'existe à l'heure actuelle aucun logiciel de simulation orienté vers l'électrotechnique ou l'électricité des courants forts, intégré à une solution de schématique électrique professionnelle.

C'est dans ce contexte que se situent ces travaux de recherche réalisés sous la cotutelle du laboratoire IXL de l'université de Bordeaux 1 et du laboratoire LIPSI-ESTIA, dans le cadre d'un étroit partenariat avec la société Algo'Tech Informatique. Ces travaux ont débouché sur la conception et la réalisation d'un simulateur de circuits électrotechnique commercialisé : Simul'Elec.

Nous proposons une alternative aux logiciels de type SPICE que nous avons conçue pour faciliter la simulation du comportement des circuits électrotechniques et pour être plus proche de la démarche de conception des installations électrotechniques ou électriques. Au travers de notre mode d'analyse événementielle notamment, nous avons voulu offrir un outil fiable et simple d'utilisation permettant d'estimer le comportement des installations afin d'optimiser leur coût de conception et de valider leur fonctionnement.
Simul'Elec est conçu dans l'optique d'être accessible au plus grand nombre, en particulier à ceux qui ne sont pas familiers de la simulation.

Simul'Elec est commercialisé par Algo'Tech Informatique au sein de sa solution de CAO électrique et suscite déjà un fort intérêt auprès des industriels et du corps enseignant. Différents projets sont à l'étude, notamment :

- La validation fonctionnelle du simulateur dans le cadre de l'enseignement technique. Un partenariat avec le lycée Louis de Foix se met en place et vise à définir les besoins d'adaptation du logiciel aux différents niveaux d'étude et aux différentes filières d'enseignement.
- L'interfaçage avec CATIA V5 pour la simulation et la localisation de défauts dans un aéronef.

- La simulation fine du comportement des câbles, torons et harnais utilisés en aéronautique en prenant en compte les échanges thermiques entre les fils et avec la structure de l'aéronef. Ce travail de recherche et développement se déroulerait dans le cadre d'un projet européen en partenariat avec un leader français du câblage aéronautique. Son objectif sera la création d'un outil d'aide au dimensionnement d'équipements, à destination des techniciens.

Les bons résultats de la validation nous ont permis de confirmer la pertinence des modèles et des modes d'analyse que nous avons développés. Nous avons pu vérifier ainsi la justesse de nos modes d'analyse qui ont su se montrer plutôt performants, notamment par rapport à Pspice, dans des configurations caractéristiques de l'électrotechnique. De plus, nous avons été confortés sur le domaine de validité des modèles innovants que nous avons développés. L'analyse comparative des données issues de la simulation, réalisée en particulier pour le cas des disjoncteurs, s'est révélée satisfaisante. D'une manière générale, ceux-ci reproduisent fidèlement le comportement attendu, principalement la sélectivité, étant donné la relative précision garantie par les constructeurs. La simulation donne d'ailleurs de meilleurs résultats que les méthodes de calcul théorique utilisées pour le dimensionnement de ces équipements. Nous avons pu isoler les rares cas divergents et proposer des pistes d'améliorations.
Ces conclusions encourageantes, nous permettent de nous fixer des objectifs plus ambitieux pour le perfectionnement de Simul'Elec. Nous avons déterminé un certain nombre d'axes de développement pour l'amélioration des modes d'analyses et des modèles.

Un point essentiel, dont nous avons estimé la faisabilité au cours de nos recherches, est l'ouverture des modèles à l'utilisateur par l'utilisation du langage VHDL-AMS [VHD1]. Cela nécessitera l'implémentation d'un traducteur VHDL-AMS vers Delphi qui permettra au simulateur d'interpréter les modèles définis par l'utilisateur. Cette étape sera suivie par la création d'une interface graphique permettant une création simple des modèles et générant du code VHDL-AMS.
Sachant que les domaines de l'hydraulique et du pneumatique sont régis par les mêmes équations fondamentales, nous envisageons à moyen terme, d'adapter nos techniques de simulation pour traiter non seulement des circuits électriques, mais aussi pneumatiques, hydrauliques et mixtes, courants en électrotechnique. Cette extension sera grandement facilitée par les possibilités de modélisation multi-domaines et multi-technologiques du langage VHDL-AMS.

Les modes d'analyses actuellement disponibles permettent de réaliser une étude rapide sur de nombreux circuits électrotechniques. Il conviendra toutefois de les faire évoluer ainsi que d'étendre la gamme des composants simulables dans l'optique de traiter davantage de cas et de réaliser des études plus complètes. Cela concerne notamment la correction des modèles de disjoncteurs et la création de nouveaux modèles apparentés comme les disjoncteurs différentiels ou les relais de protection thermique des moteurs.

La présence de phénomènes transitoires dans des circuits électrotechniques peut avoir une influence non négligeable sur le fonctionnement normal d'une installation. Ces phénomènes sont généralement bien connus et sont particulièrement importants lorsqu'il s'agit de surtensions ou de surintensités. On retrouvera par exemple des effets de surcharges, des déclenchements de protections ou des destructions liées à des courants d'appels trop élevés qui peuvent être le résultat d'un démarrage de moteur ou de la mise sous tension d'inductances. Dès aujourd'hui, nous avons la possibilité d'intégrer ces phénomènes transitoires dans l'analyse événementielle à la manière d'un niveau de détail supplémentaire.

Ainsi, dans le modèle de certains composants, il serait possible de définir un comportement transitoire sous la forme d'une succession d'événements qui en faisant varier un paramètre du modèle permet de reproduire le comportement attendu. On peut imaginer par exemple de faire varier, durant sa phase de démarrage, l'impédance caractéristique d'un moteur pour simuler son courant d'appel et ainsi observer ses effets sur ses équipements de protection.
La prise en compte des phénomènes transitoires peut donc passer par ce moyen atypique où le modèle prédéfinit le comportement qu'il attend. Dans ce cas, c'est essentiellement l'effet du transitoire que l'on souhaite étudier et pas nécessairement la cause. Ce moyen permet, dans l'esprit de Simul'Elec d'obtenir des résultats plus intuitifs et plus rapidement bien que la méthode utilisée puisse sembler moins rigoureuse. Afin d'exploiter ce potentiel, il sera nécessaire de développer les modèles de composants intégrant ces caractéristiques.
Afin d'enrichir les possibilités du simulateur, il serait toutefois nécessaire de compléter ces possibilités par un « vrai » mode d'analyse transitoire comme on peut en trouver dans les simulateurs tels que Spice. Dans ce cas, le simulateur calcule point après point l'évolution temporelle des tensions et des courants. Pour cela, il résout un système d'équations différentielles en utilisant des algorithmes spécifiques et connus (algorithmes d'intégration : trapézoïdal, Gear, backward-Euler…). Ce type de mode d'analyse requiert une description beaucoup plus fine et plus complexes des modèles.

Actuellement, le simulateur prend en compte un certain nombre de discontinuités liées généralement aux valeurs efficaces des tensions et des courants. Ces discontinuités sont simulées par l'analyse événementielle à partir de leur description au sein des modèles des composants. C'est par exemple le cas du cycle d'hystérésis qui définit l'état logique d'un relais en fonction de la valeur efficace de la tension aux bornes de la bobine.
Il serait intéressant de pouvoir enrichir les possibilités des modes d'analyse en leur permettant de traiter d'autres comportements non linéaires touchant notamment aux valeurs instantanées des signaux comme le seuillage, l'écrêtage, la saturation.
Ainsi, nous envisageons également d'intégrer des modèles de composants typiquement non linéaires (diodes, transistor, thyristor…) pour étendre notre champ d'application à l'électronique de puissance. Nous les traiterions dans un premier temps comme des composants idéalisés ayant un comportement d'interrupteurs commandés.

En introduisant des variables externes ou environnementales telles que la température ambiante, nous pourrions observer l'influence de l'environnement sur une installation. Il est nécessaire de définir quelles peuvent être ces variables et de quelles manières nous devons les faire intervenir dans la modélisation des différents composants.

Sur un plan plus ergonomique, nous envisageons également l'amélioration de l'interactivité entre la simulation et le schéma. Des travaux en ce sens ont été initiés, en parallèle de ce travail de thèse, par Algo'Tech Informatique depuis début 2003. Il s'agit d'appliquer des stimuli directement sur le schéma ainsi que de placer des sondes pour mesurer les grandeurs caractéristiques. On retrouvera les valeurs de tensions et de courants affichés sommairement au niveau du schéma, ainsi que les états logiques et les branches parcourues ou non par un courant.

Références bibliographiques

A

[ABI03] H. Abitbol, **Validation d'un simulateur de circuit électrotechnique - Rapport de mission en entreprise**, Projet 2ème année ESTIA, 2003

[ADV] **ADVance MS reference manual**, Mentor Graphics, 2000.

[AIR98] R. Airiau & al., **VHDL langage, modélisation, synthèse**, 2ème édition, Presses polytechniques et universitaires romandes, 1998 ISBN 2-88074-361-3

[ALA98] O. Alali, **Modélisation VHDL-AMS analogique et Simulation Spice**, Thèse ENST, 1998

[ALG] **Algo'Tech Informatique**, http://www.algotech.fr

B

[BAN] Institut Schneider Formation, **Banc de sélectivité**, http://www.schneiderformation.com/didactic.fr/fiches/index.php?G_id=5

[BEA] BEAMS, **Behavioural modelling of Analogue and Mixed Systems**, a non-lucrative association for the promotion of behavioural modelling and simulation with HDL languages (mainly VHDL-AMS), http : //www.beams.asso.fr

[BLA82] E. Blanc & al., **Guide de l'installation électrique**, Merlin-Gerin, 1982

[BOR02] BORLAND, **Delphi 7 Guide du développeur**, 2002

[BOU92] R. Bourgeois, D. Cogniel, **Mémotech Electrotechnique**, 4ème édition, Casteilla, 1992 ISBN 2-7135-1219-0

[BOY81] P. Boye, A. Bianciotto, **Le schéma en électrotechnique**, Delagrave, 1981 ISBN 2-206-00112-8

C

[CEI88] **Norme CEI 900 Calcul des courants de court-circuit dans les réseaux triphasés à courant alternatif**, UTE, 1988

[CIT94] **Schémathèque Technologies du contrôle industriel**, CITEF, 1994 , ISBN 2-907314-21-1

[COG00] Collectif Cogisoft**, Delphi5 / XML**, CampusPress, 2000, ISBN 2-7440-0926-1

[COU99] M. Cousineau, **Modélisation des systèmes analogiques**, Thèse INP Toulouse, 1999

[COV] XML Cover pages, **XML and Electronic Design Automation** (EDA), http://xml.coverpages.org/xmlAndEDA.html

[CRO] **Crocodile physics** ,
http://www.crocodile-clips.com/french/crocodile/physics/index.htm

D

[DEL] **Delphi**, http://www.borland.fr/delphi/index.html

[DEP91] Y. Déplanche, **Mémo Formulaire**, Casteilla, 1991 ISBN 2-7135-1162-3

[DEP1] N. Depail, **Validation d'un simulateur de circuit électrotechnique - Rapport de mission en entreprise**, Projet 2ème année ENSEIRB, 2003

[DEP2] N. Depail, H. Abitbol, **Validation d'un simulateur de circuit électrotechnique - Dossier technique global**, 2003

E

[EDM97] J. Edminister, M. Nahvi, **Circuits électriques, théorie et problèmes**,
3ème édition, McGraw-Hill, 1997 ISBN 2-7042-1291-0

[ELE] Elec Planet, http://www.elec-planet.com/
Calcul d'installation électriques en ligne.

[EIT] Pôle Technologique EITICA (Electronique, Informatique et TIC en Aquitaine),
http://www.eitica.org

[ERA96] P.-J. Erard, P. Déguénon, Simulation par événement discret, 1ère édition,
Presses polytechniques et universitaires romandes, 1996 ISBN 2-88074-295-1

[ETO] **Traducteurs EDA-XML**, e-tools,
http://www.e-tools.com/content/xml_translators.html

F, G, H

[FIT] **Schémaplic**, Fitec, http://www.fitec.fr/interactif.htm

[FOU84] J.-M. Fouchet**, Electricité pratique**, Dunod, 1984 ISBN 2-04-015912-6

[GAR01] J. M. Garrido, **Object Oriented Discrete-Event Simulation with Java**,
Kluwer Academic/Plenum publishers, 2001 ISBN 0-306-46688-0

[GIL98] N. Gilardi, **Création d'un module de simulation Electrique SiCi pour un logiciel de Schématique**, Projet DESS SPIA, 1998

[HAM] **HAMSTER**, http:/hamster-ams.com

[HER] Y. HERVE, **GRAMMAIRE VHDL-AMS (forme BNF)**,
http://www-ensps.u-strasbg.fr/coursen/Option3A/ams_bnf.html

[HER02] Y. Hervé, **VHDL-AMS Applications et enjeux industriels**,
Dunod, 2002 ISBN 2-10-005888-6

I, J, K

[IED] **Time warp**, Institute of electronic design automation, Allemagne,
www.eda.ei.tum.de/forschung.simul/englisch/timewarp.html

[JAM] S. James, **Initiation à XML et Delphi 6**, http://sjames.developpez.com/xml/

[JEM03] S. Jemmali, **Contribution à l'amélioration de méthodologie et d'outils d'aide à la conception de systèmes multitechnologiques** , Thèse ENST, 2003

[KIE98] R. Kielkowsji, **Inside Spice**, 2ème édition, McGRaw-Hill, 1998 ISBN 0-07-913712-1

L

[LAR97] P. Larcher, **VHDL introduction à la synthèse logique**, Eyrolles, 1997 ISBN 2-212-09584-8

[LAW91] A. M. Law, W. D. Kelton , **Simulation modeling & analysis**, 2ème édition, McGRaw-Hill, 1991 ISBN 0-07-100803-9

[LEG00] **Appareillage électrique d'installation**, LEGRAND, 2000

[LEG03] F. Legrand, N. Couture, R. Briand, H. Lévi, **Simulation de schémas électriques ou électrotechniques par utilisation de l'analyse événementielle**, CIAI3 (Conférence Internationale sur l'Automatisation Industrielle), Montréal, Canada, juin 2003

[LEG04] F. Legrand, H. Lévi, N. Couture, J.J. Charlot, **VHDL-AMS Modeling and Library Building for Power Electrical Engineering**, AMC'04 (the 8th IEEE International Workshop on Advanced Motion Control), Kawasaki Japon, Mars 2004

[LIT97] V. Litovski, M. Zwolinski**, VLSI Circuit Simulation and Optimization**, Chapman & Hall, 1997 ISBN 0-412-63860-6

M

[MATL] **MatLab – Simulink,** http://www.mathworks.com/

[MATH] **MathML,** http://www.w3.org/Math/

[MEN] **Mentor graphics**, http://www.mentor.com/

[MER98] R. Mérat & al., **Génie électrotechnique STI-BTS**, Editions Nathan, 1998 ISBN 2-09-177980-6

[MIL02] N. Milet-Lewis, S. Snaidero, Y. Hervé, G. Monnerie, D. Geoffroy, A. Fakhfakh, and H. Levi, **Behavioural Library Development :Models Documentation and Qualification**, Forum on Design Languages, FDL'02, Septembre 23-27, 2002, Marseille, France

[MIL97] N. Millet-Lewis, **Contribution à la modélisation comportementale des circuits analogiques**, Thèse Bordeaux I, 1997

N, O, Q, R

[NAG75] L.W. Nagel, **SPICE2 : a computer program to simulate semiconductor circuits**, University of California, Berkeley, 1975 Mémorandum n° ERL-M520

[NEW78] R. Newton, **The simulation of large scale integrated circuits**, University of California, Berkeley, 1978 Mémorandum n°UCB/ERL M78/52

[NFC98] **Norme NFC15-100 Installations électrique basse tension**, UTE, 1998

[ORC] **OrCAD, Simulation PSPICE**, http://www.orcad.com/products/pspice/

[QUE88] J.L. Queyrel, J. Mesplède, **Précis de physique - Electronique**, Bréal, 1988 ISBN 2-85394-264-3

[RAP98] J. Rappaz, M. Picasso, **Introduction à l'analyse numérique**, Presses polytechniques et universitaires romandes, 1998 ISBN 2-88074-363-X

S

[SAB] **Saber**, http://www.analogy.com/products/mixedsignal/saber/saber.html

[SCH01] **Automatismes Industriels télémécanique**, Schneider Electric, 2001

[SEY00] F. Seyler, **Analyse d'un simulateur de circuit générique intégré**, Mémoire de DEA-LABRI, 2000

[SIM] **ANSOFT, Simplorer**, Simulation software for multi-domain designs in automotive, aerospace, power electronics, and electric drive systems http://www.ansoft.com/products/em/simplorer/

[SOC00] **Catalogue général - Cahier technique**, SOCOMEC, 2000

[SPI1] Spice, **The Spice page**, Université de Berkeley, http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/

[SPI2] **Spice3F5, Spice Latest version**, Université de Berkeley, http://www.eecs.berkeley.edu/IPRO/Software/Catalog/Description/spice3f5

[SVG] **SVG**, scalable Vector Graphics, http://www.w3.org/TR/SVG/

V, W

[VHD1] **IEEE Standard VHDL Analog and Mixed-Signal Extensions**, IEEE Std 1076.1 – 1999.

[VHD2] **Groupe de travail IEEE VHDL-AMS**, http://www.eda.org/vhdl-ams/

[VLA81] A. Vladimirescu & al., **Spice Version 2g user's guide**, University of California, Berkeley, 1981

[W3C] W3C, **World Wide Web Consortium** , eXtensible Markup Language, http://www.w3.org/XML/

[WOO01] K. Wood, **Delphi Developer's guide to XML**, Wordware publishing, 2001 ISBN 1-55622-812-0

ANNEXES

SIMULATION DE SCHEMAS ELECTRIQUES OU ELECTROTECHNIQUES PAR UTILISATION DE L'ANALYSE EVENEMENTIELLE

Fabien LEGRAND[1], Nadine COUTURE[1], Renaud BRIAND[1], Hervé LÉVI[2]

[1]LIPSI-ESTIA
Technopole Izarbel 64210 Bidart, France E-mail: [f.legrand, n.couture ; r.briand]@estia.fr

[2] IXL - Université Bordeaux I (UMR CNRS 5818)
351, cours de la Libération 33405 Talence, Cedex France. E-mail : levi@ixl.u-bordeaux.fr

Abstract : We present an electrical circuit simulator integrated in a CAD software for industry. It simulates the reaction of an electrical circuit to user defined stimuli. It returns voltages, currents and logic states values of simulated devices. This simulator uses an original event driven analysis mode which simulates relatively long period events without any transient analysis to keep simulation speed.

Résumé : Nous présentons un simulateur de circuit électrique intégré au logiciel de schématique électrique pour l'industrie, Elec'view. Il simule les réactions d'un circuit électrotechnique à des stimuli définis par l'utilisateur et donne comme résultats les valeurs de tensions, de courants et les états logiques des appareils simulés. Il utilise un mode d'analyse événementielle original qui permet de simuler des événements relativement long sans recourir à une analyse transitoire afin de préserver la rapidité de calcul.

Mots clés : Conception Assistée par Ordinateur, Simulateur de circuit électrotechnique, analyse événementielle, modélisation de composants pour l'électrotechnique, optimisation des installations électriques.

1. INTRODUCTION

Nous partons de trois constats. D'une part, les logiciels de schématique électrique dédiés à la conception des installations électriques ne permettent pas, à ce jour, leur simulation. D'autre part, les logiciels de simulation électrotechnique sont surtout utilisés pour l'étude du fonctionnement des machines électriques, mais pas pour l'étude d'une installation complète. Enfin, le logiciel de simulation électrique SPICE [1] qui est la référence actuelle en matière de simulation est surtout dédié à la simulation de circuits intégrés en vue de leur fabrication. De ce fait, SPICE et les simulateurs issus de SPICE sont mal adaptés pour simuler le comportement de circuits ou systèmes sur des durées importantes, surtout si les événements provoquant des modifications de l'état du système sont relativement espacés dans le temps. En effet, ces simulateurs effectuent des analyses transitoires avec un pas de calcul suffisamment fin par rapport aux constantes de temps du circuit pour conserver une bonne précision des résultats. En électrotechnique, les événements ont généralement des durées importantes (minutes, secondes) devant ce pas ce qui conduit souvent à des temps de calcul rédhibitoires.

Nous proposons donc un simulateur électrique et électrotechnique, intégré à un logiciel de schématique, résolvant les problèmes énoncés ci-dessus.

2. LE SIMULATEUR DE CIRCUIT ELECTROTECHNIQUE

Nous présentons ici le module de simulation intégré au logiciel de DAO-CAO électrique Elec'view [2] . Son but est de permettre la simulation de réseaux électriques, intégrant la partie puissance et la partie commande, afin d'estimer le comportement physique des installations et ainsi optimiser leur coût de conception. La mise en œuvre du logiciel a nécessité une réflexion sur les modes d'analyse mais également sur la modélisation des composants de l'électrotechnique (moteurs, transformateurs, disjoncteurs, relais, sources…).

Ainsi, notre simulateur se démarque des simulateurs classiques (analogiques, numériques ou mixtes) par le type de composants qu'il permet de simuler. Comme avec SPICE, on peut simuler les composants électriques de base tels que résistances, condensateurs, selfs, sources, transformateurs et amplificateurs parfaits. Avec notre simulateur, il est possible de simuler également des composants spécifiques à l'électrotechnique comme les relais, sectionneurs, fusibles, disjoncteurs, etc. Ces composants sont particuliers car ils ont d'une part un comportement analogique (impédance caractéristique par exemple) et d'autre part un comportement « Tout Ou Rien » lié à l'état logique que l'on peut leur associer (ouvert, fermé, enclenché, déclenché, etc.).

La création du module de simulation du logiciel de DAO-CAO électrique Elec'view a nécessité la réalisation d'une interface graphique permettant l'accès aux différents modes d'analyses (régimes permanent, événementiel et fréquentiel). Parmi ces modes d'analyses, nous présentons un mode d'analyse événementiel original. Il permet de simuler le comportement séquentiel des équipements, en réaction à des stimuli définis par l'utilisateur, tout en conservant des durées de simulation acceptables.

3. LE MODE D'ANALYSE EN REGIME PERMANENT

Nous allons présenter l'analyse en régime permanent afin de mieux appréhender par la suite le mode d'analyse événementiel. Cet mode d'analyse en régime permanent nous donne un état stable de l'installation en une seule résolution du système d'équations modélisant le circuit. Il ne s'agit pas ici d'un régime continu mais plutôt d'un régime sinusoïdal permanent. En effet, on obtient des valeurs complexes de tensions et de courants avec pour chacune la notion d'amplitude et de déphasage.

Pour l'obtention de chaque régime établi, on modélise le circuit sous forme d'un système d'équations linéaires (avec des termes complexes). Il est construit selon la méthode d'analyse nodale modifiée [3] également utilisée par SPICE. Il est contenu dans le système de matrices creuses :

$$\mathbf{G.U = I}$$

En simplifiant, on peut considérer que G représente la matrice des admittances (contributions des impédances des différents composants du circuit), U est la matrice des tensions (potentiel des différents nœuds du circuit, inconnues) et I est la matrice des courants (intensités traversant les branches, excitations du circuit). Le système d'équations ainsi obtenu est résolu par des méthodes classiques d'analyse numérique [4] (décomposition LU suivi d'une élimination de Gauss).

S'il existe différentes fréquences d'excitation dans le circuit, on calcule le régime permanent correspondant à chaque fréquence en éteignant les sources de fréquences différentes. Le principe de superposition est ensuite utilisé pour ajouter les résultats obtenus. Dans le cas particulier d'une excitation continue, le principe reste le même, on calcule le régime permanent en considérant que la fréquence d'excitation est nulle.

4. PRINCIPE DE L'ANALYSE EVENEMENTIELLE

Notre préoccupation est d'offrir au concepteur de l'installation un mode d'analyse piloté par événements qui permette de suivre l'évolution naturelle ou forcée du circuit traité. L'objectif de cette analyse est de reproduire le fonctionnement de l'installation avec, en outre, la possibilité de simuler des interventions humaines, des pannes ou des courts-circuits dans l'installation.

Ainsi, l'utilisateur peut planifier un certain nombre de modifications successives ou simultanées à appliquer sur le circuit sous forme d'événements. Il est donc possible de tester de multiples configurations dans le but de vérifier le fonctionnement de l'installation ou de dimensionner au mieux des composants, par exemple. Il existe 2 types d'interventions :

- Changement de la valeur d'un paramètre d'un composant. Tous les paramètres d'un modèle sont modifiables. Ces changements peuvent être effectués par l'utilisateur qui définit manuellement les valeurs des paramètres mais également de manière automatique en prévoyant des gammes de valeurs selon des échelles linéaires, logarithmiques ou en respectant des séries normalisées (ex : E12,…).
- Forçage de l'état logique des composants (fermeture ou ouverture d'un interrupteur, déclenchement d'un disjoncteur, etc.). Le terme forçage indique que l'état définit par l'utilisateur est prioritaire sur l'état naturel du composant.

L'analyse événementielle repose sur l'enchaînement de plusieurs analyses en régime permanent. Cet enchaînement est piloté par un calendrier d'événements. Un événement traduit un changement d'état du système. A la fin de l'étude, toutes les solutions successivement obtenues décrivent tous les états stables pris par le circuit.

L'algorithme traduisant l'analyse événementielle s'appuie uniquement sur une résolution analogique. Il n'y a donc pas besoin de synchroniser deux processus concurrents de simulation analogiques et numériques comme cela existe dans des simulateurs mixtes [5], il s'agit simplement d'un cadencement.

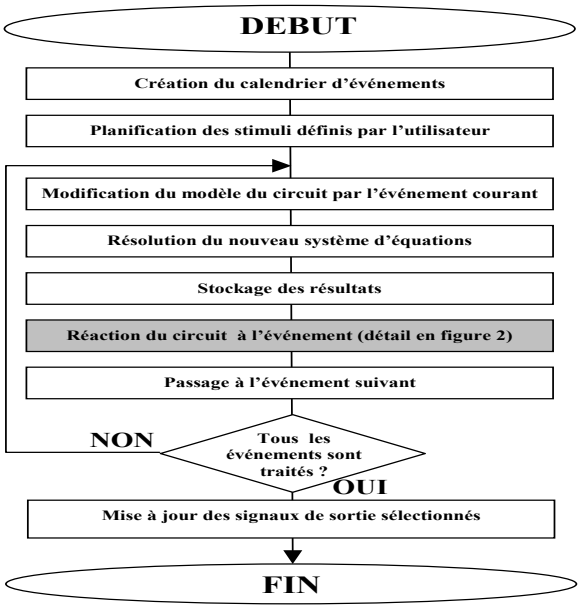


Fig.1 : Algorithme de l'analyse événementielle.

Ainsi, cet algorithme, représenté figure 1, détermine l'instant où les paramètres du circuit seront modifiés afin de représenter un changement d'état. Une nouvelle simulation est alors lancée. Cette procédure est répétée à chaque événement. Deux types d'événements peuvent être distingués. Les événements peuvent être provoqués soit par l'utilisateur en définissant des stimuli via l'interface graphique ou bien par une réaction du système à l'état en cours (déclenchement d'un disjoncteur, activation d'un relais). De façon plus détaillée, le calendrier d'événements est initialisé à partir des stimuli définis par l'utilisateur. Ensuite, pour chaque événement planifié, l'état du circuit évolue et une nouvelle résolution est déclenchée. En fonction de ces résultats, le simulateur évalue le comportement du circuit face à ce nouvel état (voir détail fig. 2), qui se traduit par l'apparition de nouveaux événements alors insérés dans le calendrier. Lorsque l'analyse est terminée, l'affichage des signaux est mis à jour.

En fin de simulation, on peut exploiter les différents résultats de l'analyse soit :

- Les tensions et courants mesurés (sous forme de courbes ou de valeurs efficaces).
- Les états logiques résultants. Ils peuvent être différents des états prédéfinis dans les zones sans forçage.

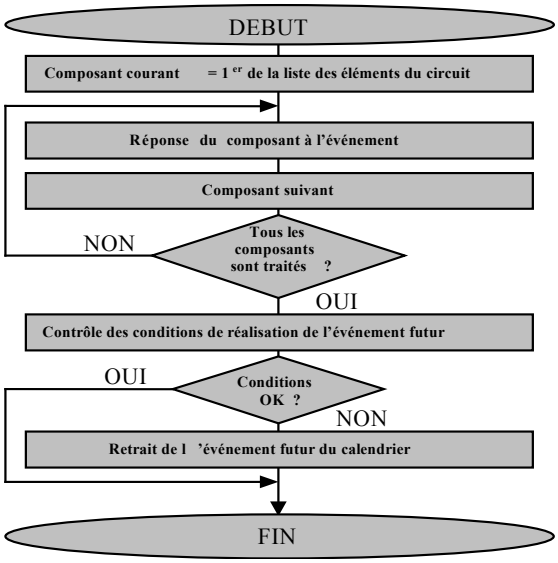


Fig 2 : Algorithme de réaction du circuit à l'événement.

5. MODELISATION DE COMPOSANT : CAS DU DISJONCTEUR

Pour être exploitable dans le mode d'analyse événementiel, il faut définir le modèle d'un composant selon 2 aspects :

- L'aspect électrique définit la participation de chaque composant en terme d'impédance et d'excitation. Dans chaque mode d'analyse, cet aspect est utilisé pour la construction des matrices dont le détail est donnée dans la partie 3.
- L'aspect événementiel décrit la réaction d'un composant suite à un événement (variation de tension, de courant, d'état logique…). Cette réponse est généralement un changement d'état logique qui va provoquer une modification du modèle électrique. Certains composants n'ont pas de modèle événementiel car leur comportement est identique en toutes circonstances (ex : résistance, condensateur, self, source…) quand on considère qu'ils fonctionnent dans des conditions nominales.

L'analyse événementielle trouve tout particulièrement son intérêt dans la simulation du comportement d'un disjoncteur magnétothermique. Elle permet de contrôler :

- Le fonctionnement individuel de chaque disjoncteur.
- Le bon dimensionnement de ces équipements (type, calibre) par rapport au circuit à protéger.
- Le respect des règles de sélectivité entre les équipements de protection amonts et avals d'une installation.

Du point de vue de l'aspect électrique, le disjoncteur est modélisé par une résistance. Elle a une valeur faible (quelques mΩ) représentant la résistance de son contact lorsqu'il est fermé et une valeur très élevée (de l'ordre du MΩ) quand il est ouvert qui prend en compte la résistance de l'air. L'objectif est, dans un premier temps, d'obtenir un modèle simple qui permette de simuler fidèlement le fonctionnement normal des équipements. Les solutions retenues permettent de caractériser ces matériels en utilisant seulement deux paramètres : la référence constructeur du matériel et le calibre.
La référence du matériel permet d'associer une courbe de déclenchement normalisée $T_d = f(I/I_n)$ au composant à partir des documentations constructeur, dont on trouve la représentation en figure 3**.** Le calibre I_n permet de retrouver le temps de déclenchement T_d du disjoncteur en fonction du courant I qui le parcourt. Ainsi la mesure du courant efficace permet de planifier un éventuel déclenchement dans un temps déterminé à partir de ce courant.

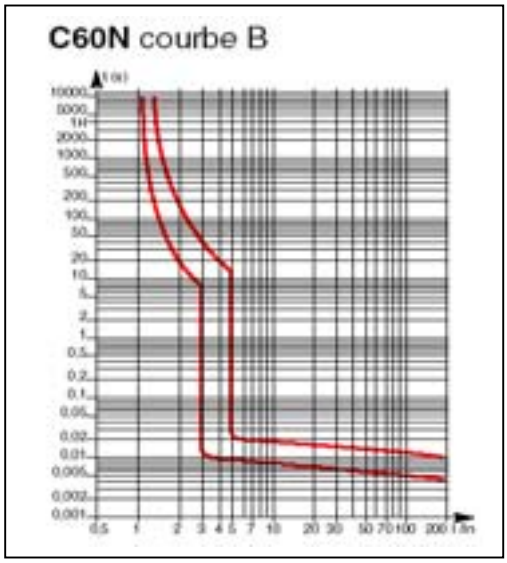


Fig 3 : courbe de déclenchement d'un disjoncteur magnétothermique (Merlin-Gérin C60N courbe B).

La figure 4 détaille l'algorithme du modèle du disjoncteur pour planifier les déclenchements lors de chaque événement. Avant tout, il est essentiel de distinguer la planification d'un événement qui correspond à sa mise en file d'attente, de sa réalisation qui ne sera effective que si toutes les conditions de déclenchement sont maintenues, dans ce cas, la surintensité. Il faut qu'entre l'instant de la planification et le moment de déclenchement le courant n'ai pas changé. Dans le cas contraire, l'événement prévu est annulé et est remplacé par un événement correspondant à la nouvelle valeur de courant.

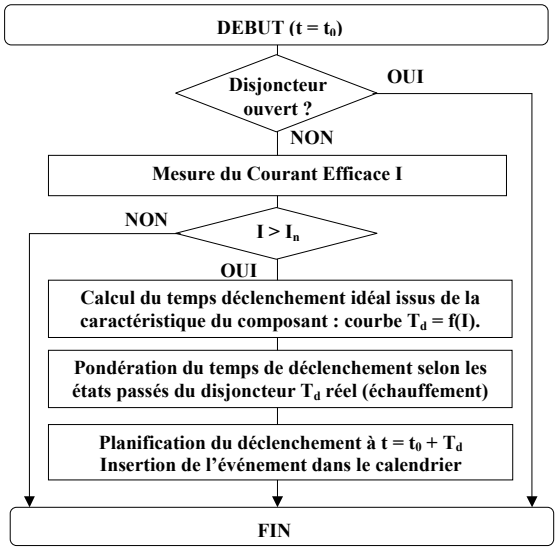


Fig 4 : Algorithme du comportement du disjoncteur

Pour un courant efficace constant, la courbe caractéristique de la figure 3 nous donne la valeur du temps qu'il faudra pour déclencher le disjoncteur. Cependant, si la valeur efficace du courant évolue, tout en restant supérieure au courant nominal, il est difficile de prévoir le comportement du disjoncteur d'après la seule courbe de déclenchement. Pour cela, le modèle du disjoncteur pondère la valeur de T_d (idéal) donnée par la courbe en tenant compte du passé du disjoncteur. Nous obtenons alors une valeur de T_d (réelle) inférieure à celle donnée par la courbe. De cette manière, on peut prendre en considération, par exemple l'échauffement subit par le disjoncteur lors de l'événement précédent et aboutir, comme on peut l'observer sur la figure 5, à un déclenchement plus rapide.

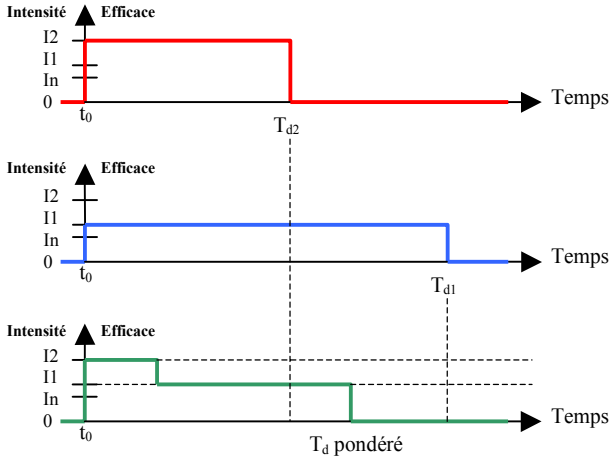


Fig. 5 : déclenchement du disjoncteur pour différentes valeurs de courant

6. CONCLUSION

Le travail réalisé a non seulement enrichi le logiciel de DAO-CAO Elec'view de la société Algo'Tech Informatique mais propose une alternative aux logiciels de type SPICE mal adaptés pour simuler le comportement des circuits suite à des événements relativement longs. Notre simulateur permet donc la simulation de réseaux, intégrant la partie puissance et la partie commande, afin d'estimer le comportement physique des installations et ainsi optimiser leur coût de conception. Il s'appuie sur une méthode d'analyse événementielle originale permettant de simuler le comportement séquentiel des équipements ainsi que certains phénomènes transitoires, tout en conservant des durées de simulation acceptables.

Sachant que les domaines de l'hydraulique et du pneumatique sont régis par les mêmes équations fondamentales, nous envisageons en guise de perspectives d'adapter les mêmes techniques de simulation et de développement de modèles pour traiter également des circuits électriques, pneumatiques, hydrauliques ou mixtes. Un autre point d'investigation possible est d'offrir à l'utilisateur l'accès aux modèles et à leur développement. Dans ce but, deux voies semblent prometteuses : l'utilisation du langage de modélisation comportementale *IEE1076.1* VHDL-AMS [6] ou encore la création d'un langage XML [7] (*IETF W3C Proposed Standard 2001*) de représentation des circuits. Certains travaux [8] ont conduit à exprimer les modèles des composants en XML. Notre approche consisterait à intégrer ces travaux pour proposer un langage XML permettant à la fois de décrire la topologie du circuit et son comportement.

RÉFÉRENCES

[1] L.W. Nagel « *SPICE2 : A computer program to simulate semiconductor circuits*», Memorandum n° UCB/ERL M520, Electronics Research Laboratory, University of California, Berkeley, 1975.

[2] Algo'Tech informatique SA http://www.algotech.fr/Le_simulateur_elecview.htm

[3] V. Litovski, M. Zwolinski « VLSI Circuit simulation and optimization » Ed. Chapman & Hall 1997, ISBN 0-412-63860-6

[4] J. Rappaz, M. Picasso *« Introduction à l'analyse numérique »* Ed. Presses polytechniques et universitaires romandes 1998, ISBN 2-88074-363-X

[5] P-J. Erard, P. Déguénon *« Simulation par événements discrets »*.Ed. Presses polytechniques et universitaires romandes 1996, ISBN 2-88074-295-1

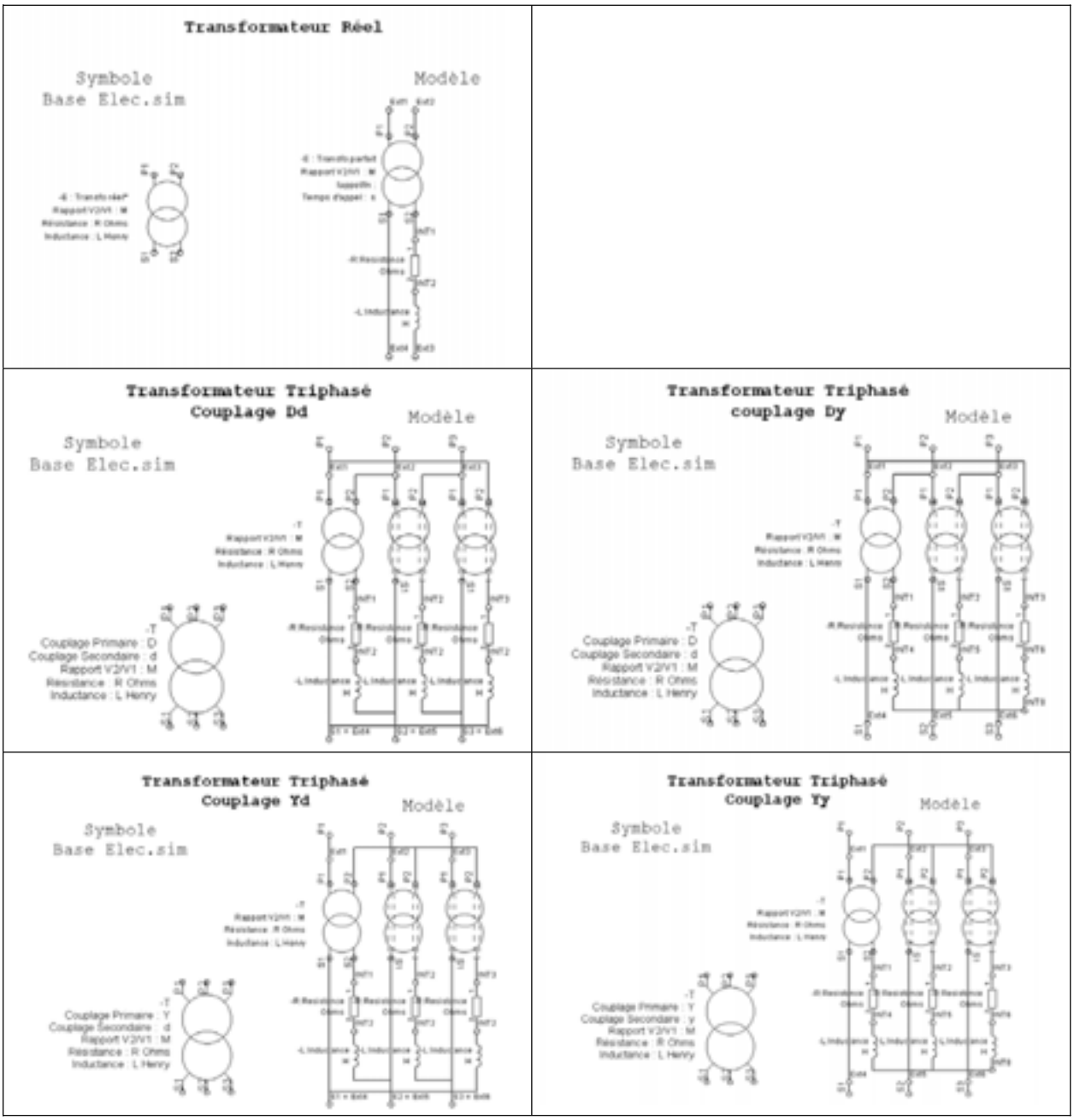
[6] VHDL-AMS : *« un outil pour la simulation des systèmes électroniques et multi-technologiques »* N. Milet-Lewis, H. Lévi, T. Zimmer, et J-L. Battaglia CNFM2000, Sixièmes Journées Pédagogiques du CNFM, 29-30 Nov - 1er Déc 2000, Saint-Malo, France

[7] Extensible Markup Language (XML) http://www.w3.org/XML/

[8] EdaXML : Electronic Design Automation and XML http://www.e-tools.com/content/xml_translators.html

Annexe 1 : Macro Modèles simples.

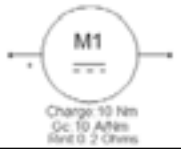
Transformateurs monophasés et triphasés



Moteurs et génératrices

Moteur à courant Continu à Excitation constante

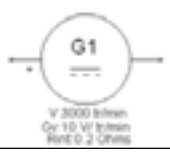
Symbole Base Elec.sim



Modèle

Génératrice à courant Continu à Excitation constante

Symbole Base Elec.sim

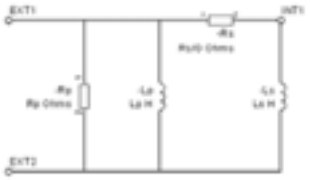


Modèle

Moteur Asynchrone Monophasé

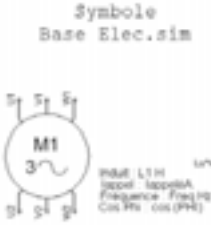
Symbole Base Elec.sim

Modèle

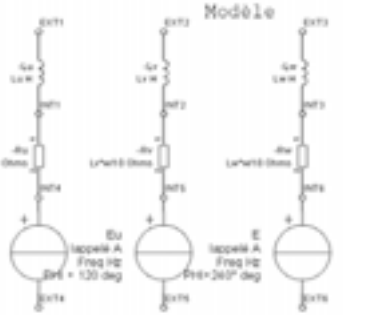


Moteur Synchrone Triphasé Sans Couplage interne

Symbole Base Elec.sim



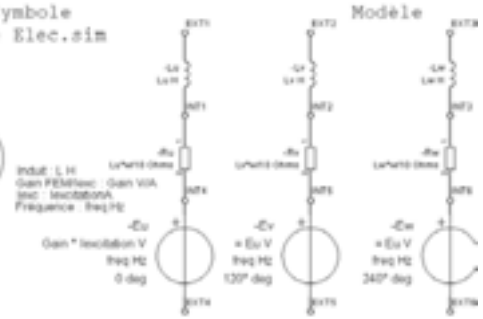
Modèle



Génératrice Synchrone Triphasé Sans Couplage interne

Symbole Base Elec.sim

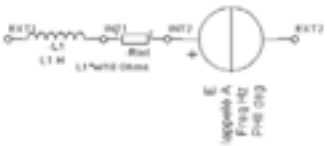
Modèle



Moteur Synchrone Monophasé

Symbole Base Elec.sim

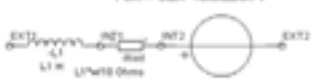
Modèle



Génératrice Synchrone Monophasée : Alternateur

Symbole Base Elec.sim

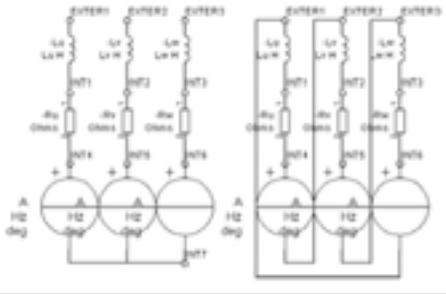
Modèle



Moteur Synchrone Triphasé Avec Couplage Interne

Symbole Base Elec.sim

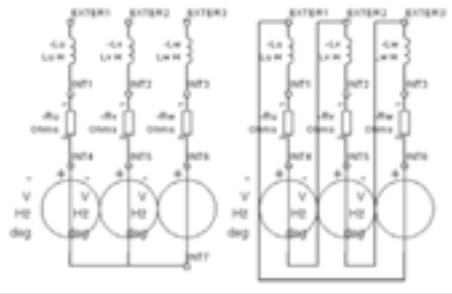
Modèle



Géné Synchrone Triphasé Avec Couplage Interne

Symbole Base Elec.sim

Modèle



Annexe 2 : Courbe de fusion des fusibles

Source LEGRAND catalogue 2000 Appareillage électrique d'installation.

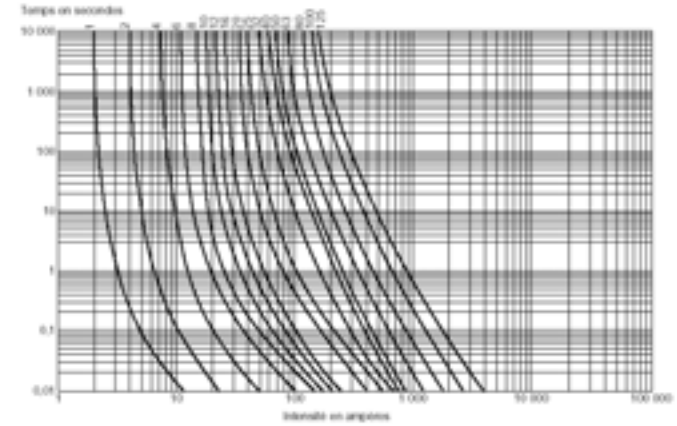


Fig. 1 Cartouche fusibles cylindriques type gG

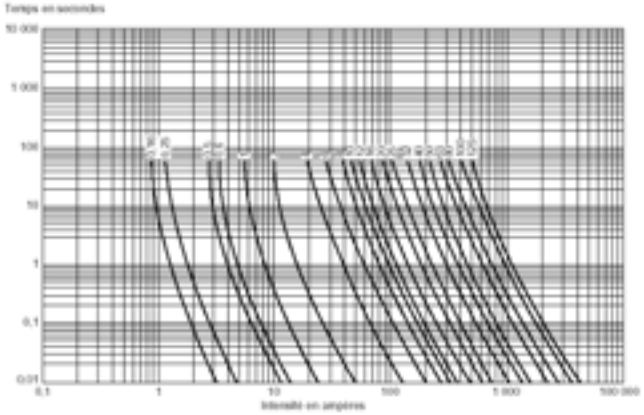


Fig. 2 Cartouche fusibles cylindriques type aM

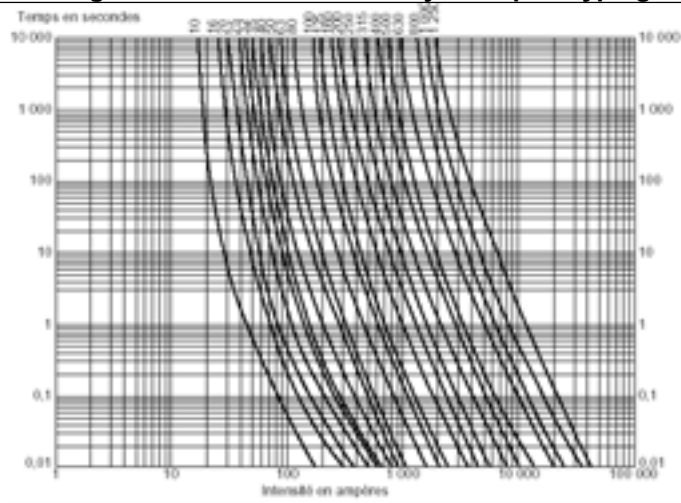


Fig. 3 Cartouche fusibles à couteaux type gG

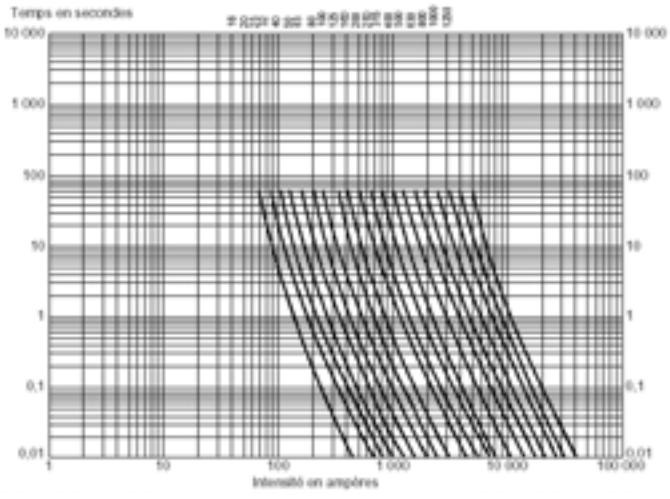


Fig. 4 Cartouche fusibles à couteaux type aM

Annexe 3 : Courbes de déclenchement des disjoncteurs magnétothermiques Merlin-Gérin C60.

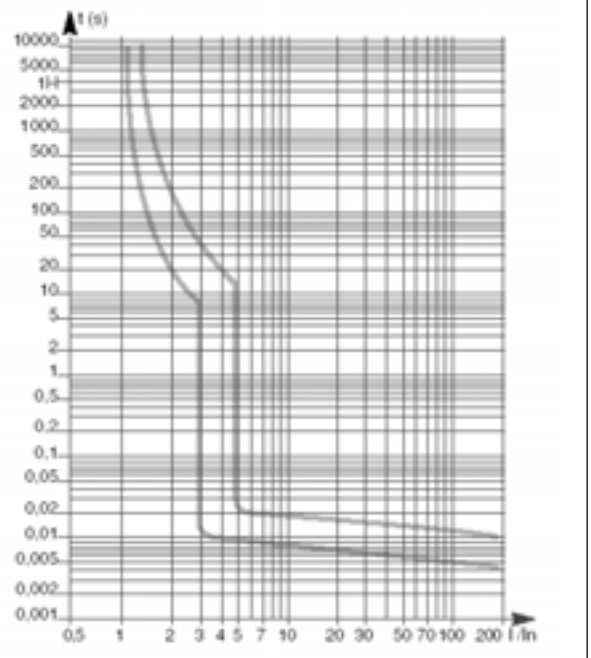


Fig. 5 Type B

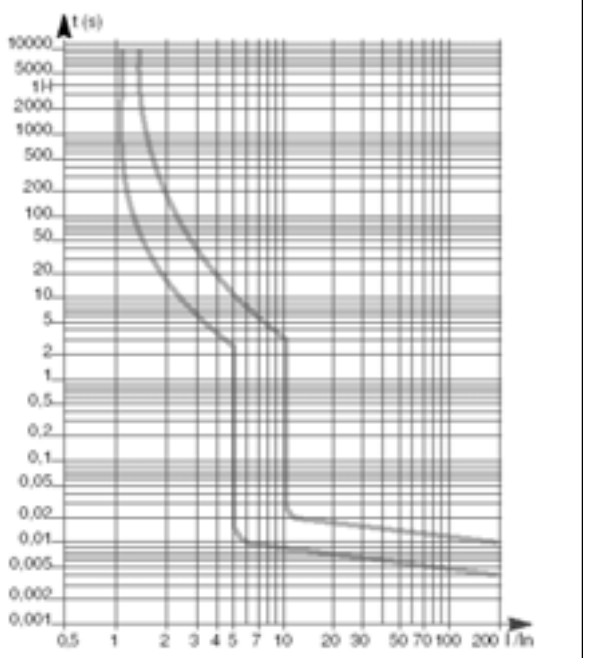


Fig. 6 Type C

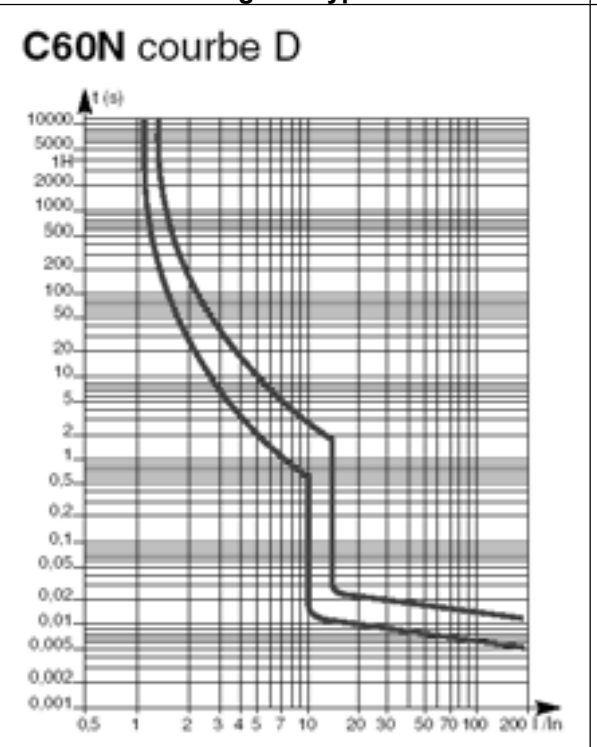


Fig. 7 Type D

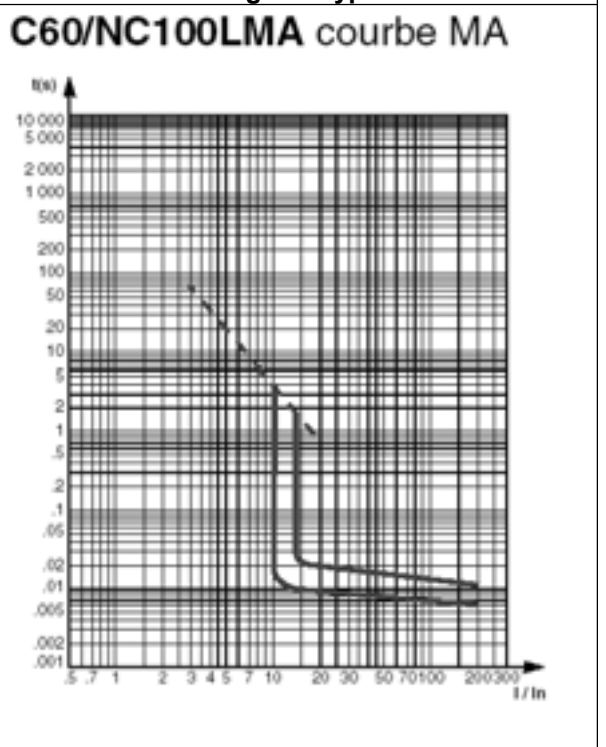


Fig. 8 Type MA

Annexe 4 : DTD « SchneiderML »

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 U (http://www.xmlspy.com) by Francis (Focal) -->
<!-- edited with XML Spy v2.5 NT  - http://www.xmlspy.com -->
<!--Description de rÃ©seau sur la base Composant/ Connection-->
<!ELEMENT Network (ListeComponents, ListeConnections)>
<!ATTLIST Network
	Name CDATA #REQUIRED
	Frequency CDATA #REQUIRED
	Voltage CDATA #REQUIRED
>
<!ELEMENT ListeComponents (Component*)>
<!ELEMENT Component (Property*)>
<!ATTLIST Component
	ComponentId ID #REQUIRED
	Type CDATA #REQUIRED
	Name CDATA #REQUIRED
>
<!ELEMENT Property EMPTY>
<!ATTLIST Property
	Code CDATA #REQUIRED
	Type (short | long | double | string) #REQUIRED
	Value CDATA #REQUIRED
	ShortLabel CDATA #IMPLIED
	LongLabel CDATA #IMPLIED
	MKSAUnit (none | m | mm2 | s | V | A | VA | W | VAr | Percentage)
#REQUIRED
>
<!ELEMENT ListeConnections (Connection*)>
<!ELEMENT Connection EMPTY>
<!ATTLIST Connection
	Name CDATA #REQUIRED
	NameRef CDATA #REQUIRED
	IdComponent1 IDREF #REQUIRED
	PinType1 (E | S | C) #REQUIRED
	PinNum1 CDATA #REQUIRED
	IdComponent2 IDREF #REQUIRED
	PinType2 (E | S | C) #REQUIRED
	PinNum2 CDATA #REQUIRED
>
```

Annexe 5 : Liste des fonctions non-supportées par hAMSter

Implementation Restrictions

The IEEE 1076.1 language, formally known as VHDL-AMS, is a superset of IEEE Std 1076-1993 (VHDL) that provides capabilities for describing and simulating analog and mixed-signal systems.
The documents contains the restrictions of the language subset in comparison with the IEEE 1076.1 standard.

Group	Description
	No configuration / configuration statements, the default settings are valid
Language structures	No behavior in entity describable; with the exception of assert
	No port map and generic map between blocks
Data types	No file , record , pointers
	No composite natures
	Range constraints
Statements	No generate
	No alias
	No group
	No userdefined attributes
	No resolution function, disconnections, postponed und guarded expressions
	No limit and tolerance
	No file I/O functions
	No attributes relating to type , nature , entity and terminal
Predefined attributes	No Q'Tolerance
	No Q'ZTF

Annexe 6 : Démarche de publication des modèles VHDL-AMS de l'association BEAMS.

La bibliothèque contient des modèles comportementaux écrits en VHDL-AMS. La documentation du modèle est considérée comme étant de la première importance afin de rendre le modèle vraiment « ré-utilisable ». Ainsi, à chaque modèle développé pour cette librairie sont associés un répertoire « nom_du_modèle » et plusieurs sous répertoires organisés comme suit (voir figure x) :

- un sous répertoire (*model*) dédié à la description du modèle (fichier *model_name.vhd*). Il contient aussi le fichier « *data sheet* » écrit selon le format habituel utilisé pour les « data-sheets » des circuits ou composants commerciaux. Les performances et limitations du modèle y sont clairement exposées, ainsi que la compatibilité avec le simulateur, et, de façon plus générale, toutes les informations qui permettent de rendre le modèle facile à utiliser.
- plusieurs autres sous répertoires (*test1, test2,… ,testn*) contenant la description aussi complète que possible des différentes procédures de test (test-bench) qui permettent de caractériser et valider le comportement du modèle. Le répertoire *testn* contient les fichiers suivants :
    - *testn.txt* est un fichier texte qui décrit les procédures de test (quelles sont les sources, les charges, les stimuli et les types d'analyse) et leurs objectifs
    - *testn.vhd* est le fichier source du modèle (ici en VHDL-AMS)
    - *testn.cmd* est le fichier de commande qui contient la définition de l'analyse (pour les simulateurs « Spice-like »)
    - *testn.ps* est un fichier postcript qui montre les courbes résultats de simulation pour le « testbench » concerné.

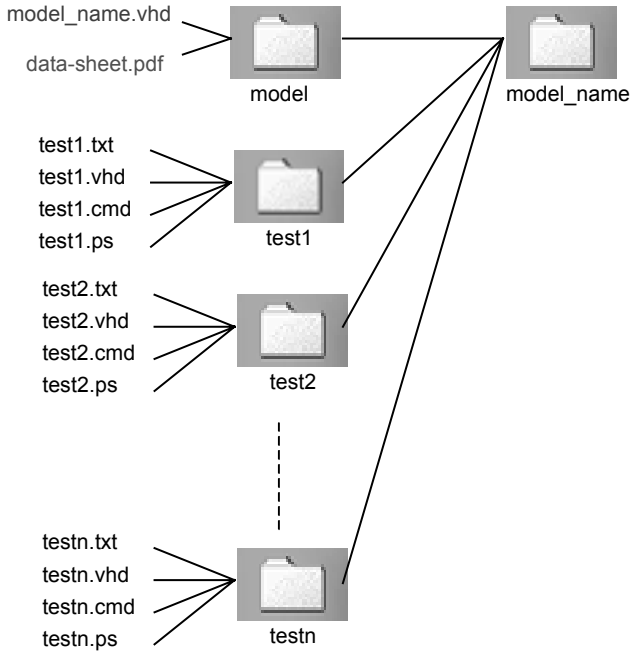


Fig. 9 Figure x: organisation du répertoire « model_name »

Annexe 7 : Modèles VHDL-AMS

```
-- Modèle VHDL-AMS d'un potentiomètre dont on connaît la résistance totale
-- et la position du curseur
-- Fabien LEGRAND

-- FL_POTENTIOMETRE_R1R2
-- potentiomètre constitué de 2 résistances en série dont on donne les valeurs en paramètre

-- Note les paramètres des composants sont des port (quantity) et pas des generic pour pouvoir leur
-- transmettre une quantity : exemple potentiomètre dont les résistances sont calculée à partir
-- de la résistance totale et de la position du curseur


LIBRARY DISCIPLINES;
LIBRARY IEEE;
LIBRARY WORK ;

USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;


ENTITY fl_potentiometreR1R2 IS
        PORT (  Quantity R1 : real := 1.0 ;    -- résistance avant le curseur
                Quantity R2 : real := 1.0 ;    -- résistance après le curseur
                TERMINAL Borne1,Curseur, Borne2  : ELECTRICAL); -- les 3 bornes du pot
END fl_potentiometreR1R2;

ARCHITECTURE fl_PotentiometreR1R2_BODY OF fl_potentiometreR1R2 IS
BEGIN
        Pot_R1 : ENTITY fl_resistance PORT MAP (R1, Borne1, Curseur) ;
        Pot_R2 : ENTITY fl_resistance PORT MAP (R2, Curseur, Borne2) ;
END fl_PotentiometreR1R2_BODY;

-- FL_POTENTIOMETRE_RtCur
-- Potentiomètre dont on donne en paramètre la résistance totale et la position du curseur en %

LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ENTITY FL_POTENTIOMETRE_RtCur IS

        PORT (  QUANTITY Rtotal : real := 1.0 ;    -- résistance totale
                QUANTITY PosCur : Real := 0.0 ;    -- position du curseur en %
                TERMINAL Borne1, Curseur, Borne2  : ELECTRICAL); -- les 3 bornes du pot
END FL_POTENTIOMETRE_RtCur;

ARCHITECTURE Pot_BODY_RR OF FL_POTENTIOMETRE_RtCur IS
-- potentiomètre constitué de deux résistances en série
   quantity  ValR1 , ValR2 : real ;
BEGIN
        ValR1 == Rtotal * PosCur / 100.0 ;
        ValR2 == Rtotal * (1.0 - (PosCur / 100.0)) ;
        Pot_R1 : ENTITY fl_resistance PORT MAP (ValR1, Borne1, Curseur) ;
        Pot_R2 : ENTITY fl_resistance PORT MAP (ValR2, Curseur, Borne2) ;
END Pot_BODY_RR;

ARCHITECTURE Sub_Pot_BODY OF FL_POTENTIOMETRE_RtCur IS
--potentiomètre utilisant le potentiomètre défini plus haut
  quantity ValR1 , ValR2 : real ;
BEGIN
        ValR1 == Rtotal * PosCur / 100.0 ;
        ValR2 == RTotal * (1.0 - PosCur / 100.0)   ;
        PtitPot : ENTITY fl_potentiometreR1R2 PORT MAP (ValR1, ValR2, Borne1, Curseur, Borne2) ;
END Sub_Pot_BODY ;
```

Fig. 10 Modèle VHDL-AMS potentiomètre

```
-------------------------------------------------------------------
-- test bench potentiometre
-- le potentiometre est alimenté par une source sinusoidale

LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL  ;

ENTITY TestbenchPot IS
END TestbenchPot;

ARCHITECTURE TestbenchPot_Body OF TestbenchPot IS
        TERMINAL n1,n2       : ELECTRICAL;
BEGIN
        Mon_Pot : ENTITY FL_POTENTIOMETRE_RtCur  (Pot_BODY_RR)
                                                PORT MAP (1.0, 20.0, n1, n2,  electrical_ground) ;
        Source  : ENTITY fl_sineSource PORT MAP ( n1, electrical_ground) ;
END TestbenchPot_Body;
```

Fig. 11 Testbench du potentiomètre

```
-------------------------------------------------------------------
-- source de tension sinusoidale triphasée
-- 3 sources de tension avec noeud commun et dephasée de 120°
--
-------------------------------------------------------------------

LIBRARY DISCIPLINES;
LIBRARY IEEE;

USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL ;
USE IEEE.MATH_REAL.ALL;

ENTITY fl_VoltTriphaseSource IS
            PORT( Quantity Veff, freq :real ;         -- valeur efficace, frequence, la phas est toujours nulle.
                        TERMINAL L1,L2,L3,N : ELECTRICAL );  -- bornes du composant 3 phases et neutre.
END;

ARCHITECTURE fl_VoltTriphaseSource_BODY OF fl_VoltTriphaseSource IS

BEGIN
        Ph1 : ENTITY fl_voltsinesource (Veff_Freq_Deg) PORT MAP ( Veff, Freq,    0.0, L1, N );
        Ph2 : ENTITY fl_voltsinesource (Veff_Freq_Deg) PORT MAP ( Veff, Freq, -120.0, L2, N );
        Ph3 : ENTITY fl_voltsinesource (Veff_Freq_Deg) PORT MAP ( Veff, Freq,  120.0, L3, N );
END fl_VoltTriphaseSource_BODY;
```

Fig. 12 Modèle VHDL-Ams d'une source sinusoïdale triphasée

```
-------------------------------------------------------------------
-- test bench source triphasee

LIBRARY DISCIPLINES;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;


ENTITY network IS
END;

ARCHITECTURE behav OF network IS
   TERMINAL n1,n2,n3: ELECTRICAL;
BEGIN
   Source  : ENTITY fl_VoltTriphaseSource PORT MAP ( 230.0, 50.0, n1,n2,n3, electrical_ground) ;
   R1: ENTITY FL_resistance    PORT MAP ( 1000.0, n1,electrical_ground);
   R2: ENTITY FL_resistance    PORT MAP ( 1000.0, n2,electrical_ground);
   R3: ENTITY FL_resistance    PORT MAP ( 1000.0, n3,electrical_ground);
END;
```

Fig. 13 Testbench de la source triphasée

```
-- Transformateur  : modèle de Kapp
-- l'impédance du tranfo est ramenée au secondaire
-- une résistance et une self en série avec l'enroulement du secondaire
--------------------------------------------------------------------------

LIBRARY DISCIPLINES;
LIBRARY IEEE;

USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;


ENTITY fl_transfo_kapp IS
        PORT (Quantity m : real := 1.0 ;     -- rapport de transformation = V1/V2
              Quantity  R : real := 1.0 ;     -- résistance ramenée au secondaire
              Quantity  L : real := 1.0 ;     -- inductance ramenée au secondaire

END fl_transfo_kapp;

ARCHITECTURE fl_transfo_kapp_BODY OF fl_transfo_kapp IS
        TERMINAL n1,n2: ELECTRICAL;

BEGIN
        Tr  : ENTITY fl_transfo    PORT MAP (m, p1, p2, N1, s2) ;
        Res : ENTITY fl_resistance PORT MAP (R,n1, n2) ;
        Ind : ENTITY fl_inductance PORT MAP (L, n2,S1) ;

END fl_transfo_kapp_BODY;
```

Fig. 14 Modèle VHDL-AMS d'un transformateur réel (modèle de Kapp)

```
-----------------------------------------------------------------
--testbench  Transformateur et sources controlee

LIBRARY DISCIPLINES;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;


ENTITY network IS
END;

ARCHITECTURE behav OF network IS
   TERMINAL n1,n2,n3,N4: ELECTRICAL;
BEGIN

        Valim: ENTITY fl_VoltSineSource (Veff_Freq_Deg) PORT MAP ( 10.0, 50.0, 0.0, n1, electrical_ground ) ;
        Re: ENTITY FL_resistance PORT MAP ( 1000.0, n1,n3);
        AMP : ENTITY fl_stcc PORT MAP ( 2.0 , n3, electrical_ground,n2,electrical_ground );
        RS: ENTITY FL_resistance PORT MAP ( 1000.0, n2,electrical_ground);
-- la simulation provoque une erreur 70136 "singular system matrix" : il faut rajouter une impedance d'entree
-- ou encore relier l'entree et la sortie des quadripoles à la masse

END;
```

Fig. 15 Testbench du transformateur réek

Annexe 8 : Banc de sélectivité des protections TBT

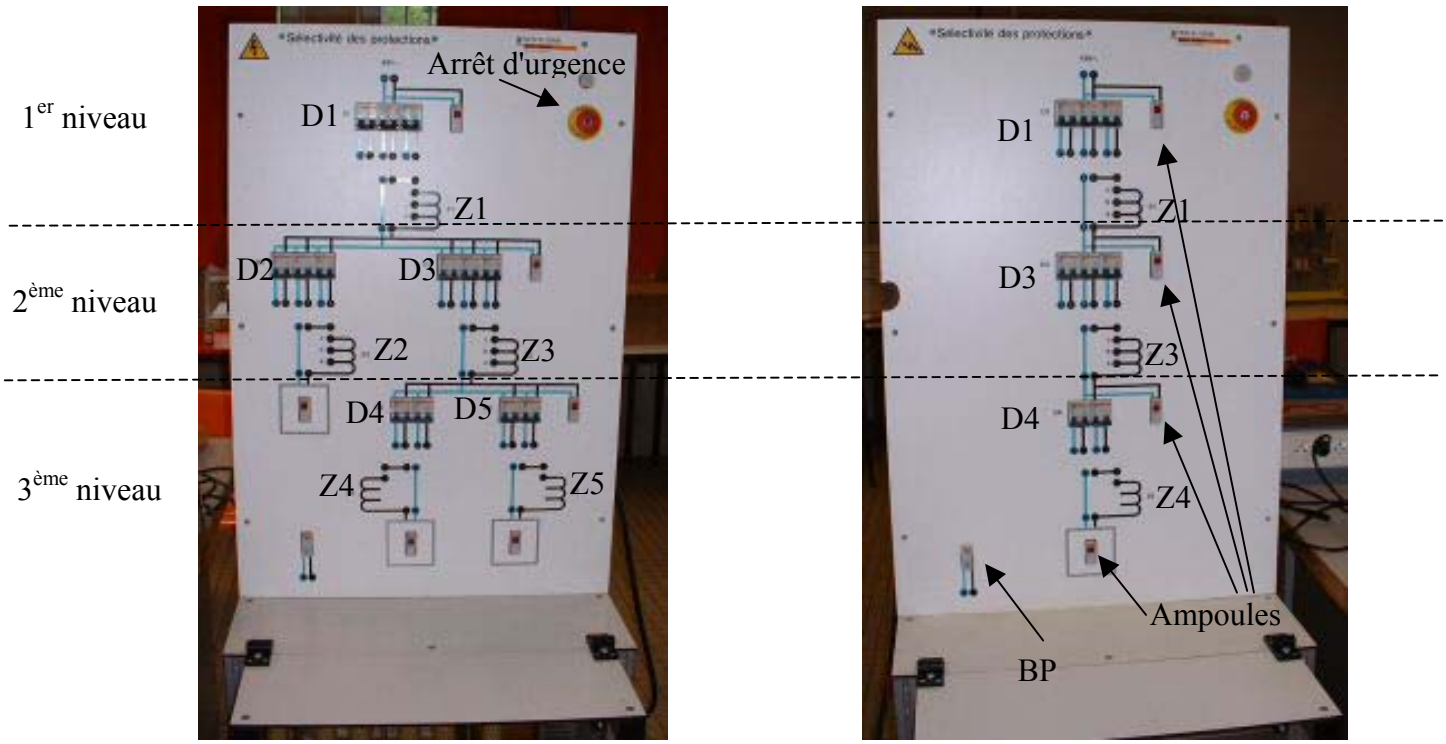


Fig. 16 Banc de sélectivité Merlin-Gérin coté 1

Fig. 17 Banc de sélectivité Merlin-Gérin coté 2

Annexe 9 : Impédances des éléments de banc didactique

Source (Ω)	Za	0,5300
	Ra	0,1000
	Xa	0,5200

Transformateur au secondaire (Ω)	Zs	0,3600
	Rs	0,0208
	Xs	0,0838

Disjoncteurs (par pôles) (Ω)	(inductif)
D1 (20A)	0,0074
D2 (10A)	0,0196
D3 (10A)	0,0196
D4 (6A)	0,1000
D5 (4A)	0,1500

Source au secondaire (Ω)	Z'a	0,0212
	R'a	0,0004
	X'a	0,0207

		a	b	c
Impédance Z1 (Ω)	Z1	0,0810	0,1773	0,2707
	R1	0,0800	0,1700	0,2600
	X1	0,0126	0,0502	0,0754

Impédances Z2 & Z3 (Ω)	Z2	0,0509	0,1048	0,1847
	R2	0,0500	0,1000	0,1700
	X2	0,0094	0,0313	0,0721

Impédances Z4&Z5 (Ω)	Z4	0,4720
	R4	0,3780
	X4	0,2826

Annexe 10 : Courant de court-circuit et temps de déclenchement théoriques pour le montage à un disjoncteur

			Valeurs totales des impédances utilisées (mΩ)		Impédance totale de défaut(mΩ)	**Courant de défaut(A)**	**Temps de déclenchement (ms)**	
Charge	Disj	Courbe	Réelle	Imaginaire	Zcc	**Icc**	min	max
Z2c	D1	B	170	72,1	273	**176**	10	17
		C	170	72,1	273	**176**	10	1800
		D	170	72,1	273	**176**	700	1800
	D2	B	170	72,1	273	**165**	8	20
		C	170	72,1	273	**165**	8	20
		D	170	72,1	273	**165**	8	20
	D3	B	170	72,1	273	**165**	8	20
		C	170	72,1	273	**165**	8	20
		D	170	72,1	273	**165**	8	20
Z4	D4	B	378	282,6	569	**67**	9	19
		C	378	282,6	569	**67**	9	19
	D5	B	378	282,6	569	**60**	8	20
		C	378	282,6	569	**60**	8	20

Annexe 11 : Courant de court-circuit, temps de déclenchement et sélectivité théoriques pour le montage à deux disjoncteurs

Courbe Disj D1	Courbe Disj D2	Impédances utilisées	Valeurs totales des impédances utilisées (mΩ)		Impédance totale de défaut (mΩ)	**Courant de défaut (A)**	Sélectivité	**Temps de déclenchement (ms)**			
			Réelle	Imaginaire	Zcc	**Icc**		**min D1**	**max D1**	**min D2**	**max D2**
B	B	Z1b, Z2c, Z4	718	404,9	932	**51**	Totale	8000	28000	10,5	29
		Z1c, Z2c	430	147,5	548	**88**	Partielle	13	6000	9	23
		Z1c, Z2a	310	84,8	414	**116**	Aucune	13	19	8,5	20,5
		Z1a, Z2b	180	43,9	288	**167**	Aucune	11	18	8	20
	C	Z1b, Z2c, Z4	718	404,9	932	**51**	Totale	8000	28000	3100	8000
		Z1c, Z2b	360	106,7	467	**103**	Aucune	12	20	9	21
		Z1a, Z2c	250	84,7	367	**131**	Aucune	11	19	8	20
		Z2a, Z3a	100	18,8	217	**221**	Aucune	9	14	7	18
	D	Z1c, Z2c	430	147,5	548	**88**	Aucune	13	6000	1400	2900
		Z1b, Z2b	270	81,5	380	**126**	Aucune	11	19	7,5	2000
		Z1a, Z2a	130	22	238	**202**	Aucune	10	15	6,5	19,5
C	B	Z1c, Z2c, Z4	808	430,1	1020	**47**	Totale	9000	30000	10,5	1000
		Z1b, Z2c	340	122,3	461	**104**	Totale	2000	5000	8,9	21
		Z1a, Z2a	130	22	238	**202**	Aucune	10	18	7,5	19,5
		Z1a	80	12,6	201	**239**	Aucune	9	17	7	18,5
	D	Z1b, Z2a	220	59,6	328	**146**	Partielle	11	2200	7	20
		Z1a, Z2a	130	22	238	**202**	Aucune	10	18	6,5	19,5
		Z1a	80	12,6	201	**239**	Aucune	9	17	6	18
D	B	Z1b, Z2a, Z4	598	342,2	799	**60**	Totale	5500	18000	10	19
		Z1a	80	12,6	201	**239**	Partielle	9	1100	7	18,5
	C	Z1c, Z2a	310	84,8	414	**116**	Totale	1800	3900	8,5	20,5
		Z1a	80	12,6	201	**239**	Partielle	9	1100	7	18
	D	Z1b, Z2c	340	122,3	461	**104**	Partielle	2000	5000	9	2300
		Z1a, Z2a	130	22	238	**202**	Partielle	10	1300	6,5	19,5

Annexe 12 : Courant de court-circuit, temps de déclenchement et sélectivité théoriques pour le montage à trois disjoncteurs

Courbe Disj D1	Courbe Disj D3	Courbe Disj D4	Impédances utilisées	Valeurs totales des impédances utilisées (mΩ)		Impédance totale de défaut (mΩ)	**Courant de défaut (A)**	Sélectivité	**Temps de déclenchement (ms)**					
				Réelle	Imaginaire	Zcc	**Icc**		min D1	max D1	min D3	max D3	min D4	max D4
D1B	D3B	D4B	Z1c, Z4, Z5	1016	640,6	1443	**33**	Part. (3/4)	30000	130000	11	19000	12	22
			Z1c, Z2c, Z4	808	430,1	1147	**42**	Part. (3/4)	13000	50000	10,5	10000	11	21
			Z1c, Z2c	430	147,5	680	**71**	Part. (1/3/4)	13	9000	10	27	9	20
			Z2b, Z3b	200	62,6	477	**101**	NS	12	20	9	22	8	19
		D4C	Z1c, Z2c, Z4	808	430,1	1147	**42**	NS (3/4)	13000	50000	10,5	10000	11	6500
			Z2a, Z4	428	292	793	**61**	NS (3/4)	6300	12000	10	29	9,5	20
			Z1b, Z2a	220	59,6	485	**99**	NS (1/3/4)	12	20	9	22	7	17
			Z1a, Z2a	130	22	411	**117**	NS (1/3/4)	11	19	7	21	6	16
	D3C	D4B	Z1c, Z2a, Z4	688	367,4	1017	**47**	Totale	10000	30000	3100	8000	10,5	20,5
			Z2c, Z3c	340	144,2	621	**77**	Part. (1/3/4)	13	8000	9,5	3500	9	19,5
			Z1b, Z2a	220	59,6	485	**99**	NS (1/3/4)	12	20	9	29	8	19
			Z1a, Z2a	130	22	411	**117**	NS (1/3/4)	11	19	7,5	21	7	14
		D4C	Z1a, Z4	458	295,2	813	**59**	Part. (3/4)	6000	20000	2200	5500	10	2400
			Z1b, Z2b	270	81,5	530	**91**	Part. (1/3/4)	12	5000	9	2800	8	18
			Z2b, Z3b	200	62,6	477	**101**	NS (1/3/4)	12	20	9	29	7	17
			Z1a, Z2a	130	22	411	**117**	NS (1/3/4)	11	19	7,5	21	6	16
	D3D	D4B	Z2b, Z4	478	313,9	840	**57**	Totale	6000	20000	2700	5800	10	20
			Z1b, Z2a	220	59,6	485	**99**	NS (1/3) Part. (1/4)	12	20	29	2500	8	19
			Z1a, Z2a	130	22	411	**117**	NS (1/3/4)	11	19	8	2000	7	14
			Z2a, Z3a	100	18,8	397	**121**	NS (1/3/4)	11	19	8	2000	7	14
		D4C	Z2b, Z4	478	313,9	840	**57**	Part. (1/3)	6000	20000	2800	7000	10	2400
			Z1b, Z2a	220	59,6	485	**99**	NS (1/3) Part. (1/4)	12	20	29	2500	7	17
			Z1a	80	12,6	386	**124**	NS (1/3/4)	11	19	8	2000	6	16

Annexe 13 : Mesure des courant de défaut et temps de déclenchement relevés sur un circuit à un seul disjoncteur.

			TEST1 19/06/03 Matin 20°				TEST2 19/06/03 Matin 22°				TEST3 20/06/03 Matin 20°				**MOYENNE**			
			Banc côté 1		Banc côté 2		Banc côté 1		Banc côté 2		Banc côté 1		Banc côté 2		**Banc côté 1**		**Banc côté 2**	
Charge	Disj	Courbe	Icc max (A)	Temps de coupure (ms)	Icc max (A)	Temps de coupure (ms)	Icc max (A)	Temps de coupure (ms)	Icc max (A)	Temps de coupure (ms)	Icc max (A)	Temps de coupure (ms)	Icc max (A)	Temps de coupure (ms)	**Icc max (A)**	**Temps de coupure (ms)**	**Icc max (A)**	**Temps de coupure (ms)**
Z2c	D1	B	192	6,22	196	6,20	184	5,92	160	7,11	188	5,89	192	10,70	**188,0**	**6,0**	**182,7**	**8,0**
		C	196	1480,00	200	25,00	188	1,52	196	26,78	196	1460,00	204	17,00	**193,3**	**980,5**	**200,0**	**22,9**
		D	192	1300,00	196	1220,00	188	1250,00	192	1140,00	188	1290,00	200	1240,00	**189,3**	**1280,0**	**196,0**	**1200,0**
	D2	B	168	4,70			162	4,92			164	4,44			**164,7**	**4,7**		
		C	160	4,86			160	8,92			170	10,86			**163,3**	**8,2**		
		D	164	6,84			164	15,44			164	6,89			**164,0**	**9,7**		
	D3	B	156	4,80	172	4,57	148	4,44	144	9,00	160	4,78	172	4,42	**154,7**	**4,7**	**162,7**	**6,0**
		C	160	5,82	168	5,00	156	7,44	168	5,89	156	4,96	172	10,25	**157,3**	**6,1**	**169,3**	**7,0**
		D	160	668,00	172	13,10	158	693,80	172	16,10	158	664,00	168	11,22	**158,7**	**675,3**	**170,7**	**13,5**
Z4	D4	B	84	5,73	96	5,73	92	5,89	96	7,11	96	11,81	96	5,49	**90,7**	**7,8**	**96,0**	**6,1**
		C	94	6,62	94	6,62	94	13,01	92	16,33	74	13,53	94	6,24	**87,3**	**11,1**	**93,3**	**9,7**
	D5	B	76	4,82			76	4,92			76	4,51			**76,0**	**4,8**		
		C	72	4,91			76	5,11			72	4,95			**73,3**	**5,0**		

Annexe 14 : Courant de défaut et temps de déclenchement relevés sur un circuit à deux disjoncteurs

			TEST1 19/06 Côté2 Aprem 23°			TEST2 20/06 Côté2 Matin 21°			TEST3 24/06 Côté2 Matin 21°			**MOYENNE**		
Courbe Disj D1	Courbe Disj D2	Impédances utilisées	Icc mesuré (A)	Disjoncteur qui coupe	Temps de coupure (ms)	Icc mesuré (A)	Disjoncteur qui coupe	Temps de coupure (ms)	Icc mesuré (A)	Disjoncteur qui coupe	Temps de coupure (ms)	**Icc mesuré (A)**	**Disjoncteur qui coupe**	**Temps de coupure (ms)**
B	B	Z1b, Z2c, Z4	68	2	24,9	68	2	16,4	64	2	16,3	**66,67**	**2**	**19,22**
		Z1c, Z2c	92	2	7,5	96	2	8,4	92	2	6,8	**93,33**	**2**	**7,54**
		Z1c, Z2a	112	2	5,7	112	2	11,3	96	2	5,4	**106,67**	**2**	**7,47**
		Z1a, Z2b	156	2	4,6	158	2	4,3	148	2	8,6	**154,00**	**2**	**5,82**
	C	Z1b, Z2c, Z4	64	2	5720,0	68	2	5750,0	68	2	5737,0	**66,67**	**2**	**5735,67**
		Z1c, Z2b	108	2	28,0	108	2	21,3	108	2	31,2	**108,00**	**2**	**26,82**
		Z1a, Z2c	132	2	6,2	130	2	5,8	124	2	10,4	**128,67**	**2**	**7,47**
		Z2a, Z3a	172	1&2	4,0	174	1&2	4,0	168	2	4,7	**171,33**	**1&2/1&2/2**	**4,24**
	D	Z1c, Z2c	96	2	1780,0	94	2	1770,0	96	2	1892,0	**95,33**	**2**	**1814,00**
		Z1b, Z2b	132	1	27,3	130	1	37,3	124	1	25,3	**128,67**	**1**	**29,95**
		Z1a, Z2a	176	1&2	12,5	174	1&2	5,4	170	1&2	13,0	**173,33**	**1&2**	**10,30**
C	B	Z1c, Z2c, Z4	56	2	23,4	58	2	18,3	60	2	27,2	**58,00**	**2**	**22,97**
		Z1b, Z2c	108	2	7,3	112	2	5,8	112	2	10,9	**110,67**	**2**	**8,00**
		Z1a, Z2a	176	2	4,2	128	2	3,7	170	2	3,9	**158,00**	**2**	**3,93**
		Z1a	188	2	16,0	228	2	3,5	204	2	6,8	**206,67**	**2**	**8,76**
	D	Z1b, Z2a	136	2	811,0	138	2	814,0	138	2	847,0	**137,33**	**2**	**824,00**
		Z1a, Z2a	172	2	16,0	172	2	14,3	170	2	6,7	**171,33**	**2**	**12,35**
		Z1a	224	2	4,0	228	2	3,8	216	2	3,9	**222,67**	**2**	**3,90**
D	B	Z1b, Z2a, Z4	72	2	14,2	72	2	7,6	74	2	10,2	**72,67**	**2**	**10,65**
		Z1a	152	2	3,3	216	2	4,3	224	2	3,8	**197,33**	**2**	**3,78**
	C	Z1c, Z2a	112	2	18,9	116	2	15,6	116	2	16,7	**114,67**	**2**	**17,07**
		Z1a	224	2	3,6	232	2	3,5	218	2	3,8	**224,67**	**2**	**3,63**
	D	Z1b, Z2c	108	2	1290,0	112	2	1300,0	108	2	1360,0	**109,33**	**2**	**1316,67**
		Z1a, Z2a	172	2	5,9	176	2	5,6	172	2	5,9	**173,33**	**2**	**5,79**

Annexe 15 : Courant de défaut et temps de déclenchement relevés sur un circuit à trois disjoncteurs

				TEST1 20/06 Aprem 23°			TEST2 23/06 Aprem 21°			TEST3 24/06 Aprem 22°			**MOYENNE**		
Courbe Disj D1	Courbe Disj D3	Courbe Disj D4	Impédances utilisées	Icc mesuré (A)	Disjoncteur qui coupe	Temps de coupure (ms)	Icc mesuré (A)	Disjoncteur qui coupe	Temps de coupure (ms)	Icc mesuré (A)	Disjoncteur qui coupe	Temps de coupure (ms)	**Icc mesuré (A)**	**Disjoncteur qui coupe**	**Temps de coupure (ms)**
D1B	D3B	D4B	Z1c, Z4, Z5	44	4	16,5	48	4	17,1	46	4	17,1	**46,0**	**4**	**16,9**
			Z1c, Z2c, Z4	54	4	12,2	54	4	11,4	54	4	8,7	**54,0**	**4**	**10,8**
			Z1c, Z2c	78	3 et 4	10,0	76	3&4	5,2	78	3&4	5,4	**77,3**	**3&4**	**6,8**
			Z2b, Z3b	104	3 et 4	4,1	92	3&4	4,2	104	3&4	5,0	**100,0**	**3&4**	**4,4**
		D4C	Z1c, Z2c, Z4	54	3	60,5	56	3	114,4	56	3	61,6	**55,3**	**3**	**78,8**
			Z2a, Z4	80	4	7,5	76	3&4	6,8	76	3&4	6,7	**77,3**	**4/3&4/3&4**	**7,0**
			Z1b, Z2a	102	3 et 4	10,3	104	3&4	4,4	104	3&4	9,1	**103,3**	**3&4**	**7,9**
			Z1a, Z2a	122	3 et 4	6,8	120	3&4	6,2	124	3&4	3,6	**122,0**	**3&4**	**5,6**
	D3C	D4B	Z1c, Z2a, Z4	58	4	6,8	56	4	6,9	56	4	6,7	**56,7**	**4**	**6,8**
			Z2c, Z3c	84	4	7,5	84	4	5,0	82	4	10,4	**83,3**	**4**	**7,7**
			Z1b, Z2a	104	4	3,4	104	4	7,3	106	4	4,0	**104,7**	**4**	**4,9**
			Z1a, Z2a	122	4	2,8	126	4	3,4	126	4	3,3	**124,7**	**4**	**3,2**
		D4C	Z1a, Z4	74	4	6,1	76	4	15,4	76	4	11,4	**75,3**	**4**	**11,0**
			Z1b, Z2b	98	4	5,0	78	4	7,1	96	4	4,4	**90,7**	**4**	**5,5**
			Z2b, Z3b	104	4	4,5	104	4	4,3	104	4	4,4	**104,0**	**4**	**4,4**
			Z1a, Z2a	122	4	5,5	124	4	3,8	118	4	8,5	**121,3**	**4**	**5,9**
	D3D	D4B	Z2b, Z4	74	4	6,1	58	4	5,4	74	4	6,2	**68,7**	**4**	**5,9**
			Z1b, Z2a	100	4	8,3	106	4	3,7	100	4	8,4	**102,0**	**4**	**6,8**
			Z1a, Z2a	116	4	6,4	104	4	2,9	124	4	3,5	**114,7**	**4**	**4,3**
			Z2a, Z3a	106	4	2,9	124	4	3,5	124	4	3,6	**118,0**	**4**	**3,3**
		D4C	Z2b, Z4	72	4	7,7	76	4	12,6	74	4	7,4	**74,0**	**4**	**9,2**
			Z1b, Z2a	104	4	4,0	106	4	5,4	104	4	4,6	**104,7**	**4**	**4,7**
			Z1a	142	4	3,3	142	4	3,1	142	4	3,2	**142,0**	**4**	**3,2**

Annexe 16 : Banc de test évolutif, comparatif des valeurs théoriques simulées et mesurées pour le déclenchement sous 230V

		Valeurs Théoriques			Valeurs Relevées			Valeurs Simulées	
Disjoncteur	Nombre d'ampoule	Courant eff (A)	Temps de déclenchement min (s)	Temps de déclenchement max (s)	Courant eff (A)	Courant crête (A)	Temps de déclenchement (s)	Courant eff (A)	Temps de déclenchement (s)
C - 1A	1	0,43	infini	infini	0,42	0,6	infini	0,45	infini
C - 0,5A		0,43	infini	infini	0,42	0,6	infini	0,45	infini
D - 0,5A		0,43	infini	infini	0,42	0,6	infini	0,45	infini
C - 1A	2	0,87	infini	infini	0,92	1,3	infini	0,90	infini
C - 0,5A		0,87	20	400	0,87	1,23	60	0,90	21,3
D - 0,5A		0,87	40	400	0,87	1,23	47	0,90	47,4
C - 1A	3	1,30	40	1000	1,30	1,84	infini	1,35	102
C - 0,5A		1,30	9	60	-	-	instantané	1,35	7,45
D - 0,5A		1,30	9	50	1,24	1,75	14	1,35	8,65
C - 1A	4	1,74	20	400	1,68	2,37	94	1,80	21,3
C - 0,5A		1,74	4	25	-	-	instantané	1,80	4,26
D - 0,5A		1,74	4	23	1,61	2,28	9,5	1,80	4,05
C - 1A	5	2,17	10	90	2,33	3,3	37,5	2,25	10,9
C - 0,5A		2,17	3	17	-	-	instantané	2,25	3,1
D - 0,5A		2,17	2,5	13	2,33	3,3	6	2,25	2,62
C - 1A	6	2,61	9	60	2,72	3,84	26	2,71	7,5
C - 0,5A		2,61	0,01	9	-	-	instantané	2,71	0,051
D - 0,5A		2,61	1,5	7	2,53	3,58	5	2,71	1,88

Légende :	Correspondant à la théorie		Proche de la théorie			Eloigné de la théorie		

Annexe 17 : Mode de pose des câbles pour la mesure de la chute de tension

Fig. 18 Déploiement câble en accordéon

Fig. 19 Déploiement câble en spires

Annexe 18 : Captures d'écran de Simul'Elec

Fig. 20 Interface de Simul'Elec en régime permanent

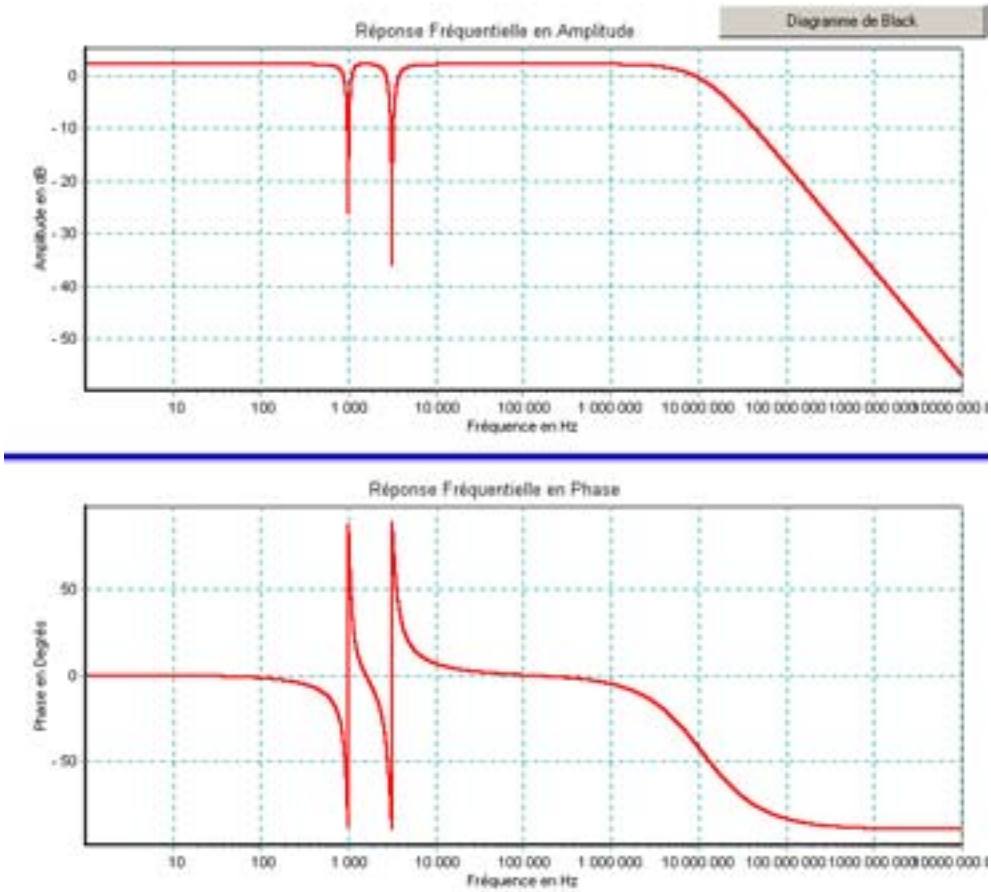


Fig. 21 Affichage des diagrammes de Bode en analyse fréquentielle de Simul'Elec

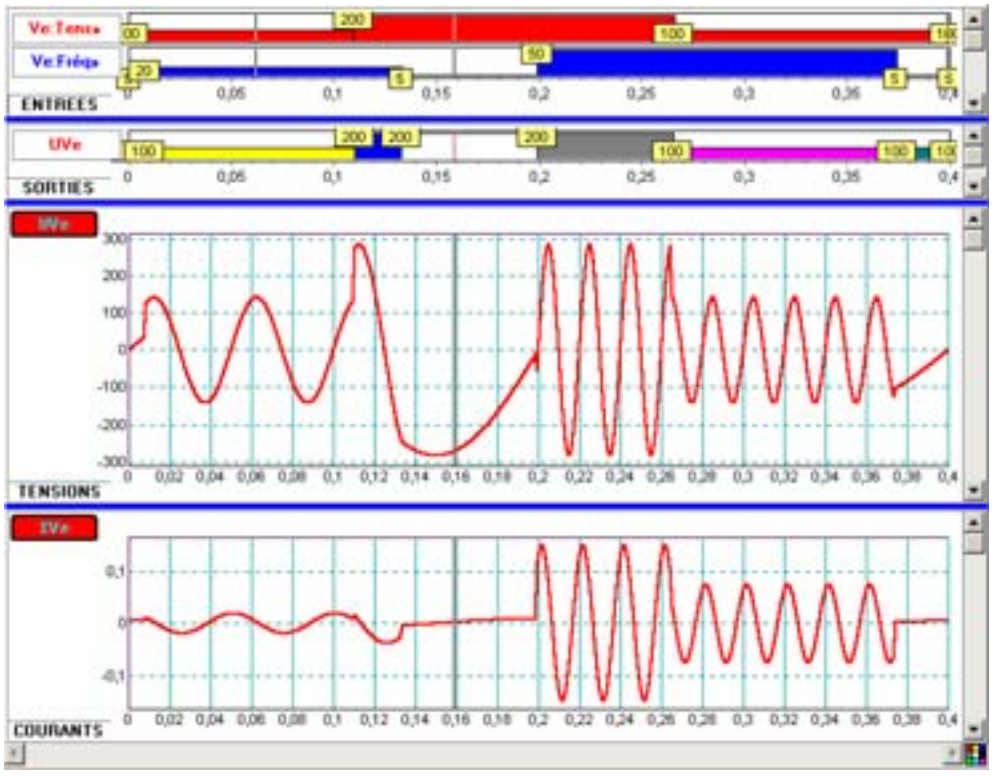


Fig. 22 Analyse événementielle de Simul'Elec

Lexique

Circuit (ou macro-composant) : un assemblage de plusieurs composants élémentaires ou de circuits. On peut décomposer un circuit en un ensemble de composants élémentaires disposé sur plusieurs niveaux.

Composant (électrique) : d'un manière générale, désigne un élément d'un circuit électrique. Une notion plus précise appliquée à Simul'Elec désigne un composant basique qui n'est pas un assemblage de différents éléments électriques.

Electronique : La partie de la science et de la technique qui étudie les phénomènes de conduction dans le vide, dans les gaz ou dans les semi conducteurs et qui utilise les dispositifs basés sur ces phénomènes. Par extension, l'électronique est l'ensemble des techniques qui utilisent des signaux électriques pour capter, transmettre et exploiter une information.

Electronique de puissance : Application de l'électronique à la conversion électrique - électrique de l'énergie.

Electrotechnique : Ensemble des applications industrielles de l'électricité. Cela comprend la production, le transport, la distribution et l'usage de l'énergie électrique.

Modèle : Représentation, physique ou mathématique, des relations qui existent réellement ou qui, par hypothèse, semblent exister entre des phénomènes ou entre les différents éléments d'un système, en vue d'études analytiques ou expérimentales (simulations) propres à faciliter la compréhension de certains mécanismes, notamment par la validation d'hypothèses.

Netlist : désigne dans le simulateur SPICE, une liste de composants et de leur interconnections permettant de décrire la structure d'un circuit électrique.

Schématique : Représentation d'un système par un dessin. La schématique électrique est une discipline de l'électrotechnique.

SED : Simulation par Evénements discrets, technique de simulation pilotée par l'apparition d'événements.

Simulateur : Logiciel permettant de reproduire fidèlement les conditions et les situations réelles qui remplace tout ou partie d'un dispositif ou de son environnement à des fins d'étude ou d'essai.

TestBench : Banc de test. En modélisation VHDL-AMS, c'est un circuit permettant de valider un modèle en le mettant

MODELISATION DE CIRCUITS ELECTROTECHNIQUES EN VUE DE LEUR SIMULATION - REALISATION D'UN SIMULATEUR

RESUME

L'objectif de la thèse est la réalisation d'un simulateur de circuits électrotechniques et réseaux électriques. Il permet d'estimer le comportement physique des installations et ainsi d'améliorer leur fiabilité tout en optimisant leur coût de conception. Privilégiant la simplicité d'utilisation, ce simulateur est présenté comme une alternative aux simulateurs classiques (comme SPICE) pour rendre la simulation accessible à un public plus large. Après avoir présenté l'état de l'art du domaine, une bibliothèque de modèles de composants et des modes d'analyses innovants sont développés pour ce simulateur. La mise en place d'un banc d'essais et les tests associés ont permis de confirmer la pertinence des modèles et l'efficacité des modes d'analyse. Une ouverture vers le langage de modélisation VHDL-AMS est proposée afin de permettre ensuite au simulateur de disposer de modèles standards et d'utiliser des descriptions multi-niveaux, multi-technologique et multi-domaines.

MOTS CLES

Simulation, modélisation, électrotechnique, schématique électrique.

ABSTRACT

The aim of this thesis is the implementation of a electrotechnical circuits or electrical networks simulator. It enables to estimate physical behaviour of electrical installations and thus to improve its reliability, optimizing design costs. Prioritizing the use easiness, this simulator is shown as an alternative to classical simulators (such as SPICE) to give access to simulation to a wider audience. After a presentation of a state of the art of the discipline, a component models library and innovative analysis modes are developed for this simulator. The setting up of a test bench and the associated tests allowed to confirm models pertinence and analysis modes efficiency. An opening to VHDL-AMS modelling language is proposed so as to enable then the simulator to dispose of standard models and to use multi levels, multi technological or multi disciplines descriptions.

KEY WORDS.

Simulation, modelling, electrical engineering, electrical drawing.