UNIVERSITÉ LIBRE DE BRUXELLES

FACULTÉ DES SCIENCES SOCIALES,
POLITIQUES ET ÉCONOMIQUES

SECTION INFORMATIQUE ET SCIENCES HUMAINES

Elaboration d'une application de la méthode Activity Based Costing utilisant les technologies XML.

Jean Mairesse

Mémoire présenté en vue de l'obtention du grade de Licencié en Informatique et Sciences humaines, sous la direction de:

Année académique 2008-2009

Monsieur Luc Bernard

Table des matières

Objectifs du mémoire.

L'objectif de base de ce mémoire est de mettre au point une application de comptabilité analytique se basant sur la méthode Activity Based Costing.

Cet outil analytique est destiné à être utilisé par des TPE (Très Petite Entreprise) et PME (Petite et Moyenne Entreprise) qui sont bien souvent démunie en matière de comptabilité analytique et de tableau de bord de gestion.

Le second objectif est de concevoir cette application Activity Based Costing en faisant usage du métalangage eXtensible Markup Langage (XML).
Cela implique que nous devrons concevoir une représentation des données comptables et analytiques en respect avec les règles émises par le World Wide Web Consortium (W3C).

Les résultats analytiques relèvent de transformations et calculs sur les données de base (en XML). Nous souhaitons produire ces résultats et les présenter à l'aide d'outils de recherche et transformation issus monde XML (XQuery), et non pas en développant une solution logicielle dédiée.
Notre démarche sera illustrée par un exemple de comptabilité TPE.

Un troisième objectif, tant pour la partie rédactionnelle de ce mémoire, que pour le développement XML, est d'utiliser des logiciels disposant d'une licence libre.
Une recherche et évaluation de solutions logicielles XML sous licence libre sera entreprise.

Nous nous proposons d'organiser notre travail de la façon suivante.

Chapitre 1: nous détaillons la méthode Activity Based Costing.
Chapitre 2: nous présentons eXtensible Markup Language (XML).
Chapitre 3: les logiciels libres sont traités.
Chapitre 4: modélisations ERD et hiérarchique.
Chapitre 5: nous créons le fichier ABC/XML et la DTD.
Chapitre 6: nous présentons XQuery.
Chapitre 7: les requêtes XQuery sont développées.
Chapitre 8: présentation des résultats.
Chapitre 9: suite du mémoire et conclusions.

Chapitre 1. La méthode Activity Based Costing.

Dans ce chapitre, nous présentons le contexte comptable qui est celui des TPE et PME, distinguons la comptabilité générale de la comptabilité analytique, et en décrivons brièvement quelques méthodes avant de détailler la méthode de comptabilité analytique Activity Based Costing (ABC). Ensuite, sur base d'un exemple inspiré d'une entreprise réelle, nous élaborons la mise au point d'une application de la méthode ABC qui servira de base au travail informatique que nous présenterons dans les prochains chapitres.

Contexte comptable.

Toute entreprise se doit de répondre aux obligations légales en matière de comptabilité.

Dans le plus classique des cas, la démarche comptable suivie par les TPE et PME - nous considérons le cas des structures de petite taille, qui ne sont pas dotées d'un service de comptabilité interne - en matière de gestion comptable, consiste à remettre les pièces comptables (factures de ventes et factures d'achats) à un comptable externe, qui est chargé de tenir une comptabilité.

Le comptable externe, outre la tenue de la comptabilité, établi mensuellement ou bien trimestriellement, une « déclaration à la TVA », qui est un résumé des « opérations à la sortie » (opérations de vente - TVA due à l'administration) et des « opérations à l'entrée » (opérations d'achat - TVA due et déductible); le but de cette déclaration est de déterminer la différence entre TVA due et TVA déductible.
Une « déclaration à l'impôt » est établie une fois par an.

Nous mentionnons ces deux déclarations car elles sont l'occasion pour l'entreprise d'avoir accès à ses résultats comptables.
Outre le fait de pouvoir évaluer l'ampleur du bénéfice (ou la perte), ces résultats permettent de prendre conscience du poids des postes de charge dans le résultat.
Ils ne permettent par contre pas d'appréhender la rentabilité effective des produits ou services de l'entreprise.

Il est cependant tout à fait possible, à partir de ces mêmes pièces comptables, de déterminer la rentabilité de chacun de ces produits ou services.
C'est ce que nous tenterons de réaliser au fil de ce mémoire.

Méthodes de comptabilité.

Il existe différentes formes de comptabilité.
Nous distinguerons la comptabilité générale qui est encadrée par la loi, de la comptabilité analytique.
Nous les présentons ci-dessous.

La comptabilité générale.

La comptabilité générale est un classement par nature des pièces comptables, qui offre une vue sur les mouvements financiers, permet de comparer les situations comptables entre périodes et entreprises, et l'élaboration de statistiques économiques.

La comptabilité générale est encadrée par la loi, qui impose depuis le 17 juillet 1975 à toute entreprise de tenir une comptabilité conforme aux dispositions en matière de comptes annuels et de présentation.

Les comptes annuels: à partir de pièces comptables probantes, les factures d'achat et de vente, sont établis les bilans, comptes de résultats et annexes.

Pour ce qui concerne la présentation, la législation défini un plan comptable, le « Plan Comptable Minimum Normalisé » (PCMN).

Signalons que sous réserve d'être considérée comme une PME (occuper moins de 250 employés, et soit moins de 40 millions d'Euro de chiffre d'affaire ou soit moins de 27 millions d'Euro au total du bilan et respecter le critère d'indépendance), une entreprise peut utiliser un schéma comptable abrégé.
Les commerçants et personnes physiques peuvent tenir une comptabilité simplifiée si leur chiffre d'affaire annuel ne dépasse pas 500.000 Euro (hors TVA).

Pour le développement ultérieur de ce mémoire, nous retiendrons de la comptabilité générale la notion de « Plan Comptable Minimum Normalisé » (PCMN).

La comptabilité analytique.

La comptabilité analytique peut contribuer à déterminer le prix de revient d'un produit, un service ou une commande.
Elle utilise la notion de charges directes et indirectes.

Les charges directes sont celles qui peuvent imputées directement au prix de revient du produit ou service de l'entreprise.
Ce sont les matières premières ou fournitures que l'on retrouve dans le produit final de l'entreprise ainsi que des coûts salariaux de production.

Les charges indirectes sont les frais n'intervenant pas directement dans le produit final.
Nous y trouvons les frais relatifs à l'immeuble où opère l'entreprise, les coûts de son outil de production, les frais publicitaires,etc...
Les charges indirectes sont liées à l'ensemble de l'exercice de l'entreprise, et doivent donc être transformées pour intégrer le prix de revient du produit final.

La comptabilité analytique propose un classement par destination.
Exemple: en première destination, nous avons l'entretien d'un ascenseur, qui en seconde destination sert dans le cadre de la production..

La comptabilité analytique qui est un outil interne à l'organisation, n'est pas encadrée légalement.

La comptabilité analytique revêt diverses formes ou méthodes.
Nous présentons quelques méthodes de détermination de prix de revient.

La méthode des coûts complets.

Elle prend en compte toutes les charges affectables à un produit/service, de sa production à sa commercialisation, tant les charges directes que les charges indirectes, qu'une analyse de l'entreprise juge opportune d'y incorporer.

Cette méthode est lourde à mettre en oeuvre, et la répartition des charges indirectes entre unités de production par des clés de répartition est arbitraire et peu précise, voire peu objective.

Principe de fonctionnement de la méthode des coûts complets: les charges directes sont réparties parmi les produits de l'entreprise (produire le meuble « a » a consommé une quantité « x » de charges directes, le meuble « f » une quantité « y »).
Les charges indirectes sont ensuite réparties selon des clés de répartition comme la proportion de matière première consommée.

C'est une méthode très ancienne qui répondait à des formes d'entreprises (manufactures) qui deviennent plus rares de nos jours, et n'avaient pas de contraintes de marchés...

Les sections homogènes.

Les charges directes sont imputées aux produits de l'entreprise.
Pour répartir les charges indirectes au sein de ces produits, l'entreprise est divisée en sections (ateliers, magasins,...)

A chaque section correspondent des coûts (des charges indirectes); ces charges indirectes sont réparties dans toutes les sections proportionnellement à l'effort qu'elles (les sections) ont consacré aux produits de l'entreprise (exemple: l'atelier représente 48% des charges indirectes des produits de l'entreprise, le magasin 17%, etc..).
Une distinction est opérée entre sections principales (atelier, service commercial,..) et sections auxiliaires (gestion, entretien,..).
Les sections auxiliaires sont réparties au sein des sections principales selon des clés de répartition (la section auxiliaire entretien se réparti dans les sections principales atelier et service commercial).

Le coût de chaque section est connu, il reste à le diviser par la quantité de produits ayant été traités. Ce sera l'unité d'oeuvre de la section considérée, et son coût.
La somme des unités d'oeuvre de chacune des sections donnera le prix de revient indirect du produit.

Homogène car chaque section est bien délimitée et ne peut être confondue avec une autre.
La section homogène est une forme de coût de revient complet.

Le Direct Costing.

Direct Costing utilise les notions de coûts variables et coûts fixes.
Les coûts fixes sont ceux qui ne dépendent pas du volume fabriqué; typiquement, l'outil de production qu'il faut payer, qu'on l'utilise ou pas.
Les charges variables sont celles qui varient quand varie le volume de l'activité; une charge variable

peut être une charge directe ou une charge indirecte.

Le Direct Costing permet de connaître combien coûte de produire une unité supplémentaire (marginale).

Le Direct Costing impute aux produits les seuls coûts variables.
Dans cette optique, l'entreprise doit couvrir ses frais fixes, qui sont connus, par la marge réalisée sur ses produits: prix de vente – coût variable.
Cela permet de connaître le volume de vente à réaliser pour couvrir ces coûts fixes.

La méthode Activity Based Costing.

La méthode Activity Based Costing propose de répartir les charges au sein des produits, services d'une entreprise, proportionnellement à ce qu'ils en ont réellement consommé.

La méthode Activity Based Costing fait intervenir les concepts d'« activités » et de « processus ».

Activity Based Costing découpe une entreprise en entités consommatrices de ressources (services, départements,...).
Ces entités effectuent diverses tâches, qui lorsqu'elles sont complémentaires, sont regroupées au sein d'« activités ».
Une « activité » peut donc être définie comme étant un ensemble de tâches complémentaires.

Enfin, un processus est un ensemble d' « activités » complémentaires menant l'entreprise à atteindre son objectif final, qui est généralement de produire et vendre des biens et/ou services.
Plusieurs processus peuvent intervenir simultanément ou successivement.

Lorsqu'ont été définies les activités, les charges de l'entreprise y sont distribuées.
Les activités ont alors un coût.
Il convient de répartir ce coût au sein des produits ou services de l'entreprise au prorata de ce qu'ils en ont consommé.

Pour cette répartition, des fractions d'« activités » sont créées.
Par exemple, une heure de production, une heure de travail administratif, etc...
Ces fractions sont des « unités d'oeuvre », qui ont un coût.

Il suffit alors de procéder à des mesurages au fil du(es) processus d'élaboration suivi par tel produit ou commande jusqu'à son aboutissement, afin de comptabiliser le nombre d'unités d'oeuvre des différentes « activités » qui ont été utilisées.

La somme de la multiplication des quantités consommées d'« unités d'oeuvre » par leur coût respectif permet de connaître la part de coût indirect à attribuer à la commande ou au produit.

Mise en oeuvre de la méthode Activity Based Costing.

A titre d'exemple, nous prenons le cas d'une TPE active dans le secteur des énergies renouvelables, plus précisément la vente et placement d'installations solaires thermiques.
Il s'agit d'un secteur ayant probablement de l'avenir, mais dont le démarrage est lent.
Le mode de fonctionnement en est la réalisation à la commande.

C'est à dire qu'un prix est fixé par avant pour réaliser tel type de commande, il ne variera pas quels que soient les aléas de production.

Notre choix porte sur la méthode Activity Based Costing pour son côté dynamique.
Les méthodes que nous décrivions précédemment sont plus adaptées à des entreprises qui, dans le passé, pouvaient planifier à moyen voire long terme.
Alors que le domaine de l'entreprise en exemple permet de planifier à quelques mois.

La notion de processus au fil duquel sont réparties les charges, comme le propose Activity Based Costing, permet d'envisager une modification de ce processus sans devoir repenser intégralement la méthode d'analyse de coûts en place.

Application pratique.

Les ouvrages que nous avons consulté pour cette partie ABC prenaient des exemples de type « General Electric », « Renault », « La Poste », et d'autres entreprises de taille moindre, mais qui ne sont pas des PME.
Nous devons donc simplifier la méthode en ce que nous qualifierons d'application de la méthode Activity Based Costing.

En matière de processus, nous sommes tentés de définir l'entièreté de l'entreprise comme un seul processus. Il n'y a pas de département, ni division, l'entreprise est une seule entité.
Il y a par contre bel et bien des activités.
Notre application consistera donc en un processus qui comporte des activités.

Notre démarche sera la suivante:
- En une première étape, nous déterminons quelles sont les charges indirectes.
  Nous regroupons ensuite ces charges indirectes par destination en « famille de coûts ».
- En seconde étape, le fonctionnement de l'entreprise est examiné afin d'en cerner le processus.
- Du processus sont déduites les activités au sein desquelles devront être réparties les familles de coût.

Charges indirectes et familles de coût.

Afin de créer les familles de coût, nous nous appuyons sur la classification de la comptabilité générale, classification correspondant au plan comptable minimum normalisé (pcmn).

L'entreprise en exemple utilise les postes pcmn suivants pour l'imputation des charges indirectes.

601300: outillage
606140: frais transport
611130: location outillage
611150: location véhicule
611300: entretien locaux
611350: entretien véhicule
612000: énergie immeuble
612160: carburant
612300: documentation
612400: imprimés fournitures bureau

612500: petit mat. bureau
613250: honoraire comptable
613530: assurance véhicule
613540: assurance RC
613550: assurance immeuble
615100: frais représentation
615200: publicité annonces
615220: foires expositions
616100: frais postaux
616200: téléphone
616300: gsm
616400: internet
630210: dotation amortissement immeuble
630220: dotation amortissement outil
630231: dotation amortissement mobilier
630232: dotation amortissement mat. roulant
640100: taxe vehicule
640200: précompte immobilier

Nous pouvons distinguer des charges indirectes relatives au véhicule/transport, à un immeuble, à de la télécommunication, à de la publicité, à l'outil de production, des assurances.

Nous créons des familles de coût en regroupant les charges indirectes par destination.
Sauf exception, il n'est pas cohérent de regrouper des amortissement de véhicule et des factures ayant trait à l'immeuble.
Il est par contre logique de rassembler les frais relatifs à l'amortissement de véhicules avec des frais de maintenance véhicule, carburant véhicule et assurances véhicules.

Famille1: publicité

615100: frais représentation
615200: publicité annonces
615220: foires expositions

Famille2: bureau

612400: imprimés fournitures bureau
612500: petit mat. bureau
616100: frais postaux
630231: dotation amortissement mobilier

Famille3: immeuble

611300: entretien locaux
612000: énergie immeuble
613550: assurance immeuble
630210: dotation amortissement immeuble
640200: précompte immobilier

Famille4: formation documentation

612300: documentation

Famille5: honoraires

613250: honoraire comptable

Famille6: transport.

606140: frais transport
611150: location véhicule
611350: entretien véhicule
612160: carburant|
613530: assurance véhicule
630232: dotation amortissement mat. roulant
640100: taxe véhicule

Famille7: outil et production.

601300: outillage
611130: location outillage
613540: assurance RC
630220: dotation amortissement outil

Famille8: telecommunication.

616200: téléphone
616300: gsm
616400: internet

Ces regroupements effectués, nous constatons avoir des frais d'immeuble et de bureau.
Bureau et immeuble sont liés, dès lors que ce qui a trait à un bureau se pratique dans un immeuble.
Nous somme tentés de relier ces frais, et créons un regroupement charges indirectes bureau + charges indirectes de l'immeuble.

L'immeuble a toutefois un second utilisateur: il tient également lieu de stockage.
Nous créons un second regroupement: charges indirectes stockage + charges indirectes de l'immeuble.

Les charges indirectes de l'immeuble étant divisées, nous choisissons une proportion de répartition qui sera : 40% pour le bureau et 60% pour le stockage, soit la proportion des superficies d'utilisation respectives.

Notons que nous avons choisi de ne pas intégrer les coûts de téléphonie parmi les frais relatifs au bureau.
Ce choix est guidé par le fait que ce poste (téléphonie) est relativement important, et que son utilisation ne se confine pas au domaine du « bureau ».

Nous créons la « famille stock », et supprimons la famille3 immeuble; les charges indirectes relatives à l'immeuble sont intégrées dans la famille2 bureau et identiquement dans la « famille stock » qui devient famille3 stock**.**
Les familles de coût 2 et 3 deviennent:

Famille2: bureau

611300: entretien locaux
612000: énergie immeuble
612400: imprimés fournitures bureau
612500: petit mat. bureau
613550: assurance immeuble
616100: frais postaux
630210: dotation amortissement immeuble
630231: dotation amortissement mobilier
640200: précompte immobilier

Famille3: stock

611300: entretien locaux
612000: énergie immeuble
613550: assurance immeuble
630210: dotation amortissement immeuble
630231: dotation amortissement mobilier
640200: précompte immobilier

Le poste pcmn 630231 « dotation amortissement mobilier » se réparti entre les Famille2: bureau, et Famille3: stock.
Nous choisissons un taux de répartition de 80% en Famille2: bureau, et 20% en Famille3: stock.

Processus et activités.

Nous devons identifier les activités que nous utiliserons pour l'analyse ABC.
Ces activités comme décrit plus haut, ressortent de l'examen du processus de l'entreprise.
Nous examinons ce processus ci-après.

Activité1 communication

Un mix publicitaire est le préambule à ce secteur d'activité , comme la présence lors d'événements commerciaux, une présence internet, l'établissement contacts commerciaux, etc.
Il en ressort des contacts avec des clients potentiels.

Activité2 démarchage

La demande du contact est étudiée: visite sur place, étude de faisabilité technique, rédaction de une ou plusieurs offre(s), et ensuite si passation de commande il y a, des réunions ont lieu.

Activité3 logistique

Une commande est passée à l'entreprise.
Une sélection du matériel nécessaire est établie, suivie de passation de commande chez divers fournisseurs.
Une partie du matériel est stockée dans les locaux de l'entreprise, d'autres composants sont livrés sur les lieux d'exécution de la commande.

Activité4 transport

Le transport de personnes et matériel est une composante non négligeable dans l'exercice de ce type d'entreprise.

Activité5 production

La production qui assure la réalisation de la commande est l'activité principale de l'entreprise.

Ces cinq activités sont une interprétation du processus de l'entreprise.
Cependant, pour être complet, nous choisissons d'y ajouter deux autres activités qui n'apparaissent pas dans le processus décrit ci-dessus.

Activité6 comptabilité / ABC

Une Activité comptabilité / ABC qui sera l'utilisation de la solution que nous tentons de développer, et qui additionnée au coût du comptable externe, nous permettra d'appréhender le budget d'une démarche comptable de qualité.
Répercussion ou pas dans les commandes passées à l'entreprise?
Nous choisissons d'imputer ce coût dans les commandes, par le biais des factures de vente émises qui relèvent de la comptabilité.

Activité7 télécommunication

Nous évoquions plus haut le coût important de ce poste télécommunication.
Nous en faisons une activité à part entière.

Remarques.

Nous appliquerons un coût horaire à l'activité production.
Par contre, nous choisissons de ne pas répercuter de coût horaire aux autres activités.
Ce choix est motivé par les raisons suivantes.

D'une part, durant un exercice fiscal, la majorité des heures prestées le sont dans le cadre de l'activité5 production. Le coût horaire est alors une charge salariale, et donc une charge directe au même titre que les matières premières.

D'autre part, en ce qui concerne les autres activités, comme la plupart des TPE, l'entreprise dont exemple n'utilise pas de personnel administratif ou marketing; ces taches sont souvent réalisées en dehors des heures de production par les administrateurs / gérants, et sont très souvent non rémunérées.
Nous ne les rémunérerons pas et n'en tiendrons pas compte dans les mesurages.

La valeur des activités.

Pour déterminer la valeur des activités, nous devons y incorporer les familles de coûts (dans

lesquelles nous avons incorporé les charges indirectes).
Seule la connaissance de l'entreprise permet d'en fixer les règles.
Plusieurs familles de coûts peuvent entrer dans une activité, la même famille de coût peut faire partie de plusieurs activités.

Plusieurs solutions sont possibles, nous retenons la suivante.
L' activité1 communication est composée de la Famille1: publicité, d'une partie de la Famille2: bureau, et d'une partie de la Famille4: formation documentation.

L'activité2 démarchage est composée d'une partie des coûts de la Famille4: formation documentation, d'une partie de la Famille2: bureau.

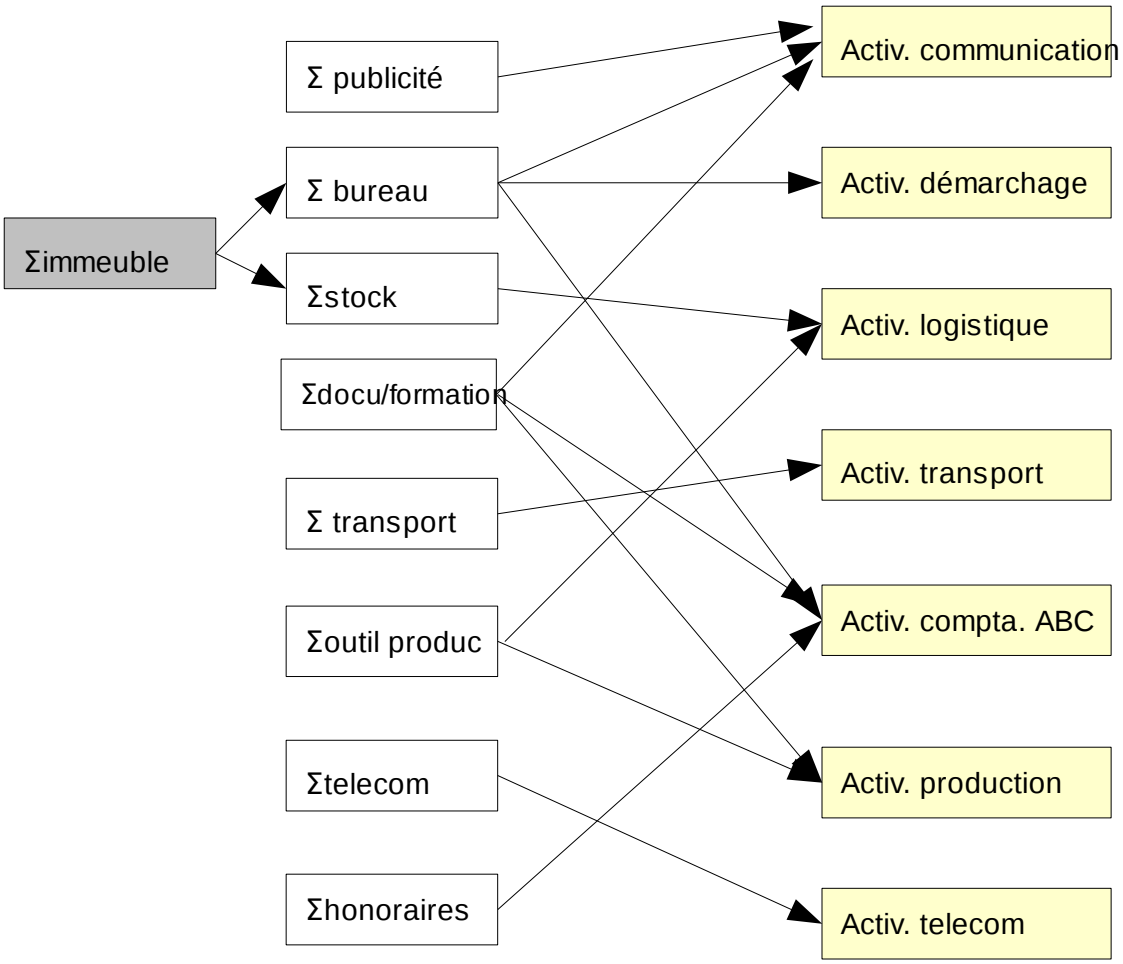
L'activité3 logistique est composée des coûts de la Famille3: stock.

L'activité4 transport est composée des coûts de la Famille6: transport.

L'activité5 production est composée coûts de la Famille7: outil et production, et d'une partie de la Famille4: formation documentation.

L'activité6 comptabilité / abc est composée d'une partie de la Famille4: formation documentation, de laFamille5: honoraires.

L'activité7 télécommunication est composée de la Famille8: télécommunication.



Proportion d'intervention des familles au sein des activités.

Nous devons choisir dans quelle proportion interviendront les familles de coût partagées entre plusieurs activités dans leurs activités respectives.

La famille regroupant les frais relatifs au bureau est partagée entre les activités « communication », « démarchage » et « comptabilité/ABC ».
Par connaissance de l'entreprise, nous choisissons une proportion de 30% en activité « communication », 35% en activité « démarchage » et 25% en activité « comptabilité /ABC ».

La famille regroupant les frais « formation et documentation » se réparti entre les activités « démarchage », « comptabilité /ABC » et « production » .
Nous choisissons une proportion de 42% en activité « démarchage », 34% en activité « production », et 24% en activité « comptabilité /ABC ».

La famille regroupant les coûts « d'outil de production » se réparti entre les activités « production » et « logistique ».
Nous choisissons une proportion de 27% en activité « logistique » et 73% en activité « production »

Le tableau ci-dessous reprend les proportions de répartition des différentes familles de coût au sein des activités.

	Act1	Act2	Act3	Act4	Act5	Act6	Act7
Fam1	1						
Fam2	0,3	0,4				0,3	
Fam3			1				
Fam4		0,42			0,34	0,24	
Fam5						1	
Fam6				1			
Fam7			0,27		0,73		
Fam8							1

Exception.

La famille « honoraires » n'est destinée qu'à ABC. Nous supposons de façon simplifiée que seul un comptable présente ses honoraires.
Si un avocat devait présenter ses honoraires pour un litige lié à la production, une solution serait de répartir cet honoraire au sein de l' activité5 production.

ABC est une démarche souple et dynamique...

Les unités d'oeuvre.

Nous devons choisir quelles seront les unités d'oeuvre utilisées pour chaque activité.

ACTIVITE	**Unité d'oeuvre**
Activité1 communication	Heure de communication
Activité2 démarchage	Heure de démarchage
Activité3 logistique	Article unitaire
Activité4 transport	Kilomètre
Activité5 production	Heure de production
Activité6 comptabilité / abc	Heure de comptabilité ABC
Activité7 télécommunication	Minute de télécommunication

L'unité revenant le plus régulièrement sera le temps, l'heure ou la minute.
L'heure de production, l'heure de démarchage, l'heure de communication, l'heure de comptabilité ABC.
Pour l'activité télécommunication, nous utiliserons la minute.

Pour l'activité logistique, la quantité d'articles utilisés.
En ce qui concerne cette activité, un choix a dû être déterminé.
Les articles volumineux ou chers sont livrés sur le lieux d'exécution de la commande en mode « just in time ».
D'autres articles sont livrés anticipativement au siège de l'entreprise et font l'objet d'un stockage.
Enfin, une troisième catégorie d'articles provient d'un achat immédiat de dépannage chez un fournisseur local.

La première catégorie « just in time » ne coûte rien si ce n'est l'envoi d'un mail de confirmation.
La seconde catégorie coûte l'utilisation du stockage.
La troisième catégorie est consommatrice de temps et de kilomètres.

A notre sens, nous avons le choix entre les unités d'oeuvre suivantes: la note d'envoi correspondant à un acte d'achat, ou l'article unitaire.
Il faut trancher: nous avons choisi l'article unitaire. Si l'expérience montre que ce n'est pas optimal, on peut en changer dans le temps.

L'application ABC XML.

Nous avons présenté en détail une application de la méthode ABC à un cas d'entreprise précis.
Pour la suite de ce travail qui traite de l'implémentation de la méthode ABC en utilisant des données comptables au format XML, nous devons retenir de la description ci-dessus les composants suivant.

- Les postes pcmn qui désigneront les charges directes et indirectes.
- Les regroupement pcmn en familles de coûts.
- Les activités.
- Les unités d'oeuvre.

Nous devrons également procéder à différents calculs qui devront livrer le résultat de la commande.
Ce résultat devra nous indiquer la marge brute et la marge nette avant impôts.

- Marge brute: Chiffre d'affaire de la commande – charges directes.

- Marge nette (avant impôts): Marge brute – charges indirectes.

Le chiffre d'affaire est connu, les charges directes facilement identifiables.
Nous devons donc calculer la part de charges directes imputable à cette commande, que nous appellerons « coût indirect ».

Notre application effectuera les transformations suivantes:

- Coût d'une famille: somme des montants « pcmn » imputés * proportion d'intervention du poste pcmn dans la famille.
- Coût d'une activité: somme des familles qui la composent * proportion d'intervention de la famille dans l'activité.
- Valeur de l'unité d'oeuvre d'une activité: coût total de l'activité divisé par la quantité totale d'unités d'oeuvres se rapportant à cette activité.
- Coût indirect d'une commande: somme de (pour chaque activité utilisée, quantité d'unités d'oeuvre * valeur d'unité d'oeuvre)
- Calcul des marges brutes et nettes

Exemple chiffré.

Notre objectif n'est pas de produire un résultat portant sur l'encodage d'une année ou bien d'un trimestre de pièces comptables.
Nous créerons quelques commandes client, encoderons quelques charges directes et indirectes.

Les commandes.

1. Commande1 CA: 12000 Charges Directes: 4756 + 322
2. Commande3 CA: 1852 Charges Directes: 640
3. Commande2 CA: 2310 Charges Directes: 801
4. Commande4 CA: 6400 Charges Directes: 604

Nous utiliserons un tarif de 25€ par heure de production.
Ce montant est déterminé par les coûts d'assurance sociale, assurance vie et accidents, salaire,...

Nous imputerons les charges indirectes suivantes.

PCMN: 601300 montant: 850
PCMN: 611300 montant: 1150
PCMN: 615200 montant: 1600
PCMN: 615100 montant: 710.2
PCMN: 612400 montant: 466
PCMN: 612300 montant: 157.85
PCMN: 612160 montant: 310
PCMN: 630232 montant: 760
PCMN: 630220 montant: 350
PCMN: 630210 montant: 423
PCMN: 613250 montant: 600
PCMN: 612000 montant: 240
PCMN: 612160 montant: 211

PCMN: 616200 montant: 118.7
PCMN: 612300 montant: 560
PCMN: 611130 montant: 422
PCMN: 616300 montant: 58
PCMN: 616300 montant: 74.6

Les postes de dotation aux amortissements (630231, 630210, 630232, 630220): nous imputons des montants correspondant à 2/12 du total annuel.
La raison en est que nous prendrons en exemple des commandes dont la durée porte sur une période de deux mois.

Notons que nous travaillerons avec des montants sans TVA. Nous parlerons donc de montant hors TVA (htva).

Sources.

- Professeur D.Helbois, Fucam: Syllabus « Comptabilité analytique d'exploitation » et « Techniques de calcul et de répartition ».
- F. Guerra E. DeHaan: « Comptabilité les procédures comptables et comptes annuels » ISBN 2-8041-0954-2
- H.Bouquin: « Comptabilité de gestion » ISBN 2-7178-4026-5

Chapitre2. Extensible Markup Language (XML) .

Le langage XML que nous utiliserons pour la suite du développement de ce mémoire, est issu des travaux du World Wide Web Consortium.
Dans ce chapitre, nous décrivons au moyen d'un exemple ce qu'est XML et quelques principes qui nous seront utiles.

Le World Wide Web Consortium (W3C).

Le World Wide Web Consortium a été fondé par Tim Berners-Lee (inventeur du World Wide Web) en octobre 1994 au Massachusetts Institute of Technology.
W3C est un groupe de normalisation et standardisation dédié aux technologies du Web.

Cette standardisation vise à permettre le développement du Web en rendant compatibles les technologies dédiées, et aux différentes solutions logicielles y accédant de travailler ensemble. L'interopérabilité Web étant une des missions du W3C.

Les standards de langages et protocoles Web publiés par le W3C le sont en Open source afin d'éviter la fragmentation du marché et par conséquence du Web.

Recommandation W3C: au terme d'un processus de plusieurs phases (brouillon de travail, dernier appel, candidat à la recommandation, recommandation proposée) , un document devient une Recommandation W3C.

Citons quelques recommandations:
- 1996 Portable Network Graphics (PNG) 1.0
- 1996 Separating content from structure, CSS Level 1 is published cascading styled sheets
- 1998 XML 1.0
- 2001 XML Schema
- ...

Source: W3C (www.w3.org).

Extensible Markup Language (XML).

Extensible Markup Language ou XML est un langage permettant le balisage de documents.
A l'origine est le SGML (Standard Generalized Markup Language) qui a été développé par IBM dans les années 70 et est devenu un standard ISO.
Le principe de base du SGML est de dissocier la structure d'un document de sa présentation.
SGML est efficace, mais complexe, son utilisation a surtout été industrielle et militaire.

Le W3C souhaitait qu'il soit possible de transmettre et traiter du SGML générique sur le Web. Cette forme simplifiée du SGML est le Extensible Markup Language, qui est devenu une recommandation du World Wide Web Consortium le 10 février 1998.

Dès le début, l'engouement était au rendez-vous, et depuis une dizaine d'années, des milliers de

documents et applications ont été portés en XML.

L'application la plus populaire de XML est le XHTML fortement utilisé sur le Web.

Le principe de XML est d'être un métalangage permettant de baliser un contenu, et donc de décrire ce contenu.
XML se veut souple d'utilisation: chacun peut créer son propre balisage relatif à des données fort différentes, des images, du texte, des bases de données, des catalogues, des formules mathématiques, des données chiffrées,..

Par contre, XML est plus rigoureux que HTML quant au respect des règles syntaxiques; un document XML doit être bien formé: il contient un ou plusieurs éléments, la balise de chaque élément est fermée correctement, les éléments sont case sensitive, les attributs ne sont pas vides et sont entre guillemets.

XML permet le découplage des données et de la présentation, XML n'enferme pas les données dans un format propriétaire qui les rend peu utilisables sous d'autres formats.

En aucun cas XML n'est à même de transformer les données qu'il contient.
XML ne propose pas de traitement, et il n'est pas possible de créer d'algorithme en XML pour produire un résultat.

Définir un document XML.

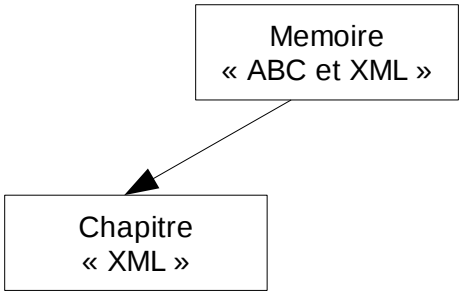
Nous définirons un document XML comme un ensemble d'informations balisées ou « encapsulées » dans des tags décrivant leur contenu.
Ces informations peuvent être un texte, un catalogue, des images, des données comptables dans ce cas précis,...
Le terme de document XML est utilisé pour décrire une telle structure.

Un exemple pour illustrer notre propos, serait celui d' un mémoire qui comporte des chapitres.
Des balises peuvent être créées pour décrire ces différentes parties.
Nous définissons « mémoire » et « chapitre » au moyen des balises <memoire> </memoire> et <chapitre> </chapitre>
Les balises sont doubles car la première ouvre <memoire> , la seconde referme sur le contenu </memoire>: en XML, toute balise ouverte doit être refermée.

Structure hiérarchique: l'arbre XML.

Nous voyons apparaître une structure hiérarchique.
Très logiquement, « mémoire » se trouve hiérarchiquement au-dessus de « chapitre ».
Un élément « mémoire » comporte un ou plusieurs élément(s) « chapitre(s) ».
Nous obtenons une notion parent-enfant(s): un élément « chapitre » est enfant d'un élément « mémoire ».

De cette représentation hiérarchique de XML provient la notion d'arbre XML.
Un document XML est un arbre XML.



Eléments et attributs.

Pour être plus précis, nous indiquons que « ABC et XML » est le titre du mémoire, que « Explication de XML » est le titre du chapitre.
En XML, nous utiliserons la notion d'« élément » et d'« attribut ».

Les éléments sont:
<memoire> et <chapitre>.
Les attributs sont:
<titre_memoire> qui est attribut de l'élément <memoire>
<titre_chapitre> qui est attribut de l'élément <chapitre>

Notre exemple peut se représenter sous cette forme dans un document XML.

```
<memoire>
<titre_memoire>ABC et XML</titre_mémoire>
</memoire>
<chapitre>
<titre_chapitre>XML</titre_chapitre>
</chapitre>
```

Nous devons alors intégrer la notion de hiérarchie. Pour ce, nous déplaçons la balise de fermeture </memoire> en fin de ce document XML.

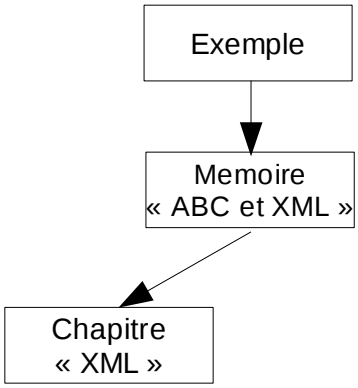
```
<memoire>
<titre_memoire>ABC et XML</titre_memoire>
<chapitre>
<titre_chapitre>XML</titre_chapitre>
</chapitre>
</memoire>
```

La recommandation XML impose qu'un arbre XML soit « encadré » par un élément racine, parfois appelé « élément bidon » ou « root ».
L'élément racine est l'élément parent de tous les autres éléments du document XML.
Il est vide, et dépourvu d'attribut.
Nous créons donc l'élément « exemple » qui remplira ce rôle.

Nous obtenons alors la représentation XML suivante.

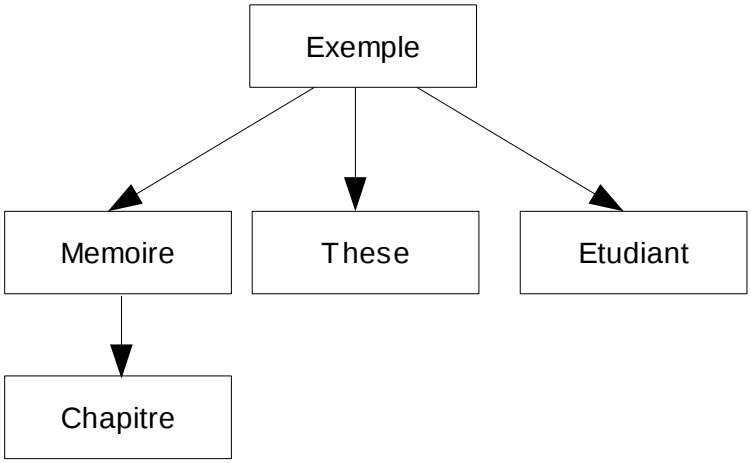
En examinant sa structure, nous constatons que les balises <chapitre> et </chapitre> contiennent les balises <titre_mémoire/>, et sont elles même contenues au sein des balises <memoire></memoire>
C'est de cette façon que se représente la hiérarchie au sein d'un document XML: l'élément parent inclus les éléments enfants.

```
<exemple>
<memoire>
<titre_memoire>ABC et XML</titre_memoire>
<chapitre>
<titre_chapitre>XML</titre_chapitre>
</chapitre>
</memoire>
</exemple>
```



L'utilité de cet élément racine est de pouvoir d'une part accéder l'arbre XML , et d'autre part de pouvoir créer d'autres éléments de même niveau hiérarchique (que « memoire » dans ce cas).
Si nous n'avons que l'élément « memoire », « memoire » est la racine de l'arbre XML.
Nous avons créé la racine <exemple> souhaitant créer d'autres éléments de même niveau hiérarchique que celui de « memoire », qui sont : « these » et « etudiant ».

Nous obtenons alors l'arbre XML ci-dessous, où nous voyons que les éléments « memoire » « these » et « etudiant » sont de niveau hiérarchique équivalent.



Le niveau hiérarchique identique de ces trois éléments apparaît dans le document XML.
</memoire, </these> et </etudiant> sont intégrés de façon identique au sein de </exemple>.

```
<exemple>
<memoire>
<titre_memoire>ABC et XML</titre_memoire>
<chapitre>
<titre_chapitre>XML</titre_chapitre>
</chapitre>
</memoire>
<these>
<titre_these>thèse1</titre_these>
</these>
<etudiant>
<nom_etudiant> Dupondt</nom_etudiant>
</etudiant>
</exemple>
```

Validation d'un document XML.

Afin de déterminer le rôle de chaque composant et son niveau hiérarchique au sein d'un document XML, une validation est nécessaire.

Un document XML peut être validé par une DTD ou bien par un schéma XML.

Nous présenterons brièvement la DTD ayant choisi cette alternative pour la suite du développement de ce travail.

Data Type Definition.

Une Data Type Definition (DTD) est un document qui peut être externe au document XML, ou bien en faire partie intégrante.

Une DTD est rédigée dans un formalisme et une syntaxe précise, définie par le W3C.
Elle défini le vocabulaire du document XML, et décrit quels en sont les éléments, quels sont les attributs de chaque élément.

La validation du document XML est assurée par un parseur ou via le site du W3C**.**
Durant cette opération, le document XML est comparé à sa DTD, afin de vérifier la présence et concordance des éléments et attributs.
Un document XML validé par une DTD est un document instance de la DTD.

Reprenant notre exemple, nous créons une DTD externe que nous appelons memoire.dtd .
Pour cette alternative que nous utiliserons plus loin, le document XML doit explicitement faire appel à la DTD via la ligne suivante qui se trouve en entête du document XML :

```
<!DOCTYPE exemple SYSTEM "/home/jean/Memoireulb/XML/memoire.dtd">
```

Définir Elément et Attributs.

Elément est décrit dans une DTD par: </ELEMENT (le nom de l'élément) (contenu)>
Pour créer un élément « chapitre », donc le jeu de balise <chapitre> </chapitre>, nous rédigeons en en termes de DTD: </ELEMENT chapitre (contenu)>.

Le contenu peut être de type vide , contenir des sous-éléments ou éléments enfants (une séquence de sous-éléments), proposer un choix de sous-éléments, du texte, du contenu mixte.

Nous utiliserons par la suite les éléments vides, les éléments comportant des sous-éléments et les éléments comprenant une description.

Un élément comportant une description se défini: </ELEMENT (nom_élément) (#CDATA)>
L'élément « chapitre » contenant une description de type CDATA :
</ELEMENT chapitre (#CDATA)>.

Un élément vide se défini par : </ELEMENT nom_élément (EMPTY)>.
L'élément chapitre qui serait vide est défini par: </ELEMENT chapitre (EMPTY)>.
Un élément contenant un sous-élément se défini par :
</ELEMENT exemple (nom_élément) (enfant ou sous-élément+)>.
L'élément « chapitre » qui contient un élément enfant « paragraphe »:
</ELEMENT exemple (chapitre) (paragraphe+)>.
« + » signifiant que l'élément chapitre peut contenir un ou plusieurs sous-éléments « paragraphe ».

Les attributs.

Une déclaration d'attribut commence par une balise <!ATTLIST (élément) (nom(s) d'attribut(s)>.
Les attributs peuvent être de différents types.
- ID, IDREF: identificateurs uniques, qui peuvent être mis en relation par référence (IDREF).
- LIST (ou ENUMERATION): choix d'attributs imposé.
- ENTITY, ENTITIES: entité(s) déclarée(s) et non analysée de la DTD.
- NMTOKEN, NMTOKENS: unité(s) lexicale nominale XML.
- CDATA: une chaîne de caractères.

Sources:

W3C: Extensible Markup Language (XML) 1.0
The XML 1.1 Bible ISBN: 0-7645-4986-3

Activity Based Costing et Extensible Markup Language.

Nous avons au fil du chapitre précédent décrit l'application de la méthode Activity Based Costing que nous souhaitons développer dans ce mémoire; dans ce chapitre, nous avons brièvement présenté XML.
Nous relierons ces deux matières au sein de ce travail, en développant notre application ABC en utilisant les technologies XML du W3C.
Ce choix est motivé par le souhait de ne pas enfermer nos données en un format propriétaire, et être ensuite tributaire des possibilités techniques offertes par ce format.
Un autre risque lié au format propriétaire et de plus pouvoir réutiliser les données dans le temps, par exemple à cause d'un changement de format lors d'une évolution logicielle.

D'autre part, le monde XML offre une palette de possibilités technologiques qui sont complémentaires .
Il doit dès lors être possible de les utiliser pour transformer des données comptables saisies en XML afin de mener une démarche analytique.
C'est ce que nous tentons au fil des chapitres suivants.

Chapitre 3. Les logiciels libres.

Un des objectifs de ce travail est d'utiliser des solutions logicielles issues du monde du libre.
Nous devons examiner les logiciels libres disponibles en matière de XML et les évaluer.

Défini sommairement: un logiciel libre est livré en Open Source, c'est à dire avec son code source, la plupart du temps gratuitement, sans pour autant signifier qu'un logiciel gratuit est libre.
Il est soumis à un droit d'auteur, qui stipule les droits octroyés à l'utilisateur: la licence libre.
Toutefois, l'auteur conserve ses droits d'auteur, bien qu'il renonce à une grande partie de ceux-ci.
Un logiciel commercial est au contraire soumis au droit d'auteur, son code source n'est pas fourni, il ne peut être rediffusé, et est accompagné d'un contrat de licence payant.

Définir les besoins.

L'intégralité de ce mémoire, de la recherche d'information à la rédaction sans oublier le traitement XML, sera réalisée sous l' Operating System Linux en version Mandriva 2008.

En partie rédactionnelle, nous utiliserons OpenOffice.org Writer, pour la réalisations de schémas: OpenOffice.org Draw, les saisies d'écran proviennent de Ksnapshot, sans oublier Mozilla Firefox comme navigateur Web, et des utilitaires anonymes.

En ce qui concerne XML, des logiciels de traitement et transformation de données sont nécessaires.

Nous devons disposer d'outils permettant de réaliser les opérations ci-dessous.

- Créer la DTD
- Créer le fichier XML
- Valider XML et DTD
- Opérer des recherches au sein de notre document XML afin d'en extraire les données nécessaires au calcul des résultats ABC.
  Nous utiliserons pour ces recherches XQuery.
  Nous souhaitons une application XQuery disposant d'un environnement visuel semblable à celui d'un compilateur dédié à un langage de programmation , permettant de saisir les requêtes, mettant en évidence les erreurs de syntaxe, et muni d'une fonction de déboguage donc, analysant la requête et affichant un résultat.
- Présenter les résultats; nous utiliserons XSL-FO.

Recherche logicielle.

La création de la DTD et du fichier XML peut être réalisée avec un simple éditeur.
Nous avons utilisé Kate, éditeur disponible sous Mandriva Linux.

La validation de la syntaxe XML1.0 et de la conformité à la DTD est possible en ligne via une page de la Brown University.

Nous voyons déjà apparaître une division des solutions qui n'est pas très pratique; DTD et document XML avec un éditeur, par contre, XQuery nécessite un logiciel et non pas un éditeur.

Nous en serions donc à deux logiciels et au recours à un site externe.
L'idéal serait une suite pouvant assurer l'ensemble des taches.

La démarche pour trouver un processeur XQuery ou bien une suite logicielle XML a été de mener une recherche en ligne qui ponctuellement menait au W3C (http://www.w3.org/XML/Query/)
Le W3C répertorie sur son site une cinquantaine de possibilités.

Licence	Database	XQuery	Conversion
Payant (30jours)	9	3	
Libre	8	8	2
Site indisponible	2	6	
Apple	2	1	
MicroSoft	1	1	
Autres	2	4	

Nous les avons examinées, le tableau non exhaustif ci-dessus en propose un simple aperçu, certaines solutions pouvant figurer dans plusieurs cases.

Les implémentations répertoriées proposent des Database managées par XQuery, ou bien des outils purement XQuery, ou enfin des outils de conversion.
Parmi les conversions, citons XQ2XML permettant de convertir XQuery en XML ou en XSLT

Les Licences étaient soit libres , soit payantes, le plus souvent proposées en période d'essai de 30 à 90 jours.

Certains sites étaient indisponibles, par exemple Rainbowcore, ou Relational XQuery.

D'autres solutions sont conçues exclusivement pour Microsoft ou bien MacOS X (Sherlock).

Sous la rubrique « autres », nous reprenons des consultant sans logiciel, ou une solution en ligne comme XQuery demo.

Parmi les XQuery payants, nous trouvons Saxon, StylusStudio, Xquantum, ...

Les XQuery en licence libre: Saxon à nouveau, open xquery, Xqilla, Berkeley Lab's NUX, Zorba, MXQuery.
Des projets, dont un probablement oublié car datant de 2002: php xml classes, ou en cours comme XQP.
AltovaXML peut remplir plusieurs cases: il est payant ou libre, mais sous Windows.

En Database libres, notons XML Global, eXist, Qizx/db.
Enfin, notons que certaines solutions intègrent XBRL (Saxon, StylusStudio).

L'installation des logiciels sous Linux.

L'installation sous Linux de logiciels peu répandus révèle une caractéristique des Linux du début: la technique de « l'essai-erreur » est de mise, on travaille en ligne de commande dans une console, il manque toujours une librairie que l'on doit trouver en ligne, et qui quand elle est partiellement installée, réclame une autre librairie, etc.
Pour parfois ne pas fonctionner au final.
Le double-clic sur une icône du monde Windows n'est pas de mise.

La procédure consiste à télécharger sur le site de l'éditeur un fichier archivé en .tar.gz ou .tgz ou .zip puis à le décompresser dans un répertoire , parfois au moyen de la console, parfois par un outil permettant de désarchiver sans ligne de commande (il faut essayer si cela fonctionne), ensuite, lecture de la notice au format .txt si elle existe, ou opérer des tentatives sur des fichiers .jar, .sh, ou qui semblent par expérience être exécutables.

Saxon.

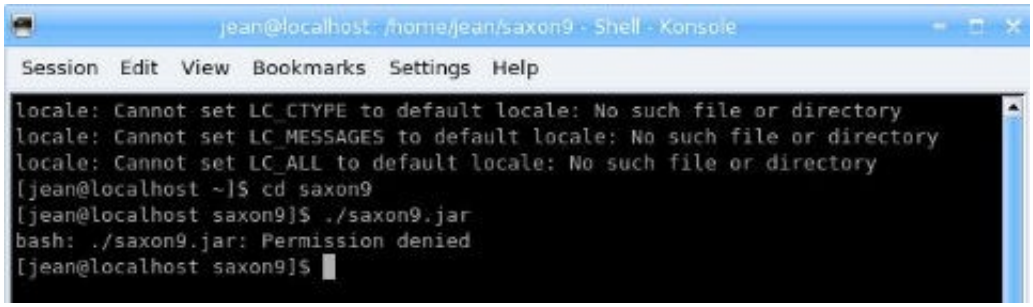
Une première tentative a été menée avec un des logiciels de base du monde XQuery et XML: Saxon.
Saxon développé par Michael Kay, est une suite comprenant un processeur XSLT2.0, un processeur Xpath2.0, un processeur XQuery 1.0, un processeur XmlSchema1.0 .
Saxon est multi-plateformes, java, .NET .
Saxon utilise des parseurs DOM et SAX qui sont deux approches de traitement de document XML.

Les licences de Saxon.

En 2004, Michael Kay après avoir quitté Software AG qui contribuait au développement de Saxon, a fondé la société Saxonica où il continue de développer Saxon.
A partir de 2004 et Saxonica, Saxon 8 a été décliné en deux versions: SaxonB et Saxon SA.
SaxonB est en « Mozilla Public Licence » qui est une variante de licence libre, Saxon SA est un produit commercial, payant et au code non ouvert.

Nous avons tenté la version Saxon9.0 B (donc libre).

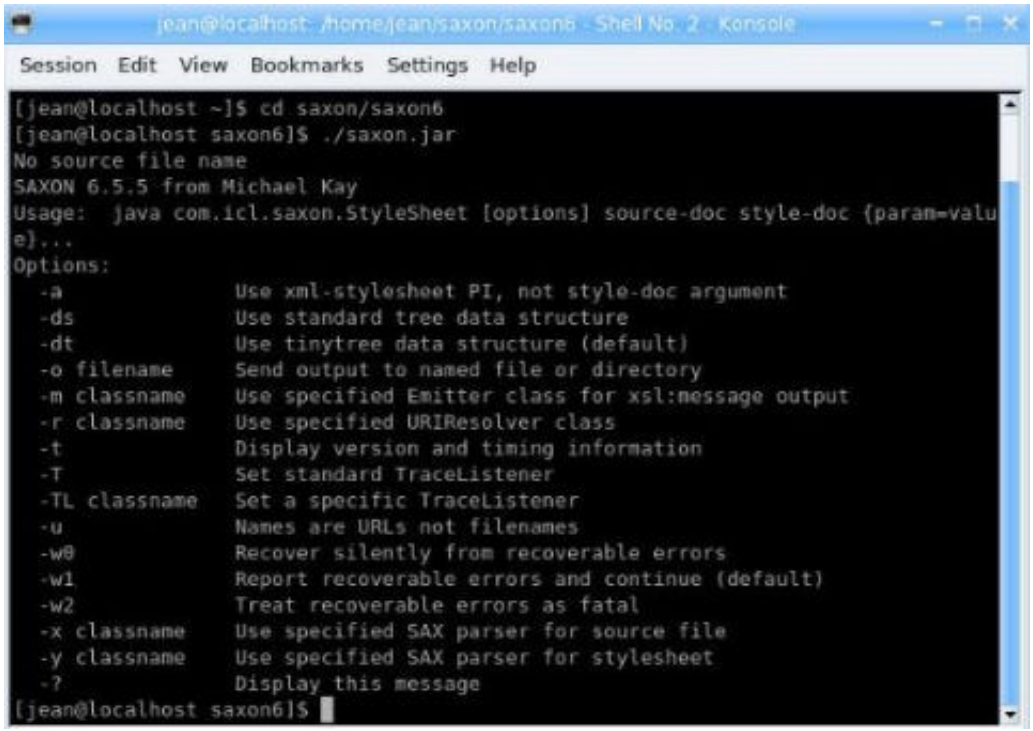
Au lancement, Saxon9 a proposé ceci: « ./saxon9.jar: Permission denied »



La documentation de Saxon disponible en ligne ne permet pas de résoudre ce qui n'était pas un problème de configuration de permissions.
Des recherches en ligne sur « *Permission denied saxon9.jar »* ne proposaient que 75 résultats sur Google, et laissaient supposer une configuration Linux/ Java à modifier.

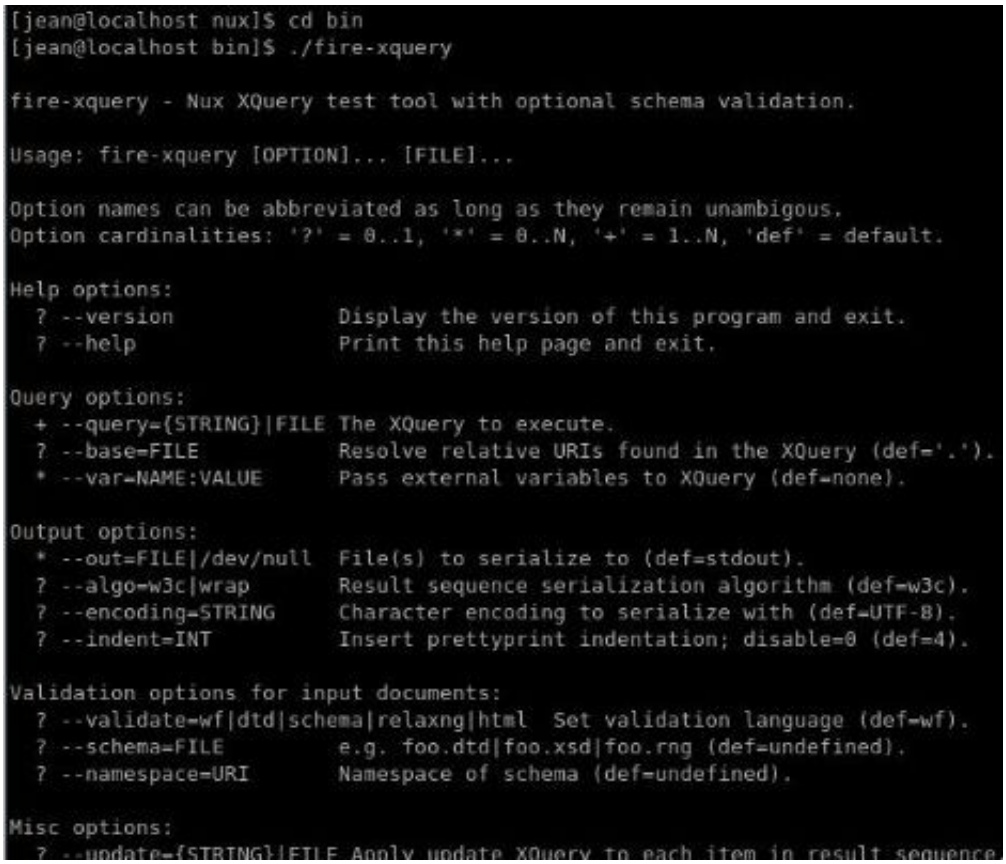
La plate-forme Java disponible pour Linux n'est probablement pas tant « up-to-date » que celle dédiée à Windows.

Diverses tentatives Linux n'ayant pas amélioré le résultat, Saxon6 a été testé.
Saxon6 s'ouvre, et propose une interface ... en ligne de commande.



Souhaitant une solution ergonomique, ce qui n'est pas le cas avec une fenêtre monochrome de saisie de ligne de commande, nous poursuivons les essais de logiciels.

La recherche a ensuite porté sur une éventuelle interface graphique.
Le choix s'est révélé être relativement restreint.
Kernow pour Saxon9 est une interface graphique, que nous avons utilisée au début du travail d'ébauche XML. Pour l'abandonner ensuite, car de l'aveu du concepteur, Kernow n'est pas encore très développé pour sa partie XQuery.



D'autres tentatives ont été menées, retenons essentiellement NUX qui présente le même type d'interface (ci-dessus), et MxQuery dont nous reparlerons ci-après.

Solution payante.

Face à ces déconvenues, les ressources temps étant limitées et trop de temps ayant été consacré à cette recherche partiellement vaine, la décision a été prise de renoncer à une solution libre, pour une suite commerciale afin de pouvoir continuer à aller de l'avant.
Plusieurs solutions existent, celle qui paraissait être la plus attrayante « Stylus Studio » du fait d'un outil graphique permettant de créer les requêtes XQuery, était disponible uniquement sous Windows.
Il y avait également Saxon SA, et d'autres.
L'essentiel étant de prendre une décision, nous avons opté pour Oxygen, « <oXygen/>XML » pour être précis.
« <oXygen/>XML » : pour sa convivialité d'installation inespérée sous Linux, son ergonomie, la somme modique de la licence.

<oXygen/> XML.

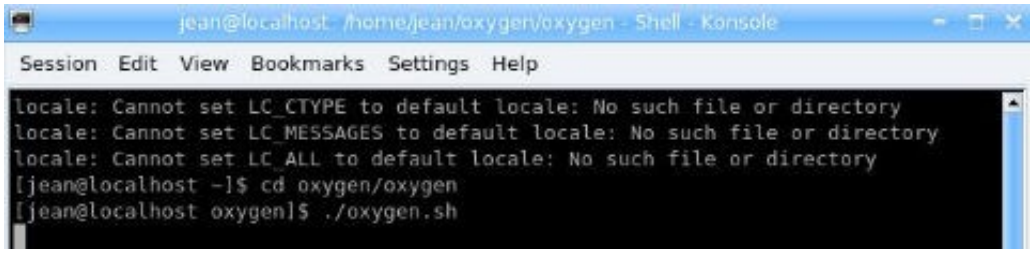
<oXygen/>XML est un éditeur complet dédié à XML.
L'auteur en est la société roumaine SyncRO Soft.

Outre un éditeur XML DTD, nous y trouvons des outils Xpath et XQuery avec fonction débogueur (<oXygen/> XML intègre SaxonSA) XSLT, XSL-FO et d'autres possibilités dont nous ne ferons pas usage comme XML Schema, Relax NG, SOAP; <oXygen/>XML peut également traiter des databases relationelles XML.

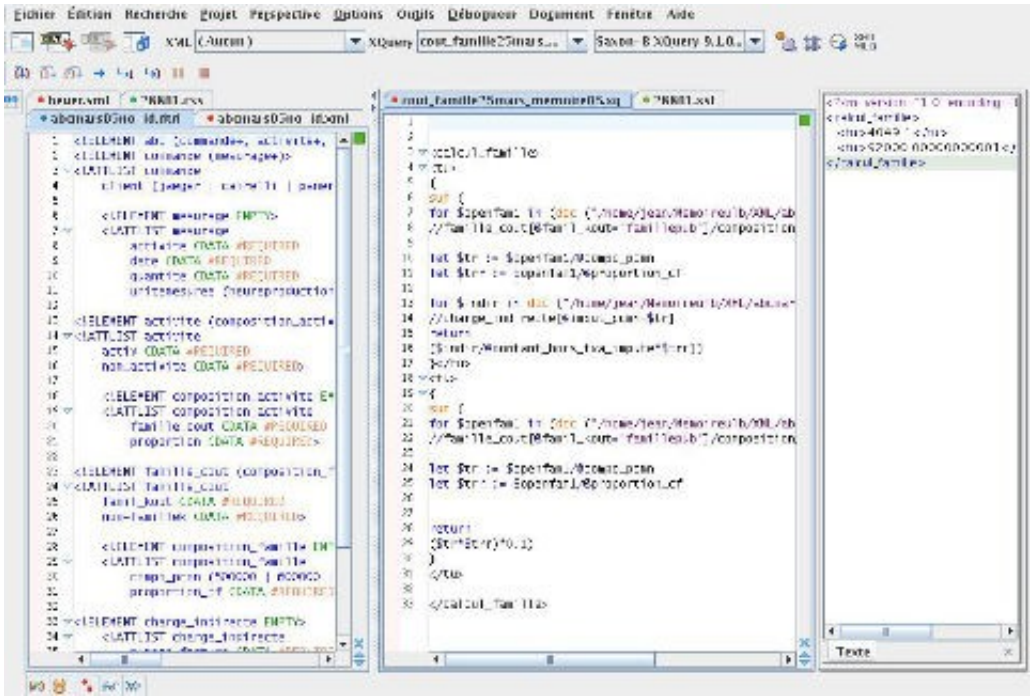
<oXygen/>XML est un programme écrit en java, ce qui lui permet une indépendance quant à la plate-forme utilisée: Windows, Linux, MacOS. Son fonctionnement est lié à la présence d'une Machine Virtuelle Java adéquate à la plate-forme.
<oXygen/>XML existe sous forme d'une suite autonome, ou bien en plug-in de la plate-forme Eclipse.

Enfin, <oXygen/>XML est documenté, il existe une communauté permettant de solutionner les problèmes liés à l'apprentissage.

Le lancement s'effectue au moyen d'une console qui cette fois n'affiche pas de message d'erreur...



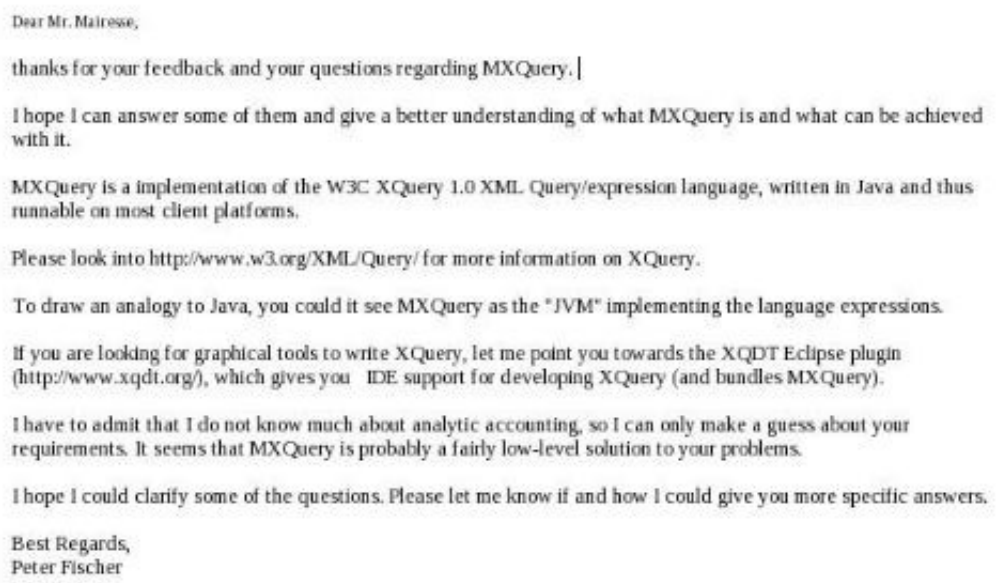
Quant à l'aspect de <oXygen/>XML , convivial et ergonomique.



MxQuery et XQDT.

Sans vouloir transformer ce chapitre en saga, nous revenons sur MxQuery.
MxQuery est un outil XQuery issu d'un projet de l'ETH Zurich.
Ce projet consiste à créer un « XQuery engine » nécessitant une faible mémoire afin de le porter sur des supports mobiles.
Les participants de ce projet sont actifs dans le monde W3C.
Suite à un résultat peu ergonomique comme ceux détaillés ci-dessus, et un manque de documentation, la question quant à une utilisation graphique a été posée à Monsieur Peter Fischer, co-auteur du projet.

Il a très gentillement conseillé de se tourner vers Eclipse, un projet de plate-forme patronné par IBM, pour lequel un plugin XQDT a été développé, et qui intègre Mxquery.



Dear Mr. Mairesse,

thanks for your feedback and your questions regarding MXQuery.

I hope I can answer some of them and give a better understanding of what MXQuery is and what can be achieved with it.

MXQuery is a implementation of the W3C XQuery 1.0 XML Query/expression language, written in Java and thus runnable on most client platforms.

Please look into http://www.w3.org/XML/Query/ for more information on XQuery.

To draw an analogy to Java, you could it see MXQuery as the "JVM" implementing the language expressions.

If you are looking for graphical tools to write XQuery, let me point you towards the XQDT Eclipse plugin (http://www.xqdt.org/), which gives you IDE support for developing XQuery (and bundles MXQuery).

I have to admit that I do not know much about analytic accounting, so I can only make a guess about your requirements. It seems that MXQuery is probably a fairly low-level solution to your problems.

I hope I could clarify some of the questions. Please let me know if and how I could give you more specific answers.

Best Regards,
Peter Fischer

Nous retranscrivons ci-dessous pour plus de lisibilité.

Dear Mr. Mairesse
thanks for your feedback and your questions regarding MXQuery.

I hope I can answer some of them and give a better understanding of what MXQuery is and what can be achieved with it.

MXQuery is a implementation of the W3C XQuery 1,0 XML Query/expression language, written in java and thus runnable on most client platforms.

Please look into http://www.w3org/XML/Query/ for more information on XQuery.

To drawn an analogy to java, you coud it see MXQuery as the « JVM » implementing the language expressions.

If you are looking for graphical tools to write XQuery, let me point you towards the XQDT Eclipse plugin (http://www.xqdt.org/) which gives you IDE support for developing XQuery (and bundles MXQuery).

I have to admit that I do not know much about analytic accounting, so I can only make a guess about your requirements. It seems that MXQuery is probably a fairly low-level solution to your problems.

I hope I could clarify some of the questions. Please let me know if and how I could give you more specific answers.

Best Regards,
Peter Fischer

Au moment de rédiger ces lignes, la licence Oxygen était acquise et une bonne partie du travail réalisée.
Nous avons toutefois tenté XQDT qui s'installe sous Linux de façon automatique et simple.
Toutefois, XQDT s'appuie sur le logiciel XQuery Zorba (existant en .rpm) dont l'installation nécessitait l'installation de plusieurs dizaines de librairies qui à leur tour allaient en requérir d'autres.

La tentative s'est soldée par un échec pour le déploiement de Zorba, la question reste donc ouverte.

XQuery low-level.

Nous notons la remarque de Monsieur Fischer: « It seems that MXQuery is probably a fairly low-level solution to your problems ».
N'étant pas à même de mesurer la portée de la remarque, nous en prenons note et poursuivons notre démarche.

Echec de la solution libre.

Pour conclure cette partie logicielle, nous avons dû renoncer à utiliser des logiciels libres par manque de connaissance probablement, par manque de documentation.
Linux n'a pas simplifié la démarche, il est possible qu'un Operating System plus répandu aurait ouvert d'autres possibilités en matière d'installation logicielle.
Nous utiliserons donc une solution commerciale: Oxygen.

Chapitre 4. Modélisation ERD et hiérarchique.

En cette partie consacrée à la modélisation, nous devons transposer la description technique de la méthode Activity Based Costing en concepts XML, servant de base à la création du document XML reprenant les informations ABC.

Nous l'avons déjà évoqué: XML a une structure hiérarchique.
La modélisation des données se fera à l'aide d'un Entity Relationship Diagram (ERD), que nous transformerons ensuite en modèle hiérarchique.

Modèle Entity Relationship Diagram.

Nous représentons les données et les relations qui les lient au moyen d'un modèle Entity Relationship Diagram (ERD).
Nous créerons des tables et attributs nécessaires; un attribut ayant un rôle index et noté « id_ ... » sera créé pour chaque table.
Nous prenons comme convention de ne pas utiliser, tant pour nommer les tables que les attributs , ni de majuscules, ni d'accent, qui sont sources de confusions et erreurs dans le développement informatique de ce travail.

L'application ABC que nous avons proposée met en oeuvre des charges directes, des charges indirectes, des regroupements en familles de coûts, et des activités.

Nous pouvons créer les tables correspondantes, soit: « charge_directe », « charge_indirecte », « famille_cout » et « activite ».

Les quatre tables que nous venons de définir peuvent prendre les attributs suivants:

table « charge_directe »: « id_charge_d », « montant_dir_htva_impute », « quantite »
table « charge_indirecte »: « id_chargeindirecte », « montant_htva_impute »
table « famille_cout » : « id_famille », « nom_famille »
table « activite » : « id_activité », « nom_activite », « unite_oeuvre »

Nb. « htva » que nous trouvons pour les attributs « montant_dir_htva_impute » et « montant_htva_impute » signifie « hors TVA ».
La relation comptable entre charge indirecte et famille de coût étant articulée autour des postes du pcmn, nous créons également une table « pcmn » dont les attributs seront le code pcmn (6 chiffres) et l'intitulé correspondant.

table **« pcmn »** : « id_pcmn », « intitule »

Ces quatre tables, nous permettent d'inscrire et regrouper les données comptables.

D'autres tables seront dédiées à l'utilisation de ces données comptables de départ, que nous devons répartir au sein de commandes passées par des clients à l'entreprise.
Pour ce, comme évoqué plus haut, nous devons effectuer des mesurages au fil du déroulement du processus de l'entreprise afin de répartir nos coûts d'activités.

Il convient donc de créer une table « commande », une table « mesurage », et une table « client ».

table « commande » : « id_commande », « descriptif », « chiffre_affaire »
table « client » : « id_client », « nom » et « adresse »

Les tables de relation.

Nous devons établir des relations entre ces tables de base du système.
C'est-à-dire une relation entre « pcmn » et « charge_indirecte », entre « charge_indirecte » et « famille_cout », entre « famille_cout » et « activite », entre « activite » et « commande », entre « client » et « commande », entre « charge_directe » et « commande », entre « charge_directe » et « pcmn ».

Relation des tables « charge_indirecte » et « pcmn ».

Ces deux tables ont une relation de N à M ou plusieurs à plusieurs.
Nous désignons par le terme « charge indirecte » une pièce comptable contenant une charge indirecte.
Plusieurs destinations d'imputation sont possibles pour une même pièce: un même fournisseur peut reprendre au sein de la même facture des articles que nous destinons à des imputations différentes.

Nous créons la table de relation « imputation » .

La table « imputation » comprendra les attributs suivants:

table « imputation »: « id_imputation », « id_charge_indirecte », « id_pcmn_imputation », « montant_htva_impute ».

« id_imputation » est l'identificateur propre de la table « imputation », deux attributs sont les identificateurs des deux tables que nous mettons en relation: « id_charge_indirecte » et « id_pcmn_imputation », le dernier attribut est le montant hors TVA imputé.

Relation des tables « pcmn » et « famille_cout ».

La relation entre ces deux tables est également de N à M.
Une Famille de coûts est constituée de un ou plusieurs postes pcmn intervenant en proportions différentes dans le calcul de sa valeur finale.
Un poste pcmn peut être intégré dans une ou plusieurs familles.
La table « composition_famille » sera la relation entre les tables « pcmn » et « famille_cout ».

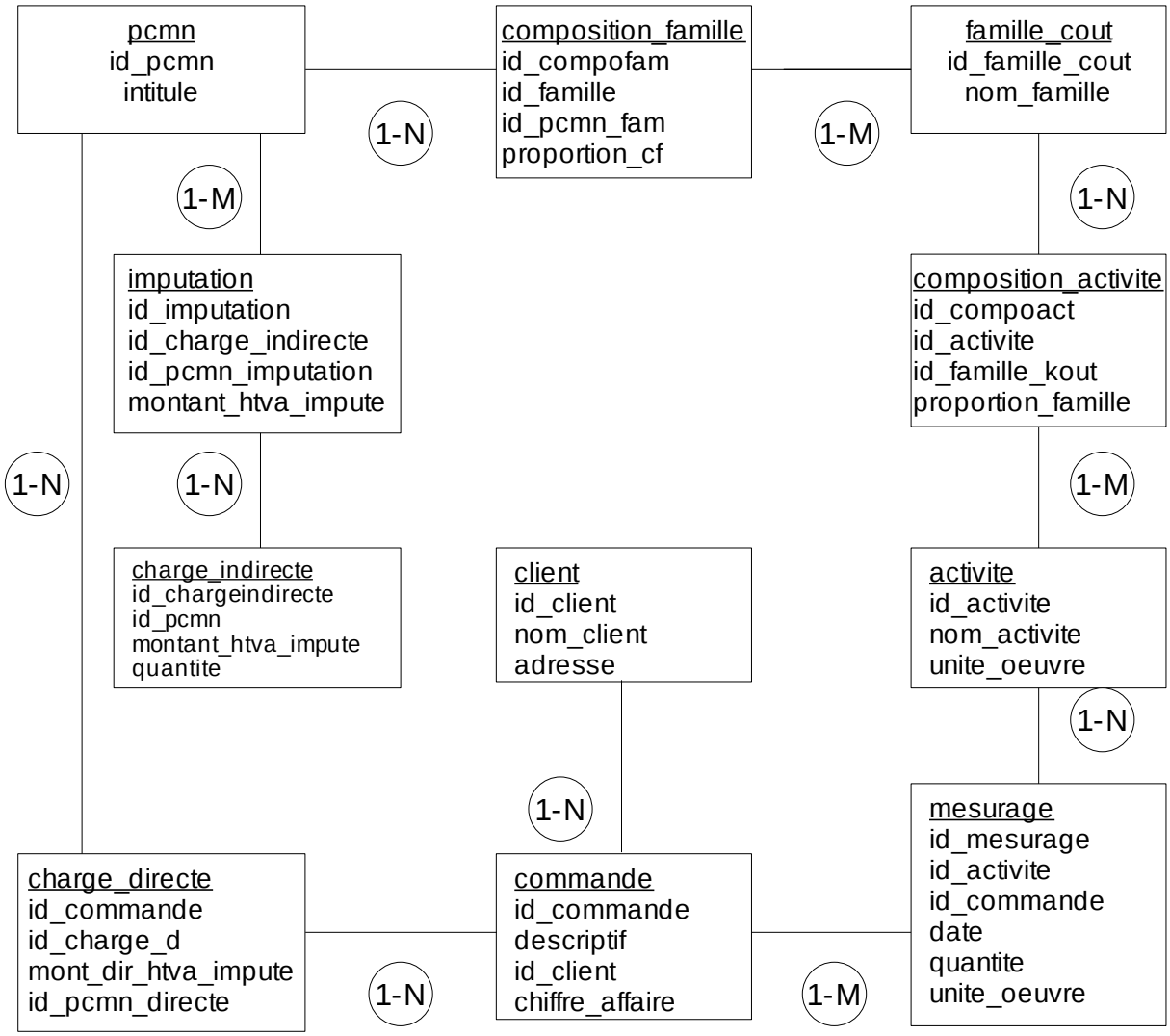
Elle aura pour attributs:

table « composition_famille »: « id_compofam », « id_famille », « id_pcmn_fam », « proportion_cf »

Nous avons utilisé des abréviations: « id_compofam » pour id_composition_famille , « id_pcmn_fam » pour id pcmn (dans famille_cout), et « proportion_cf » pour proportion cout famille.

L'attribut « proportion_cf » contiendra la valeur de répartition d'un poste pcmn intervenant dans

plusieurs familles.
Si un poste pcmn est repris intégralement dans le calcul du coût d'une famille, l'attribut sera 1.
Si il intervient pour 67%, l'attribut sera de 0.67 .



Relation des tables « famille_cout » et « activite ».

La relation entre ces deux tables est de N à M.
Une famille de coûts peut être composante de une ou plusieurs activités , qui elle peut inclure une ou plusieurs familles de coûts.
Exemple: l'activité production inclus la famille7: outil et production et la famille4: formation documentation.

La table « composition_activite » sera la relation entre les tables « famille_cout » et « activite ».
Les attributs en seront:

table « composition_activite »:« id_compoact », « id_activite ». « id_famille_kout », « proportion ».

L'abréviation « id_compoact » est pour id_composition_activite.
Comme précédemment, cet attribut « proportion » permet de fixer la proportion d'intervention

d'une famille de coûts au sein d'une activité.
Relation des tables « activite » et « commande ».

Une commande peut mettre en oeuvre une ou plusieurs activités, une activité est utilisée par une ou plusieurs commande, ce qui implique une relation de N à M.
La table de relation que nous utiliserons sera utilisée pour effectuer les mesurages d'unités d'oeuvre des différentes activités nécessaires à l'analyse ABC.
Ce sera la table « mesurage », comportant les attributs:

table « mesurage »: « id_commande », « id_activite », « id_mesurage », « date », « unite_oeuvre », « quantite ».

Les champs « id_commande », « id_activite » sont des champs de relation.

Relation des tables « client » et « commande ».

Ces deux tables ont une relation de 1 à N.
Un client peut passer une ou plusieurs commandes, une commande correspond à un seul client.
La relation s'établira en incluant l'identificateur de client au sein de commande.
La table « commande » devient:

table « commande »: « id_commande », « descriptif », « id_client », « chiffre_affaire »

Relation des tables « charge_directe » et « commande ».

Ces deux tables ont une relation de 1 à N.
Une commande peut se voir imputer plusieurs charges directes, une charge directe est imputée à une seule commande.
La relation est établie en incluant l'identificateur de la commande dans la table charge_directe.

Relation des tables « charge_directe » et « pcmn ».

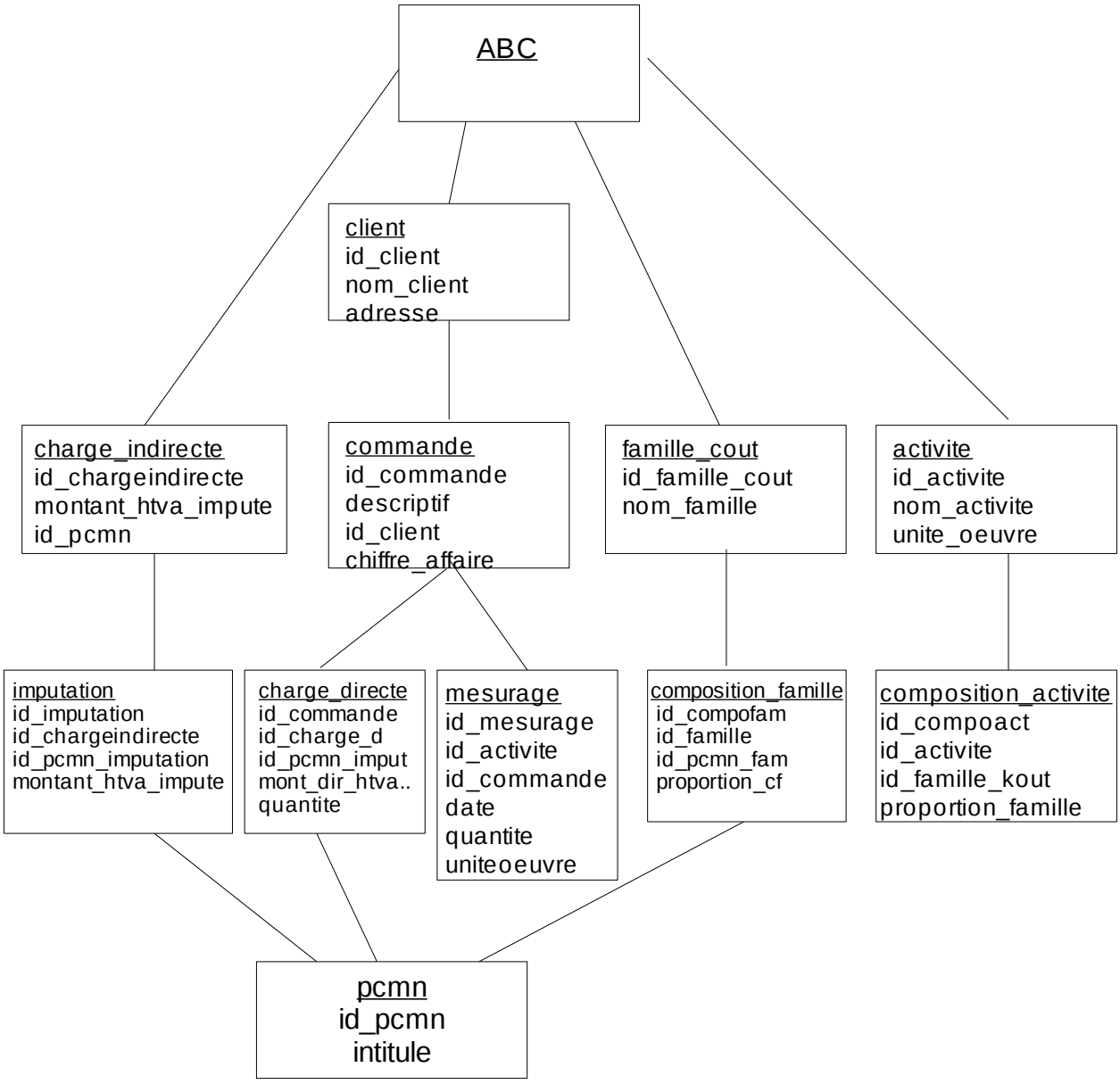
Ces deux tables ont une relation de 1 à N.
Un poste pcmn correspond à une seule charge directe; une charge directe peut recevoir plusieurs attributions pcmn.
Nous incluons l'identificateur du poste pcmn que nous appelons « id_pcmn_directe »
La table « charge_directe » devient:

table « charge_directe »: « id_charge_d », « id_commande », « id_pcmn _directe », « montant_dir_htva_impute », « quantite »

Nb. Dans quelques cas, nous avons choisi de modifier le nom de l'attribut comme par exemple « id_pcmn_directe » qui est dans la table « pcmn » id_pcmn et dans la table « composition_famille »: « id_pcmn_fam.
Cela ne modifie en rien la valeur de l'attribut, et nous permettra dans la suite de ce travail de pouvoir distinguer les trois attributs.

Modélisation hiérarchique.

Comme nous l'avons évoqué plus haut, XML propose une représentation hiérarchique des données, qui forme un arbre XML.
Un élément racine est le point de départ obligatoire d'une modélisation hiérarchique. Nous ne disposons pas parmi les tables du modèle ERD que nous venons de développer, d'élément racine (ou parent) de tous les autres éléments (tables).
Nous créons cet élément: « ABC » (en majuscules exception faite pour l'élément racine) qui sera un élément vide.



Le premier élément hiérarchiquement sous la racine « ABC » est la table « client ».

Au niveau hiérarchique suivant se trouvent les quatre tables principales de notre travail, « charge_indirecte », « commande », « famille_cout » et « activite ».
Elles sont descendantes de « ABC », sauf la table « commande » qui est enfant de « client ».
Elles se positionnent comme élément parent des tables de relation que nous avons créées dans le modèle ERD.

« charge_indirecte » est parent de « imputation »: à une « chargeindirecte » correspond 1 où N instances d' « imputation ».

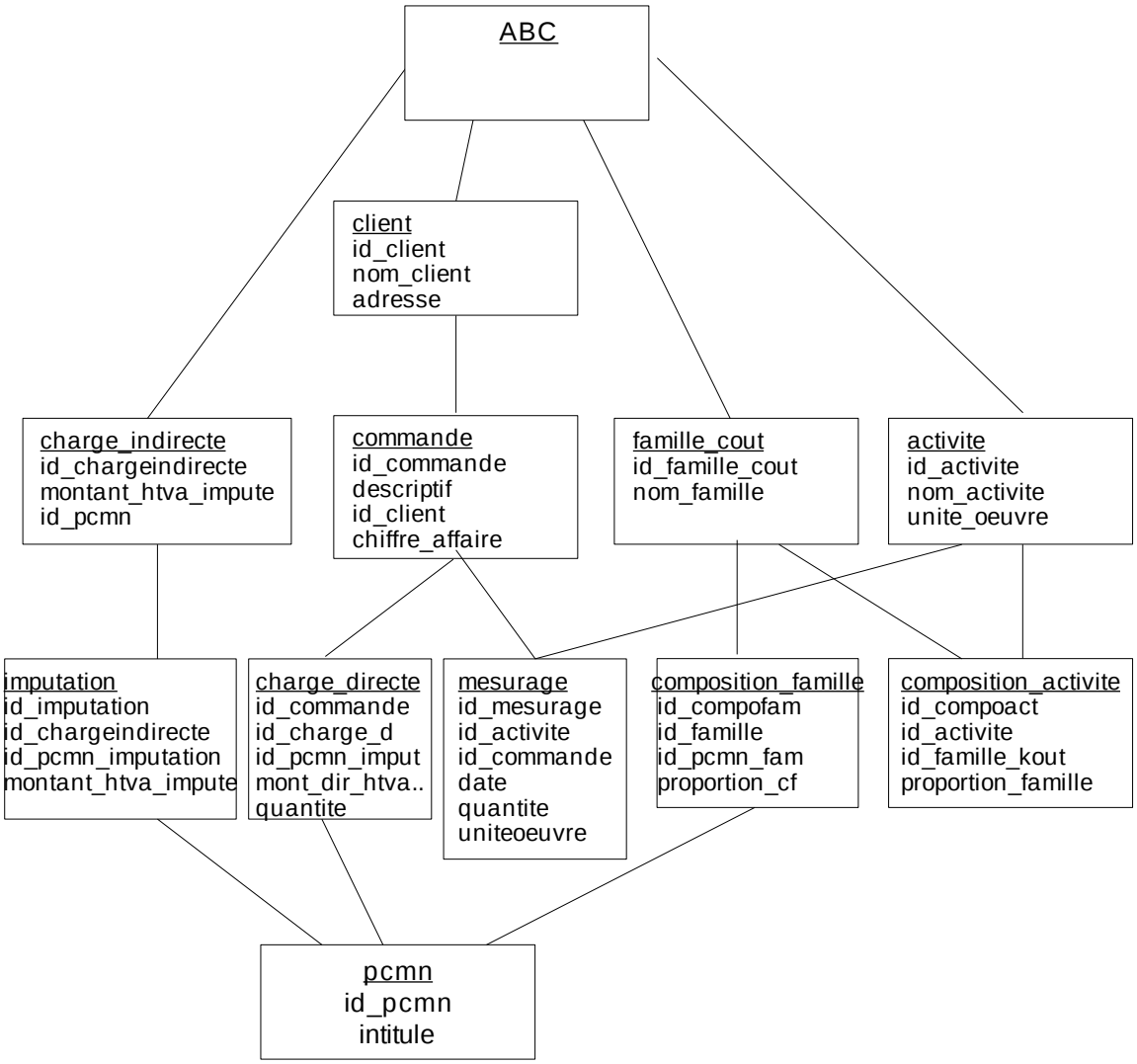
« commande » est parent de « mesurage »: à une « commande » correspond 1 où N instances de « mesurage ».
« commande » est également parent de « charge_directe »: à une « commande » correspond 1 où N instances de « charge_directe ».

« famille_cout » est parent de « composition_famille »: à une « famille_cout » correspond 1 où N instances de « composition_famille ».

« activite » est parent de « composition_activite »: à une « activite » correspond 1 où N instances de « composition_activite ».

Eléments à parents multiples.

Notre représentation hiérarchique ne serait pas complète sans les éléments ayant des parents multiples.



Il s'agit des trois tables descendantes de plus de une table.

Nous voyons que « composition_activite » a pour parent « activite » comme nous venons de le voir, mais également « famille_cout »: à une « famille_cout » correspond 1 où N instances de « composition_activite ».

La table de relation « mesurage » est descendante de « commande » et « activite ».

Enfin, au dernier rang hiérarchique, la table « pcmn » est une table enfant à la fois d'« imputation » « charge_directe » et de « composition_famille ».

Chapitre 5. Le fichier ABC/XML et la DTD.

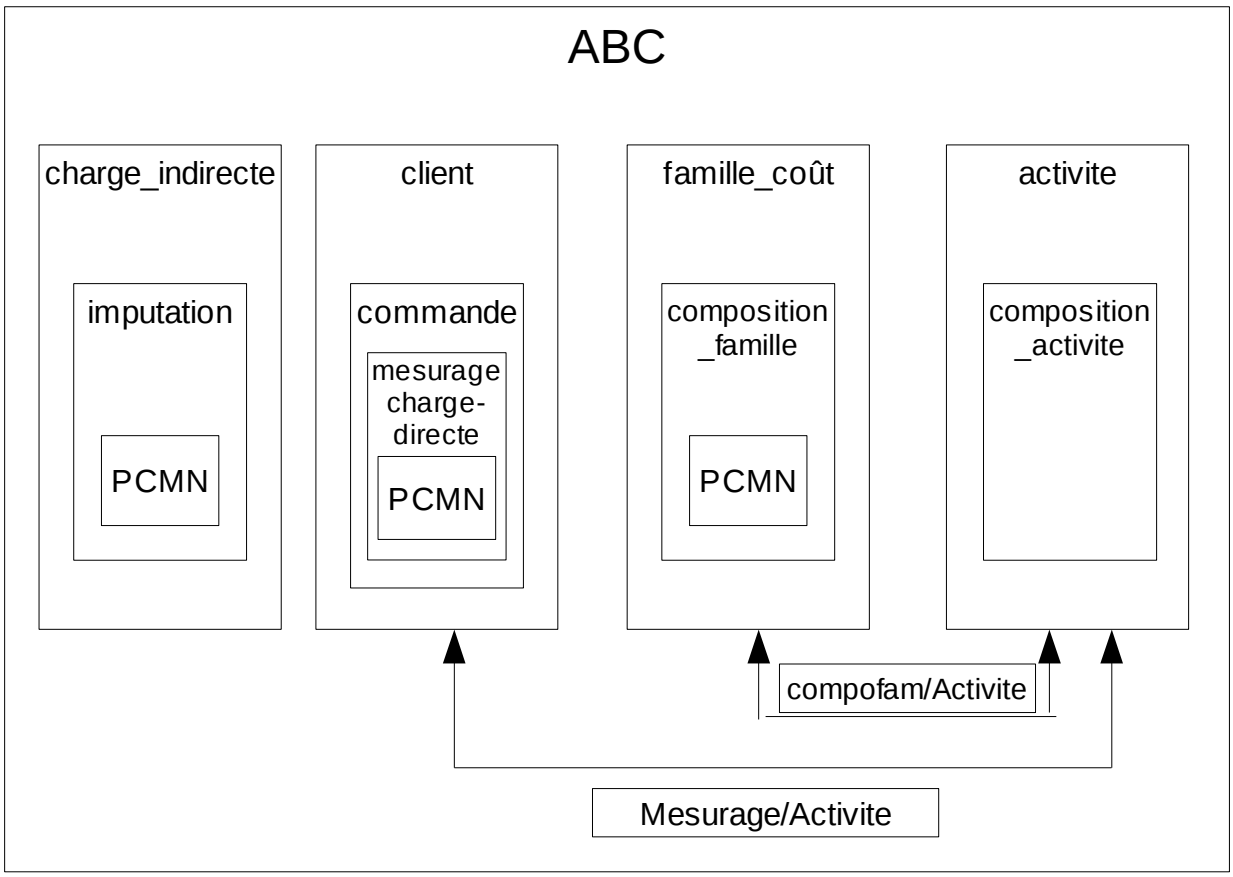
Ayant modélisé hiérarchiquement l'application ABC, nous devons la transposer sous forme de document XML.
Lors d'une première étape, nous créerons la DTD qui comme nous l'avons vu plus avant, validera notre document XML.
Ensuite, la seconde étape consistera à élaborer le document XML correspondant, et à y saisir de l'information.

La DTD.

Dans la DTD, nous devons traduire les tables issues des modélisations ERD et Hiérarchiques.
XML retient la notion d'élément, nos tables deviendront des éléments.
Le terme d'attribut restera, un élément comportant des attributs, tout comme une table comporte des attributs.
Hormis le fait de créer des éléments et leurs attributs, nous devons également définir dans la DTD la structure des relations entre éléments et leur ordonnancement.

Les relations entre éléments.

En examinant le modèle hiérarchique, nous distinguons que les tables se répartissent en quatre « colonnes » distinctes, qui sont quatre relations parent/enfants d'éléments.



Ces quatre relations sont reliées entre elles par les trois éléments ayant deux parents, « mesurage », « composition_activite » et « pcmn » comme nous l'avons vu dans la modélisation hiérarchique.

XML autorise deux types de liens entre éléments: l'imbrication des éléments ou les pointeurs.

L'imbrication des éléments.

L'imbrication d'éléments consiste à inclure l'(es) élément(s) enfant(s) dans l'élément parent.
Le défaut de cette méthode est une lourdeur du document due à la redondance de l'information.
XML est par nature verbeux, ce n'est donc pas anormal.

Nous imbriquerons les quatre « colonnes » du modèle hiérarchique.

- L'élément racine (vide) « ABC » inclut les éléments « client », « activite », « famille_cout », « charge indirecte » et « pcmn ».
- L'élément « client » inclut « commande » qui inclut « mesurage » et « charge_directe ».
- L'élément « Activite » inclut « composition_activite ».
- L'élément « Famille_cout » inclut « composition_famille ».
- L'élément « charge indirecte » inclut « imputation ».

Les pointeurs.

Les pointeurs utilisent généralement les types ID IDREF(S) pour mettre des éléments en relation.
Pour établir la relation entre les éléments pointés par les attributs ID IDREF(S) situé dans des éléments distincts, l'outil de requête compare la valeur de ces attributs.

Exemple: si l'attribut IDREF d'un élément « livre » prend pour valeur « memoire1 » et l'attribut ID d' un élément « bibliothèque » prend pour valeur « roman10 », l'évaluation comparative ne permettra pas de liaison.
Si par contre l'attribut ID devient « memoire1 », la liaison entre les éléments par le pointeur sera établie.

Comme nous le verrons plus bas, la notion de pointeur peut relever de la comparaison d'attributs identifiant sans recourir à ID IDREF.

Si les pointeurs permettent un découplage des éléments, il génèrent un inconvénient: la lenteur de traitement.
Le parseur ne navigue pas facilement entre les ID IDREF, la relation ID IDREF est unidirectionnelle: de IDREF vers ID s'opère facilement, mais l'inverse est plus laborieux.

Cela implique que le parseur doit parfois effectuer plusieurs passages au sein du document si l'élément comportant le ID est positionné avant celui contenant IDREF.

Ce processus peut devenir laborieux pour les fichiers XML de grande taille.
Nous aurons recours à cette méthode, notre fichier ABC_XML ayant peu de chances d'atteindre une taille critique.

(source: XML et bases de données. ISBN: 2-212-09282-2 page 34-35).

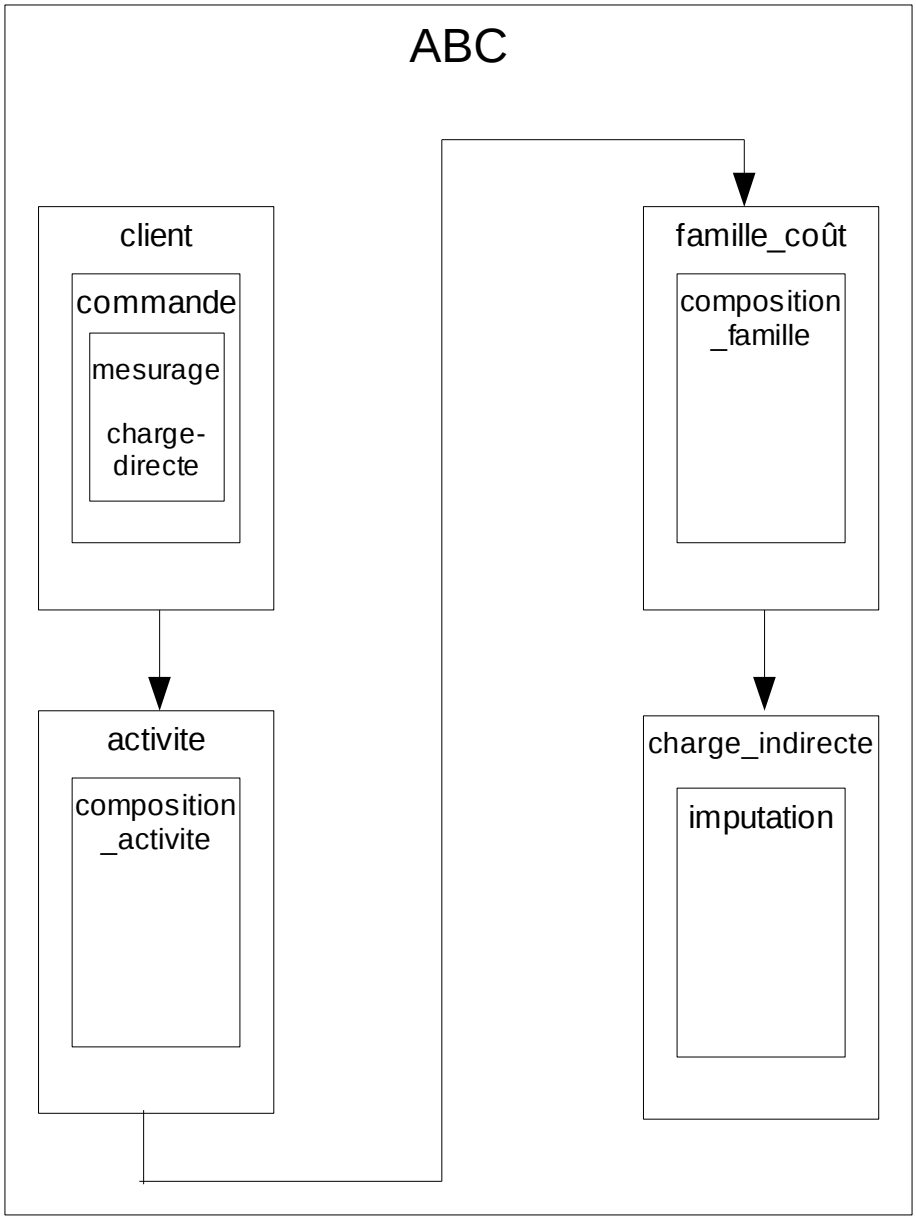
Nous utiliserons des pointeurs pour relier les quatre groupes d'éléments de relation définis ci-dessus

- « mesurage » pointe vers « activite »
- « composition_activite » pointe vers « famille_cout »
- « imputation » pointe vers « pcmn »; « composition_famille » pointe vers « pcmn »; « charge _directe » pointe vers « pcmn ».

Nous remarquons que les pointeurs portent sur trois les éléments « multi-parents » évoqués en modélisation hiérarchique.

Organisation de l'ordre des éléments.

Nous devons choisir l'ordre dans lequel apparaîtront ces éléments imbriqués.
Nous optons une représentation lisible.
A savoir: partir de client et de sa commande qui fait l'objet de mesurages ayant trait à des activités constituées de familles de coûts elles mêmes composées de charges indirectes.



La DTD en pratique.

Elément racine.

La première ligne de la DTD déclare l'élément racine qui est « ABC » ainsi que les éléments imbriqués dans ABC

```
<!ELEMENT ABC (client+, activite+, famille_cout+, charge_indirecte+, pcmn+)>
```

Plusieurs instances de ces éléments sont autorisées dans le document XML du fait de la présence du signe + suivant chacun des éléments déclarés (client+,..etc..)

Elément « client ».

```
<!ELEMENT client (commande+)>
```

L'élément « client » est déclaré, il inclut un élément « commande » qui peut faire l'objet de plusieurs instantiations.

```
<!ATTLIST client
      id_client (jaeger01| cairelli02| panerai03) #REQUIRED
      nom (jaeger| cairelli| panerai) #REQUIRED
      adresse CDATA #REQUIRED>
```

Pour chaque élément (sauf « ABC » qui est vide), nous déclarons la liste des attributs correspondant.
La liste des attributs de l'entité « client », soit <!ATTLIST client> :

- id_client, dont le type est une liste de choix entre les différentes valeurs possibles pour cet attribut, soit « jaeger01 » ou « cairelli02 » ou bien « panerai03 ». Il n'est pas possible d'encoder dans le futur document XML une valeur autre que ces trois, autorisées par la DTD.
- nom, où nous saisirons le nom du client avec logiquement trois possibilités, soit « jaeger » ou « cairelli » ou bien « panerai ».
- adresse, où figurera l'adresse du client, de type CDATA.

Elément « commande ».

```
<!ELEMENT client (commande+)>
.............

<!ELEMENT commande (mesurage+, charge_directe+)>
   <!ATTLIST commande
         id_commande CDATA #REQUIRED
         descriptif CDATA #REQUIRED
         chiffre_affaire CDATA #REQUIRED>
```

La déclaration de l'élément commande intervient au sein de « client », lui-même comportant les éléments mesurage et charge_directe (plusieurs instantiations de mesurage et de charge_directe sont possibles) , l'élément commande étant un élément imbriqué dans l'élément client.
Les attributs de l'élément « commande »: « id_commande », « descriptif » et « chiffre_affaire »,

tous trois en format CDATA.
Notons que nous n'avons pas utilisé un attribut « id_client » dans l'élément « commande », contrairement à ce qu'indiquent les schémas de modélisation ERD et hiérarchique, où il avait un rôle de pointeur qui n'est plus nécessaire, les entités « client » et « commande » étant imbriquées.

Elément « mesurage ».

```
<!ELEMENT client (commande+)>
.............

<!ELEMENT commande (mesurage+)>
.............

<!ELEMENT mesurage EMPTY>
        <!ATTLIST mesurage
            id-mesurage CDATA #REQUIRED
            id_aktivite IDREF #REQUIRED
            date CDATA #REQUIRED
            quantite CDATA #REQUIRED
             unite_oeuvre (heure_communication| heure_demarchage| article_unitaire| kilometre|
                           heure_production| heure_comptabilite_abc| minute_telecom)
                          #REQUIRED >
```

Nous déclarons ensuite l'élément « mesurage » qui ne comporte pas d'élément imbriqué, et est donc « EMPTY ».

Les attributs de cette élément: « id_mesurage », « id_aktivite », « date » (du relevé), « quantite » (d'unités d'oeuvre) et « unite_oeuvre » avec une liste de choix imposés.
Nb. Il a été choisi d'orthographier « id_aktivite » avec un « k » pour être en mesure de la distinguer d'un second attribut « id_activite ».

Notons que l'attribut « id_activite » est de type IDREF: il s'agit du pointeur reliant l'élément « mesurage » à l'élément « activite ».

Ici aussi, du fait de l'imbrication, l'attribut « id_commande » des modélisations précédentes n'a pas été créé.

Elément « charge_directe ».

```
<!ELEMENT charge_directe EMPTY>
        <!ATTLIST charge_directe
            id_charge_d CDATA #REQUIRED
            id_pcmn (601000| 620000) #REQUIRED
            montant_dir_htva_impute CDATA #REQUIRED
            quantite CDATA #REQUIRED>
```

Nous déclarons enfin l'élément « charge_directe ».
Ses attributs sont: « id_charge_d », « id_pcmn » qui a une liste de choix de postes pcmn autorisés, « montant_dir_htva_impute » et « quantite ».
Sauf « id_pcmn », les attributs sont de type CDATA.

Comme pour l'élément « mesurage », l'attribut « id_commande » n'est plus repris.

Elément « activite ».

```
<!ELEMENT activite (composition_activite+)>
        <!ATTLIST activite
            id_activite ID #REQUIRED
            nom_activite (act1_communication| act2_demarchage| act3_logistique| act4_transport|
                        act5_production| act6_comptabilite_abc| act7_telecom ) #REQUIRED>
```

L'élément « activite » qui inclut un élément « composition_activite » peut être instancié une ou plusieurs fois. Les attributs en sont « id_activite » et « nom_activite », avec une liste de choix; « id_activite » est de type ID.
Il assure le lien avec l'attribut IDREF que nous décrivons quelques lignes plus haut.
Le lien est effectué par comparaison de la valeur prise par ces deux attributs.

Element « composition_activite ».

```
<!ELEMENT activite (composition_activite+)>
.............

<!ELEMENT composition_activite EMPTY>
        <!ATTLIST composition_activite
            id_compoact CDATA #REQUIRED
            id_famille_kout IDREF #REQUIRED
            proportion CDATA #REQUIRED>
```

L'élément « composition_activite » n'inclut pas d'autre élément (EMPTY).
Ses attributs: « id_compoact » de type CDATA; chaque instanciation de l'élément « composition _activite » aura une valeur unique pour cet attribut; id_famille_kout de type IDREF est un pointeur vers l'élément « famille_cout »; proportion est de type CDATA.

Elément « famille_cout ».

```
<!ELEMENT famille_cout (composition_famille+)>
        <!ATTLIST famille_cout
            id_famille_cout ID #REQUIRED
            nom_famille  (fam1_publicite| fam2_bureau| fam3_stock| fam4_formation_
            documentation| fam5_honoraires| fam6_transport| fam7_outil_production|
            fam8_telecom ) #REQUIRED>
```

Définissons l'élément « famille_cout » qui inclut l'élément « composition_famille ».
Ses attributs: « id_famille_cout », de type ID qui assure la relation avec l'élément « composition_activite » (ci-dessus) et recevra une valeur unique à chaque instantiation; « nom_famille » imposant une liste de choix.

Elément « composition_famille ».

```
<!ELEMENT famille_cout (composition_famille+)>
.....

<!ELEMENT composition_famille EMPTY>
        <!ATTLIST composition_famille
            id_compofam CDATA #REQUIRED
            id_pcmn_fam (601300| 606140| 611130| 611150| 611300| 611350| 612000| 612160|
                         612300| 612400| 612500| 613250| 613530| 613540| 615100| 615200|
                         615220| 616100| 616200| 616300| 616400| 630210| 630220| 630231|
                         630232| 640100| 640200) #REQUIRED
            proportion_cf CDATA #REQUIRED>
```

« composition_famille » est l'élément imbriqué dans « composition_famille ». L'attribut « id_pcmn_fam » liste les postes pcmn autorisés par leur code en 6 chiffres, « proportion_cf » de type CDATA reprendra la part d'intervention de tel poste pcmn dans la composition de famille.

Elément « charge_indirecte ».

```
<!ELEMENT charge_indirecte (imputation+)>
<!ATTLIST charge_indirecte
        id_chargeindirecte CDATA #REQUIRED
        montant_htva_total CDATA #REQUIRED>
```

Nous déclarons l'élément « charge_indirecte » incluant l'élément « imputation » dont une ou plusieurs instances sont possibles.
Ses attributs: « id_chargeindirecte » et « montant_htva_total » tous deux au format CDATA.

Elément « imputation ».

```
<!ELEMENT charge_indirecte (imputation+)>
.....

<!ELEMENT imputation EMPTY>
        <!ATTLIST imputation
            id_imputation CDATA #REQUIRED
            id_pcmn_imputation (601300| 606140| 611130| 611150| 611300| 611350| 612000|
                                612160| 612300| 612400| 612500| 613250| 613530| 613540|
                                615100| 615200| 615220| 616100| 616200| 616300| 616400|
                                630210| 630220| 630231| 630232| 640100| 640200) #REQUIRED
            montant_htva_impute CDATA #REQUIRED>
```

L'élément « imputation » incorporé à « charge_indirecte » a pour attributs « id_imputation » de type CDATA, pour lequel chaque instanciation recevra une valeur unique et une liste de choix portant sur les codes pcmn comme précédemment (la liste est identique) pour l'attribut « id_pcmn_imputation ».

Elément « pcmn ».

```
<!ELEMENT pcmn EMPTY>
        <!ATTLIST pcmn
            id_pcmn (601300| 606140| 611130| 611150| 611300| 611350| 612000| 612160| 612300|
                612400| 612500| 613250| 613530| 613540| 615100| 615200| 615220| 616100|
                616200| 616300| 616400| 630210| 630220| 630231| 630232| 640100| 640200)
                #REQUIRED
            intitule (601300_outillage|606140_frais_transport|611130_location_outillage|
                611150_location_vehicule| 611300_entretien_locaux | 611350_entretien_
                vehicule| 612000_energie_immeuble| 612160_carburant|
                612300_documentation| 612400_imprimes_fourn_bureau|
                612500_petit_mat_bureau| 613250_honoraire_comptable| 613530
                _assurance_vehicule | 613540_assurance_RC| 615000_frais_deplacement|
                615100_frais_representation| 615200_publicite_annonces| 615220
                _foires_expositions| 616100_frais_postaux| 616200_telephone| 616300_gsm|
                616400_internet| 630210_dot_amort_immeuble| 630220_dot_amort_outil|
                630231_dot_amort_mobilier| 630232_dot_amort_matroulant|
                640100_dot_taxe_vehicule| 640200_precompte_immobilier) #REQUIRED>
```

Nous présentons ci-dessous la DTD.

```
<!ELEMENT ABC (commande+, activite+, famille_cout+, charge_indirecte+, client+, pcmn+ )>

<!ELEMENT commande (mesurage+)>
<!ATTLIST commande
        id_commande CDATA #REQUIRED
        id_client (jaeger| cairelli| panerai) #REQUIRED
        descriptif CDATA #REQUIRED
        chiffre_affaire CDATA #REQUIRED>

        <!ELEMENT mesurage EMPTY>
        <!ATTLIST mesurage
            id-mesurage CDATA #REQUIRED
            id_activite IDREF #REQUIRED
            date CDATA #REQUIRED
            quantite CDATA #REQUIRED
            unite_oeuvre (heure_communication| heure_demarchage| heure_logistique| kilometre|
             heure_production| heure_comptabilite_abc| minute_telecom) #REQUIRED >

       <!ELEMENT charge_directe EMPTY>
        <!ATTLIST charge_directe
            id_charge_d CDATA #REQUIRED
            id_pcmn (601000| 620000) #REQUIRED
            montant_dir_htva_impute CDATA #REQUIRED
            quantite CDATA #REQUIRED>

<!ELEMENT activite (composition_activite+)>
<!ATTLIST activite
        id_activite ID #REQUIRED
        nom_activite ( act_communication| act_demarchage| act_logistique| act_transport| act_production
                | act_comptabilite_abc| act_telecom ) #REQUIRED>

        <!ELEMENT composition_activite EMPTY>
```

```
        <!ATTLIST composition_activite
                id_compoact CDATA #REQUIRED
                id_famille_cout IDREF #REQUIRED
                proportion CDATA #REQUIRED>

<!ELEMENT famille_cout (composition_famille+)>
<!ATTLIST famille_cout
        id_famille_cout ID #REQUIRED
        nom_famille (fam_publicite| fam_bureau| fam_stock| fam_formation_documentation|
        fam_honoraires| fam_transport| fam_outil_production| fam_telecom ) #REQUIRED>

        <!ELEMENT composition_famille EMPTY>
        <!ATTLIST composition_famille
                id_fam CDATA #REQUIRED
                id_pcmn_fam (601300| 611150| 612000| 612300| 612400| 612500| 613250| 613530|
                        615200| 616200| 616300) #REQUIRED
                proportion_cf CDATA #REQUIRED>

<!ELEMENT charge_indirecte (imputation+)>
<!ATTLIST charge_indirecte
        id_chargeindirecte CDATA #REQUIRED
        montant_htva_total CDATA #REQUIRED>

        <!ELEMENT imputation EMPTY>
        <!ATTLIST imputation
                id_imputation CDATA #REQUIRED
                id_pcmn_imputation (601300| 611150| 612000| 612300| 612400| 612500| 613250| 613530|
                615200| 616200| 616300) #REQUIRED
                montant_htva_impute CDATA #REQUIRED>

<!ELEMENT client EMPTY>
        <!ATTLIST client
                id_klient (jaeger| cairelli| panerai) #REQUIRED
                adresse CDATA #REQUIRED>

<!ELEMENT pcmn EMPTY>
        <!ATTLIST pcmn
                id_pcmn (601300| 611150| 612000| 612300| 612400| 612500| 613250| 613530| 615200|
                616200| 616300) #REQUIRED
                intitule (601300_outillage| 611150_location_vehicule| 612000_energie_immeuble| 6
                12300_documentation| 612400_imprimes_fourn_bureau| 612500_petit_mat_bureau|
                613250_honoraire_comptable| 613530_assurance_vehicule| 615000_frais_deplacement|
                615200_frais_representation| 615200_publicite_annonces| 616200_telephone|
                616300_gsm)             #REQUIRED
```

Le document XML.

Nous rédigeons le document XML.
Comme nous l'avons déjà évoqué plus haut, la DTD doit être respectée scrupuleusement.
Si au lieu de saisir « pcmn », nous saisissons « pmcn », le parseur décèlera l'erreur qui sera une erreur bloquante, c'est à dire qu'aucune opération n'est possible tant qu'il n'y a pas correction.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE ABC SYSTEM "/home/jean/Memoireulb/XQ27avril/abc_avril27_id01.dtd">
```

Des deux premières lignes du document XML , la première est la déclaration XML qui spécifie quelle est la version XML utilisée (1.0) et le type d'encodage du document (ISO-8859-1).
La seconde ligne indique que l'élément racine est « ABC », et précise où se trouve la DTD dans le système de fichiers de l'ordinateur.

```
<ABC>
   <client id_client="" nom="" adresse="">
```

L'ensemble des éléments sont inclus dans l'élément racine <ABC> qui est la première balise ouverte.
Nous avons ensuite l'élément <client> qui inclut l'élément <commande> qui inclut les éléments <mesurage> et <charge_directe>.
La balise </commande> est refermée après l'élément <charge_directe>, la balise </client> est refermée après l'élément </commande>.

```
<ABC>
   <client id_client="" nom="" adresse="">
      <commande id_commande="" descriptif="" chiffre_affaire="">
         <mesurage id_mesurage="" id_aktivite="" date="" quantite="" unite_oeuvre=""/>
         <charge_directe id_charge_d="" id_pcmn="" montant_dir_htva_impute="" quantite=""/>
      </commande>
   </client>
```

Au même niveau hiérarchique que l'élément <client>, l'élément <activité> qui inclut l'élément <composition_activite>.
La balise </activite> est refermée après l'élément </composition_activite>.

```
<activite id_activite="" nom_activite="" unite_oeuvre="">
      <composition_activite id_compoact="" id_famille_kout="" proportion=""/>
 </activite>
```

L'élément <famille_cout> inclut l'élément <composition_famille>; la balise </famille_cout> est refermée après l'élément </composition_famille>.

```
<famille_cout id_famille_cout="" nom_famille="">
      <composition_famille id_compofam="" id_pcmn_fam="" proportion_cf=""/>
</famille_cout>
```

L'élément <charge_indirecte> inclut l'élément <imputation>, la balise </charge_indirecte> est

refermée après l'élément </imputation>.

```
<charge_indirecte id_chargeindirecte="" montant_htva_total="">
      <imputation id_imputation="" id_pcmn_imputation="" montant_htva_impute=""/>
</charge_indirecte>
```

Enfin, le dernier élément <pcmn> qui est lui aussi au même niveau hiérarchique que les éléments <client>, <activité>, <famille_cout> et <charge_indirecte>.
La balise de l'élément racine </ABC> est refermée.

```
 <pcmn id_pcmn="" intitule=""/>
</ABC>
```

Les données du document XML.

Disposant de la structure du document XML, nous devons y saisir les données comptables que nous déclarions au premier chapitre, ainsi que quelques mesurages et autres données client.

Nous distinguons deux types de données au sein du document XML:
- les données relevant de la pratique de l'entreprise
- les données nécessaires à l'analyse ABC.

La seconde catégorie consistera en une information servant de base aux calculs ABC. Elle sera accèdée plus rarement en écriture, et cet accès devrait être idéalement restreint.
Il est nécessaire de bien comprendre les conséquences sur les résultats de l'application ABC avant de procéder à une modification de ces données.

Les données d'analyse ABC.

Ce sont les données des éléments <activite> et <composition_activite>, <famille_cout> et <composition_famille>.

Les activités.

Nous y transcrivons activité par activité quelles familles de coût les composent, et dans quelle proportion.
Cela a été défini dans le chapitre consacré à la méthode ABC, et synthétisé par un tableau que nous reproduisons.

	Act1	Act2	Act3	Act4	Act5	Act6	Act7
Fam1	1						
Fam2	0,3	0,4				0,3	
Fam3			1				
Fam4		0,42			0,34	0,24	
Fam5						1	
Fam6				1			

Fam7			0,27		0,73		
Fam8							1

Pour le premier élément traitant de l'activité « communication », les valeurs d'attributs sont:

id_activite: « act1 »
nom_activite: « act1_communication » (conformément à la liste de choix de la DTD)
unité_oeuvre: « heure_communication » (conformément à la liste de choix de la DTD)

Les valeurs d'attribut des deux éléments <composition_activite> inclut dans l'élément <activite> sont:

id_compoact: « compoact001 »
id_famille_kout: « fam1 »
proportion: « 1 »

et

id_compoact: « compoact002 »
id_famille_kout: « fam2 »
proportion: « .3 »

Ce qui se lit: l'activité1 est composée de la famille de coûts « fam1 » qui intervient à 100%, et de la famille de coûts « fam2 » qui intervient à 30% dans le calcul de sa valeur.

```
<activite id_activite="act1" nom_activite="act1_communication" unite_oeuvre="heure_communication">
        <composition_activite id_compoact="compoact001" id_famille_kout="fam1" proportion="1"/>
        <composition_activite id_compoact="compoact002" id_famille_kout="fam2" proportion=".3"/>
</activite>
```

Le même raisonnement est valable pour les sept activités décrites dans le chapitre ABC.
Nous reproduisons la partie du document XML relative aux activités ci-dessous.

```
<activite id_activite="act1" nom_activite="act1_communication" unite_oeuvre="heure_communication">
        <composition_activite id_compoact="compoact001" id_famille_kout="fam1" proportion="1"/>
        <composition_activite id_compoact="compoact002" id_famille_kout="fam2" proportion=".3"/>
</activite>

<activite id_activite="act2" nom_activite="act2_demarchage" unite_oeuvre="heure_demarchage">
        <composition_activite id_compoact="compoact003" id_famille_kout="fam2" proportion=".4"/>
        <composition_activite id_compoact="compoact004" id_famille_kout="fam4" proportion=".42"/>
</activite>

<activite id_activite="act3" nom_activite="act3_logistique" unite_oeuvre="article_unitaire">
        <composition_activite id_compoact="compoact005" id_famille_kout="fam3" proportion="1"/>
<composition_activite id_compoact="compoact006" id_famille_kout="fam7" proportion=".27"/>

</activite>

<activite id_activite="act4" nom_activite="act4_transport" unite_oeuvre="kilometre">
```

```
        <composition_activite id_compoact="compoact008" id_famille_kout="fam6" proportion="1"/>
</activite>

<activite id_activite="act5" nom_activite="act5_production" unite_oeuvre="heure_production">
        <composition_activite id_compoact="compoact009" id_famille_kout="fam4" proportion=".34"/>
        <composition_activite id_compoact="compoact010" id_famille_kout="fam7" proportion=".73"/>
</activite>

<activite id_activite="act6" nom_activite="act6_comptabilite_abc"
unite_oeuvre="heure_comptabilite_abc">
        <composition_activite id_compoact="compoact011" id_famille_kout="fam5" proportion="1"/>
        <composition_activite id_compoact="compoact012" id_famille_kout="fam2" proportion=".3"/>
        <composition_activite id_compoact="compoact013" id_famille_kout="fam4" proportion=".24"/>
</activite>

<activite id_activite="act7" nom_activite="act7_telecom" unite_oeuvre="minute_telecom">
        <composition_activite id_compoact="compoact002" id_famille_kout="fam8" proportion="1"/>
</activite>
```

Les familles de coût.

Les familles de coût sont également décrites dans le chapitre1 dédié à ABC.
Chacune de ces huit familles sont constituées de postes pcmn qui interviennent dans des proportion parfois différentes.
Le fragment du document XML relatif aux familles de coût se lit de la même façon que celui ci-dessus traitant des activités.

La famille1 est constituée des postes pcmn « 615100 », « 615200 » et « 615220 » qui interviennent chacun à 100%.
Dans la famille2, nous trouvons le poste pcmn « 611300 » pour 40%, etc..

```
<famille_cout id_famille_cout="fam1" nom_famille="fam1_publicite">
        <composition_famille id_compofam="compofam01" id_pcmn_fam="615100" proportion_cf="1"/>
        <composition_famille id_compofam="compofam02" id_pcmn_fam="615200" proportion_cf="1"/>
        <composition_famille id_compofam="compofam03" id_pcmn_fam="615220" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam2" nom_famille="fam2_bureau">
        <composition_famille id_compofam="compofam04" id_pcmn_fam="611300" proportion_cf=".4"/>
        <composition_famille id_compofam="compofam05" id_pcmn_fam="612000" proportion_cf=".4"/>
        <composition_famille id_compofam="compofam06" id_pcmn_fam="612400" proportion_cf="1"/>
        <composition_famille id_compofam="compofam07" id_pcmn_fam="612500" proportion_cf="1"/>
        <composition_famille id_compofam="compofam08" id_pcmn_fam="616100" proportion_cf="1"/>
        <composition_famille id_compofam="compofam09" id_pcmn_fam="630210" proportion_cf=".4"/>
        <composition_famille id_compofam="compofam10" id_pcmn_fam="630231" proportion_cf=".8"/>
        <composition_famille id_compofam="compofam30" id_pcmn_fam="640200" proportion_cf=".4"/>
</famille_cout>

<famille_cout id_famille_cout="fam3" nom_famille="fam3_stock">
        <composition_famille id_compofam="compofam11" id_pcmn_fam="611300" proportion_cf=".6"/>
        <composition_famille id_compofam="compofam12" id_pcmn_fam="612000" proportion_cf=".6"/>
        <composition_famille id_compofam="compofam13" id_pcmn_fam="630210" proportion_cf=".6"/>
```

```
        <composition_famille id_compofam="compofam14" id_pcmn_fam="630220" proportion_cf=".2"/>
        <composition_famille id_compofam="compofam15" id_pcmn_fam="630231" proportion_cf=".2"/>
        <composition_famille id_compofam="compofam31" id_pcmn_fam="640200" proportion_cf=".6"/>
</famille_cout>

<famille_cout id_famille_cout="fam4" nom_famille="fam4_formation_documentation">
        <composition_famille id_compofam="compofam16" id_pcmn_fam="612300" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam5" nom_famille="fam5_honoraires">
        <composition_famille id_compofam="compofam17" id_pcmn_fam="613250" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam6" nom_famille="fam6_transport">
        <composition_famille id_compofam="compofam18" id_pcmn_fam="606140" proportion_cf="1"/>
        <composition_famille id_compofam="compofam19" id_pcmn_fam="611150" proportion_cf="1"/>
        <composition_famille id_compofam="compofam20" id_pcmn_fam="611350" proportion_cf="1"/>
        <composition_famille id_compofam="compofam21" id_pcmn_fam="612160" proportion_cf="1"/>
        <composition_famille id_compofam="compofam22" id_pcmn_fam="613530" proportion_cf="1"/>
        <composition_famille id_compofam="compofam23" id_pcmn_fam="630232" proportion_cf="1"/>
        <composition_famille id_compofam="compofam24" id_pcmn_fam="640100" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam7" nom_famille="fam7_outil_production">
        <composition_famille id_compofam="compofam25" id_pcmn_fam="601300" proportion_cf="1"/>
        <composition_famille id_compofam="compofam26" id_pcmn_fam="611130" proportion_cf="1"/>
        <composition_famille id_compofam="compofam27" id_pcmn_fam="613540" proportion_cf="1"/>
        <composition_famille id_compofam="compofam28" id_pcmn_fam="630220"proportion_cf=".8"/>
</famille_cout>

<famille_cout id_famille_cout="fam8" nom_famille="fam8_telecom">
        <composition_famille id_compofam="compofam29" id_pcmn_fam="616200" proportion_cf="1"/>
        <composition_famille id_compofam="compofam32" id_pcmn_fam="616300" proportion_cf="1"/>
        <composition_famille id_compofam="compofam33" id_pcmn_fam="616400" proportion_cf="1"/>
</famille_cout>
```

Les données d'exercice de l'entreprise.

Ces données concernent les clients, commandes, mesurages et charges directes et indirectes que nous imputons.

Les clients et commandes.

Nous avons ouvert trois éléments <client>.
Le premier élément <client> a pour attributs:

id_client: « jaeger01 » (conformément à la liste de choix de la DTD)
nom: « jaeger » (conformément à la liste de choix de la DTD)
adresse: « rue du bois 36 1000 bruxelles »

Ce client a passé une commande, que nous lisons dans l'élément <commande> qui est inclu (dans

l'élément <client> que nous venons de décrire).

```
<commande
id_commande: « com001 »
descriptif: « systeme solaire »
chiffre_affaire= « 12000 »>
```

La lecture est identique pour chaque élément <client>.
Le id de cette commande est « com001 », son objet consistera à vendre et placer un systeme solaire, le chiffre d'affaire en est de 12000 (Euro)

```
<client id_client="jaeger01" nom="jaeger" adresse="rue du bois 36 à 1000 bruxelles">
        <commande id_commande="com001" descriptif="systeme solaire" chiffre_affaire="12000">
<client id_client="cairelli02" nom="cairelli" adresse="grand-place 1 à 1200 Bruxelles">
        <commande id_commande="com002" descriptif="extension système" chiffre_affaire="2310">
<client id_client="panerai03" nom="panerai" adresse="av. des Arts 2 1200 Bruxelles">
        <commande id_commande="com004" descriptif="système solaire" chiffre_affaire="6400">
```

Les mesurages et charges directes.

Revenons au premier élément <client>.
Nous lui avons attribué deux commandes (deux éléments <commande> inclus dans les tags <client>).
Il a ensuite été procédé à des mesurages d'unités d'oeuvre des activités utilisées.
Dans le document XML, nous avons incorporé à l'élément <commande> dont l'attribut id_commande est « com001 », seize éléments <mesurage> pour seize mesurages réalisés pour cette commande.

La lecture du premier mesurage:

```
id_mesurage="mes001"
id_aktivite="act2"
date="01 feb 2009"
quantite="4"
unite_oeuvre="heure_demarchage"/>
```

Le mesurage1 se rapportant à l'activité2 a été réalisé le 01 février 2009; 4 unités d'oeuvre de type « heure_demarchage » ont été consommées.

```
<client id_client="jaeger01" nom="jaeger" adresse="rue du bois 36 à 1000 bruxelles">
 <commande id_commande="com001" descriptif="rediger exemple" chiffre_affaire="12000">
 <mesurage id_mesurage="mes001" id_aktivite="act2" date="01 feb 2009" quantite="4"
  unite_oeuvre="heure_demarchage"/>
 <mesurage id_mesurage="mes002" id_aktivite="act7" date="01 feb 2009" quantite="23"
  unite_oeuvre="minute_telecom"/>
 <mesurage id_mesurage="mes003" id_aktivite="act4" date="01 feb 2009" quantite="42"
  unite_oeuvre="kilometre"/>
 <mesurage id_mesurage="mes004" id_aktivite="act5" date="15 feb 2009" quantite="8"
  unite_oeuvre="heure_production"/>
```

```
<mesurage id_mesurage="mes005" id_aktivite="act4" date="01 feb 2009" quantite="40"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes006" id_aktivite="act3" date="13 feb 2009" quantite="24"
 unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes007" id_aktivite="act4" date="01 feb 2009" quantite="22"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes008" id_aktivite="act5" date="16 feb 2009" quantite="8.5"
 unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes009" id_aktivite="act4" date="01 feb 2009" quantite="38"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes010" id_aktivite="act5" date="15 feb 2009" quantite="6"
 unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes011" id_aktivite="act4" date="01 feb 2009" quantite="40"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes012" id_aktivite="act5" date="15 feb 2009" quantite="10"
 unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes013" id_aktivite="act3" date="13 feb 2009" quantite="11"
 unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes014" id_aktivite="act2" date="01 feb 2009" quantite="1.5"
 unite_oeuvre="heure_demarchage"/>
<mesurage id_mesurage="mes015" id_aktivite="act6" date="01 feb 2009" quantite="1.5"
 unite_oeuvre="heure_comptabilite_abc"/>
<mesurage id_mesurage="mes016" id_aktivite="act7" date="01 feb 2009" quantite="51"
 unite_oeuvre="minute_telecom"/>

<charge_directe id_charge_d="fact0020" id_pcmn="601000" montant_dir_htva_impute="4756"
                quantite="1"/>
<charge_directe id_charge_d="fact0023" id_pcmn="601000" montant_dir_htva_impute="322"
                quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25"
                quantite="32.5"/>
</commande>

<commande id_commande="com003" descriptif="seconde commande" chiffre_affaire="1852">
<mesurage id_mesurage="mes022" id_aktivite="act4" date="12 avril 2009" quantite="42"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes023" id_aktivite="act5" date="12 avril 2009" quantite="2"
 unite_oeuvre="heure_production"/>

<charge_directe id_charge_d="fact0041" id_pcmn="601000" montant_dir_htva_impute="640"
                quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25" quantite="2"/>
</commande>

</client>

<client id_client="cairelli02" nom="cairelli" adresse="grand-place 1 à 1200 Bruxelles">
<commande id_commande="com002" descriptif="production exemple" chiffre_affaire="2310">
<mesurage id_mesurage="mes017" id_aktivite="act4" date="25 mars 2009" quantite="12"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes018" id_aktivite="act5" date="25 mars 2009" quantite="8"
 unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes019" id_aktivite="act4" date="28 mars 2009" quantite="20"
 unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes020" id_aktivite="act5" date="28 mars 2009" quantite="7.25"
 unite_oeuvre="heure_production"/>
```

```
<mesurage id_mesurage="mes021" id_aktivite="act6" date="02 avril 2009" quantite=".5"
 unite_oeuvre="heure_comptabilite_abc"/>
<charge_directe id_charge_d="fact0005" id_pcmn="601000" montant_dir_htva_impute="801"
                quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25"
                quantite="15.25"/>
</commande>
</client>

<client id_client="panerai03" nom="panerai" adresse="av. des Arts 2 1200 Bruxelles">
<commande id_commande="com004" descriptif="système solaire" chiffre_affaire="6400">
<mesurage id_mesurage="mes024" id_aktivite="act2" date="06 mars 2009" quantite="2.5"
          unite_oeuvre="heure_communication"/>
<mesurage id_mesurage="mes025" id_aktivite="act4" date="06 mars 2009" quantite="61"
          unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes026" id_aktivite="act5" date="28 mars 2009" quantite="10"
          unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes027" id_aktivite="act3" date="13 feb 2009" quantite="14"
          unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes028" id_aktivite="act4" date="06 mars 2009" quantite="17"
          unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes029" id_aktivite="act3" date="13 feb 2009" quantite="04"
          unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes030" id_aktivite="act4" date="06 mars 2009" quantite="85"
          unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes031" id_aktivite="act5" date="28 mars 2009" quantite="7"
          unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes032" id_aktivite="act4" date="06 mars 2009" quantite="64"
          unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes033" id_aktivite="act5" date="28 mars 2009" quantite="9"
          unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes034" id_aktivite="act7" date="26 mars 2009" quantite="41"
          unite_oeuvre="minute_telecom"/>
<mesurage id_mesurage="mes035" id_aktivite="act6" date="26 mars 2009" quantite="1"
          unite_oeuvre="heure_comptabilite_abc"/>
<charge_directe id_charge_d="fact0018" id_pcmn="601000" montant_dir_htva_impute="3406"
                quantite="1"/>
<charge_directe id_charge_d="fact0030" id_pcmn="601000" montant_dir_htva_impute="604"
                quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25"
                quantite="26"/>
</commande>
</client>
```

Toujours dans le cadre de notre premier élément <client>, à la suite des éléments <mesurage>, nous créons trois éléments <charge_directe>

```
<client id_client="jaeger01" nom="jaeger" adresse="rue du bois 36 à 1000 bruxelles">
 <commande id_commande="com001" descriptif="rediger exemple">
 <mesurage id_mesurage="mes001" id_aktivite="act2" date="01 feb 2009" quantite="4"
.....
.....
 <mesurage id_mesurage="mes013" id_aktivite="act3" date="13 feb 2009" quantite="11"
```

```
            unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes014" id_aktivite="act2" date="01 feb 2009" quantite="1.5"
            unite_oeuvre="heure_demarchage"/>
<mesurage id_mesurage="mes015" id_aktivite="act6" date="01 feb 2009" quantite="1.5"
            unite_oeuvre="heure_comptabilite_abc"/>
<mesurage id_mesurage="mes016" id_aktivite="act7" date="01 feb 2009" quantite="51"
            unite_oeuvre="minute_telecom"/>
<charge_directe id_charge_d="fact0020" id_pcmn="601000" montant_dir_htva_impute="4756"
                  quantite="1"/>
<charge_directe id_charge_d="fact0023" id_pcmn="601000" montant_dir_htva_impute="322"
                  quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25"
                  quantite="32.5"/>
</commande>
```

La lecture du dernier élément « charge_directe » est: la charge directe dont le id est « fact9999 » a été imputée sous le numéro pcmn 620000; le montant imputé est de 25 (Euro), la quantité 32,5 (unités).

Trois charges directes sont imputées à cette commande
- des achats de fournitures pour un montant de 4756 (Euro), et 322 (Euro).
- des heures de production, dont la quantité de 32,5 heures correspond aux mesurages de « production » de la commande. Le montant de 25 (Euro) par heure est une valeur fournie par l'utilisateur.

Les charges indirectes.

Cette partie du document XML concerne les charges indirectes et la méthode selon laquelle nous les imputons.

Précisions.

Une même pièce comptable peut contenir des charges directes et indirectes, et les charges indirectes d'une même pièce comptable peuvent être incorporées dans différents postes pcmn.
L'attribut « id_chargeindirecte » prendra la même valeur que dans la comptabilité générale.
Autrement dit, si la pièce comptable porte dans la comptabilité générale le numéro 001, la même valeur sera appliquée pour l'attribut « id_chargeindirecte » : « fact001 ».

Nous ouvrons un élément <charge_indirecte> pour chaque pièce comptable comportant une charge indirecte.
Pour chaque imputation de cette pièce comptable, nous ouvrons un élément <imputation>, inclu dans l'élément <charge_indirecte>.

```
<charge_indirecte id_chargeindirecte="fact001" montant_htva_total="2000">
 <imputation id_imputation="imp0001" id_pcmn_imputation="601300" montant_htva_impute="850"/>
 <imputation id_imputation="imp0002" id_pcmn_imputation="611300" montant_htva_impute="1150"/>
</charge_indirecte>
```

Examinons le premier élément <charge_indirecte> de ce fragment du document XML:

```
id_chargeindirecte="fact001"
montant_htva_total="2000"
```

y sont inclus deux éléments <imputation>:

```
id_imputation="imp0001"
id_pcmn_imputation="601300"
montant_htva_impute="850 »
```

et

```
imputation id_imputation="imp0002"
id_pcmn_imputation="611300"
montant_htva_impute="1150"
```

La lecture en est: la facture fact001 dont le montant total hors TVA est de 2000 (Euro), est imputée en deux fois, dans le poste pcmn 601300 pour un montant hors TVA de 850 (Euro), et dans le poste pcmn 611300 pour un montant hors TVA de 1150 (Euro).

```
<charge_indirecte id_chargeindirecte="fact005" montant_htva_total="2401">
 <imputation id_imputation="imp0003" id_pcmn_imputation="615200" montant_htva_impute="1600"/>
</charge_indirecte>
```

La lecture de ce second élément: la facture fact005 dont le montant total hors TVA est de 2401(Euro) est imputée dans le poste pcmn 615200 pour un montant hors TVA de 1600 (Euro). Cette facture comportait une charge directe pour un montant hors TVA de 801 (Euro), que nous avons imputé dans la commande « com002 » en charge_directe.

La lecture de tous les éléments <charge_indirecte> dans le fragment de document XML ci-dessous est identique.

```
<charge_indirecte id_chargeindirecte="fact001" montant_htva_total="2000">
 <imputation id_imputation="imp0001" id_pcmn_imputation="601300" montant_htva_impute="850"/>
 <imputation id_imputation="imp0002" id_pcmn_imputation="611300" montant_htva_impute="1150"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact005" montant_htva_total="2401">
 <imputation id_imputation="imp0003" id_pcmn_imputation="615200" montant_htva_impute="1600"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact011" montant_htva_total="710">
 <imputation id_imputation="imp0004" id_pcmn_imputation="615100"montant_htva_impute="710.2"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact015" montant_htva_total="623.85">
 <imputation id_imputation="imp0005" id_pcmn_imputation="612400" montant_htva_impute="466"/>
 <imputation id_imputation="imp0006" id_pcmn_imputation="612300"
                montant_htva_impute="157.85"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact016" montant_htva_total="310">
 <imputation id_imputation="imp0007" id_pcmn_imputation="612160" montant_htva_impute="310"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact028" montant_htva_total="4562">
 <imputation id_imputation="imp0008" id_pcmn_imputation="630232" montant_htva_impute="4562"/>
</charge_indirecte>
```

```
<charge_indirecte id_chargeindirecte="fact031" montant_htva_total="2100">
 <imputation id_imputation="imp0009" id_pcmn_imputation="630220" montant_htva_impute="2100"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact032" montant_htva_total="2542">
 <imputation id_imputation="imp0010" id_pcmn_imputation="630210" montant_htva_impute="2542"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact054" montant_htva_total="600">
 <imputation id_imputation="imp0011" id_pcmn_imputation="613250" montant_htva_impute="600"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact055" montant_htva_total="240">
 <imputation id_imputation="imp0012" id_pcmn_imputation="612000" montant_htva_impute="240"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact056" montant_htva_total="211">
        <imputation id_imputation="imp0013" id_pcmn_imputation="612160" montant_htva_impute="211"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact062" montant_htva_total="118.7">
        <imputation id_imputation="imp0014" id_pcmn_imputation="616200"
          montant_htva_impute="118.7"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact071" montant_htva_total="560">
        <imputation id_imputation="imp0015" id_pcmn_imputation="612300" montant_htva_impute="560"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact072" montant_htva_total="422">
        <imputation id_imputation="imp0016" id_pcmn_imputation="611130" montant_htva_impute="422"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact075" montant_htva_total="58">
        <imputation id_imputation="imp0017" id_pcmn_imputation="616300" montant_htva_impute="58"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact100" montant_htva_total="74.6">
        <imputation id_imputation="imp0018" id_pcmn_imputation="616300" montant_htva_impute="74.6"/>
</charge_indirecte>
```

La dernière partie du document XML concerne l'élément <pcmn>
Nous avons créé ceux qui sont autorisés par la DTD.

```
<pcmn id_pcmn="601300" intitule="601300_outillage"/>
<pcmn id_pcmn="611350" intitule="611350_entretien_vehicule"/>
<pcmn id_pcmn="612160" intitule="612160_carburant"/>
<pcmn id_pcmn="612300" intitule="612300_documentation"/>
<pcmn id_pcmn="612400" intitule="612400_imprimes_fourn_bureau"/>
<pcmn id_pcmn="613250" intitule="613250_honoraire_comptable"/>
<pcmn id_pcmn="611130" intitule="611130_location_outillage"/>
<pcmn id_pcmn="611150" intitule="611150_location_vehicule"/>
<pcmn id_pcmn="611300" intitule="611300_entretien_locaux"/>
<pcmn id_pcmn="612500" intitule="612500_petit_mat_bureau"/>
<pcmn id_pcmn="613530" intitule="613530_assurance_vehicule"/>
<pcmn id_pcmn="613540" intitule="613540_assurance_RC"/>
<pcmn id_pcmn="615000" intitule="615000_frais_deplacement"/>
<pcmn id_pcmn="615100" intitule="615100_frais_representation"/>
<pcmn id_pcmn="615200" intitule="615200_publicite_annonces"/>
<pcmn id_pcmn="615220" intitule="615220_foires_expositions"/>
<pcmn id_pcmn="616100" intitule="616100_frais_postaux"/>
<pcmn id_pcmn="616200" intitule="616200_telephone"/>
<pcmn id_pcmn="616300" intitule="616300_gsm"/>
<pcmn id_pcmn="616400" intitule="616400_internet"/>
<pcmn id_pcmn="630210" intitule="630210_dot_amort_immeuble"/>
<pcmn id_pcmn="630220" intitule="630220_dot_amort_outil"/>
```

```
<pcmn id_pcmn="630231" intitule="630231_dot_amort_mobilier"/>
<pcmn id_pcmn="612000" intitule="612000_eau_gaz_elec"/>
<pcmn id_pcmn="630232" intitule="630232_dot_amort_matroulant"/>
<pcmn id_pcmn="640100" intitule="640100_dot_taxe_vehicule"/>
<pcmn id_pcmn="640200" intitule="640200_precompte_immobilier"/>
</ABC>
```

La lecture du dernier élément « pcmn » est: le poste dont le id_pcmn est 640200 est intitulé 640200 precompte immobilier. La lecture est identique pour les éléments précédents.

Enfin, le document XML est fermé par la balise clôturant l'élément racine: </ABC>

Le document ABC/XML entier.

Nous reproduisons ci-dessous l'entièreté du document XML.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE ABC SYSTEM "/home/jean/Memoireulb/XQ27avril/abc_avril27_id01.dtd">

<ABC>
<client id_client="jaeger01" nom="jaeger" adresse="rue du bois 36 à 1000 bruxelles">
<commande id_commande="com001" descriptif="systeme solaire" chiffre_affaire="12000">
<mesurage id_mesurage="mes001" id_aktivite="act2" date="01 feb 2009" quantite="4"
            unite_oeuvre="heure_demarchage"/>
<mesurage id_mesurage="mes002" id_aktivite="act7" date="01 feb 2009" quantite="23"
            unite_oeuvre="minute_telecom"/>
<mesurage id_mesurage="mes003" id_aktivite="act4" date="01 feb 2009" quantite="42" unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes004" id_aktivite="act5" date="15 feb 2009" quantite="8"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes005" id_aktivite="act4" date="01 feb 2009" quantite="40" unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes006" id_aktivite="act3" date="13 feb 2009" quantite="24"
            unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes007" id_aktivite="act4" date="01 feb 2009" quantite="22" unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes008" id_aktivite="act5" date="16 feb 2009" quantite="8.5"
           unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes009" id_aktivite="act4" date="01 feb 2009" quantite="38" unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes010" id_aktivite="act5" date="15 feb 2009" quantite="6"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes011" id_aktivite="act4" date="01 feb 2009" quantite="40" unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes012" id_aktivite="act5" date="15 feb 2009" quantite="10"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes013" id_aktivite="act3" date="13 feb 2009" quantite="11"
            unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes014" id_aktivite="act2" date="01 feb 2009" quantite="1.5"
            unite_oeuvre="heure_demarchage"/>
<mesurage id_mesurage="mes015" id_aktivite="act6" date="01 feb 2009" quantite="1.5"
            unite_oeuvre="heure_comptabilite_abc"/>
<mesurage id_mesurage="mes016" id_aktivite="act7" date="01 feb 2009" quantite="51"
            unite_oeuvre="minute_telecom"/>
<charge_directe id_charge_d="fact0020" id_pcmn="601000" montant_dir_htva_impute="4756" quantite="1"/>
<charge_directe id_charge_d="fact0023" id_pcmn="601000" montant_dir_htva_impute="322"  quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25" quantite="32.5"/>
</commande>

<commande id_commande="com003" descriptif="seconde commande" chiffre_affaire="1852">
<mesurage id_mesurage="mes022" id_aktivite="act4" date="12 avril 2009" quantite="42"
            unite_oeuvre="kilometre"/>
```

```
<mesurage id_mesurage="mes023" id_aktivite="act5" date="12 avril 2009" quantite="2"
            unite_oeuvre="heure_production"/>
<charge_directe id_charge_d="fact0041" id_pcmn="601000" montant_dir_htva_impute="640" quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25" quantite="2"/>
</commande>
</client>

<client id_client="cairelli02" nom="cairelli" adresse="grand-place 1 à 1200 Bruxelles">
<commande id_commande="com002" descriptif="extension système" chiffre_affaire="2310">
<mesurage id_mesurage="mes017" id_aktivite="act4" date="25 mars 2009" quantite="12"
            unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes018" id_aktivite="act5" date="25 mars 2009" quantite="8"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes019" id_aktivite="act4" date="28 mars 2009" quantite="20"
            unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes020" id_aktivite="act5" date="28 mars 2009" quantite="7.25"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes021" id_aktivite="act6" date="02 avril 2009" quantite=".5"
            unite_oeuvre="heure_comptabilite_abc"/>
<charge_directe id_charge_d="fact0005" id_pcmn="601000" montant_dir_htva_impute="801"  quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25" quantite="15.25"/>
</commande>
</client>

<client id_client="panerai03" nom="panerai" adresse="av. des Arts 2 1200 Bruxelles">
<commande id_commande="com004" descriptif="système solaire" chiffre_affaire="6400">
<mesurage id_mesurage="mes024" id_aktivite="act2" date="06 mars 2009" quantite="2.5"
            unite_oeuvre="heure_communication"/>
<mesurage id_mesurage="mes025" id_aktivite="act4" date="06 mars 2009" quantite="61"
            unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes026" id_aktivite="act5" date="28 mars 2009" quantite="10"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes027" id_aktivite="act3" date="13 feb 2009" quantite="14"
            unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes028" id_aktivite="act4" date="06 mars 2009" quantite="17"
            unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes029" id_aktivite="act3" date="13 feb 2009" quantite="04"
           unite_oeuvre="article_unitaire"/>
<mesurage id_mesurage="mes030" id_aktivite="act4" date="06 mars 2009" quantite="85"
           unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes031" id_aktivite="act5" date="28 mars 2009" quantite="7"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes032" id_aktivite="act4" date="06 mars 2009" quantite="64"
            unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes033" id_aktivite="act5" date="28 mars 2009" quantite="9"
            unite_oeuvre="heure_production"/>
<mesurage id_mesurage="mes034" id_aktivite="act7" date="26 mars 2009" quantite="41"
            unite_oeuvre="minute_telecom"/>
<mesurage id_mesurage="mes035" id_aktivite="act6" date="26 mars 2009" quantite="1"
            unite_oeuvre="heure_comptabilite_abc"/>
<charge_directe id_charge_d="fact0018" id_pcmn="601000" montant_dir_htva_impute="3406"  quantite="1"/>
<charge_directe id_charge_d="fact0030" id_pcmn="601000" montant_dir_htva_impute="604"  quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25" quantite="26"/>
</commande>
</client>

</client>

<activite id_activite="act1" nom_activite="act1_communication" unite_oeuvre="heure_communication">
        <composition_activite id_compoact="compoact001" id_famille_kout="fam1" proportion="1"/>
```

```
        <composition_activite id_compoact="compoact002" id_famille_kout="fam2" proportion=".3"/>
</activite>

<activite id_activite="act2" nom_activite="act2_demarchage" unite_oeuvre="heure_demarchage">
        <composition_activite id_compoact="compoact003" id_famille_kout="fam2" proportion=".4"/>
        <composition_activite id_compoact="compoact004" id_famille_kout="fam4" proportion=".42"/>
</activite>

<activite id_activite="act3" nom_activite="act3_logistique" unite_oeuvre="article_unitaire">
        <composition_activite id_compoact="compoact005" id_famille_kout="fam3" proportion="1"/>
        <composition_activite id_compoact="compoact006" id_famille_kout="fam7" proportion=".27"/>
</activite>

<activite id_activite="act4" nom_activite="act4_transport" unite_oeuvre="kilometre">
        <composition_activite id_compoact="compoact008" id_famille_kout="fam6" proportion="1"/>
</activite>

<activite id_activite="act5" nom_activite="act5_production" unite_oeuvre="heure_production">
        <composition_activite id_compoact="compoact009" id_famille_kout="fam4" proportion=".34"/>
        <composition_activite id_compoact="compoact010" id_famille_kout="fam7" proportion=".73"/>
</activite>

<activite id_activite="act6" nom_activite="act6_comptabilite_abc" unite_oeuvre="heure_comptabilite_abc">
        <composition_activite id_compoact="compoact011" id_famille_kout="fam5" proportion="1"/>
        <composition_activite id_compoact="compoact012" id_famille_kout="fam2" proportion=".3"/>
        <composition_activite id_compoact="compoact013" id_famille_kout="fam4" proportion=".24"/>
</activite>

<activite id_activite="act7" nom_activite="act7_telecom" unite_oeuvre="minute_telecom">
        <composition_activite id_compoact="compoact002" id_famille_kout="fam8" proportion="1"/>
</activite>

<famille_cout id_famille_cout="fam1" nom_famille="fam1_publicite">
        <composition_famille id_compofam="compofam01" id_pcmn_fam="615100" proportion_cf="1"/>
        <composition_famille id_compofam="compofam02" id_pcmn_fam="615200" proportion_cf="1"/>
        <composition_famille id_compofam="compofam03" id_pcmn_fam="615220" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam2" nom_famille="fam2_bureau">
        <composition_famille id_compofam="compofam04" id_pcmn_fam="611300" proportion_cf=".4"/>
        <composition_famille id_compofam="compofam05" id_pcmn_fam="612000" proportion_cf=".4"/>
        <composition_famille id_compofam="compofam06" id_pcmn_fam="612400" proportion_cf="1"/>
        <composition_famille id_compofam="compofam07" id_pcmn_fam="612500" proportion_cf="1"/>
        <composition_famille id_compofam="compofam08" id_pcmn_fam="616100" proportion_cf="1"/>
        <composition_famille id_compofam="compofam09" id_pcmn_fam="630210" proportion_cf=".4"/>
        <composition_famille id_compofam="compofam10" id_pcmn_fam="630231" proportion_cf=".8"/>
        <composition_famille id_compofam="compofam30" id_pcmn_fam="640200" proportion_cf=".4"/>
</famille_cout>

<famille_cout id_famille_cout="fam3" nom_famille="fam3_stock">
        <composition_famille id_compofam="compofam11" id_pcmn_fam="611300" proportion_cf=".6"/>
        <composition_famille id_compofam="compofam12" id_pcmn_fam="612000" proportion_cf=".6"/>
        <composition_famille id_compofam="compofam13" id_pcmn_fam="630210" proportion_cf=".6"/>
        <composition_famille id_compofam="compofam14" id_pcmn_fam="630220" proportion_cf=".2"/>
        <composition_famille id_compofam="compofam15" id_pcmn_fam="630231" proportion_cf=".2"/>
        <composition_famille id_compofam="compofam31" id_pcmn_fam="640200" proportion_cf=".6"/>
</famille_cout>

<famille_cout id_famille_cout="fam4" nom_famille="fam4_formation_documentation">
        <composition_famille id_compofam="compofam16" id_pcmn_fam="612300" proportion_cf="1"/>
```

```
</famille_cout>

<famille_cout id_famille_cout="fam5" nom_famille="fam5_honoraires">
	<composition_famille id_compofam="compofam17" id_pcmn_fam="613250" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam6" nom_famille="fam6_transport">
	<composition_famille id_compofam="compofam18" id_pcmn_fam="606140" proportion_cf="1"/>
	<composition_famille id_compofam="compofam19" id_pcmn_fam="611150" proportion_cf="1"/>
	<composition_famille id_compofam="compofam20" id_pcmn_fam="611350" proportion_cf="1"/>
	<composition_famille id_compofam="compofam21" id_pcmn_fam="612160" proportion_cf="1"/>
	<composition_famille id_compofam="compofam22" id_pcmn_fam="613530" proportion_cf="1"/>
	<composition_famille id_compofam="compofam23" id_pcmn_fam="630232" proportion_cf="1"/>
	<composition_famille id_compofam="compofam24" id_pcmn_fam="640100" proportion_cf="1"/>
</famille_cout>

<famille_cout id_famille_cout="fam7" nom_famille="fam7_outil_production">
	<composition_famille id_compofam="compofam25" id_pcmn_fam="601300" proportion_cf="1"/>
	<composition_famille id_compofam="compofam26" id_pcmn_fam="611130" proportion_cf="1"/>
	<composition_famille id_compofam="compofam27" id_pcmn_fam="613540" proportion_cf="1"/>
	<composition_famille id_compofam="compofam28" id_pcmn_fam="630220" proportion_cf=".8"/>
</famille_cout>
<famille_cout id_famille_cout="fam8" nom_famille="fam8_telecom">
	<composition_famille id_compofam="compofam29" id_pcmn_fam="616200" proportion_cf="1"/>
	<composition_famille id_compofam="compofam32" id_pcmn_fam="616300" proportion_cf="1"/>
	<composition_famille id_compofam="compofam33" id_pcmn_fam="616400" proportion_cf="1"/>
</famille_cout>

<charge_indirecte id_chargeindirecte="fact001" montant_htva_total="2000">
	<imputation id_imputation="imp0001" id_pcmn_imputation="601300" montant_htva_impute="850"/>
	<imputation id_imputation="imp0002" id_pcmn_imputation="611300" montant_htva_impute="1150"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact005" montant_htva_total="2401">
	<imputation id_imputation="imp0003" id_pcmn_imputation="615200" montant_htva_impute="1600"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact011" montant_htva_total="710">
	<imputation id_imputation="imp0004" id_pcmn_imputation="615100" montant_htva_impute="710.2"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact015" montant_htva_total="623.85">
	<imputation id_imputation="imp0005" id_pcmn_imputation="612400" montant_htva_impute="466"/>
	<imputation id_imputation="imp0006" id_pcmn_imputation="612300" montant_htva_impute="157.85"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact016" montant_htva_total="310">
	<imputation id_imputation="imp0007" id_pcmn_imputation="612160" montant_htva_impute="310"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact028" montant_htva_total="4562">
	<imputation id_imputation="imp0008" id_pcmn_imputation="630232" montant_htva_impute="4562"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact031" montant_htva_total="2100">
	<imputation id_imputation="imp0009" id_pcmn_imputation="630220" montant_htva_impute="2100"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact032" montant_htva_total="2542">
	<imputation id_imputation="imp0010" id_pcmn_imputation="630210" montant_htva_impute="2542"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact054" montant_htva_total="600">
	<imputation id_imputation="imp0011" id_pcmn_imputation="613250" montant_htva_impute="600"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact055" montant_htva_total="240">
	<imputation id_imputation="imp0012" id_pcmn_imputation="612000" montant_htva_impute="240"/>
</charge_indirecte>
```

```
<charge_indirecte id_chargeindirecte="fact056" montant_htva_total="211">
        <imputation id_imputation="imp0013" id_pcmn_imputation="612160" montant_htva_impute="211"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact062" montant_htva_total="118.7">
        <imputation id_imputation="imp0014" id_pcmn_imputation="616200" montant_htva_impute="118.7"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact071" montant_htva_total="560">
        <imputation id_imputation="imp0015" id_pcmn_imputation="612300" montant_htva_impute="560"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact072" montant_htva_total="422">
        <imputation id_imputation="imp0016" id_pcmn_imputation="611130" montant_htva_impute="422"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact075" montant_htva_total="58">
        <imputation id_imputation="imp0017" id_pcmn_imputation="616300" montant_htva_impute="58"/>
</charge_indirecte>
<charge_indirecte id_chargeindirecte="fact100" montant_htva_total="74.6">
        <imputation id_imputation="imp0018" id_pcmn_imputation="616300" montant_htva_impute="74.6"/>
</charge_indirecte>
<pcmn id_pcmn="601000" intitule="601000_achat_fournitures"/>
<pcmn id_pcmn="601300" intitule="601300_outillage"/>
<pcmn id_pcmn="611350" intitule="611350_entretien_vehicule"/>
<pcmn id_pcmn="612160" intitule="612160_carburant"/>
<pcmn id_pcmn="612300" intitule="612300_documentation"/>
<pcmn id_pcmn="612400" intitule="612400_imprimes_fourn_bureau"/>
<pcmn id_pcmn="613250" intitule="613250_honoraire_comptable"/>
<pcmn id_pcmn="611130" intitule="611130_location_outillage"/>
<pcmn id_pcmn="611150" intitule="611150_location_vehicule"/>
<pcmn id_pcmn="611300" intitule="611300_entretien_locaux"/>
<pcmn id_pcmn="612500" intitule="612500_petit_mat_bureau"/>
<pcmn id_pcmn="613530" intitule="613530_assurance_vehicule"/>
<pcmn id_pcmn="613540" intitule="613540_assurance_RC"/>
<pcmn id_pcmn="615000" intitule="615000_frais_deplacement"/>
<pcmn id_pcmn="615100" intitule="615100_frais_representation"/>
<pcmn id_pcmn="615200" intitule="615200_publicite_annonces"/>
<pcmn id_pcmn="615220" intitule="615220_foires_expositions"/>
<pcmn id_pcmn="616100" intitule="616100_frais_postaux"/>
<pcmn id_pcmn="616200" intitule="616200_telephone"/>
<pcmn id_pcmn="616300" intitule="616300_gsm"/>
<pcmn id_pcmn="616400" intitule="616400_internet"/>
<pcmn id_pcmn="620000" intitule="620000_remuneration"/>
<pcmn id_pcmn="630210" intitule="630210_dot_amort_immeuble"/>
<pcmn id_pcmn="630220" intitule="630220_dot_amort_outil"/>
<pcmn id_pcmn="630231" intitule="630231_dot_amort_mobilier"/>
<pcmn id_pcmn="612000" intitule="612000_eau_gaz_elec"/>
<pcmn id_pcmn="630232" intitule="630232_dot_amort_matroulant"/>
<pcmn id_pcmn="640100" intitule="640100_dot_taxe_vehicule"/>
<pcmn id_pcmn="640200" intitule="640200_precompte_immobilier"/>
</ABC>
```

Chapitre 6. XQuery, outil de requête.

Avant de passer en la partie du mémoire consacrée aux requêtes portant sur le document XML, nous présentons ici XQuery, outil de requête qui sera utilisé.
Nous en brossons un bref historique et tentons d'en décrire le fonctionnement, une compréhension du modèle interne pouvant aider à son utilisation.
Nb.: nous utiliserons des termes techniques en langue anglaise, les ouvrages lus à propos de XQuery étant rédigés en langue anglaise, nous préférons utiliser le vocabulaire existant que de tenter une traduction maladroite.

Historique.

Le format XML ayant été largement adopté, une profusion de documents XML, bases de données XML sont apparus. Il est devenu nécessaire de disposer d'un outil permettant d'extraire des données de ces fichiers XML, présenter des résultats, effectuer des recherches au sein de bases de données XML, etc.

Le W3C est à l'origine de cette démarche qui allait devenir XQuery, lorsqu'en décembre 1998, un groupe de travail fut créé, pour devenir officiel en septembre 1999 où il prit l'appellation de « working group for XML Query », généralement raccourci par « query working group ».

Une des premières tâches du « query working group » fut de déterminer si les moteurs de recherche existants, comme SQL parvenu à maturité et devenu un standard largement utilisé dans le monde des données relationnelles, ne pouvait être utilisé moyennant des extensions pour des données au format XML.

Des différences sont apparues au fil de cette étude.

Citons les différences entre la structure des données: deux dimensions pour les données relationnelles (tableau lignes colonnes), quand les données XML « taguées » ont une profondeur non prévisible.

Ou bien la différence entre l'homogénéité des données relationnelles qui permet les metadata pour les décrire, opposées à l'hétérogénéité des données XML dont les metadata se retrouvent dans tout le document.

Ou encore, la notion d'ordre des données qu'il est nécessaire de respecter en XML, ce qui n'est pas le cas en base de données relationnelles.

Il en est d'autres, notre liste n'est pas exhaustive.

Cela a mené le « query working group » à la décision de définir un design propre pour XQuery.

Quelques lignes conductrices de cette tâche ont été:

- XQuery sera une transformation du Query Data Model
- Compatibilité avec Xpath
- Conformité au schéma XML

- Souplesse d'application, tant à des documents validés par un XML Schema, que par une DTD ou bien non validés.
- ...

L'influence de Xpath.

Xpath, une recommandation du W3C datant de novembre 1999, était déjà largement utilisé par la communauté XML.
Xpath est un outil de recherche simple pour documents XML, permettant la navigation dans le document.
L'influence de Xpath se trouve dans l'intégration des « path » au sein de XQuery, un parcours de l'arbre qui retourne les « nodes », leur position et position relative, leur type et contenu.
Une expression Xpath assure la sélection de « nodes » au sein du document et est à même de les filtrer par le biais de prédicats.
La notion d'expression et prédicat a également été retenue.

L'influence de XML Schema.

XML SCHEMA est un langage qui crée des schémas de validation pour documents XML.
Il comporte de nombreux types (de données), permet d'en créer d'autres.
Les données d'un document XML et les types issus du schéma qui leur sont appliqués sont le « Post-Schema Validation Infoset » ou « PSVI ».

Le design de XQuery a été conçu pour une compatibilité avec le typage défini dans XML Schema.

Autres sources.

Plusieurs « query languages » existants ont influencé XQuery, des membres du « query work group » ayant auparavant contribué au développement de ces langages.
Le plus proche parent de XQuery est QUILT , une proposition de membres du « query work group ».
La proposition du langage QUILT comportait un des composants essentiels de XQuery: FLWOR (pour For-Let-Where-Order-Return), un équivalent du (select-from-where) du monde SQL.

QUILT retenait également de XML-QL la notion de « constructor » , où « where » génère un ensemble de tuples et variables liées; « construct » s'exécute pour chacun de ces tuples , ce qui génère des éléments de sortie.

XQuery recommandation W3C.

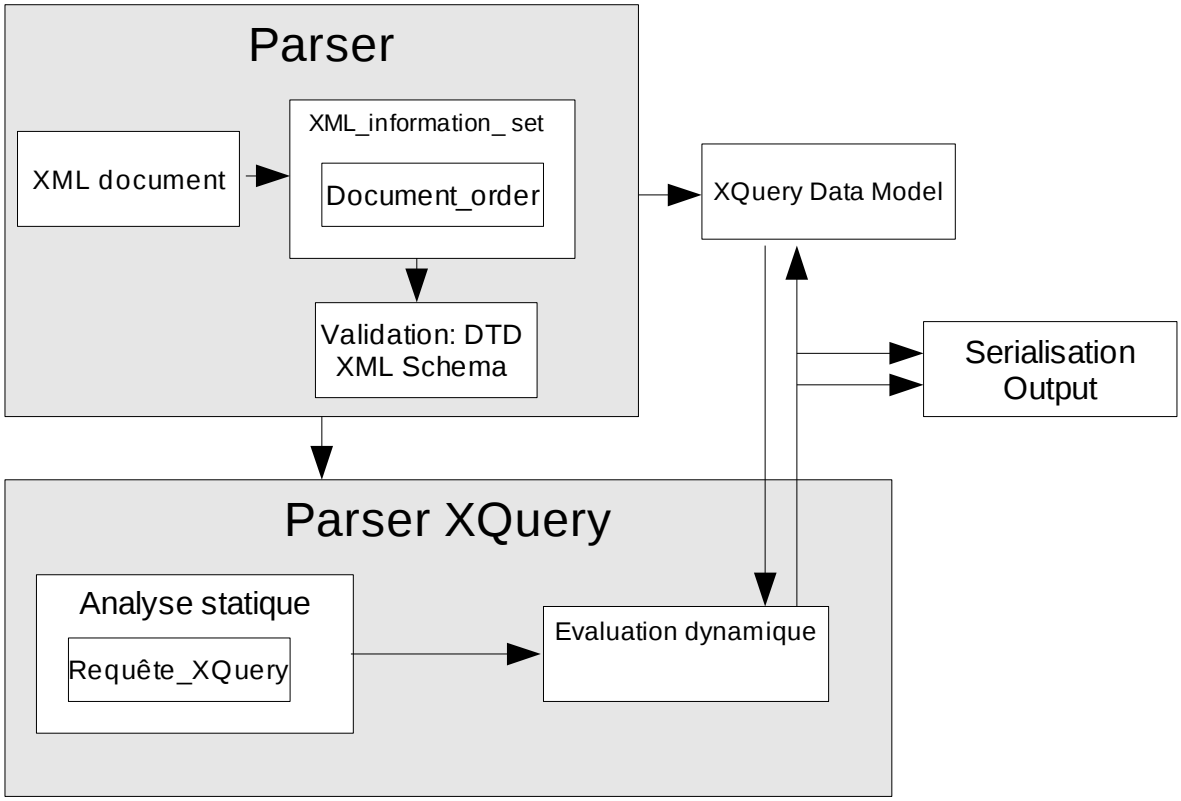
Le 23 janvier 2007, soit après neuf années, XQuery est devenu une recommandation du W3C .

Fonctionnement de XQuery.

Notre décrivons de façon très simplifiée le processus de traitement de XQuery.

Le W3C divise le processus que nous résumons en deux parties:
- l'une assimilée à une démarche « externe » à XQuery, traitant le document XML
- la seconde « interne » et coeur de XQuery traitant la requête XQuery.

Notons que ces deux « parties » se produisent au sein du même processus XQuery, et ne sont pas perceptibles pour l'utilisateur.



1. Traitement « externe ».

Un input, le document XML sur lequel porte la requête XQuery, est traité par un parser.

Lors du parsing est créé le « XML Information Set » modélisant le document sous forme d'une hiérarchie de « nodes ».
Les « nodes » sont identifiés et notés dans un « document order » complété à chaque « node » rencontré.
La finalité du « document order » est de retenir l'ordre du document XML afin que les requêtes puissent s'y référer, et de restituer cet ordre en sortie de traitement.

Le document est ensuite validé par comparaison à sa DTD ou son Schema XML.

Le « XML Information Set » est alors transformé en « XQuery Data Model ».

2. Traitement « interne ».

XQuery procède ensuite à deux phases.
Un input qui est la requête XQUERY, est parsé durant la « phase d'analyse statique ».
Si l' « analyse statique » valide la requête, intervient la « phase d'évaluation dynamique », au cours de laquelle les valeurs des expressions sont traitées .

Enfin intervient la sérialisation (la seconde phase), qui est la transformation du data model en flux d'octets, autrement exprimé, le résultat de la requête en XML.

Le processeur XQuery.

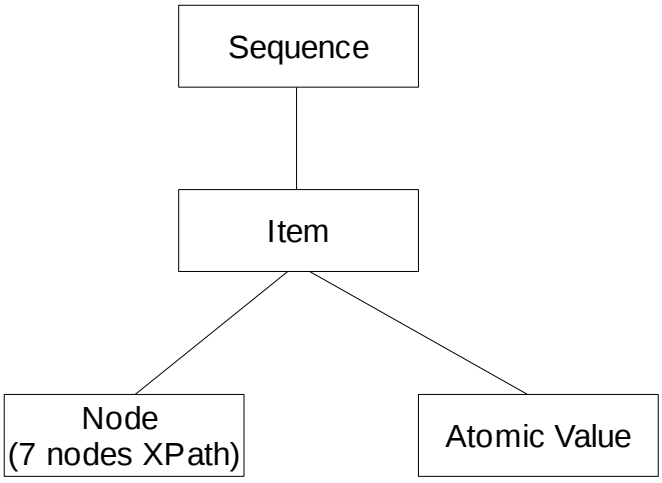
XQuery a un processeur, solution logicielle qui effectue l' « analyse syntaxique », l' « évaluation dynamique » et l'exécution de la requête.
Ce processeur se charge également de « parser » le document XML, en créer le « XML Infoset » et le « XQuery data model ».
Le processeur retourne un résultat soit au format XML (la sérialisation) qui peut être soit affiché, ou bien passé à une autre application pour un traitement ultérieur.

Le datamodel.

Le data model est une représentation formelle des données d'un document XML qui fera l'objet d'un traitement XQuery.
XQuery est défini comme une transformation d'une instance du query data model en une autre instance du data model.
Cette représentation défini chaque donnée, tant en position au sein du document qu'en termes de type.
Cette définition est utilisée en input de la requête, en output (le résultat de la requête) ainsi qu'en résultat intermédiaire.

Le data model de XQuery est « XQuery 1.0 et Xpath 2.0 » ou « XDM ».
Il est partiellement issu du data model de Xpath qui a été enrichi par des informations issues du Post-Schema Validation Infoset (PSVI de XML SCHEMA).
« XDM » est utilisé simultanément par XQuery 1.0 et Xpath 2.0 .

Le « XDM » organise et représente les données au travers de: « Sequence », « Item », « Atomic Value » et « Nodes ».



- Une « sequence » est un ensemble ordonnancé de zéro, un ou plusieurs items.
- Un « Item » est un « node » ou bien une « Atomic value ».
- Une « Atomic value » est une valeur simple, non marquée, ne contenant pas d'autre valeur, elle correspond à un type provenant des « atomic types » définis par le XML Schema.
- Un « node » est issu d'un des 7 « nodes » de Xpath (Element nodes, Attributes, Documents, Text, Comments nodes, Namespace, Processing Instruction nodes).

Dans ce mémoire, nous considérons essentiellement Les « Element nodes » et « Attributes nodes ».

Une « sequence » sera par exemple constituée de trois « item » qui sont des « nodes » de type « element ».

XQuery est un langage fort typé.
Dès lors, chaque « item » et/ou « node » doit correspondre à des types définis par le XDM ou XML Schema.
En admettant une certaine souplesse toutefois: des « items » non typés (par exemple lorsqu'il n'existe pas de document de validation) sont convertis automatiquement dans le type nécessaire à l'opération souhaitée.

« XDM » représentera le document XML hiérarchiquement.
Son élément racine est « Document node », qui sera suivi hiérarchiquement par, ou bien sera parent de, l'élément « node » racine du document XML, lui-même parent des éléments du document traité.

Les Expressions XQuery.

L'outil essentiel de XQuery pour l'évaluation est l' expression.
Plusieurs catégories d'expressions existent.

Nous ne les citons pas toutes, simplement celles qui seront utilisées pour la suite de ce travail.
- Path: une expression path permet de naviguer dans le document sur lequel porte la requête.
- Comparaisons: =,<,>, !, etc permet de comparer les valeurs, nodes.
- Conditions: if, then, else
- « Constructor »: permet d'exprimer le résultat de la requête sous forme de document XML ou XHtml, donc de baliser le résultat au moyen de tags < >, de créer des éléments et attributs. Le « Constructor » permet également de construire des valeurs atomiques.
- Arithmetic: qui permet d'additionner, soustraire, diviser, multiplier.
- FLWOR: qui est l'équivalent du SQL select-from-where, For-Let-Where-Order-Return.

Nous revenons plus en détail sur deux des expressions que nous utilisons: Path et FLWOR.

Expression Path.

Une « expression path » permet de naviguer dans le document sur lequel porte la requête.
La fonction d'appel « doc » retourne le « document node » du document XML à transformer.

```
doc ("/home/Memoireulb/abc.xml") //ABC/client/commande
```

Le « document node » de abc.xml est retourné à l'appel de la fonction « doc », « premier path » de « doc ».
La suite du « path » est évaluée par rapport à ce premier « document node » qui est un « context

node ».
Le « second step » vers « ABC »: « ABC » devient le « context node » par rapport auquel est évaluée la séquence « client ».
Le « troisième step » vers « client »: « client » devient le « context node » par rapport auquel est évaluée la séquence « commande ».

XQuery fait usage de « predicat ».
Les « predicats ». sont intégrés dans les « path expressions » afin de filtrer les « nodes » utiles.
Par exemple [@nom_activite="act3_logistique"] sélectionnera les « nodes activite » dont l'attribut « nom_activite » sera égal à « act3_logistique ».

Les requêtes FLWOR.

XQuery utilise des variables, qui sont précédées du signe $. Ex: $var.
Une variable reçoit une valeur précise lors de l'évaluation XQuery.

FOR: associe une valeur à une variable de façon itérative. FOR suivi d'un PATH comme dans l'exemple suivant:

```
for $in in doc ("/home/Memoireulb/abc.xml")//ABC/activite
```

associe à la variable $in de façon itérative une « activite » de la sequence.
La requête FLWOR est exécutée pour chaque itération de la séquence.

LET: associe la séquence entière à une variable plutôt que chacun de ses item un à un comme FOR.

Dans l'exemple let $in in doc ("/home/Memoireulb/abc.xml")//ABC/activite $in sera associé à la valeur de la séquence activité totale.

WHERE: on y défini les filtres, prédicats, conditions et jointures entre entités.
WHERE sélectionnera dans les items ceux qui correspondront au(x) prédicat(s) qu'il décrit.

ORDER BY: permet d'ordonnancer la valeur des résultats attendus (ordre croissant, décroissant, alphabétique).

RETURN suivi de l'expression qui est le résultat attendu retourne ce résultat.
Un item est retourné à chaque itération de la boucle FOR, ce qui créée une séquence d'items.

RETURN n'interrompt pas le déroulement de la boucle FOR.
Il est également tout à fait possible d'incorporer dans les sorties RETURN de nouvelles boucles conditionnelles.

Sources.

- W3C :http://www.w3.org/TR/2006/PR-xquery-20061121/
- XQuery from the Experts: A Guide to the W3C XML Query Language ISBN: 0-321-18060-7
- XQuery de Priscilia Walmsley ISBN: 0-596-00634-9

Chapitre 7. Requêtes XQuery.

Dans cette partie consacrée aux requêtes XQuery, qui devront fournir des résultats ABC, nous nous proposons de présenter notre travail de façon graduelle.
Nous débuterons par une requête élémentaire et tenterons de terminer en fin de chapitre par une requête fournissant un résultat ABC.
Nous utiliserons des requêtes de type FLWOR.

Produire une liste de clients.

Une première requête simple consiste à produire la liste des clients.

```
for $client in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC/client
return
<p>{($client/@nom)}</p>
```

Dans la première ligne, nous déclarons la variable « client » en la faisant précéder par le signe $, soit « $client ».
La variable « $client » est précédée de la clause FOR.
FOR lance une itération qui permet à la requête FLWOR de multiples évaluations (récursives).
La fonction « doc » ouvre le document sur lequel portera la requête.

```
("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC/client
```

Pour accéder au document, il est fait usage d'une « path expression ».
Une « path expression » permet de sélectionner les éléments ou attributs nécessaires.
Le point de départ en est l'élément root du file system « / », puis le répertoire « home », puis le répertoire « jean », puis le répertoire « Mémoire_ulb_2009 », puis le répertoire « group_21juin » et enfin le fichier « abc_avril27jn_id01.xml », puis la racine « / ABC » du fichier, puis la séquence « client ».

Nous lions la variable $client au premier élément de la séquence « client » dont le « context node » est « ABC ».
La boucle induite par FOR évaluera chacune des valeurs prises par chacun des éléments de la séquence « client ».

La clause « return » envoie un élément d'information à chaque itération de la requête.
Le résultat demandé est la valeur prise par l'attribut « nom » de l'élément « client ».
A chaque évaluation, « return » enverra le nom d'un client.
Les évaluations itératives de « for » cesseront lorsqu'aura été évalué le dernier élément « client ».

```
<p>{($client/@nom)}</p>
```

Nous utilisons un « constructor » XML, afin de créer un élément <p/>, et donc un résultat de requête sous forme XML.

Le résultat en est le suivant:

```
<?xml version="1.0" encoding="UTF-8"?>
<p nom="jaeger"/>
<p nom="cairelli"/>
<p nom="communication"/>
```

Chaque instanciation du résultat est un élément <p/>.

Nous pouvons modifier cette requête de façon à obtenir un arbre XML, comportant donc un élément racine, et des éléments <p> ne contenant que la valeur de l'attribut « nom ».
Nous choisissons de nommer l'élément racine <requete/>.
L'entièreté de la requête est incorporée dans les balises <requete> </requete>

```
<requete>
{
for $client in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC/client
return
<p>{data($client/@nom)}</p>
}
</requete>
```

Afin de ne conserver que la valeur atomique de l'attribut, nous faisons appel à la fonction « data » qui permet d'extraire les valeurs atomiques d'éléments et attributs.

```
<p>{data($client/@nom)}</p>
```

Le résultat est un arbre XML comportant l'élément racine « requête » parent de trois éléments <p/>.
Notons que seuls les noms des trois clients apparaissent.

```
<?xml version="1.0" encoding="UTF-8"?>
<requete>
  <p>jaeger</p>
  <p>cairelli</p>
  <p>communication</p>
</requete>
```

Nous utiliserons les « constructors » XML pour toutes les requêtes suivantes.

Les liens.

Nous devons pouvoir établir des liens entre éléments du document XML, liens équivalents à ceux définis dans les représentations par modèle hiérarchique ou ERD.

But de la requête: afficher pour chaque « activite » les « familles_cout » qui la composent.

```
<requete_lien>
{
for $in in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC
```

```
for $akty in $in/activite
return
<activite>
<nom_activite>{data($akty/@nom_activite)}</nom_activite>
{
for $famy in $in/famille_cout[@id_famille_cout=$akty/composition_activite/@id_famille_kout]
return
<famille_cout> {data($famy/@nom_famille)}</famille_cout>
}
</activite>
}
</requete_lien>
```

Un lien est à établir entre les « éléments » « activite » et « famille », par comparaison des attributs communs à « activite » et « famille ».

Pour l'élément « activite », il s'agit de l'attribut « id_famille_kout » (dans le « path » « ABC/ activite/composition _activite »).
Pour l'élément « famille cout », il s'agit de l'attribut « id_famille_cout » (dans le « path » « ABC/ famille_cout ».

Les valeurs que prennent ces attributs sont: « fam »1, « fam2 », « fam3 », « fam4 », « fam5 », « fam6 », « fam7 » et « fam8 ».

L'évaluation s'opérera en comparant un attribut « fam1 » trouvé dans un élément « activite » aux attributs « fam... » des éléments « famille_cout »; lorsque l'attribut « fam1 » sera trouvé en « famille_cout », la valeur de l'attribut « nom_famille » de cet élément « famille_cout » sera affichée.

Pour l'élément « activite » dont l'attribut « id_activite » = "act1", les familles de coût figurant dans l'élément « composition_activite » correspondant aux valeurs prises par l'attribut « id_famille _kout » sont: « fam1 » et « fam2 ».

Le résultat affiché pour cet élément devrait être « fam1_publicite » et « fam2_bureau ».
D'autres résultats seront affichés: nous ne filtrons pas

Que fait la requête?

La première boucle FOR évalue à partir du « context node » ABC le premier élément de la séquence « activite » et le lie à la variable « $akty ».
La clause « return » en restitue la valeur de l'attribut « nom_activite ».

Puis, la boucle FOR contenue dans « return » sélectionne à partir du « context node » « ABC », le premier élément de la séquence « famille_cout » et le lie à la variable « $famy » si la condition suivante est respectée.

La valeur de l'attribut « id_famille_cout » (de l'élément « famille_cout » traîté) est comparée à celle prise par l'attribut « id_famille_kout » du premier élément « composition_activite » de l'élément « activite » lié à la variable « $akty ».
Lorsque la comparaison est positive, la valeur de l'attribut « nom_famille » de l'élément « famille_cout » est affichée.

La même itération se reproduit pour la « composition_activite » suivante de la variable « $akty », et ce jusqu'à la dernière « composition_activite» de l' « activite » traitée.

Ensuite, retour à la première boucle FOR, qui reproduit les mêmes itérations pour le second élément de la séquence « activite », et ce jusqu'à ce qu'il n'y ait plus d'élément non traité dans la séquence « activite ».

Le résultat figure ci-dessous.

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_lien>
  <activite>
    <nom_activite>act1_communication</nom_activite>
    <famille_cout>fam1_publicite</famille_cout>
    <famille_cout>fam2_bureau</famille_cout>
  </activite>
  <activite>
    <nom_activite>act2_demarchage</nom_activite>
    <famille_cout>fam2_bureau</famille_cout>
    <famille_cout>fam4_formation_documentation</famille_cout>
  </activite>
  <activite>
    <nom_activite>act3_logistique</nom_activite>
    <famille_cout>fam3_stock</famille_cout>
    <famille_cout>fam7_outil_production</famille_cout>
  </activite>
  <activite>
    <nom_activite>act4_transport</nom_activite>
    <famille_cout>fam6_transport</famille_cout>
  </activite>
  <activite>
    <nom_activite>act5_production</nom_activite>
    <famille_cout>fam4_formation_documentation</famille_cout>
    <famille_cout>fam7_outil_production</famille_cout>
  </activite>
  <activite>
    <nom_activite>act6_comptabilite_abc</nom_activite>
    <famille_cout>fam2_bureau</famille_cout>
    <famille_cout>fam4_formation_documentation</famille_cout>
    <famille_cout>fam5_honoraires</famille_cout>
  </activite>
  <activite>
    <nom_activite>act7_telecom</nom_activite>
    <famille_cout>fam8_telecom</famille_cout>
  </activite>
</requete_lien>
```

Notons que nous obtenons un arbre XML ayant pour élément racine </requete_lien>, qui a pour éléments enfants les éléments </activite> lesquels contiennent les éléments </nom_activite> et </famille_cout>.

Effectuer des opérations arithmétiques.

Nous devons effectuer, pour obtenir des résultats ABC, des opérations de calcul.
Ce sont des opérations élémentaires, sommer, multiplier et diviser.

- Sommer fait usage de la fonction « SUM », qui est une fonction d'agrégation portant sur une séquence de résultats.
- La multiplication est liée à l'opérateur « * »
- La division a pour opérateurs « div » et « idiv ». Nous utiliserons « div ».

Sommer les charges .

Cette requête devrait nous retourner la somme des « montant_htva_total ».
Nous reproduisons le fragment de document XML ci-dessous.

```
<charge_indirecte id_chargeindirecte="fact001" montant_htva_total="2000">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact005" montant_htva_total="2401">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact011" montant_htva_total="710">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact015" montant_htva_total="623.85">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact016" montant_htva_total="310">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact028" montant_htva_total="4562">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact031" montant_htva_total="2100">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact032" montant_htva_total="2542">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact054" montant_htva_total="600">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact055" montant_htva_total="240">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact056" montant_htva_total="211">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact062" montant_htva_total="118.7">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact071" montant_htva_total="560">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact072" montant_htva_total="422">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact075" montant_htva_total="58">
        ... </charge_indirecte>
<charge_indirecte id_chargeindirecte="fact100" montant_htva_total="74.6">
        ... </charge_indirecte>
```

La requête:

```
<requete_somme>
{
for $chargeind in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")
//charge_indirecte
return
<Eur> {sum($chargeind/@montant_htva_total)} </Eur>
}
</requete_somme>
```

Cette requête comporte une erreur.
Nous la présentons car le résultat, qui n'est pas celui attendu, illustre très bien la notion de séquence de résultats.

Fonctionnement de la requête.

La variable « $chargeind » est liée à la séquence « charge_indirecte ».
Une boucle FOR parcourt une à une les instantiations de la séquence et les passe à « $chargeind » .
Return affiche à chaque itération l'attribut « charge_indirecte/@montant_htva_total » de la variable « $chargeind ».

Le résultat retourné.

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_somme>
  <Eur>2000</Eur>
  <Eur>2401</Eur>
  <Eur>710</Eur>
  <Eur>623.85</Eur>
  <Eur>310</Eur>
  <Eur>4562</Eur>
  <Eur>2100</Eur>
  <Eur>2542</Eur>
  <Eur>600</Eur>
  <Eur>240</Eur>
  <Eur>211</Eur>
  <Eur>118.7</Eur>
  <Eur>560</Eur>
  <Eur>422</Eur>
  <Eur>58</Eur>
  <Eur>74.6</Eur>
</requete_somme>
```

Nous constatons que le résultat n'est pas un total, mais une séquence des nombres qui auraient du être sommés.

La cause en est que nous avons intégré la fonction « SUM » dans la séquence de génération de nombres, alors que la séquence aurait dû se produire dans « SUM », « SUM » étant une fonction d'agrégation.
Si nous avions appelé la fonction « data » au lieu de la fonction « SUM », le résultat aurait été identique.

La requête doit être modifiée de façon à ce que les itérations FOR se produisent au sein de « SUM ».
Nous invoquons dès lors « SUM » au début de la requête

```
<requete_somme>
{
sum(
for $chargeind in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")
//charge_indirecte
return
<Eur> {data($chargeind/@montant_htva_total)} </Eur>
)
}
</requete_somme>
```

Nous obtenons le résultat attendu: la somme est de 17533,15 (Euro).

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_somme>17533.15</requete_somme>
```

Calculer le coût d'une unité d'oeuvre (d'activité).

Dans cette nouvelle requête, nous voulons calculer le coût d'une unité d'oeuvre pour une activité déterminée.
Nous choisissons l'activité5, « act5_production ».

La démarche suivie.

1) Calculer le coût total de l'activité choisie.
   - Filtrer les éléments « activite ».
   - Les lier aux éléments « famille_cout ».
   - Lier les éléments « famille_cout » aux éléments « charge_indirecte/imputation ».
2) Sommer la quantité de mesurages correspondant à l'activité choisie.
3) Calculer et sommer.

Le calcul de notre requête peut se résumer par:

SUM « montant_htva_impute » * « proportion_cf » * « proportion » div SUM « quantite » (de mesurages).

La requête se présente comme suit:

```
<requete_calcul_unite_oeuvre>
{
sum(
for $in in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC,
$akty in $in/activite[@id_activite="act5"],
$compoakt in $akty/composition_activite,
$famy in $in/famille_cout[@id_famille_cout=$compoakt/@id_famille_kout]/composition_famille,
$pcmn in $in/charge_indirecte/imputation[@id_pcmn_imputation=$famy/@id_pcmn_fam]
```

```
return
<activite>{data($pcmn/@montant_htva_impute
*$famy/@proportion_cf
*$compoakt/@proportion
div
sum(
for $mesur in $in/client/commande/mesurage[@id_aktivite="act5"]
return
<div>{data($mesur/@quantite)}</div>)
)}
</activite>
)}
</requete_calcul_unite_oeuvre>
```

Fonctionnement de la requête.

Nous utilisons des boucles FOR; notre point de départ sera le « context node » ABC.

La variable « $akty » est liée au premier élément de la séquence « activite » qui est sélectionné si le prédicat « id_activite » = « act5 » est respecté.
Si le prédicat n'est pas respecté, par itération le second élément est examiné, le troisième, jusqu'au dernier si nécessaire.

La variable « $compoact » est liée au premier élément de la séquence « composition_activite » dont le « context node » est $akty (qui respecte « act5 »).

La variable « $famy » est liée au premier élément de la séquence « famille_cout » si un lien est établi.
Nous établissons un lien avec l'élément « activite » par comparaison des valeurs des attributs « famille_cout/@id_famille_cout » et « activite/composition_activite/@id_famille_kout ».
Le lien est établi si il y a concordance, sinon, il y a itération vers le second élément de la séquence « famille_cout », évaluation pour lien, et ce jusqu'au dernier si nécessaire.

La variable « $pcmn » accède au premier élément « imputation » du « context node » « charge _indirecte » .
Comme ci-dessus, un lien est établi avec la variable « $famy » par comparaison des valeurs des attributs « id_pcmn_imputation » à « id_pcmn_fam ».
Une itération se produit pour le second élément « imputation » , etc, tant qu'un lien n'a pas été établi.

Return reçoit la valeur de l'attribut « montant_htva_impute » lorsqu'un lien est établi ($pcmn) et traite le premier résultat en le multipliant par la valeur de l'attribut « $famy/@proportion_cf » multiplié par la valeur de l'attribut « $compoakt/@proportion ».

```
<activite>{data($pcmn/@montant_htva_impute
*$famy/@proportion_cf
*$compoakt/@proportion
```

Nous devons diviser ce résultat par la somme des quantités de mesurages correspondant.
Return comprend une boucle FOR.
La variable « $mesur » est liée aux éléments de la séquence « mesurage » du « context node »

« commande », dont la valeur de l'attribut « id_aktivite » respecte le prédicat « id_aktivite » = « act5 ».
Cette boucle se produit à l'intérieur d'un « SUM » qui somme les valeurs prises par l'attribut « quantite » qui sont retournées.

```
div
sum(
for $mesur in $in/client/commande/mesurage[@id_aktivite="act5"]
return
<div>{data($mesur/@quantite)}</div>)
```

Le premier résultat « montant_htva_impute » * « proportion_cf » * « proportion » est ainsi divisé par la somme des « quantite ».

Après cette opération, la boucle retourne parcourir et comparer les éléments « imputation », puis retourne vers les éléments « composition_famille », et enfin retourne vers les éléments « composition_activite » jusqu'à l'épuisement des éléments « activite ».

Après le dernier « return », la séquence de résultats est sommée.
Le résultat de la requête:

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_calcul_unite_oeuvre>18.17860066006601</requete_calcul_unite_oeuvre>
```

Soit 18,178 (Euro) le coût d'une unité d'oeuvre de l'activité « act5_production ».

Nous avons voulu vérifier au moyen d'un tableur (sous Linux).
Nous y avons rapporté les données rencontrées par la requête qui parcours le fichier XML.

id_mesurage	Q mesurage	famille	proportion_fam	pcmn	proportion_pcm	montant	produit	quotient
mes004	8	fam4	0,34	612300	1	157,85	53,67	0,71
mes008	8,5	fam4	0,34	612300	1	560	190,4	2,51
mes010	6	fam7	0,73	601300	1	850	620,5	8,19
mes012	10	fam7	0,73	611130	1	422	308,06	4,07
mes023	2	fam7	0,73	613540	1		0	0
mes018	8	fam7	0,73	630220	0,8	350	204,4	2,7
mes020	7,25							
mes026	10							
mes031	7							
mes033	9							
					Valeur de 1 unité d'oeuvre de l'activité5			18,18
	75,75							

Les mesurages portant sur l'activité5 sont au total de 75,75 unités.
L'activité 5 est composée des familles fam4 qui intervient à 34% et fam5 qui intervient à 73%.
La famille4 est constituée du poste pcmn 612300 qui intervient à 100% dans la famille4.
Nous trouvons dans charge_indirecte/imputation deux « montants_htva_impute », soit 157,85 (Euro) et 560 (Euro).

```
157,85 * 1 * 0,34 = 53,67 (Euro)
560 * 1 * 0,34 = 190,4 (Euro)
```

Le même raisonnement est à appliquer à la famille7.
Détaillons la dernière ligne: le poste pcmn 630220 qui constitue la famille7: le montant de 350 (Euro) est imputé; il est multiplié par 80% (intervention dans la famille) et est à nouveau multiplié par 73%, intervention de la famille dans l'activité5, soit 350 * 0,73 * 0,8 = 204,4 (Euro)

Chacun de ces produits sont divisés un à un par le total des unités mesurées, soit 75,75 .

Pour la première ligne: 53,67 / 75,75 = 0,71 (Euro)
...
La dernière ligne: 204,4 /75,75 = 2,7 (Euro)
La somme de ces quotients est bel et bien de 18,18 (Euro) , résultat fourni par la requête.

Ayant pu vérifier la justesse du résultat de la requête, nous calculons encore le coût d'une unité d'oeuvre de l'activité « act4_transport », et en retenons la valeur pour la suite: 2,652 (Euro).

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_calcul_unite_oeuvre>2.6521739130434785</requete_calcul_unite_oeuvre>
```

Calculer le coût indirect d'une commande déterminée.

Nous souhaitons obtenir le coût indirect pour une commande précise.

La démarche suivie.

Nous réutilisons la démarche précédente, mais le point de départ est une commande choisie.

1) Sélectionner une commande et calculer le coût total des activités utilisées.
   - Filtrer les éléments « commande ».
   - Sélectionner les éléments « activite » utilisés par la commande choisie.
   - Les lier aux éléments « famille_cout ».
   - Lier les éléments « famille_cout » aux éléments « charge_indirecte/imputation ».
2) Sommer la quantité de mesurages correspondant aux activités traitées.
3) Calculer et sommer.

En cours d'élaboration de cette requête, est apparue une erreur du fait de la nature différente des activités, erreur que nous détaillons.

Prenons comme exemple la commande ayant le « id_commande » com003.
Cette commande comporte deux mesurages, l'un concernant l'activité « id_aktivite » = « act4 »,

l'autre, l'activité « id_aktivite » = « act5 ».
Les unités d'oeuvre utilisées pour ces deux activités sont fondamentalement différentes.
l'activité4 transport a pour unité de mesure le kilomètre parcouru.
L'activité5 se mesure elle en heure de production.

id_activite	mesurage	activité	produit	quotient
act4	45	act4	**600**	**5**
act4	62	act5	**1000**	**8,33**
act5	6			
act5	7			
Somme	**120**			

Ce tableau synthétise l'information telle qu'elle sera traitée par la requête.
Nous imaginons qu'activité4 (act4) et activité5 (act5) ont respectivement un coût de 600 (Euro) et 1000 (Euro).
La colonne « produit » doit son appellation au fait qu'il s'agit du résultat de l'opération:
« montant_htva _impute » * « proportion_cf » * « proportion ».

Du fait de la nature différente des unités d'oeuvre, nous aurons beaucoup plus de kilomètres que d'heures prestées.
Nous avons 107 kilomètres (45 + 62) parcourus pour 13 heures de travail (6 + 7).
La requête sommera 120 unités mesurées, sans distinction quant à leur différence de nature.
Le calcul qui sera effectué pour calculer le coût d'une unité d'oeuvre sera: 600 (Euro) / 120 et 1000 (Euro) / 120, soit 5 (Euro) et 8,33 (Euro).

Ce qui est évidement faux.

L'opération à effectuer est de diviser l'activité4 par la somme d'unités s'y rapportant, et pratiquer de même pour l'activité5.

id_activite	mesurage	activité	produit	quotient
act4	45	act4	**600**	**5,61**
act4	62			
Somme	**107**			

Pour l'activité4, 600 (Euro) / 107 = 5,61 (Euro) par unité d'oeuvre

id_activite	mesurage	activité	produit	quotient
act5	6	act5	**1000**	**76,92**
act5	7			
Somme	**13**			

Pour l'activité5, 1000 (Euro) / 13 = 76,92 (Euro) par unité d'oeuvre

La requête devra donc intégrer la possibilité de distinguer les différents types d'unité d'oeuvre possibles.

Calculer le coût indirect de la commande « com003 ».

Nous traiterons l'exemple qui consiste à afficher le coût indirect de la commande dont la valeur de « id_commande » est « com003 ».

Nous reproduisons le fragment du document XML ci-dessous.

```
<commande id_commande="com003" descriptif="seconde commande" chiffre_affaire="1852">
<mesurage id_mesurage="mes022" id_aktivite="act4" date="12 avril 2009" quantite="42"
        unite_oeuvre="kilometre"/>
<mesurage id_mesurage="mes023" id_aktivite="act5" date="12 avril 2009" quantite="2"
        unite_oeuvre="heure_production"/>
<charge_directe id_charge_d="fact0041" id_pcmn="601000" montant_dir_htva_impute="640"
            quantite="1"/>
<charge_directe id_charge_d="fact9999" id_pcmn="620000" montant_dir_htva_impute="25"
            quantite="2"/>
</commande>
```

Nous utilisons comme base la requête précédente de calcul d'une unité d'oeuvre pour une activité donnée.
Nous y avons ajouté le prédicat nécessaire à parcourir le document XML à partir de la commande « com003 » .

La requête se présente comme suit.

```
<cout_indirect_commande>
{
sum(
for $in in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")//ABC,
$commande in $in/client/commande[@id_commande="com003"],
$akty in $in/activite[@id_activite=$commande/mesurage/@id_aktivite],
$compoakt in $akty/composition_activite,
$famy in $in/famille_cout[@id_famille_cout=$compoakt/@id_famille_kout]/composition_famille,
$pcmn in $in/charge_indirecte/imputation[@id_pcmn_imputation=$famy/@id_pcmn_fam]

return
<activite>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage/@quantite
return
<uo_total>{data ($uototal)}</uo_total>) )}</activite>
)
```

```
}
</cout_indirect_commande>
```

Fonctionnement de la requête.

En boucle FOR, nous déclarons les variables.
La variable « $commande » est liée au premier élément de la séquence « commande » du « context node » ABC respectant le prédicat « id_commande » = « com003 ».
La variable « $akty » est liée au premier élément de la séquence « activite » qui est sélectionné si un lien peut être établi avec l'élément de la séquence « commande ».
Le lien est effectif si la comparaison des attributs « id_activite » de l'élément activité et de « id_aktivite » du premier élément de la séquence mesurage ayant pour « context node » « $commande » abouti.
Si le lien n'est pas établi, une itération examine le second élément de la séquence « activite », etc...

La variable « $compoact » est liée au premier élément de la séquence « composition_activite » dont le « context node » est $akty (qui est liée à « $commande »).

La variable « $famy » est liée au premier élément de la séquence « composition_famille » de la séquence « famille_cout » si un lien est établi.
Nous établissons un lien avec l'élément « activite » par comparaison des valeurs des attributs « famille_cout/@id_famille_cout » et « $compoakt/@id_famille_kout ».
Le lien est établi si il y a concordance, sinon, il y a itération vers le second élément de la séquence « famille_cout », pour une nouvelle évaluation.

La variable « $pcmn » accède au premier élément « imputation » du « context node » « charge_indirecte » .
Un lien est établi avec la variable « $famy » par comparaison des valeurs des attributs « imputation/@id_pcmn_imputation » à « $famy/@id_pcmn_fam ».
Une itération se produit pour le second élément « imputation » , etc..., tant qu'un lien n'a pas été établi.

Return fourni une séquence de résultats: « montant_htva_impute » multiplié par « proportion_cf » multiplié par « proportion » divisé par la somme de la valeur des attributs « mesurage/@quantite » sélectionnés.
Nous devons enfin multiplier ce résultat par la somme de mesurages « mesurage/@quantite » se rapportant à cette commande « com003 ».

Cette requête traitera les mesurages de la commande « com003 » identiquement, et générera l'erreur que nous évoquions ci-dessus.

Le résultat de la requête annonce un coût indirect de 446,234 (Euro) pour cette commande.

```
<?xml version="1.0" encoding="UTF-8"?>
<cout_indirect_commande>446.2341931548628</cout_indirect_commande>
```

IF THEN ELSE.

Lorsque nous examinons les mesurages de « com003 », nous voyons qu'ils sont deux.
Le premier 42 unités (kilomètres) pour l'activité4 (transport).
Le second 2 unités (heure_production) pour l'activité5 (production).

Afin que ces deux activités soient traitées individuellement, nous avons utilisé IF THEN ELSE.

Nous avons 7 activités qui doivent être traitées individuellement, et devons proposer 7 traitements individuels.
Nous avons retenu la solution suivante.

Return sera suivi d'une expression IF posant comme condition que l'activité à traiter soit l'activité1.
Si il s'agit bien de l'activité1, l'expression THEN autorise le calcul de la requête.
La différence du traitement appliqué par « Return » comparativement aux deux requêtes précédentes est que nous ne considérons plus les quantités de mesurage dans leur globalité, mais individuellement pour chaque activité.
Nous ajoutons donc un prédicat: « @id_aktivite="act1" » .

for « $uo » in « $in » /client/commande/mesurage/@quantite
devient
for « $uo » in « $in » /client/commande/mesurage**[@id_aktivite="act1"]**/@quantite

```
return
if ($akty/@id_activite="act1")
then
<activite_1>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act1"]/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act1"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>) )}</activite_1>
```

Si il ne s'agit pas de l'activité1, l'expression ELSE évalue s'il s'agit de l'activité2 au quel cas l'expression THEN autorise le traitement.

L'évaluation se fera sur base du prédicat « @id_aktivite="act2" »

```
else
if ($akty/@id_activite="act2")
then
<activite_2>{data($ ... etc...
```

Et ce répétitivement jusqu'à l'activité6 au terme de laquelle il ne reste plus comme dernière alternative que de traiter l'activité7.

```
else
<activite_7>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(

.....etc.....

<uo_total>{data ($uototal)}</uo_total>)
)}</activite_7>
)
}
</cout_abc_commande>
```

L'entièreté de cette requête est imbriquée dans SUM( ), afin de sommer la séquence de résultats.

Nous reproduisons la requête complète ci-dessous.

```
<cout_indirect_commande>
{
sum(
for $in in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC,
$commande in $in/client/commande[@id_commande="com003"],
$akty in $in/activite[@id_activite=$commande/mesurage/@id_aktivite],
$compoakt in $akty/composition_activite,
$famy in $in/famille_cout[@id_famille_cout=$compoakt/@id_famille_kout]/composition_famille,
$pcmn in $in/charge_indirecte/imputation[@id_pcmn_imputation=$famy/@id_pcmn_fam]

return
if ($akty/@id_activite="act1")
then
<activite_1>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act1"]/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act1"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>) )}</activite_1>

else
if ($akty/@id_activite="act2")
then
<activite_2>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act2"]/@quantite
return<uoeuvre>{data ($uo)}
</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act2"]/@quantite
```

```
return
<uo_total>{data ($uototal)}</uo_total>)
)}</activite_2>

else
if ($akty/@id_activite="act3")
then
<activite_3>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act3"]/@quantite
return<uoeuvre>{data ($uo)}
</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act3"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
)}</activite_3>

else
if ($akty/@id_activite="act4")
then
<activite_4>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act4"]/@quantite
return<uoeuvre>{data ($uo)}
</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act4"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
)}</activite_4>

else
if ($akty/@id_activite="act5")
then
<activite_5>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act5"]/@quantite
return<uoeuvre>{data ($uo)}
</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act5"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
)}</activite_5>
else
if ($akty/@id_activite="act6")
then
<activite_6>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
```

```
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act6"]/@quantite
return<uoeuvre>{data ($uo)}
</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act6"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
)}</activite_6>

else
<activite_7>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act7"]/@quantite
return<uoeuvre>{data ($uo)}
</uoeuvre>)
*
sum(
for $uototal in $commande/mesurage[@id_aktivite="act7"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
)}</activite_7>
)
}
</cout_indirect_commande>
```

Le résultat généré est cette fois de 147,748 (Euro).

```
<?xml version="1.0" encoding="UTF-8"?>
<cout_indirect_commande>147.7485056679581</cout_indirect_commande>
```

Les « function ».

En procédant de la sorte, nous obtenons un résultat différent, mais au prix d'une requête longue (121 lignes) et complexe à lire.

Nous utiliserons les « functions » ou fonctions proposées par XQuery afin d'alléger le code.

Le bloc suivant calculant la somme des mesurages pour une activité se répète à sept reprises.

```
sum(
for $uo in $in /client/commande/mesurage[@id_aktivite="act1"]/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
```

Nous le transformons en « function ».

Une « function » comporte les parties suivantes:
- la déclaration de « function » suivie de son nom « declare function local: ici le nom ».
- les paramètres de la « function » et le type retourné par la « function » (entre parenthèses).
- le corps de la « function » contenant les instructions de traitement.

Nous déclarons la « function » que nous nommons : « activite »:

```
declare function local:activite
```

Nous avons un seul paramètre qui est « act1 » ou « act2 », ..., ou « act7 ».
La variable « $ak » que nous déclarons prendra comme argument un des sept « act... »
Ce paramètre est de type « atomic », et le résultat de la « function » sera également une valeur « atomic » (un nombre).

La déclaration de « function » complète est:

```
declare function local:activite($ak as xs:anyAtomicType?)as xs:anyAtomicType?
```

Le corps de la « function » est la partie de imbriquée dans SUM.

```
{
sum(
for $uo in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")
//client/commande/mesurage[@id_aktivite=$ak]/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
}
```

La « function » complète ci-dessous:

```
declare function local:activite($ak as xs:anyAtomicType?)as xs:anyAtomicType?
{
sum(
for $uo in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")
//client/commande/mesurage[@id_aktivite=$ak]/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
};
```

Nous pouvons créer une seconde « function » calculant la somme des mesurage par activité pour une commande précise.

```
sum(
for $uototal in $commande/mesurage[@id_aktivite="act7"]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
```

Nous nommons cette « function » « activite2 ».

```
declare function local:activite2($ak2 as xs:anyAtomicType?)as xs:anyAtomicType?
{
```

```
sum(
for $uototal in doc("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")//
client/commande[@id_commande="com003"]/mesurage[@id_aktivite=$ak2]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
};
```

XQuery permet qu'une « function » se trouve au sein d'un espace mémoire dédié ou bien dans la requête elle même; nous retenons cette seconde possibilité.

Les deux « functions » se trouvent en préambule de la requête.
Au sein de la requête, pour le calcul relatif à l'activité1, nous appelons la « function » « activite » par:

```
local:activite("act1")
```

et lui passons le paramètre « act1 »
et la seconde « function » qui reçoit le meme paramètre par:

```
local:activite2("act1")
```

La requête complète devient:

```
declare function local:activite($ak as xs:anyAtomicType?)as xs:anyAtomicType?
{
sum(
for $uo in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")
//client/commande/mesurage[@id_aktivite=$ak]/@quantite
return
<uoeuvre>{data ($uo)}</uoeuvre>)
};

declare function local:activite2($ak2 as xs:anyAtomicType?)as xs:anyAtomicType?
{
sum(
for $uototal in doc("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")//
client/commande[@id_commande="com003"]/mesurage[@id_aktivite=$ak2]/@quantite
return
<uo_total>{data ($uototal)}</uo_total>)
};
<cout_indirect_commande>
{
sum(
for $in in doc ("/home/jean/Mémoire_ulb_2009/group_21juin/abc_avril27jn_id01.xml")/ABC,
$commande in $in/client/commande[@id_commande="com003"],
$akty in $in/activite[@id_activite=$commande/mesurage/@id_aktivite],
$compoakt in $akty/composition_activite,
$famy in $in/famille_cout[@id_famille_cout=$compoakt/@id_famille_kout]/composition_famille,
$pcmn in $in/charge_indirecte/imputation[@id_pcmn_imputation=$famy/@id_pcmn_fam]

return
if ($akty/@id_activite="act1")
```

```
then
<activite_1>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act1")
*
local:activite2("act1")
)}</activite_1>

else
if ($akty/@id_activite="act2")
then
<activite_2>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act2")*local:activite2("act2")
)}</activite_2>

else
if ($akty/@id_activite="act3")
then
<activite_3>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act3")*local:activite2("act3")
)}</activite_3>

else
if ($akty/@id_activite="act4")
then
<activite_4>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act4")*local:activite2("act4")
)}</activite_4>

else
if ($akty/@id_activite="act5")
then
<activite_5>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act5")*local:activite2("act5")
)}</activite_5>

else
if ($akty/@id_activite="act6")
then
<activite_6>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act6")*local:activite2("act6")
)}</activite_6>

else
<activite_7>{data($pcmn/@montant_htva_impute*$famy/@proportion_cf*$compoakt/@proportion
div
local:activite("act7")*local:activite2("act7")
)}</activite_7>
)}
</cout_indirect_commande>
```

La requête est simplifiée, 85 lignes; si nous souhaitons modifier la méthode de calcul des quantités de mesurage, une seule modification de « function » sera nécessaire, alors qu'il aurait fallu modifier à sept reprises sans l'usage de « functions ».

Le résultat de cette requête est identique:

```
<?xml version="1.0" encoding="UTF-8"?>
<cout_indirect_commande>147.7485056679581</cout_indirect_commande>
```

Vérification du résultat.

Nous avons à nouveau souhaité vérifier le résultat.
Ayant déjà vérifié le coût d'une unité d'oeuvre pour l'activité5, nous vérifions celle de l'activité 4.
Commande « com003 » utilise 42 unités d'oeuvre de l'activité4 (mesurage « mes022 »).
La somme de tous les mesurages se rapportant à l'activité4 est de 483 unités d'oeuvre.

Nous obtenions précédement pour l'activité5 un coût d'unité d'oeuvre de 18,178 (Euro)

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_calcul_unite_oeuvre>18.17860066006601</requete_calcul_unite_oeuvre>
```

et de 2,652 (Euro) pour l'activité4.

```
<?xml version="1.0" encoding="UTF-8"?>
<requete_calcul_unite_oeuvre>2.6521739130434785</requete_calcul_unite_oeuvre>
```

Pour la commande « com003 », 42 unités de l'activité4 sont utilisées: 2,652 * 42 = 111,384 (Euro).
Pour l'activité5, où 2 unités d'oeuvre sont mesurées: 18,18 * 2 = 36,36 (Euro).

Sommons ces deux valeurs: 111,384 + 36,36 = 147,744 (Euro) soit le résultat fourni par la requête XQuery.

Chapitre 8. Présentation des résultats.

Parvenant au terme de ce travail, nous nous attacherons en cette dernière partie à la présentation des résultats obtenus.
Au départ d'un fichier au format XML, nous avons au travers des diverses requêtes XQuery produit des résultats sous forme d'arbres XML, qui sont lisibles, mais dont la présentation peut être améliorée.
Pour ce, nous utiliserons les solutions XSL du W3C, et plus particulièrement XSL-FO.

Comme pour la partie dédiée à XQuery, nous utiliserons le vocabulaire technique en anglais, la recommandation source du W3C étant rédigée en anglais.

XSL.

Extensible Stylesheet Language est un langage de transformation de documents XML.
XSL, devenu une recommandation du W3C le 16 décembre 1998 en version1.0 , a évolué pour être scindé en trois domaines: Xpath, XSLT et XSL-FO.

XSLT et Stylesheet.

Une transformation XSL s'opère par le recours à une « stylesheet » ou feuille de style.
XSLT est le langage utilisé pour produire les « stylesheet », ce qui revient à ce qu'une stylesheet est un document de transformation XSLT.
Une « stylesheet » permet de décrire les règles de transformation des données (éléments et attributs) d'un arbre XML afin d'aboutir en fin de traitement à leur présentation (police, taille, couleur, fond de page,..).

Une « stylesheet » est un document XML bien formé utilisant des expressions issues de l'espace de nom XML.

Transformation et Templates.

Une « stylesheet » consiste en un ensemble de « template rules ».
Un « template rule » utilise deux composants:

1) Des pattern qui recourent à Xpath et sélectionnent les éléments de l'arbre XML de base.
2) Des template qui appliqués aux éléments sélectionnés, construisent chacun une partie de l'arbre de résultat.

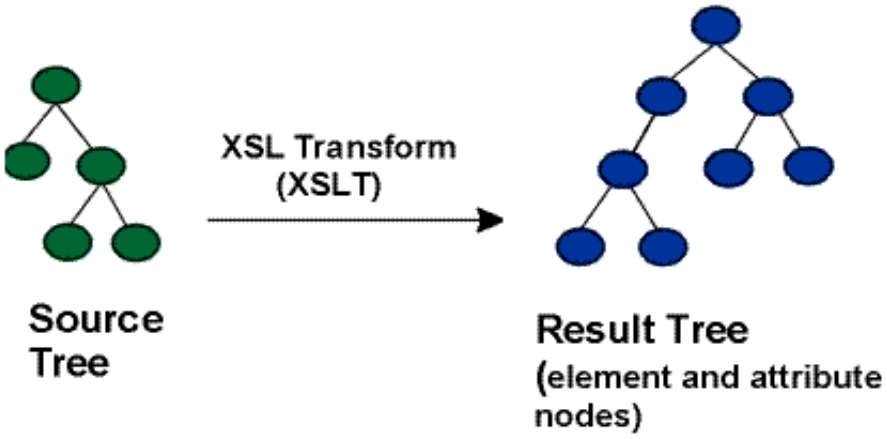
Principe de transformation.

Nous décrivons brièvement le principe de transformation d'un arbre XML en un autre afin de mieux appréhender les mécanismes mis en oeuvre.

Un processeur XSL reçoit en entrée un document XML (à traiter) et une « stylesheet ».

L'arbre XML initial est transformé par le processeur XSL en un autre arbre XML selon les indications fournies par la « stylesheet ».

Cette transformation est réalisée en deux étapes:
1) La construction de l'arbre de résultat (« tree transformation »).
2) Traitement de l'arbre de résultat selon la feuille de style (« formatting »).



Le formatting.

L'étape de « Formatting » inclus de la sémantique dans l'arbre de résultat dont les « nodes » sont des « formatting objects ».

Les « formatting objects » font partie de classes qui décrivent paragraphes, texte, tables.

« Formatting » consiste à créer un arbre « formatting object tree », dont les éléments sont des « formatting object ».

Le modèle du « formatting object tree » est celui du « area tree ».

« Formatting » revient à créer un arbre « area tree » constitué de « geometric area » .
Les « geometric area » consistent en une séquence de pages, où chaque « area » a sa position précise dans la page.
Cet « area tree » décrit la structure géométrique de la sortie.

Simplement exprimé, la transformation crée une séquence de zones géométriques ayant un contenu et une position précise au sein de séquence.

Nous retenons que les « formatting objects » peuvent être de type « block-level » ou « inline-level », ceci selon le type d' « areas » qu'ils génèrent.

XSL-FO.

XST-FO fait partie de la recommandation Extensible Stylesheet Language version 1.0 du 15 octobre 2001, modifiée en version 1.1 le 05 décembre 2006.
XSL-FO a pour finalité de produire un document imprimable, alors qu'une transformation XSLT comme nous venons de décrire, génère soit du XML, soit du HTML ou du texte.

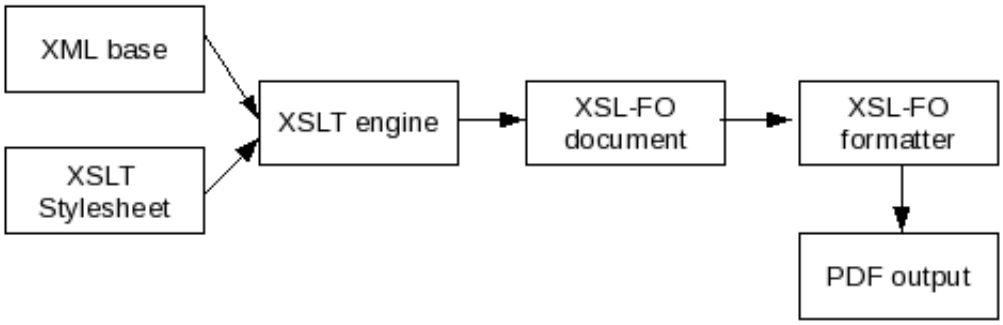
Plutôt que de devoir utiliser une multitude de spécifications correspondant chacune à un type d'imprimante existant, il a été choisi de produire un document au format PS ou PDF, ce qui le rend imprimable presque universellement.

Principe de transformation XSL-FO.

Une transformation XSL-FO est similaire à une transformation XSLT: à un document XML de base est appliquée une feuille de style XSLT qui sélectionne les éléments XML de base et leur applique les règles de transformation qui créent le document XSL-FO.

Ce document XSL-FO est alors traité par un « formatter » qui produit le document PDF imprimable.

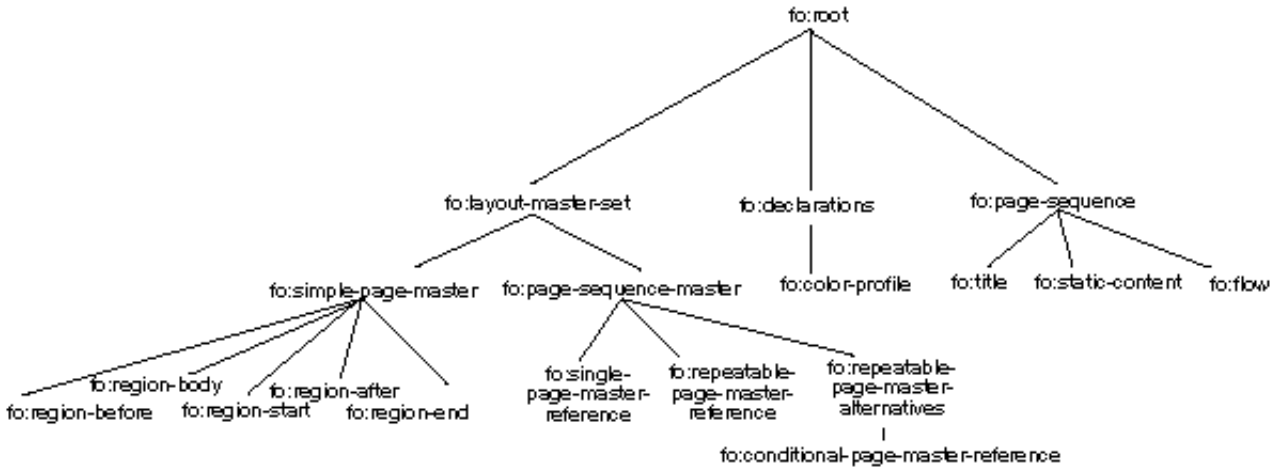
Apache propose un outil: Formatting Objects Processor - FOP - permettant la transformation en PDF d'un document XSL-FO.



XSL-FO stylesheet.

La construction d'une « stylesheet » XSL-FO repose sur l'utilisation de différents « formatting objects ».

La recommandation W3C propose le modèle ci-après.

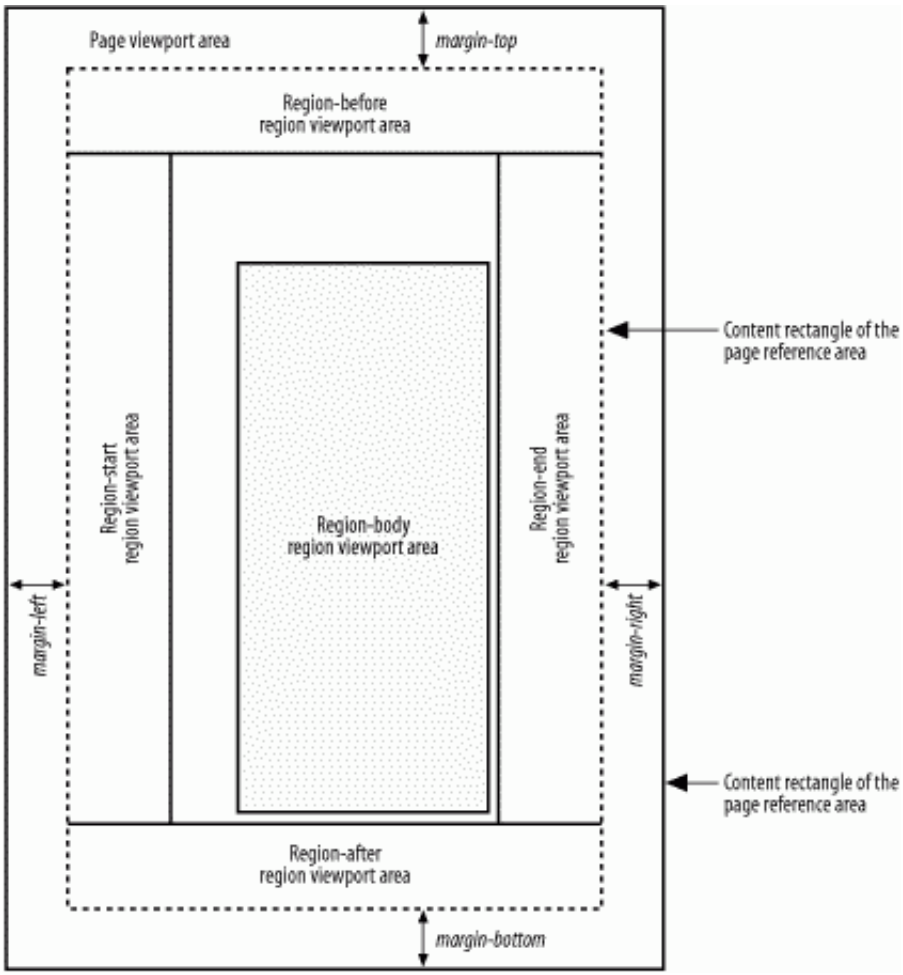


La forme d'un document XSL-FO est celle d'un arbre XML, à la base duquel se trouve un élément racine qui est le « fo:root » « formatting object ».

« fo-root » a pour éléments descendants:
- « fo:layout-master-set » qui défini la géométrie et la séquence des pages utilisées dans le document final.
- « fo:declarations » (qui est optionnel).
- une ou plusieurs « fo:page-sequences », dont les descendants sont des « flows », génèrent le contenu de ces pages.

« fo:layout-master-set » a pour enfants:
- simple-page-masters qui décrit les subdivisions des pages et leur géométrie; il faut au minimum un simple-page-master. Un simple-page-master a un attribut « master-name ».
- page-sequence-masters qui décrit la séquence de « page-masters » que le formatting du « fo:page-sequence » devra respecter. Il détermine l'ordre d'apparition des « page-masters ». Un page-sequence-master » a un attribut « master-reference ».

Représentation d'un simple-page-master model. (source: W3C)



Le « simple-page-master » permet de déterminer différentes régions (« region before », « region start », « region after », « region end », « body ») et la taille d'une page (« margin »).

D'autres propriétés ayant trait à la pagination peuvent être décrites, comme la direction d'écriture au travers des différentes pages d'un document « writing mode » et « relative direction » .

Application pratique.

Nous créons une présentation pour l'arbre XML synthétisant les résultats d'une commande, que nous développions au cours du chapitre précédent.

Nous le reproduisons ci-dessous.

```
<?xml version="1.0" encoding="UTF-8"?>
<presentation2>
   <commande>
      <id_commande>com001</id_commande>
      <descriptif>systeme solaire</descriptif>
      <nom>jaeger</nom>
      <adresse>rue du bois 36 à 1000 bruxelles</adresse>
   </commande>
   <charges>
      <cout_direct>5103</cout_direct>
      <cout_indirect_commande>3376.1968</cout_indirect_commande>
   </charges>
   <resultats>
      <chiffre_affaire>12000</chiffre_affaire>
      <marge_brute>6897</marge_brute>
      <marge_nette>3520.8032</marge_nette>
   </resultats>
   <ratios>
      <a>42.525</a>
      <b>57.475</b>
      <c>28.134974</c>
      <d>29.340027</d>
      <e>70.65997</e>
   </ratios>
</presentation2>
```

Détail de la stylesheet.

Nous détaillons la « Stylesheet » de façon fragmentaire; le document est présenté dans son entièreté plus bas.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

Nous déclarons le document XSL qui est conforme à la version XSL 1.0 et est lui même un document XML.
La mention de xmlns:xsl="http://www.w3.org/1999/XSL/Transform" indique au processeur XSLT qu'il devra utiliser le « namespace » correspondant (Transform).

```
<xsl:template match="presentation">
 <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
```

Nous sélectionnons l'élément « presentation » qui est l'élément racine du fichier XML que nous voulons transformer.
L'élément racine du fichier XSL-FO est déclaré, avec une précision pour le processeur (xmlns:fo="http://www.w3.org/1999/XSL/Format"), qui devra utiliser le « Namespace » « Format » pour la transformation FO.

```
<fo:layout-master-set>
 <fo:simple-page-master master-name="expdf" page-height="29.7cm"
 page-width="21cm" margin-top="2cm" margin-bottom="2cm"
 margin-left="2cm" margin-right="2cm">
 <fo:region-body/>
 </fo:simple-page-master>
 </fo:layout-master-set>
```

Nous déclarons le « fo:layout-master-set » contenant le « simple-page-master » dont le « master-name » est « expdf » . Nous y décrivons le format de page qui sera utilisé : une feuille A4 (21 * 29,5 cm).
Les marges seront de 2cm.

```
<fo:page-sequence master-reference="expdf">
 <fo:flow flow-name="xsl-region-body">
```

Le « page-sequence-master-reference » est déclaré, il fait référence à « expdf ».
Le flow (qui génère le contenu des pages) écrira dans la région « body ».

```
<fo:block border="2 px solid black" padding="4mm" space-after="18mm" >
 <fo:block font-weight="bold" font-family= "Helvetica"
 font-size="24pt" text-align="center"
 border="3px outset olive"  space-before = "8mm"  space-after="18mm" padding="2mm">
 Résultats financiers de la commande
 </fo:block>
```

Ici commence la sélection de ce qui devra être affiché en sortie de traitement.
La notion de « block-level » que nous évoquions plus haut est utilisée.
Dans la première ligne, nous déclarons que l'ensemble du contenu devra être encadré d'une bordure noire épaisse de 2px.
Cette première balise sera refermée après la déclaration de tous les blocs suivants.
Puis en seconde ligne nous déclarons le premier « block » qui sera le titre « Résultats financiers de la commande ».
La police, sa taille, son épaisseur, le mode d'alignement du texte sont déclarés; ce titre devra être encadré en couleur « olive ».

```
<fo:block font-weight="bold" font-family= "Helvetica"
 font-size="18pt" text-align="center"  border="2px double darkolivegreen"
 space-after="10mm" padding="2mm">
 Commande
 </fo:block>
```

```
<fo:block border="2 px solid black" padding="2mm" space-after="28mm" >
   <fo:block font-size="16pt" space-before="14pt"  space-after="2mm" line-height="20pt"
    text-align="left" start-indent="2mm" >
    - Référence Commande: <xsl:value-of select="commande/id_commande"/>
```

```
  </fo:block>

  <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
   - Client: <xsl:value-of select="commande/nom"/>
  </fo:block>

  <fo:block font-size="16pt" start-indent="2mm" space-after="2mm" line-height="20pt">
    - Détail de la commande: <xsl:value-of select="commande/descriptif"/>
  </fo:block>
 </fo:block>
```

Puis vient une zone portant le titre « Commande », suivi d'une zone reprenant les informations de l'arbre XML de départ: "commande/id_commande", "commande/nom", et "commande/descriptif". Nous sélectionnons ces éléments via l'instruction « xsl:value-of select » suivi d'une expression Xpath « commande/id_commande ».
Chaque élément sélectionné est précédé d'un commentaire: « - Référence Commande: », « - Client: », « - Détail de la commande: » dont le but est de faciliter la compréhension et d'enrichir la présentation des éléments XML de départ.

```
<fo:block font-weight="bold" font-family= "Helvetica"
   font-size="18pt" text-align="center" border="3px ridge darkolivegreen"
   space-after="10mm"  padding="2mm">
   Calcul de rentabilité
  </fo:block>
```

```
<fo:block border="2 px solid black" padding="2mm"  space-after="8mm">
    <fo:block font-size="16pt" space-before= "4mm" start-indent="2mm" space-after="2mm"
      line- height="20pt">
      Chiffre d'affaire de la commande: <xsl:value-of select="resultats/chiffre_affaire"/> Euro
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  ( - ) Charge Directe: <xsl:value-of select="charges/cout_direct"/> Euro
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  ( = ) Marge Brute: <xsl:value-of select="resultats/marge_brute"/> Euro
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  ( - ) Charge indirecte (ou coût ABC): <xsl:value-of select="charges/cout_indirect_commande"/> Euro
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt"
  padding="1mm"  border="2px solid red">
  ( = ) Marge Nette: <xsl:value-of select="resultats/marge_nette"/> Euro
 </fo:block>
</fo:block>
</fo:block>
```

```
 <fo:block font-weight="bold" font-family= "Helvetica"
   font-size="18pt" text-align="center" border="3px ridge darkolivegreen"
   space-after="4mm"  padding="2mm">    Présentation sous forme de ratios
```

```
  </fo:block>
  <fo:block border="2 px solid black" padding="2mm"  space-after="8mm">

  <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
   Proportion Charges Directes dans le chiffre d'affaire: <xsl:value-of select="ratios/a"/> %
  </fo:block>

  <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
   Proportion Marge Brute dans le chiffre d'affaire: <xsl:value-of select="ratios/b"/> %
  </fo:block>

  <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
    Proportion Charges Indirectes dans le chiffre d'affaire: <xsl:value-of select="ratios/c"/> %
  </fo:block>

  <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
    Proportion Marge Nette dans le chiffre d'affaire: <xsl:value-of select="ratios/d"/> %
  </fo:block>
  </fo:block>
</fo:flow>
 </fo:page-sequence>
 </fo:root>
 </xsl:template>
</xsl:stylesheet>
```

Nous créons une zone similaire à la précédente selon exactement les mêmes principes.

Le titre encadré devient « Calcul de rentabilité », les données sélectionnées dans l'arbre XML de résultat sont "resultats/chiffre_affaire", "charges/cout_direct", "resultats/marge_brute", "charges/cout_abc_ commande", "resultats/marge_nette".

Les commentaires sont : « Chiffre d'affaire de la commande: », « (-) Charge Directe: », « (=) Marge Brute: », « (-) Charge indirecte (ou coût ABC): » et « (=) Marge Nette: ».

La ligne dédiée à l'élément « resultats/marge_nette » est encadrée de rouge.

Enfin, un commentaire est placé à la suite de chaque élément sélectionné afin de préciser l'unité monétaire: « Euro ».

Enfin, en une troisième zone, identique, nous présentons les données issues du calcul de ratios.
Le titre: « Présentation sous forme de ratios ».
Nous présentons également les éléments: « ratios/a », « ratios/b », « ratios/c », et « ratios/d ».
Nous commentons chacun des éléments: « Proportion Charges Directes dans le chiffre d'affaire: », « Proportion Marge Brute dans le chiffre d'affaire: », « Proportion Charges Indirectes dans le chiffre d'affaire: », et « Proportion Marge Nette dans le chiffre d'affaire: ».
Chaque élément est suivi du signe « % ».
Ci-dessous, la « stylesheet » complète.

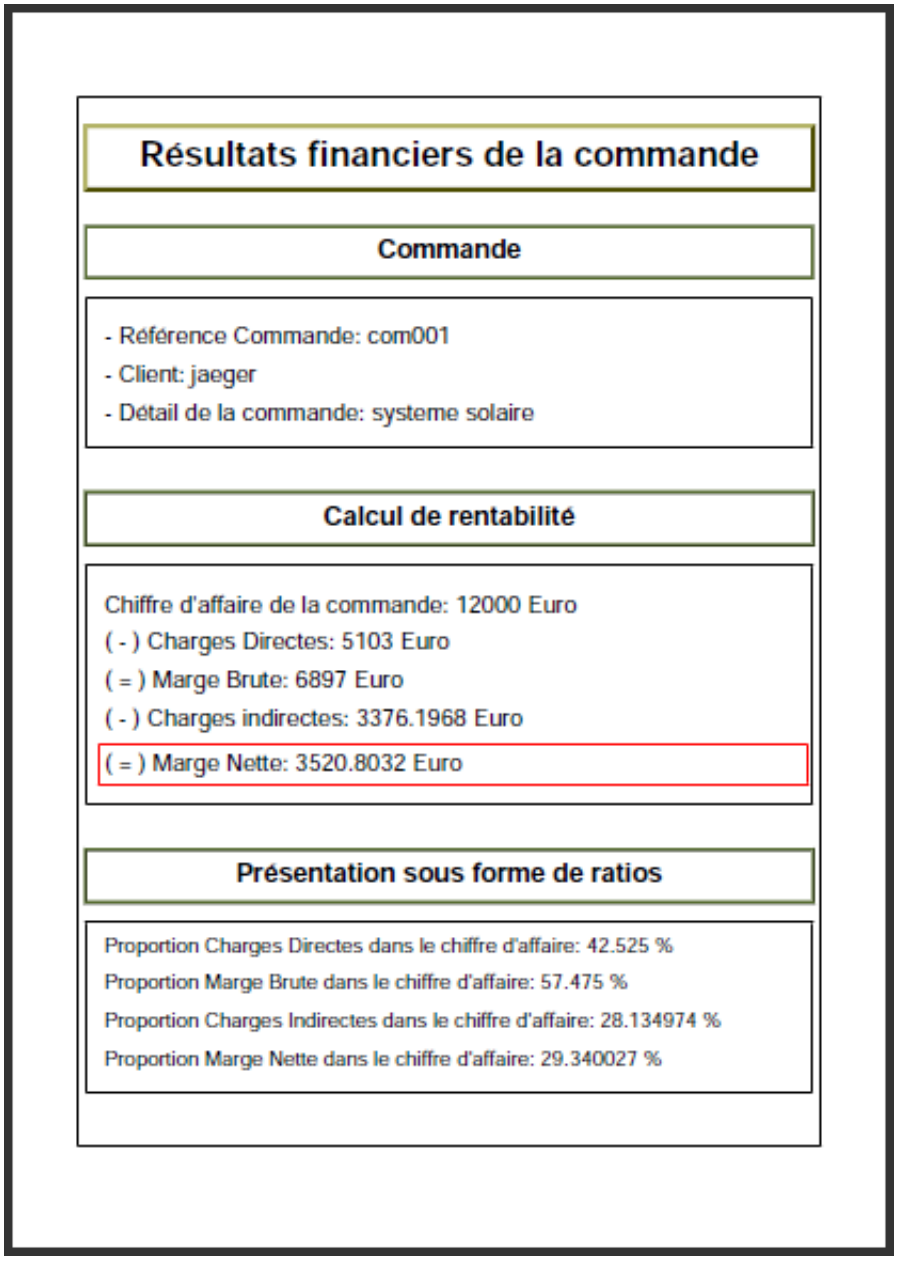
```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:fo="http://www.w3.org/1999/XSL/Format">
```

```
<xsl:template match="presentation">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">

<fo:layout-master-set>
<fo:simple-page-master master-name="expdf" page-height="29.7cm"
page-width="21cm" margin-top="2cm" margin-bottom="2cm"
margin-left="2cm" margin-right="2cm">
<fo:region-body/>
</fo:simple-page-master>
</fo:layout-master-set>

<fo:page-sequence master-reference="expdf">
<fo:flow flow-name="xsl-region-body">

<fo:block border="2 px solid black" padding="4mm" space-after="18mm" >
<fo:block font-weight="bold" font-family= "Helvetica"
font-size="24pt" text-align="center"
border="3px outset olive"   space-before = "8mm"  space-after="18mm" padding="2mm">
Résultats financiers de la commande
</fo:block>

<fo:block font-weight="bold" font-family= "Helvetica"
font-size="18pt" text-align="center"  border="2px double darkolivegreen"
space-after="10mm" padding="2mm">
Commande
</fo:block>

<fo:block border="2 px solid black" padding="2mm" space-after="28mm" >
  <fo:block font-size="16pt" space-before="14pt"  space-after="2mm" line-height="20pt"
    text-align="left" start-indent="2mm" >
    - Référence Commande: <xsl:value-of select="commande/id_commande"/>
  </fo:block>

  <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
    - Client: <xsl:value-of select="commande/nom"/>
  </fo:block>

  <fo:block font-size="16pt" start-indent="2mm" space-after="2mm" line-height="20pt">
     - Détail de la commande: <xsl:value-of select="commande/descriptif"/>
  </fo:block>
</fo:block>

<fo:block font-weight="bold" font-family= "Helvetica"
  font-size="18pt" text-align="center" border="3px ridge darkolivegreen"
  space-after="10mm"  padding="2mm">
  Calcul de rentabilité
 </fo:block>
<fo:block border="2 px solid black" padding="2mm"  space-after="8mm">
   <fo:block font-size="16pt" space-before= "4mm" start-indent="2mm" space-after="2mm"
      line- height="20pt">
      Chiffre d'affaire de la commande: <xsl:value-of select="resultats/chiffre_affaire"/> Euro
</fo:block>
 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  ( - ) Charge Directe: <xsl:value-of select="charges/cout_direct"/> Euro
```

```
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  ( = ) Marge Brute: <xsl:value-of select="resultats/marge_brute"/> Euro
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  ( - ) Charge indirecte (ou coût ABC): <xsl:value-of select="charges/cout_indirect_commande"/> Euro
 </fo:block>

 <fo:block font-size="16pt"  space-after="2mm" start-indent="2mm" line-height="20pt"
  padding="1mm"  border="2px solid red">
  ( = ) Marge Nette: <xsl:value-of select="resultats/marge_nette"/> Euro
 </fo:block>
</fo:block>
</fo:block>
 <fo:block font-weight="bold" font-family= "Helvetica"
  font-size="18pt" text-align="center" border="3px ridge darkolivegreen"
  space-after="4mm"  padding="2mm">    Présentation sous forme de ratios
 </fo:block>
 <fo:block border="2 px solid black" padding="2mm"  space-after="8mm">

 <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  Proportion Charges Directes dans le chiffre d'affaire: <xsl:value-of select="ratios/a"/> %
 </fo:block>

 <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  Proportion Marge Brute dans le chiffre d'affaire: <xsl:value-of select="ratios/b"/> %
 </fo:block>

 <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  Proportion Charges Indirectes dans le chiffre d'affaire: <xsl:value-of select="ratios/c"/> %
 </fo:block>

 <fo:block font-size="14pt"  space-after="2mm" start-indent="2mm" line-height="20pt">
  Proportion Marge Nette dans le chiffre d'affaire: <xsl:value-of select="ratios/d"/> %
 </fo:block>
 </fo:block>
 </fo:flow>
</fo:page-sequence>
</fo:root>
</xsl:template>
</xsl:stylesheet>
```

Traitement « formatter ».

La suite logicielle <oXygen/> intègre un « formatter ». Nous avons eu recours à cette solution, qui a généré le document .PDF suivant.



Résultats financiers de la commande

Commande

- Référence Commande: com001
- Client: jaeger
- Détail de la commande: systeme solaire

Calcul de rentabilité

Chiffre d'affaire de la commande: 12000 Euro
(-) Charges Directes: 5103 Euro
(=) Marge Brute: 6897 Euro
(-) Charges indirectes: 3376.1968 Euro
(=) Marge Nette: 3520.8032 Euro

Présentation sous forme de ratios

Proportion Charges Directes dans le chiffre d'affaire: 42.525 %
Proportion Marge Brute dans le chiffre d'affaire: 57.475 %
Proportion Charges Indirectes dans le chiffre d'affaire: 28.134974 %
Proportion Marge Nette dans le chiffre d'affaire: 29.340027 %

Nous y retrouvons les éléments du document XML de base, comme le nom du client, le chiffre d'affaire de la commande, la marge nette, etc.. enrichis des commentaires que nous souhaitions.

Sources.

W3C XSL Transformations (XSLT) Version 1.0
Illustrations: W3C XSL Transformations (XSLT) Version 1.0
W3C Extensible Stylesheet Language (XSL) Version 1.1
XSL-FO de Dave Pawson; ISBN : 0-596-00355-2

Chapitre 9. Suite du mémoire et conclusions.

Terminer ce mémoire.

Nous terminons ce mémoire, en ayant suivi les objectifs, excepté notre semi-échec concernant les logiciels libres.

Cependant, au fil de la réalisation de ce travail, d'autres objectifs potentiels sont apparus, ce qui nous fait dire que si ce travail est terminé par rapport aux objectifs de base, il ne l'est pas tout à fait.

Il s'agit de points qui pourraient être améliorés, ou encore de points auxquels nous n'avions pas pensé au moment de l'établissement des objectifs initiaux.
Ils pourraient faire l'objet de développement ultérieurs.

Recommandations pour la suite de ce travail.

Ce travail pourrait être poursuivi en développant les aspects que nous décrivons ci-après.

Recherche sur les logiciels libres.

Nous avons failli quant à l'objectif de n'utiliser que des logiciels libres, ayant dû acquérir une licence d'utilisation de la suite logicielle <oXygen/>XML.

Il est probable qu'en se concentrant sur ce point uniquement, une solution libre aboutisse.
Nos recherches (que nous avons dû limiter faute de temps) ont montré qu'il existe diverses solutions libres, mais non regroupées au sein d'une même suite, et surtout pas très ergonomiques et encore moins documentées.
Par regroupement, nous entendons un éditeur XML doté d'un parser , un logiciel Xquery, voire un traitement XSL et encore mieux, un formatter, le tout accessible via une plate-forme unique.
Idéalement dotée d'une interface graphique.

Simplement exprimé, la même chose que <oXygen/>XML, mais en licence libre et s'installant sous Linux.
A notre sens, XQDT (pour XQuery) sous la plate-forme Eclipse est une très bonne base moyennant un minimum de documentation.

Repenser le document XML.

La méthode que nous avons utilisée pour représenter l'ensemble des données comptables, relatives aux commandes, etc... au sein d'un même document XML pourrait être revue si le volume d'information devenait important.
Pour quelques milliers de saisies, et relativement peu d'éléments, cela convient.
La redondance de l'information (par exemple pour les postes pcmn) rend sa maintenance laborieuse et comporte un risque d'erreur.
Plus d'information et plus d'éléments augmenterait cette redondance.

Il serait alors intéressant de développer un support pour ces informations XML ayant recours à un ensemble tables plutôt qu'un document XML unique.

Travailler l'ergonomie.

Un travail portant sur l'ergonomie doit être réalisé.
L'utilisateur final ciblé, c'est à dire les gérants de TPE et PME sont très peu friands, voire réticents à toute forme de travail administratif, pour utiliser un terme global.
Cela relève très souvent de la corvée, et est sans cesse reporté au lendemain.

Si il doit être utilisé, il importe donc que ce projet ABC soit très bien présenté, et de façon la plus simple possible à manipuler.

Saisir les informations XML « manuellement » dans le document XML est impensable.
Nous l'avons pratiqué dans le cadre de ce travail, c'est très fastidieux, et générateur d'erreurs.

A notre sens, le recours à un système de fenêtres de saisie, donc une interface graphique est nécessaire.

Organisation de l'interface graphique.

Une interface graphique pourrait être organisée selon différents thèmes.

Interface orientée vers les données de fonctionnement de l'entreprise.

L'interface concernerait la saisie, lecture et correction/suppression des données relatives aux mesurages, imputations des pièces comptables, et données relatives aux commandes.

Interface orientée vers les données système.

Une autre interface pourrait donner accès aux données qui régissent le fonctionnement de l'application.
Il s'agit essentiellement des proportions de répartition des charges indirectes et des familles de coût, mais aussi de la gestion des comptes pcmn autorisés, clients autorisés.
L'action porte ici sur la DTD et fichier XML.

Interface orientée vers la structure du système.

Cette interface permettrait de modifier ou de créer des activités, familles de coût, mais aussi d'ajouter des éléments inexistants dans la forme actuelle.
L'action porte ici sur la DTD et fichier XML.

Interface orientée consultation / requêtes.

Une interface graphique accédant les requêtes XQuery est nécessaire également.
L'apprentissage de XQuery est long, et nécessite une parfaite connaissance de l'arbre XML de base.
A nouveau, ce n'est pas là tâche pour l'utilisateur final.

Il serait possible de proposer une série de requêtes par thèmes.
- Recherches par commande: des détails quant aux mesurages relatifs à telle commande, la part de

charges indirectes pour telle commande,..
- Requêtes de calcul: le coût d'une unité d'oeuvre, la valeur de toutes les charges indirectes sur une période, le coût d'une famille,...

Une alternative serait un canevas de requêtes « préfabriquées » qui seraient composables via des objets graphiques (cf. Stylus Studio).

D'autres solutions sont bien entendu possibles.

L'utilisateur final devrait idéalement voir de la requête:
- une fenêtre où saisir la(es) condition(s) de sélection imposée à la requête; un choix pourrait lui être proposé; par exemple un choix entre les identificateurs de commande.
- une icône « start » lançant la requête.

En retour, un affichage du résultat à l'écran ou l'impression d'un document de synthèse semblable à celui que nous créons dans le dernier chapitre sont proposés.

Autres fonctions des interfaces.

Ces différentes vues pourraient proposer un filtrage par utilisateur, par exemple pour éviter des modifications des règles de calcul par un utilisateur non-initié, ou ne pas autoriser l'accès aux requêtes de résultats à tous.

L'interface devrait avoir également quelques fonctions propres, comme la création automatique d'identificateurs uniques à chaque saisie (sans intervention de l'utilisateur).

Ou encore lorsque le choix des valeurs pour un attribut est défini dans la DTD, ces différentes possibilités pourrait être proposé à l'utilisateur par menu déroulant.

Technologie pour les interfaces.

Au terme d'une très brève recherche, il semble que Xul ou Xforms pourraient être des solutions pour les interfaces, dans l'esprit XML.
- Xul (XML-based User interface Language) créé pour le projet Mozilla, utilise un langage ayant une structure XML pour créer des interfaces graphiques.
- Xforms, une recommandation du W3C depuis 2003, qui est en constante évolution.

Les requêtes XQuery.

Nous restons avec une interrogation à propos de l'aspect « low level » de XQDT évoqué précédemment, que nous avons généralisé à XQuery , à tort ou à raison, partant du principe que XQDT est un outil de recherche XQuery.
Nous n'avons pas approfondi, la question reste ouverte.

Si comme nous le suggérions, l'aspect low level peut se traduire par la nécessité de devoir appliquer une série de requêtes successives sur un arbre XML pour obtenir un résultat plus raffiné que les données du document XML de base, il serait intéressant d'automatiser ces enchaînements de requêtes.
L'utilisateur ne devrait voir qu'une seule requête, la première, et lui demander de fournir les résultats raffinés ... de la dernière requête.

Conclusions.

Au terme de ce travail, force est de constater qu'il est possible de mener une démarche de comptabilité analytique en utilisant les technologies XML.
En dressant la liste des objectifs, nous avions un doute quant à la possibilité de transformer les données chiffrées sans avoir recours à un langage de programmation classique.
Parmi les solutions de sélection et transformation de données XML, nous avons retenu XQuery, qui s'est révélé suffisant pour mener à bien les opérations que nous souhaitions effectuer sur ces données chiffrées, et ainsi produire les résultats attendus.

Pour produire des ratios, donc raffiner l'information de base, nous avons dû créer une succession de requêtes FLWOR, peut-être est-ce là la limite de XQuery: ne pas pouvoir effectuer cette transformation en un seul traitement.

La méthode que nous avons tenté de développer permet grâce aux technologies XML un très intéressant découplage entre les données de base (le document XML) et le traitement (XQuery) des informations contenues.
Par rapport à une expérience précédente organisée autour d'un tableur , le risque de perte ou de modification accidentelle des informations de base est très réduit.
Et nos données ont l'énorme avantage de ne pas être tributaires d'un format propriétaire.

Par contre, notre objectif de réaliser ce mémoire en n'utilisant que des logiciels libres a failli dans une certaine mesure.
Pour une suite de logiciels assurant les différents traitements XML, nous avons dû opter pour le choix d'une solution payante.
Les solutions XML libres que nous avons pu trouver étaient soit trop complexes à installer sous Linux, soit très difficiles d'usage, du moins pour notre niveau de connaissance, ou encore trop peu documentées.
Ce point pourrait être retravaillé.

Ce travail n'est à notre sens pas entièrement terminé. Plusieurs points peuvent être approfondis. Citons l'ergonomie, qui devrait être améliorée par une interface graphique, ou la méthode de stockage des informations qui pourrait être plus efficiente sous forme de tables plutôt que sous forme d'un document XML unique.