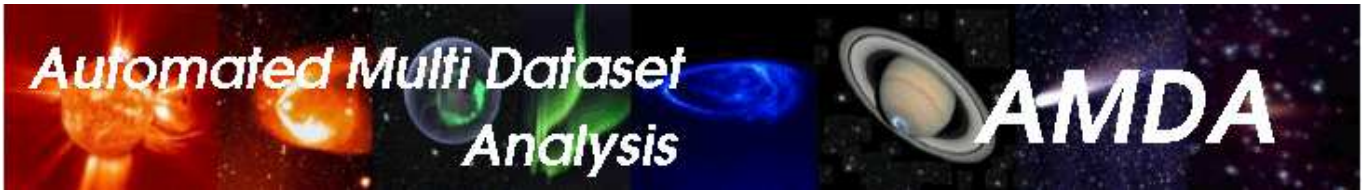
Projet : AMDA



DOSSIER
D'ARCHITECTURE DU
NOYAU D'AMDA-NG

Rédigé par : *Architecte* : *Gaëtan SCHNELLER*	Diffusé à : CNES/ IRAP
Approuvé par : Chef de projet AKKA : CASIMIR Freddy Chef de projet CNES : DUFOURG Nicolas	:

GROUPE AKKA Technologies – Pôle Informatique & Systèmes
6, rue Roger Camboulives - BP 13633 - 31036 Toulouse Cedex 1 tel : 05.34.61.92.92 fax : 05.34.61.92.22

© AKKA Technologies - Tous droits réservés

LISTE DES MODIFICATIONS DU DOCUMENT

Vers.	Date	Paragraphes modifiés	Description des modifications
0.1	21/06/2010	Tous	Création du document
01.00	19/11/2010	Tous	Finalisation du document et prise en compte des remarques du CNES
01.01	31/10/2012	§2.6 et §3.2	Prise en compte des modifications à CDPP-ST-32500-417-CES (Spécification technique industrialisation du noyau AMDA)
01.02	06/11/2012	§2.6, §6 et ANNEXES	Création de Dossier de contrôle des interfaces du noyau AMDA-NG et mise jour des logiciels réutilisés (COTS)
01.03	12/11/12	§2.3, §2.5, §2.7, §3.1, §3.2, §5.1, §6	Prise en compte des remarques de l'IRAP
02.00	16/11/12	Tous	Prise en compte des FEPS : ND1, ND6
02.01	29/11/12	Précision « liste des modifications »	Prise en compte des FEPS CNES

© AKKA Technologies - Tous droits réservés

SOMMAIRE

© AKKA Technologies - Tous droits réservés

Projet : AMDA

1 INTRODUCTION

Le Dossier d'Architecture décrit la vue technique du sous-système noyau AMDA (AMDA kernel). Il répond aux exigences techniques du système, et définit l'architecture matérielle et logicielle.

2 DESCRIPTION GÉNÉRALE

2.1 DOMAINE CONCERNE

Sur la base d'un logiciel « DD System », le CESR développe depuis 2006 une application complète d'analyse de données multi-jeux : AMDA.

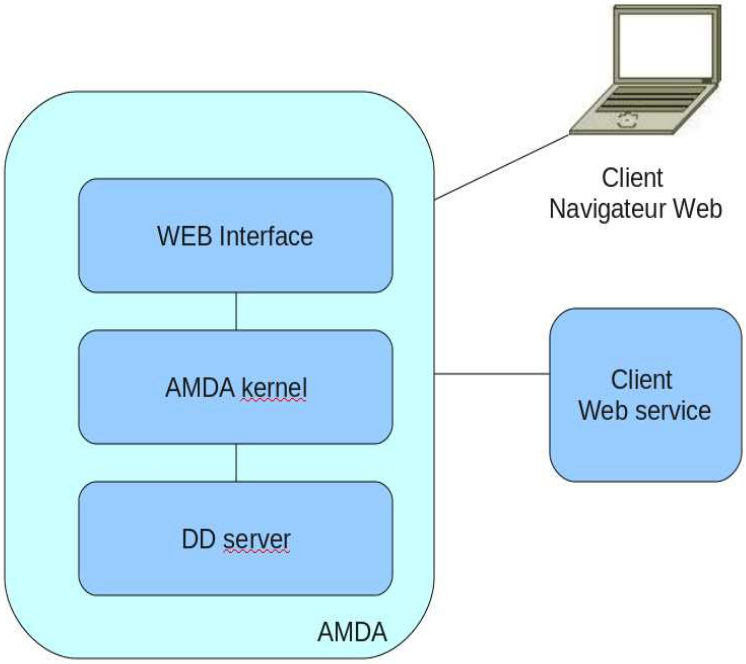
Cet outil est plébiscité par la communauté scientifique de la physique des plasmas et fait partie intégrante du CDPP. Ce succès amène tout naturellement des besoins nouveaux de fonctionnalités mais aussi de convivialité d'utilisation.

Développé depuis de nombreuses années et plus particulièrement ces deux dernières, AMDA nécessite aujourd'hui certaines améliorations de « fond » afin de lui permettre de répondre à la demande croissante des utilisateurs.

2.2 INTERFACES AVEC LES AUTRES APPLICATIONS

Le noyau AMDA est un élément dans l'architecture suivante (cf. DA-AMDA-001) :

© AKKA Technologies - Tous droits réservés Modèle DOC-FR 03.0



Le but principal du noyau AMDA est de rendre les services nécessaires au bon fonctionnement de l'interface Web AMDA.

Ses services principaux sont:

- gestion des utilisateurs (login / droit)
- gestion de l'espace utilisateur
- gestion des paramètres AMDA
- tracé des paramètres AMDA
- gestion des TimeTables
- calcul des paramètres AMDA.

Pour ce faire il peut utiliser DD server qui donne accès aux valeurs des paramètres basiques mesurés.

Ainsi deux interfaces sont définies :

- une interface avec le serveur Web d'AMDA. Celle-ci sera détaillée plus tard
- une interface avec DD server.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.3 EXIGENCES TECHNIQUES FONCTIONNELLES

Les exigences fonctionnelles découlent du dossier des cas d'utilisation d'AMDA (MU-AMDA-001), elles permettent de définir le périmètre du noyau AMDA **et ont pour vocation d'être mises à jour suite aux évolutions d'AMDA**.

Chaque exigence est identifiée par une référence sous la forme ARD_XXX_YYY où XXX est un identifiant de quelques lettres permettant de spécifier le thème, et YYY est un numéro chronologique.

2.3.1 Gestion des utilisateurs

ID exigence	*Description*	*Commentaire*
ARD_LOG_010	*L'administrateur doit pouvoir ajouter un utilisateur avec son mot de passe*	
ARD_LOG_020	*L'administrateur doit pouvoir modifier un mot de passe*	
ARD_LOG_030	*Le système doit pouvoir accepter un utilisateur connu par son login /mot de passe, on dit qu'il est identifié pour le système*	
ARD_LOG_040	*Le système doit accepter dix utilisateurs identifiés en « GUEST » au maximum en simultané*	

Toutes les exigences suivantes ne pourront être effectuées que par des utilisateurs identifiés.

2.3.2 Les workspaces

ID exigence	*Description*	*Commentaire*
ARD_WS_010	*Le système doit pouvoir retourner la liste des workspaces d'un utilisateur*	
ARD_WS_020	*Le système doit pouvoir retourner la description (arbre affiché) d'un workspace utilisateur ou de l'espace partagé*	*Cet arbre sera retourné en sous forme XML.*
ARD_WS_030	*Le système doit pouvoir sauvegarder la description (arbre affiché) d'un workspace pour un utilisateur ou pour l'espace partagé*	
ARD_WS_040	*Le système doit pouvoir copier une sous-partie d'un workspace dans un autre workspace pour un utilisateur ou pour l'espace partagé*	
ARD_WS_050	*Le système doit pouvoir supprimer une sous-partie d'un workspace pour un utilisateur ou pour l'espace partagé*	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ID exigence	Description	Commentaire
ARD_WS_060	*Le système doit pouvoir sauvegarder un élément d'un workspace (par exemple la description d'un paramètre composé, la description time table, etc.) pour un utilisateur*	
ARD_WS_070	*Le système doit pouvoir retourner un élément d'un workspace (par exemple la description d'un paramètre composé, la description time table, etc.) d'un utilisateur ou de l'espace partagé*	
ARD_WS_080	*Le système doit pouvoir organiser son workspace ou l'espace partagé sous forme d'arbre i.e. création, renommage, suppression de répertoire ou d'élément du workspace*	

2.3.3 Les paramètres

Un paramètre AMDA représente une donnée physique accessible via DD server.

Ils se présentent sous forme d'un identifiant unique (path) et de valeurs réelles associées au temps.

Ces valeurs peuvent être de deux formes : soit scalaire, soit sous la forme d'un tableau à une dimension.

ID exigence	Description	Commentaire
ARD_PAR_010	*Le système doit pouvoir retourner l'arbre des paramètres basiques*	*Cet arbre sera retourné sous forme XML.*
ARD_PAR_020	*Le système doit pouvoir valider un paramètre composé. C'est à dire valider que la formule décrivant la composition du paramètre a une syntaxe correcte.*	
ARD_PAR_030	*Le système doit pouvoir définir un alias sur un paramètre et le sauvegarder dans l'espace utilisateur*	
ARD_PAR_040	*Le système doit gérer une description d'un paramètre composé comme un élément du workspace*	

2.3.4 Les time tables

ID exigence	Description	Commentaire

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ID exigence	Description	Commentaire
ARD_TT_010	*Le système doit pouvoir fusionner des intervalles définis d'une time table*	
ARD_TT_020	*Le système doit pouvoir trier des intervalles définis d'une time table*	
ARD_TT_030	*Le système doit pouvoir faire l'intersection de deux time tables*	
ARD_TT_040	*Le système doit pouvoir faire une opération de décalage dans une time table*	
ARD_TT_050	*Le système doit pouvoir filtrer une time table selon une condition*	
ARD_TT_060	*Le système doit gérer une description d'une time table comme un élément du workspace*	

2.3.5 Les conditions (création automatique d'une time table)

ID exigence	Description	Commentaire

ARD_COND_010	*Le système doit pouvoir rechercher des intervalles de temps (timeTable) suivant une condition. La condition est donnée par une formule respectant la syntaxe présentée.*
	Le résultat de la recherche est :
	- *une time table créée et sauvegardée dans l'espace utilisateur,*
	- *deux fichiers :*
	  - *l'un contenant les données de l'intervalle répondant à la condition,*
	  - *l'autre contenant les données ne répondant pas à la condition.*
	Une description de ce résultat est retournée.	

ARD_COND_020	*Le système doit gérer une description d'une condition comme un élément du workspace*	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.3.6 Le tracé

ID exigence	Description	Commentaire
ARD_PLOT_010	*Le système doit pouvoir* - *tracer un graphique représentant les variations de paramètres sous forme de fichier PNG PDF ou POSTSCRIPT gzippé* - *retourner le fichier.*	
ARD_PLOT_020	*Le système doit pouvoir choisir automatiquement un type de tracé (spectrogramme, série, ...) pour chaque paramètre suivant un fichier de configuration*	
ARD_PLOT_030	*Le système doit pouvoir modifier son tracé via les attributs des paramètres*	
ARD_PLOT_040	*Le système doit gérer une description de tracé comme un élément du workspace*	

Remarque: Le zoom et le déplacement d'un tracé sont de la responsabilité du serverWeb en utilisant le service de tracé existant.

2.3.7 Le téléchargement

ID exigence	Description	Commentaire
ARD_LOAD_010	*Le système doit pouvoir créer un fichier contenant les données d'un paramètre composé, pour une time table donnée, dans le but d'un téléchargement. Il s'agit d'un fichier texte. La partie Web se charge de faire soit un fichier HTML, soit un fichier à télécharger*	*TBC*
ARD_LOAD_020	*Le système doit gérer une description de requête de téléchargement comme un élément du workspace*	

2.3.8 Les centre de données

ID exigence	Description	Commentaire
ARD_CDATA_010	*Le système doit pouvoir ajouter un centre de données*	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.3.9 Les catalogues

ID exigences	Description	Commentaire
ARD_CAT_010	*Le système doit gérer les catalogues comme un élément du workspace*	

2.4 EXIGENCES TECHNIQUES SYSTEME

2.4.1 Plugin

ID exigence	Description	Commentaire
ARD_PLUG_010	*Le système doit pouvoir accepter des « plugins » via un fichier de configuration*	
ARD_PLUG_020	*Le système doit pouvoir accepter une autre forme de tracé via un « plugin ».*	
ARD_PLUG_030	*Le système doit pouvoir accepter d'autres fonctions de calcul via un « plugin ».*	

2.4.2 Robustesse

ID exigence	Description	Commentaire
ARD_ROB_010	*Tolérance aux pannes: un « plantage » doit impacter de manière minimale un utilisateur en terme de perte de données et de paramétrage.*	

2.4.3 Plateforme

ID exigence	Description	Commentaire
ARD_PLA_010	*L'ensemble d'AMDA-NG doit fonctionner sous Linux mais être indépendant de la distribution de Linux.*	

2.4.4 Niveau de service requis

ID exigence	Titre	Description	Commentaire

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

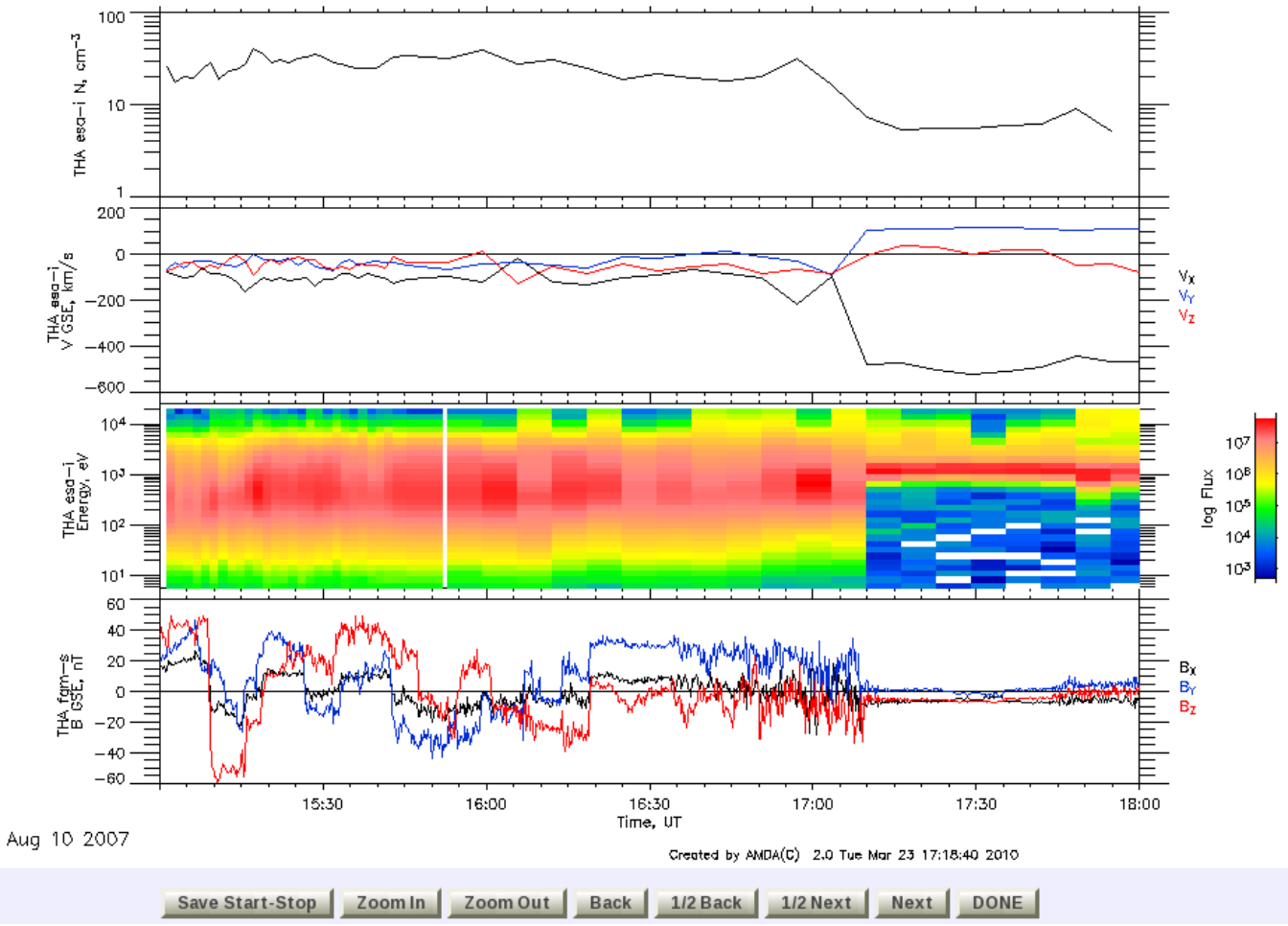
ID exigence	Titre	Description	Commentaire
ARD_NDR_010	*taux de disponibilité de 100%*	*Il ne doit pas y avoir d'arrêt de service (service 7j/7, 24h/24)*	*Origine : Observatoire virtuel*
ARD_NDR_020	*Temps de maintenance maximal autorisé*	*TBD*	

2.5 DESCRIPTION DES FORMATS DE DONNEES UTILISES

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.5.1 Description des possibilités graphiques

Ces informations ne sont données qu'à titre indicatif, les possibilités du noyau d'AMDA ne sont pas encore figées.



Remarque : formats possibles des fichiers : PNG, PDF et PostScript

2.5.1.1 Choix du graphique et des attributs possibles en fonction du paramètre à tracer

Les types de graphique, couleur de tracé, troncature des valeurs par min, max, titre du graphique, (liste non exhaustive d'attribut) sont paramétrables de plusieurs manières :

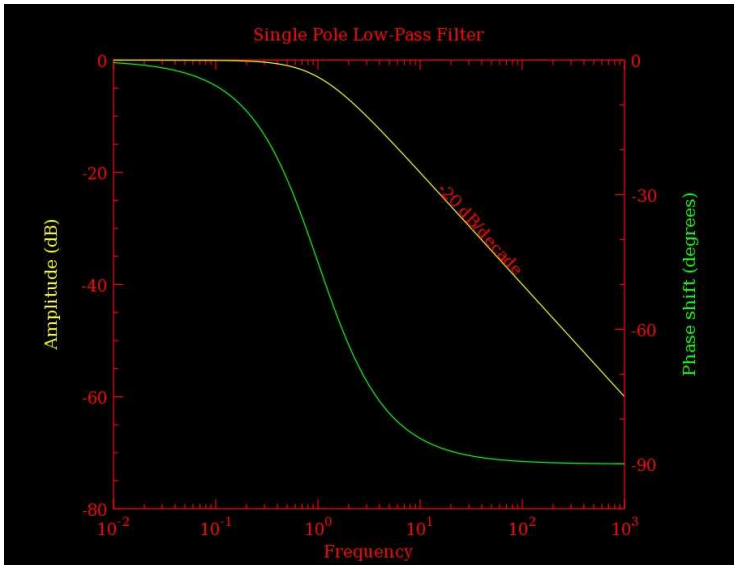
- par l'utilisateur grâce à l'IHM : attributs dans la commande en entrée
- par fichier de configuration XML : fichier XML décrivant les spécificités des paramètres basiques
- par défaut suivant le type de donnée : courbe simple par coordonnées du paramètre

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.5.1.2 Courbe simple

Le but de ce type de tracé est de représenter les valeurs d'un paramètre en fonction du temps.

Exemple fourni avec la bibliothèque plplot x29c:



2.5.1.3 Spectrogramme

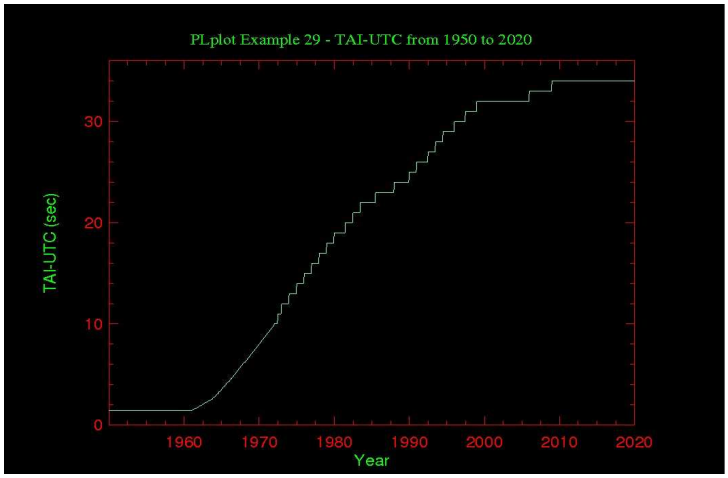
Le but de ce type de tracé est d'utiliser la couleur pour représenter une coordonnée. Pour chaque temps, une coordonnée est utilisée en ordonnée et une autre en dégradé de couleur.

2.5.1.4 Histogramme

Le but de ce type de tracé est de représenter les valeurs d'un paramètre en fonction du temps mais de ne pas relier les points de façon continue. La valeur est représentée par un trait horizontal jusqu'au point suivant.

Exemple fourni avec la bibliothèque plplot x29c:

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

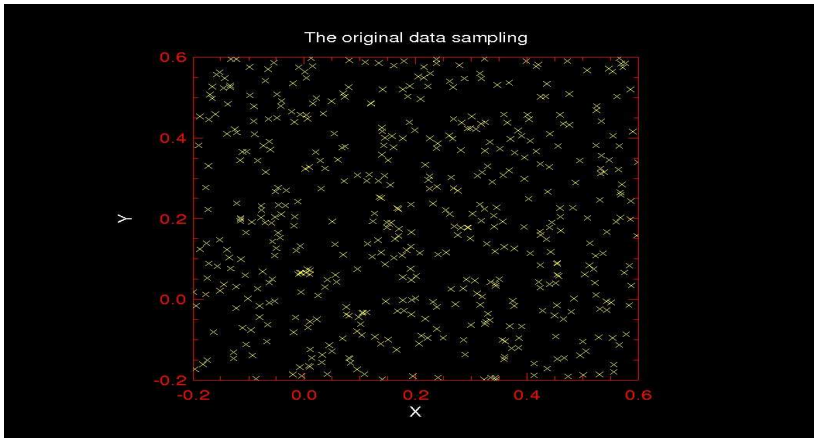


2.5.1.5 Scater / cylindrique

Le but de ce type de tracé est de représenter les valeurs d'une coordonnée d'un paramètre en fonction d'une autre coordonnée. La représentation est dite cylindrique quand l'abscisse est un rayon. Il est possible de rajouter des courbes prédéfinies représentant des frontières.

2.5.1.6 Nuage de points (y =f(x))

Exemple fourni avec la bibliothèque plplot x21c:



2.6 INTERFACES EXTERNES

Les interfaces externes d'AMDA ne sont pas décrites ici, mais pour rappel il y a trois interfaces externes :

- une interface utilisateur (IHM) via un navigateur Web. Elle est implémentée en javascript avec le COTS EXT-JS.
- une interface Web-service implémentée par le noyau AMDA-NG.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

- une interface avec DD serveur implémentée par le noyau AMDA-NG.

Les interfaces externes du noyau AMDA sont décrites dans le Document [A1]. Ne sont décrites ici que les interfaces prévisionnelles du noyau AMDA.

2.6.1 Interfaces prévisionnelles

2.6.1.1 Interface avec le serveur WEB

Le noyau AMDA doit communiquer avec le serveur Web. Cette communication sera gérée par le composant KernelService. Cf §3.2.1.1 : Composant KernelServices.

Cette communication a comme contrainte :

- Pas de mémorisation avec une requête précédente : toutes les informations utiles à la requête doivent être fournies dans le message déclenchant la requête
- Les données retournées peuvent être sous forme binaire (pour les plots, download zip) ou texte (cas général).

Étant donné que le choix du format XML a été fait dans la version précédente d'AMDA, ce format permet une évolution simple en respectant la compatibilité ascendante. Ce format est retenu pour tous les échanges de données autre que binaire.

Les données binaires seront transférées par dépôt du fichier sur un répertoire partagé, visible par NFS des deux serveurs (serveur Kernel AMDA et server Web). La politique de persistance de ces fichiers est la suivante: suppression du fichier par le consommateur. De plus, une surveillance de ces fichiers doit être réalisée pour supprimer tous les fichiers ayant une date de dernière mise à jour de plus de 24h.

Pour des raisons de performance et de simplification des processus, les données des workspaces utilisateur devront être visibles des deux serveurs (serveur Kernel AMDA et server Web). Ainsi une politique stricte d'accès à ces données doit être définie pour éviter tout conflit. Ainsi, la lecture d'un item du workspace doit être effectuée une seule fois pendant l'exécution d'un processus (un service) car il peut changer à tout moment. De plus, il ne doit pas y avoir de conflit entre une lecture et une écriture.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.6.1.2 Interface Web-service

AMDA-NG doit fournir sous forme de Web-service soap les services suivants :

Titre	Description	Commentaire
Search	Service équivalent à la fonction search d'AMDA permettant à partir d'une expression et des informations de start-stop de récupérer la description d'une timeTable	La recherche avec une timeTable en paramètre en lieu et place des informations de start-stop n'est pas demandée
Plot	Service équivalent à la fonction plot	facultatif
Download	Service équivalent à la fonction download	facultatif

Si un besoin de Web-service REST devient nécessaire, c'est le serveur web (par la couche java) qui exposera ces services.

2.6.1.3 Interface d'ajout/suppression de plugin

L'application a besoin d'un mécanisme de plugin pour évoluer de façon souple comme suit :

- Il doit permettre d'ajouter un nouveau moyen de récupérer les valeurs d'un paramètre, c'est-à-dire de proposer une nouvelle implémentation de l'interface ParamGet
- Il doit permettre d'ajouter un nouveau format de sortie des données d'un paramètre, c'est-à-dire de proposer une nouvelle implémentation de l'interface ParamOutput
- Il doit permettre d'ajouter un moyen d'enrichir les possibilités de calcul.

Ceci est fait avec la contrainte suivante : les ajouts doivent s'effectuer sans redémarrer le serveur. Ceci induit un chargement/déchargement à chaud. En effet, dans le cas d'une mise à jour d'une version de plugin, il est nécessaire de supprimer les fonctionnalités (déchargement) pour pouvoir les ajouter à nouveau (chargement)

Nous proposons ci-dessous une introduction au mécanisme que nous mettrons en œuvre.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

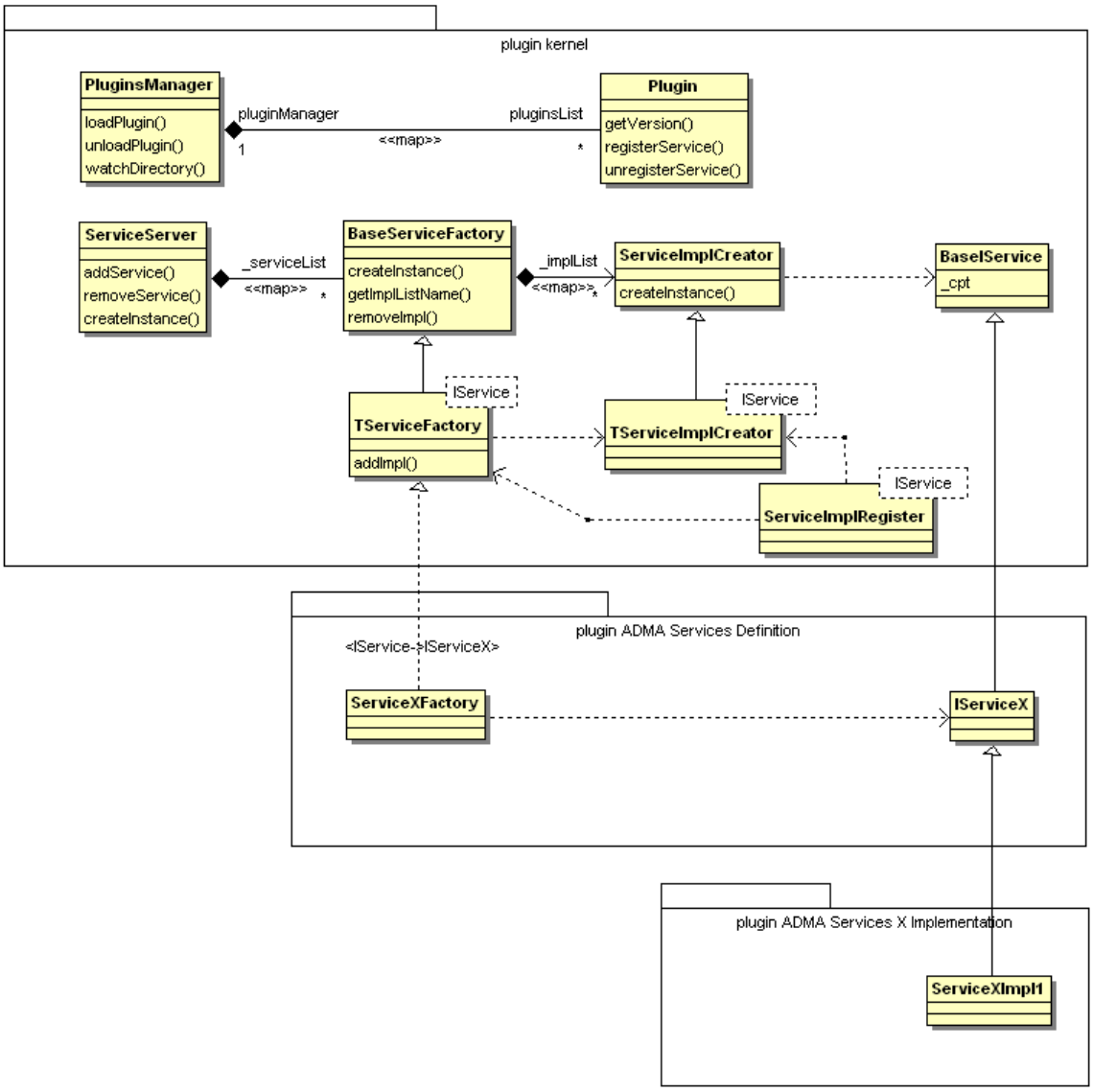


Figure 1 Plugin principe

- Plugin
  - Le mécanisme de plugin sera unique dans l'application,
  - Chaque plugin consistera en une librairie dynamique (.so) comportant trois fonctions C : l'une permettra d'enregistrer des interfaces de services ou/et des implémentations de services ; une autre permettra de retirer ces interfaces ou/et implémentations avant déchargement du plugin ;la troisième donnera la version du plugin.
  - C'est le PluginsManager qui sera responsable du chargement et déchargement des plugins. Ceci se fera via un système de surveillance d'un répertoire de librairie dynamique ou de manière programmée : il chargera et déchargera à chaud les librairies et les associera à un objet Plugin. De cet objet nous pourrons appeler les trois fonctions précitées.
  - En vue de devancer la problématique multithread, nous appliquerons le design pattern « Active Object » sur les méthodes des classes de notre mécanisme de plugin (PluginsManager, ServiceServer …). C'est-à-dire que les méthodes de ces classes ne devront être exécutées que par un seul et même thread.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

- Les services
  - Un unique serveur de services ServiceServer contiendra des BaseServiceFactory.
  - Déclarer un service se résumera à créer et enregistrer une fabrique TServiceFactory, d'une interface IServiceX que l'on aura pris soin de définir.
  - Implémenter un service se résumera à enregistrer une classe ServiceXImp, qui implémentera le IServiceX, via une macro utilisant ServiceImplRegister, dans la fabrique du service concerné (TServiceFactory< IServiceX>).
  - Pour exécuter un service :
    - appeler la méthode « get » de IServiceServer en lui passant le nom du service,
    - à partir de la liste d'implémentation obtenue, choisir l'implémentation souhaitée,
    - demander la création du Service. A cette dernière étape nous obtiendrons un pointeur intelligent sur le Service. Le serveur de services conservera une référence sur ce pointeur pour consulter le compteur de référence qui sera un des critères dans la possibilité de décharger un plugin.

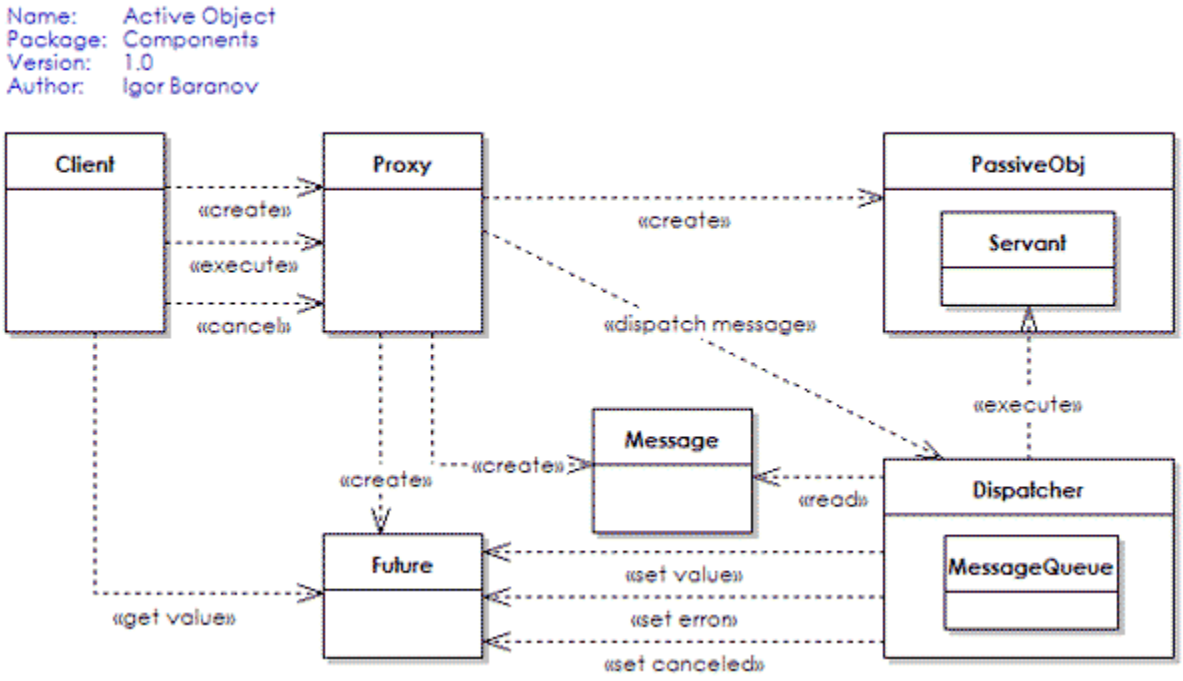
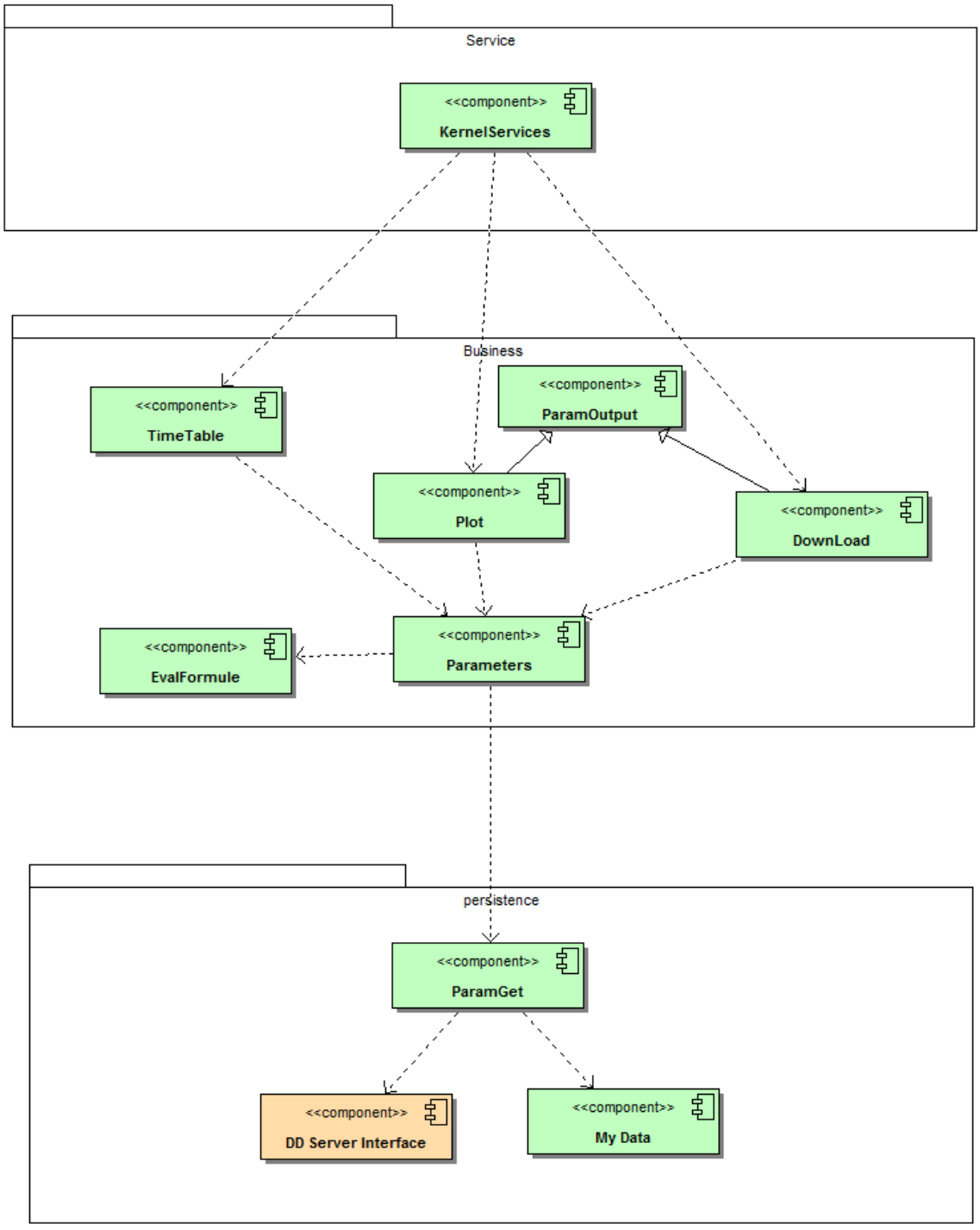


Figure 2 Active Object design pattern

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

2.7 DIAGRAMME DE COMPOSANTS

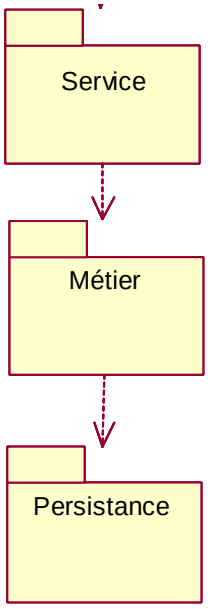
Le diagramme ci-dessous n'est donné qu'à titre indicatif, il pourra évoluer en cours de conception.



© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

3 ARCHITECTURE LOGICIELLE

3.1 MODELE DES COUCHES LOGICIELLES



La couche **« Service »** va contenir tous les objets génériques tant en terme d'architecture, que technique, voire même métier. Cette couche contiendra aussi les « framework » génériques. Dans ce cas aussi, il pourra s'agir de frameworks métier et/ou architecturaux. D'une manière plus générale cette couche va fournir à la couche «Application» une interface pour les «cas d'utilisation» métiers nécessaires à l'application. On notera cependant que ces services peuvent être utilisés par une ou plusieurs applications. Enfin cette couche «Service» a pour rôle de convertir les objets de la couche «Métier» en objets distribuables pour les Applications clientes.

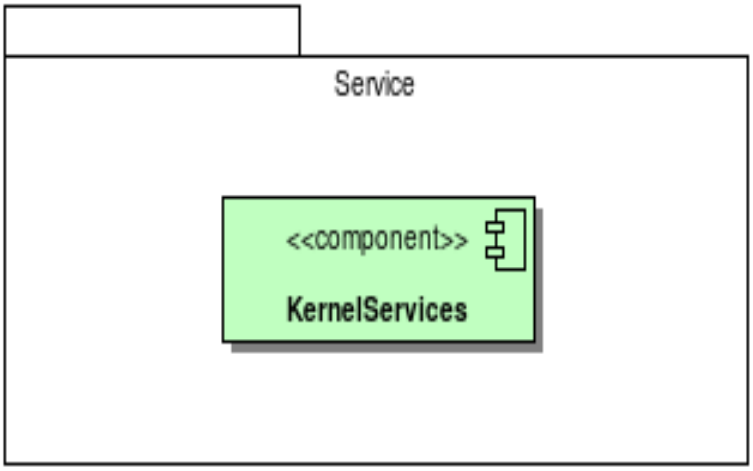
La couche **« Métier »** contient tous les objets métier. On trouvera aussi dans cette couche l'implémentation des règles de gestion définies dans le modèle objet de l'application. On trouvera de ce fait dans cette couche les vérifications sémantiques des informations issues des couches supérieures à l'aide de la logique métier définie par les règles de gestion.

La couche **« Persistance »** va permettre de rendre persistantes les données qui en ont besoin. On trouvera dans cette couche les services de base liés à la persistance (Création, Lecture, Mise à jour, Suppression) ainsi que la logique de transformation Objet-Relationnel dans le cas d'une persistance implémentée dans une base de données relationnelle.

3.2 DESCRIPTION DES COUCHES LOGICIELLES

3.2.1 Couche Service

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0



3.2.1.1 Composant KernelServices

Ce composant a pour but d'unifier l'accès à tous les services du noyau AMDA-NG. Il sera considéré comme un point d'entrée d'une architecture orientée service.

Le service doit :

- offrir un ensemble d'opérations dont les interfaces sont décrites,
- être autonome (disposer de toutes les informations nécessaires à son exécution : pas de notion d'état),
- respecter un ensemble de contrats (règles de fonctionnement),
- correspondre aux processus métier et fonctions.

Les solutions envisagées sont:

- Web Service : en s'appuyant sur les outils gSOAP: http://gsoap2.sourceforge.net/ ou Apache Axis2/C : http://ws.apache.org/axis2/c/docs/axis2c_manual.html,
- Scripts et exécutables avec une interface par fichier,
- Directement par TCP/IP avec un formalisme d'échange à définir.

Méthode	***Avantage***	***Inconvénient***

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

Méthode	*Avantage*	*Inconvénient*
WebService	• l'utilisation des services par une application tiers sera fortement simplifiée, • ne dépendra pas du langage de programmation utilisé, • gestion multi-user (par multithread) pris en charge par le COTS utilisé • aura un protocole simple référence, • évite les problèmes de proxys et pare-feu • sécurisable par ssl (https) • load balancing sur différentes machines envisageable à peu de frais	• un seul exécutable sous forme de serveur gérera l'ensemble des utilisateurs. S'il y a un « plantage », ce sont tous les utilisateurs connectés à cet instant qui n'auront pas la réponse à leur requête.
Scripte (ksh)	• simplicité de mise en œuvre	• gestion des erreurs plus difficile • l'utilisation par une application tiers nécessite d'être localisé sur la machine du serveur • gestion des ressources partagées entre les utilisateurs (en particulier l'espace partagé commun) plus difficile à gérer
Direct TCP/IP	• pas d'utilisation de COTS • rapide et optimisable à souhait. • load balancing	• formalisme d'échange à définir mais simple client/serveur échangeant des données (par exemple requête XML et retour en XML plus données binaire au format connu (*PNG, PDF ou POSTSCRIPT gzippé)*, si indiqué dans le XML. • difficilement récupérable par une autre application cliente (« protocole maison ») • gestion multi-user « multithread » à implémenter

L'inconvénient de la première solution (Web service) peut être contourné par la solution suivante:

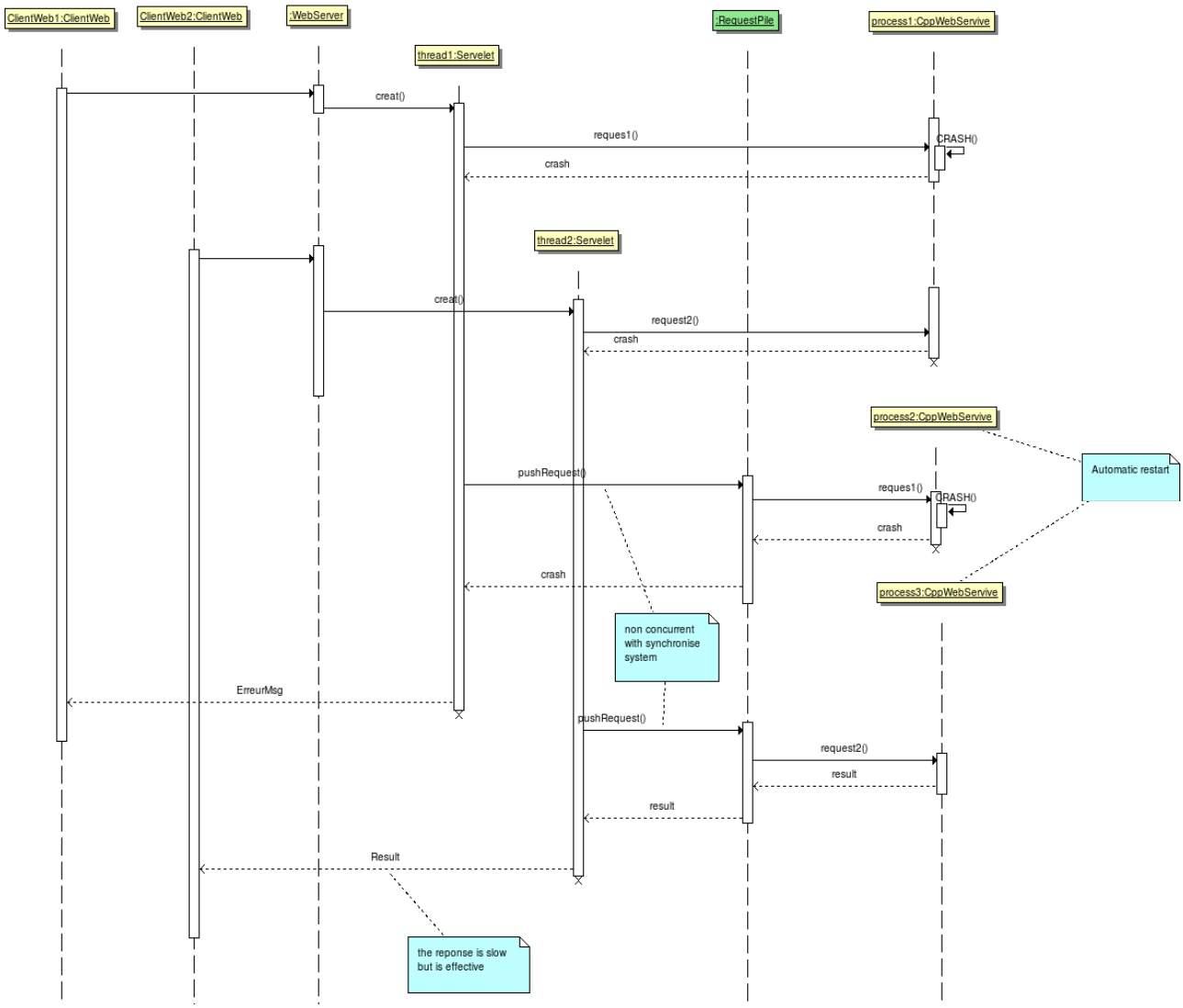
Quand le serverWeb (servlet) fait une requête au WebService, si celui-ci « plante », il demande à un objet commun à tous les threads du serverWeb de faire la requête pour lui. Ainsi si plusieurs requêtes sont en cours au moment

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

du plantage, elles se retrouvent empilées et exécutées les unes à la suite des autres (en liste FIFO). Le serverWeb peut ainsi détecter la requête faisant « planter » le serverWeb et retourner une erreur au client de la dite requête. De plus, les autres requêtes en cours auront une réponse. Le seul inconvénient est que la réponse sera plus lente qu'à l'accoutumée.

Il est bien entendu que le processus gérant le WebService doit se relancer automatiquement en cas de « plantage » avec une solution du type: « while (1) exec(ProcessWebService) ».

Ce scénario est décrit dans le diagramme de séquences suivant :



Le choix se porte donc sur le Web-Service.

Ainsi il sera simple de rendre disponible le ou les web-service(s) demandé(s) à AMDA-NG voir §2.6.1.1

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

Une comparaison d'Axis2/C et gsoap est effectuée en annexe « choix du COTS de Axis2/C et gsoap ». En résumé, le choix se porte sur gsoap car il n'y a pas besoin d'accéder au Web service de type REST (méthode GET de HTTP) et que gsoap est mieux documenté et permet de faire un serveur « stand-alone ».

Description du format de données: SOAP (WSDL)

Cette description sera fournie dans un document annexe. Elle ne pourra être détaillée que pendant la phase de design car elle demande une connaissance précise de la spécification fonctionnelle d'AMDA.

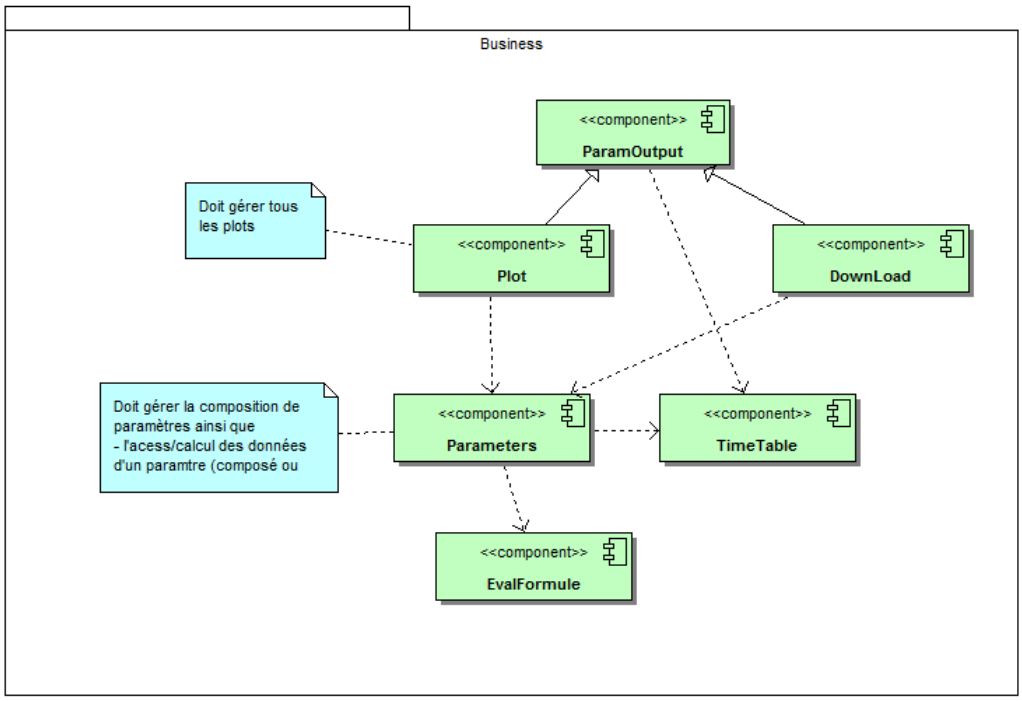
Un fichier WSDL, avec les commentaires renseignés obligatoirement, pourra faire office de document de description d'interface.

Cette couche aura aussi pour but de remonter d'éventuelles erreurs fonctionnelles ou techniques.

Dans le but de garder une indépendance entre la partie Vue (affichage sur le navigateur) et la partie modèle (noyau AMDA), la remontée des erreurs se fera via des codes d'erreurs auxquels pourront être ajoutés des paramètres indépendants de toute forme d'affichage (formatage, langue, …)

3.2.2 Couche Métier

Le diagramme ci-dessous n'est là qu'à titre indicatif, il pourra évoluer.



© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

3.2.2.1 Composant Parameters

Ce composant a pour but :

- de gérer
    - la construction de paramètres composés
    - l'accès aux données des paramètres basiques via DD server.
- de déléguer
    - la validation de la formule de composition au composant EvalFormule
    - le calcul des valeurs du paramètre composé au module EvalFormule.

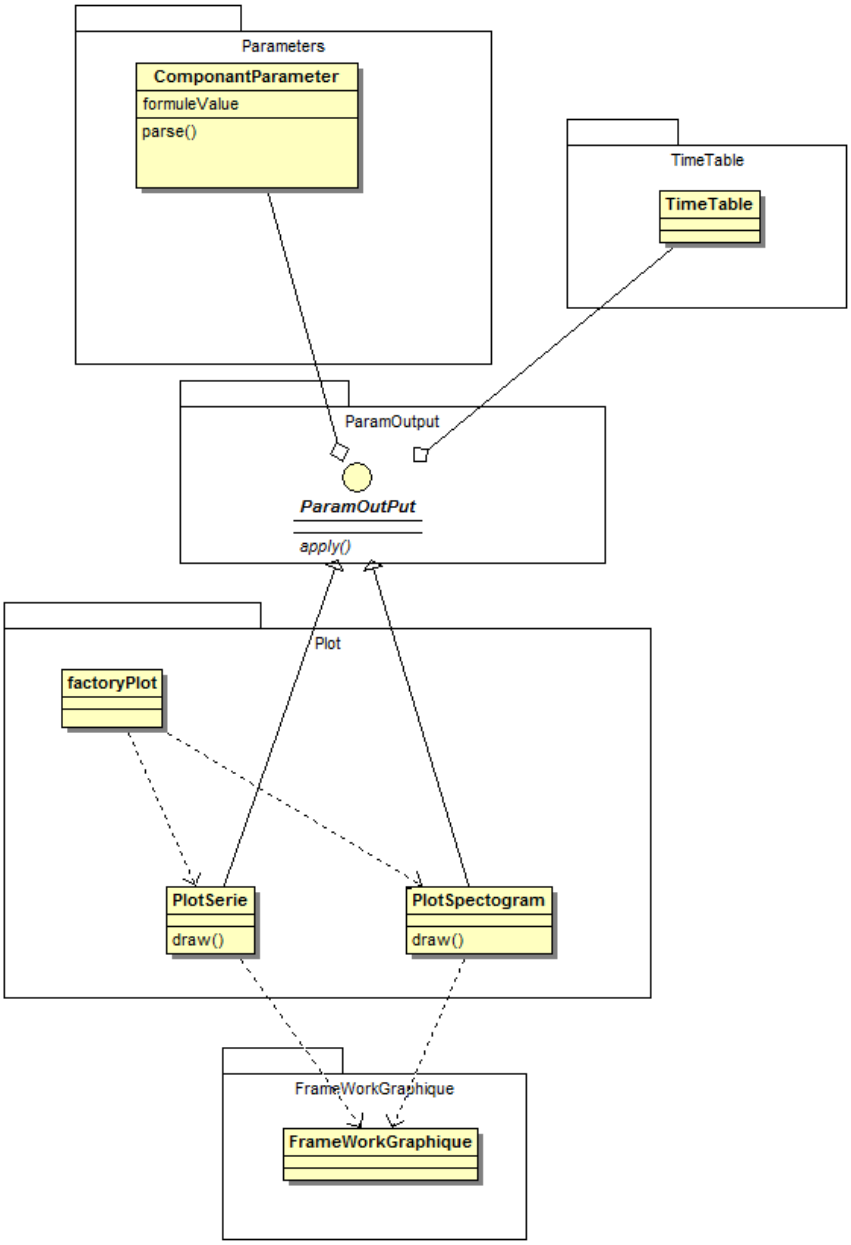
Une description détaillée du composant Parameters est donnée dans le dossier de Conception du noyau d'AMDA [R3].

3.2.2.2 Composant Plot

Ce composant est une implémentation de l'interface ParamOutput, il a pour but de gérer les fonctions de tracé.

Le diagramme UML de classe suivant explicite les interfaces du composant Plot avec les autres composants, il n'est donné qu'à titre indicatif, il pourra être amené à évoluer :

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0



L'utilisation d'un composant pris sur étagère (ou COTS) est envisagé pour le tracé.

Ce COTS doit :

- pouvoir s'exécuter sur une station Linux
- être capable de tracer des spectrogrammes comme spécifié au « 2.5.1 Description des possibilités graphiques »
- être capable de tracer des séries comme spécifié au « 2.5.1 Description des possibilités graphiques »

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

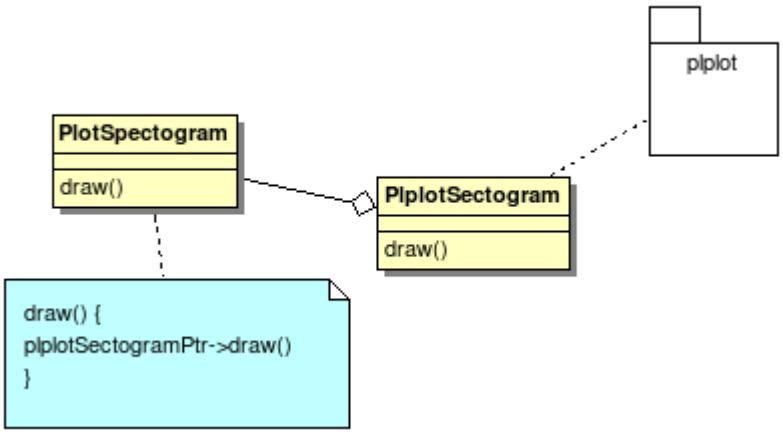
- avoir une API accessible via C++
- fournir des tracés au format PostScript ou PNG ou PDF.

Les COTS suivants sont évalués :

- Gnuplot: http://www.gnuplot.info/
- Plplot: http://plplot.sourceforge.net/

Après analyse (voir ANNEXE - A) plplot, est choisi principalement pour sa compatibilité avec le C++. Ce choix n'est toujours pas arrêté, il peut être remis en cause par les évolutions des besoins graphiques.

Le module Plot devra diminuer au maximum ces dépendances vis à vis de plplot. Ainsi chaque classe spécialisée dans un plot agrègera une classe dédiée à plplot.



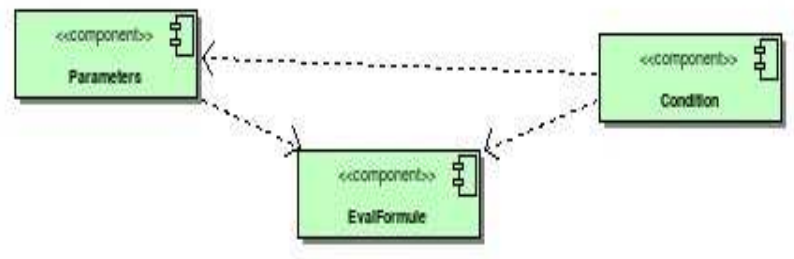
3.2.2.3 Composant EvalFormule

Ce composant a pour but de gérer :

- La validation de la formule de composition
- Le calcul des valeurs résultant de la formule de composition

Le diagramme UML de classe suivant explicite les interfaces du composant EvalFormule avec les autres composants.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0



L'utilisation d'un COTS est envisagée pour le parsing des formules de composition de paramètres et des formules de condition.

Ce COTS doit :

- pouvoir s'exécuter sur une station Linux
- être capable d'évaluer des formules mathématiques
- avoir une API accessible via C++.

Les COTS suivants sont évalués :

- Flex/bison (version gnu de lex/yacc): http://flex.sourceforge.net/ et http://www.gnu.org/software/bison/
- Boost::spirit: http://boost-spirit.com/home/

Boost a notre préférence car il ne demande pas l'apprentissage d'un nouveau langage (flex et bison), il offre par défaut les mécanismes d'analyse lexicale utiles pour le projet, comme la reconnaissance d'un réel.

La licence Boost (Cf. « ANNEXE - D Licence BOOST ») encourage une utilisation commerciale et non commerciale.
Les bibliothèques Boost ont pour objectif d'établir «la pratique existante» et fournir des implémentations de référence afin d'être adaptées à la normalisation éventuelle. Dix bibliothèques Boost sont déjà incluses dans le rapport du « C++ Standards Committee's Library Technical Report (TR1 voir : http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2005/n1745.pdf) » et seront dans le nouveau « C++0x Standard » en cours de finalisation.

C++0x comprendra également plusieurs autres bibliothèques Boost en plus de celles de TR1. D'autres bibliothèques Boost sont proposées pour TR2 (voir : http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2005/n1810.html).

Boost est très répandu, il est possible de consulter la page http://www.boost.org/users/uses_shrink.html permettant de voir tous les utilisateurs des librairies Boost.

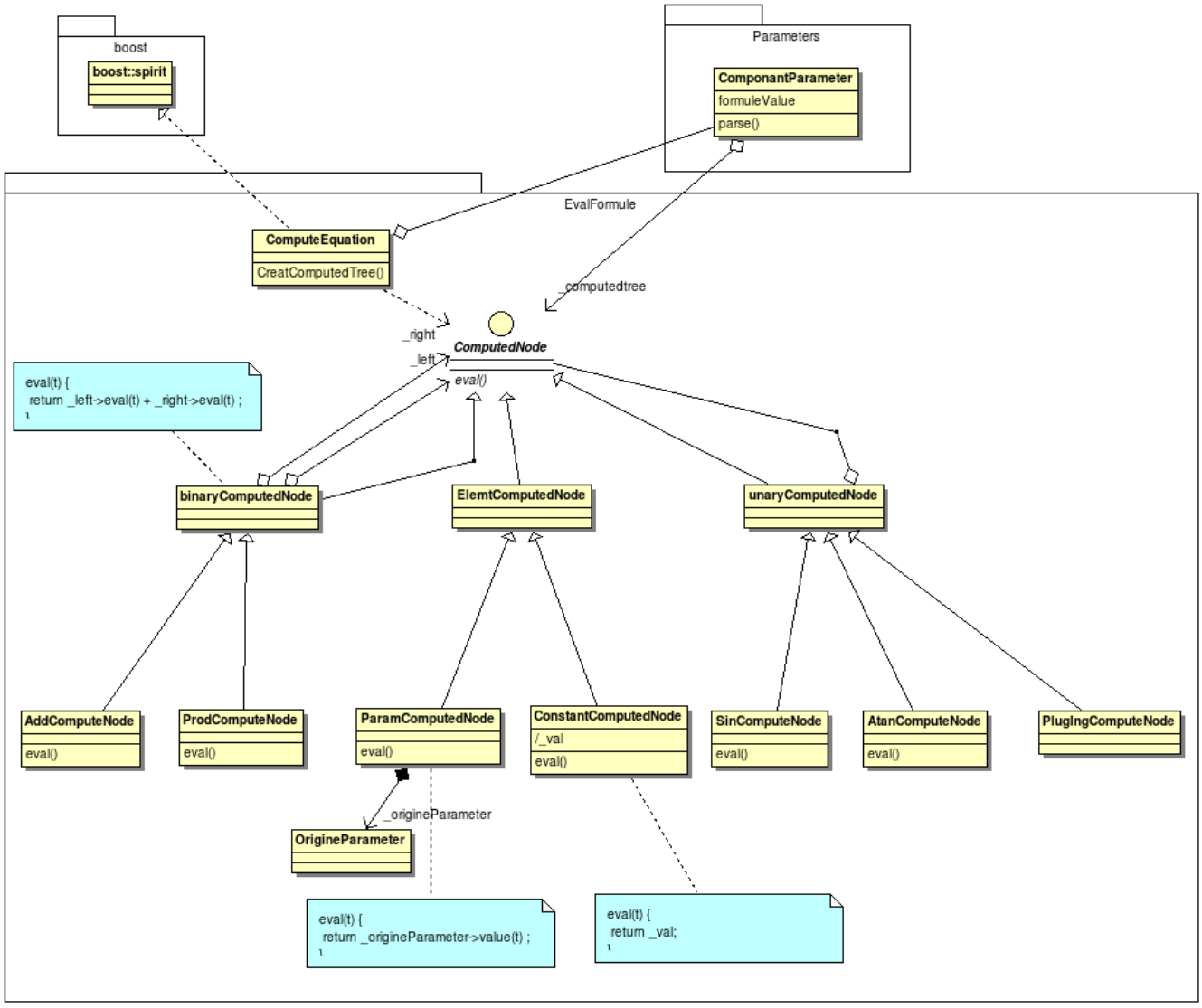
© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

Deux solutions sont envisagées pour évaluer les formules :

- évaluation par parcourt d'arbre en mémoire
- évaluation par code natif

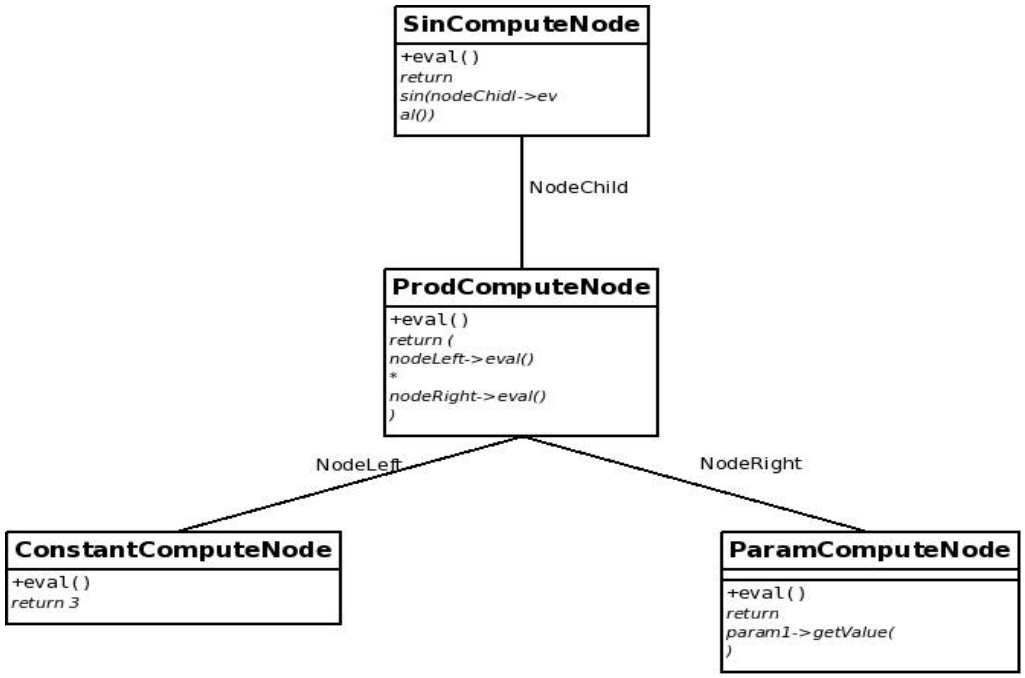
3.2.2.3.1 Evaluation par parcours d'arbre

Avec le COTS boost ::spirit : transformation d'une formule sous forme de chaîne de caractères en un arbre de computedNode.



Ainsi la formule sin(3 * Param1) générera l'arbre suivant:

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0



Le calcul s'effectue en appelant la méthode eval de l'instance de la class SinComputeNode .Cette instance évalue son « fils » : produit de 3 par param1. Enfin le calcul de sinus est effectué.

Nous avons ainsi le moyen d'effectuer les calculs très rapidement grâce à cet arbre qui est stocké en mémoire.

Pour les quatre cas particuliers des fonctions AMDA, voici le comportement proposé :

- **sans argument**
  deriv_(child) – eval(t) va utiliser les valeurs du child à t-1, t et t+1
- **avec arguments**
  - Fonctions travaillant sur une fenêtre de temps nommé T: à la première évaluation d'une de ces fonctions pour un temps entre nT et (n+1)T le calcul est effectué en parcourant l'intervalle de temps sur le nœud child puis le résultat retourné est mémorisé pour que le prochain appel retourne la valeur de manière instantanée.
  - Fonctions Shift: après évaluation de la valeur de shift en seconde (T), l'évaluation de la fonction est effectuée en utilisant le décalage de temps child->eval(T+s)
- **pseudo-vecteur**
  - Le calcul est effectué sur le vecteur, la coordonnée demandée est retournée

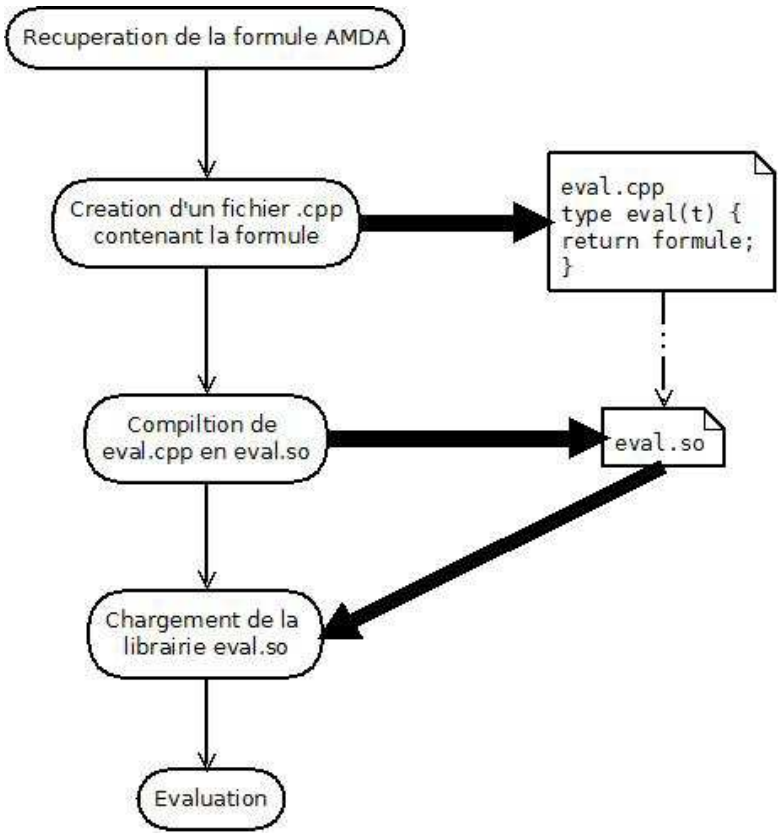
3.2.2.3.2 Evaluation par code natif

Avec le COTS boost ::spirit :

- transformation d'une formule au format AMDA en une chaîne de caractères au format C++.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

- génération d'un fichier temporaire contenant une fonction de calcul dont le code est la chaîne générée par Boost ::spirit.
- compilation de ce fichier sous forme d'une librairie dynamique, chargée en temps réel par le serveur.
- évaluation de cette fonction pour chaque pas de temps.



3.2.2.3.3 Choix du moyen d'évaluation

La méthode d'évaluation par code natif représente l'avantage certain d'être plus rapide. Elle a en revanche quelques inconvénients :

- La gestion des erreurs (division par zéro, racine carré d'un nombre négatif …) est plus difficile : utilisation du mécanisme de floating-point exception (voir `feclearexcept` et `fetestexcept` de l'include `fenv`).
- L'écriture du code C++ ne doit pas générer d'erreur car elle sera très difficilement compréhensible pour un utilisateur final.

3.2.2.1 Composant TimeTable

Ce composant a pour but de gérer les Time Tables :

- fusion de TimeTable

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

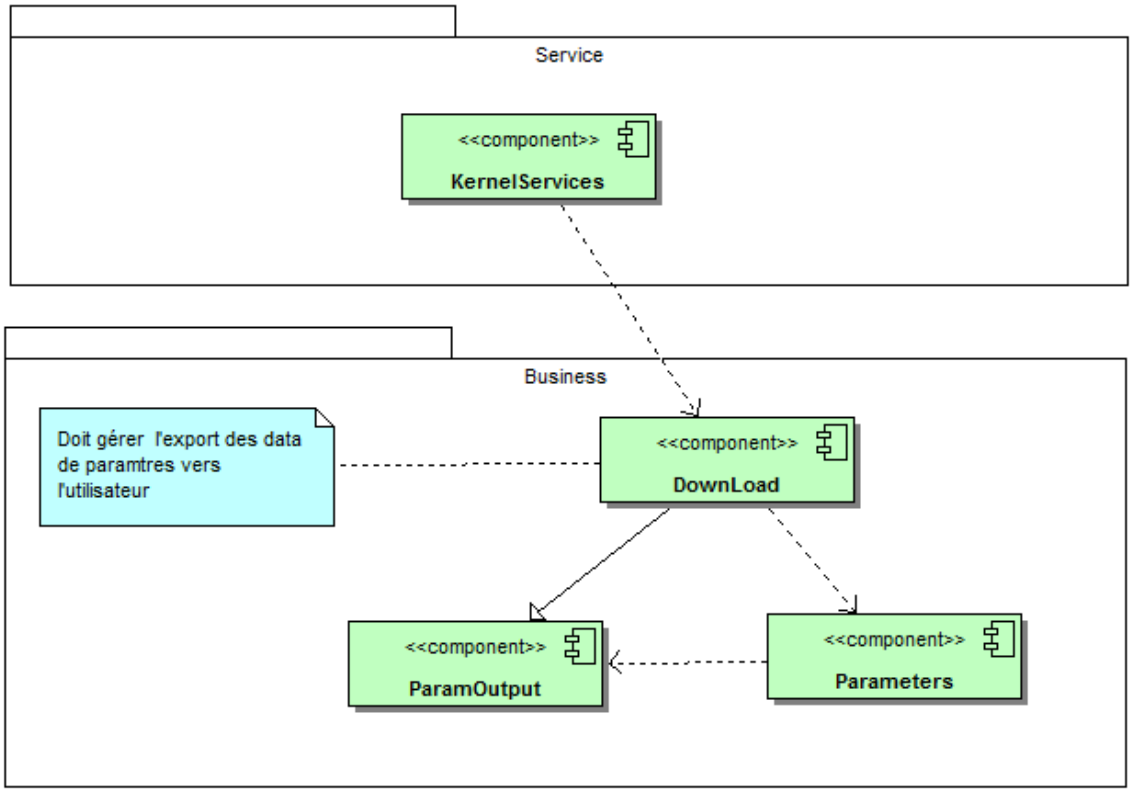
Le composant TimeTable propose principalement une fonction de merge, sort, intersection, shift, filtre et statistique de timeTable.

3.2.2.2 Composant DownLoad

Ce composant a pour but

- de mettre à disposition de « gros » fichiers de données sous forme de tar.gz ou zip

Le diagramme UML de classe suivant explicite les interfaces du composant DownLoad avec les autres composants.

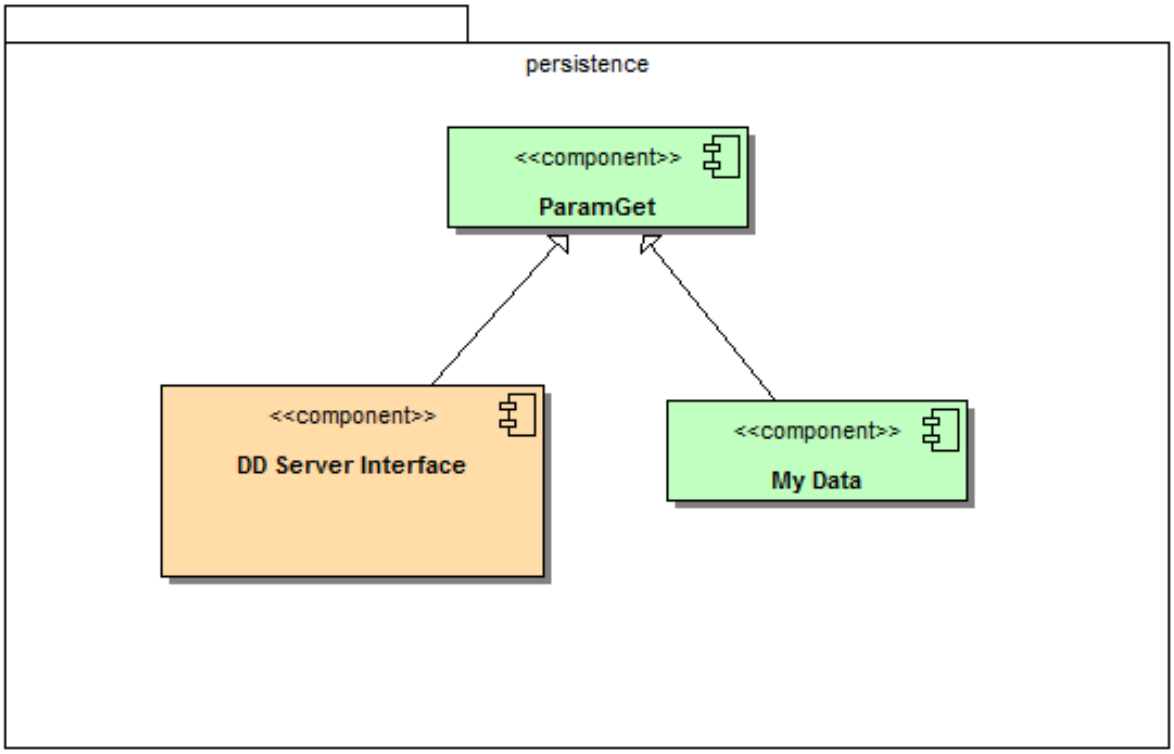


Ce composant utilisera trois COTS pour « zipper » les fichiers, plus deux solutions suivant les besoins d'optimisation:

- les exécutables: tar, gzip et zip : plus simple d'utilisation.
- les librairies tarlib et zlib: permettent de compresser au fur et à mesure de la création du fichier.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

3.2.3 Couche Persistance



3.2.3.1 Composant DD server Interface

Ce composant a pour but d'unifier l'accès à DD server. Il utilise une librairie qui communique via des sockets TCP/IP. Quatre principales étapes sont nécessaires pour récupérer des données:

- OPENINSREQ : Ouvre un dataset et retourne un identifiant unique.
- CLOSEINSREQ : ferme un dataset (identifiant en paramètre)
- TIMESETREQ: Positionne un pointeur sur le Start Time (Start Time et identifiant en paramètre).
- DATAGETREQ: A partir du start Time positionné, déplace le pointeur durant la requête par intervalle de temps (indiqué en paramètre). Le retour est sous forme d'un flux dans une socket. En en-tête est indiqué le nombre d'enregistrements et la taille des paramètres, suivent les données.

3.2.3.2 Composant My Data

Ce composant a pour but de mémoriser des données de paramètres ajoutés par l'utilisateur et de donner accès à ces données aux classes de la couche métier du noyau AMDA-NG (comme le plot).

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

3.2.3.3 Composant ParamGet

Ce composant a pour but d'unifier l'accès aux données, qu'elles soient mémorisées dans DD server ou dans My Data.

3.2.3.4 Composant transverse de trace

Un mécanisme de trace (log) doit être mise en place. Pour cela nous utiliserons le module log4cxx. C'est un « framework » libre (licence Apache License) fortement inspiré de log4j de JAVA. Log4j est maintenant le module de trace quasi standard en JAVA.

Le framework log4cxx permet d'identifier les traces suivant un niveau (TRACE, DEBUG, INFO, WARN, ERROR, FATAL).

Log4cxx a trois composants principaux: les loggers, les appenders et les layouts. Ces trois types de composants fonctionnant ensemble, permettent aux développeurs d'enregistrer les messages en fonction de leurs hiérarchies et niveaux, et de contrôler lors de l'exécution comment ces messages sont formatés et où ils sont rapportés. Ce framework prend en compte les threads qui sont gérés via le model de conception "Patterns for Logging Diagnostic Messages," in Pattern Languages of Program Design 3, édité par R. Martin, D. Riehle, and F. Buschmann (Addison-Wesley, 1997).

De plus, la notion de performance n'est pas oubliée par log4cxx, les macros proposées par log4cxx permettent de diminuer le coût à une simple évaluation de fonction et une comparaison d'entier. Il n'y a pas d'évaluation non nécessaire de construction de chaine de caractères.

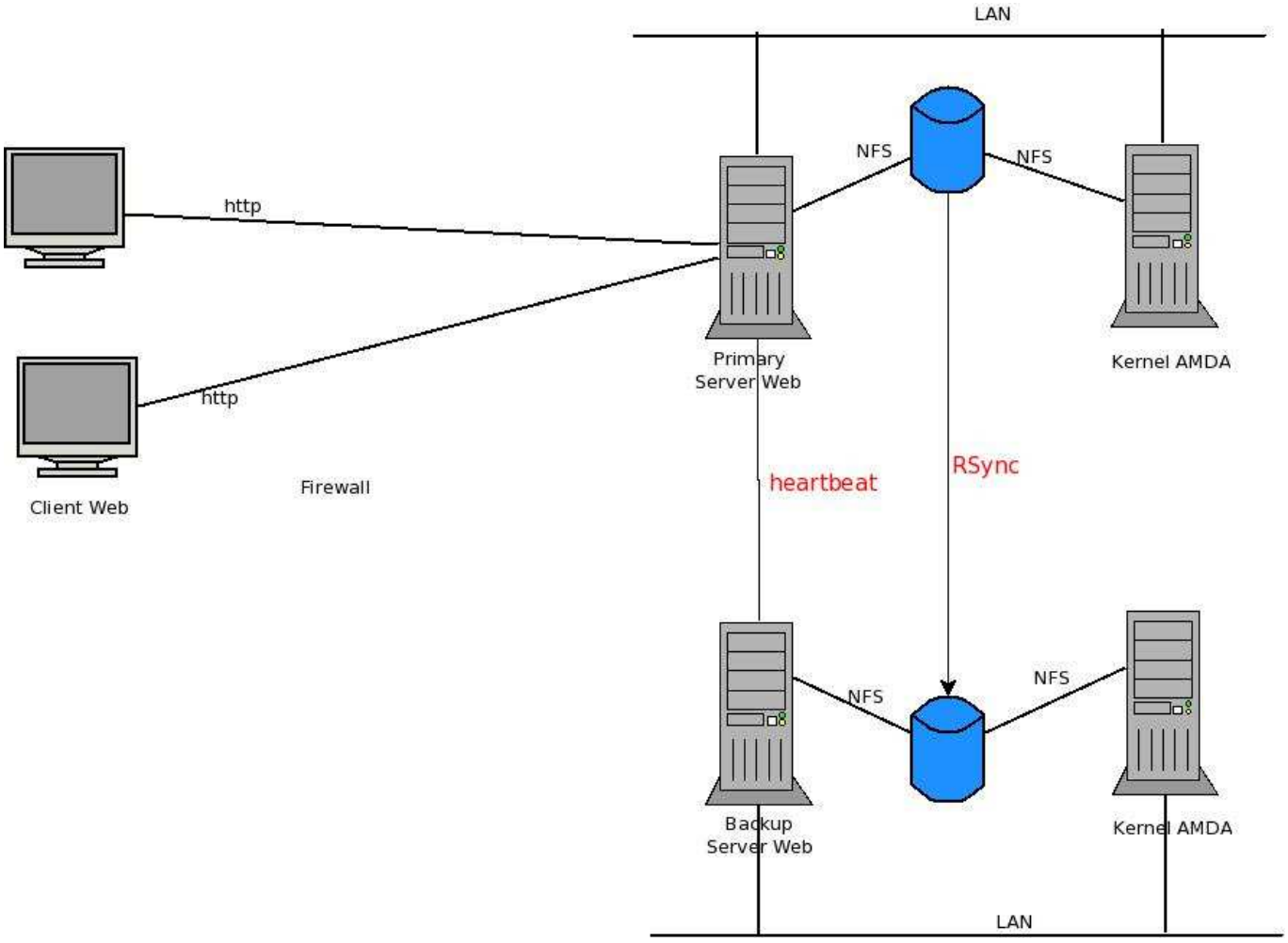
4 ARCHITECTURE PHYSIQUE

4.1 SCHEMA D'ARCHITECTURE MATERIELLE

Pour répondre au besoin de haute disponibilité de service (24h/24h) avec une installation sur deux sites distants (assurant tout risque de perturbation dû à l'environnement), les serveurs web seront installés derrière un heartbeat. Une adresse IP flottante visible des clients sera créée, ainsi si un serveur vient à « tomber », le deuxième prendra automatiquement le relais de façon transparente pour les clients.

Les données utilisateurs (user workspace) seront recopiées automatiquement via l'utilitaire RSync de façon à assurer aucune perte de données.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0



4.2 DESCRIPTION DE L'ARCHITECTURE PHYSIQUE

Chacun des serveurs Web s'exécutera sur une station linux identique (version, puissance TBD) dans un conteneur APACHE/TOMCAT.

Heartbeat gère la haute disponibilité de services qui peuvent être fournis par plusieurs serveurs (deux au minimum). Pour ce faire, chacun des serveurs est surveillé **via** un **battement de cœur** (heartbeat) diffusé sur le réseau. Au départ les services sont démarrés sur l'un des serveurs, si celui-ci n'émet plus de battement de cœur, un autre serveur prend la relève.

Le noyau d'AMDA-NG devra fonctionner sur une station linux :

- Dell PowerEdge R620 bi-proc Intel Xeon E5-2640 2.50GHz
- 15M Cache
- Turbo 6C

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

- 48Go de mémoire
- 2 disque durs de 300Go SAS + 4 de 900Go SAS
- carte RAID
- linux CentOS 64 bits.

Le serveur DD server sera embarqué dans la même station.

Le serveur AMDA-NG se présentera sous forme d'un exécutable qui devra fonctionner comme un « démon » : c'est à dire lancé au démarrage de la machine et jamais arrêté. Il sera monitoré par script.

5 DOSSIER DES LOGICIELS RÉUTILISÉS

5.1 LOGICIELS REUTILISES

Sont présentés dans le tableau ci-dessous les logiciels réutilisés.

COTS	Version	Licence	Usage	Communauté / pérennité	OS supporté	Commentaire
BOOST	V1.51	Boost Software licence (type BSD licence)	Large bibliothèque C++	A pour objectif d'être inclus dans la librairie standard C++. Voir http://www.boost.org/	Linux, Unix, Windows	
log4cxx	V0.10.0	licence Apache License, Version 2.0	Production et gestion des logue	Démarré en 2003, version chez Apache Software Foundation depuis 2008	Linux, Unix, Windows, Mac OS	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

COTS	Version	Licence	Usage	Communauté / pérennité	OS supporté	Commentaire
Libxml2	V2.7.6	MIT License	Lecture des fichiers xml en entrées du noyau AMDA	Appartient à la communauté Gnome, est un des parsers xml avec xeces le plus utilisé en C++	Linux, Unix, Windows, CygWin, MacOS, MacOS X, RISC Os, OS/2, VMS, QNX, MVS, VxWorks, ...	
gcc	V4.4.6	Free Software Foundation	Compilation à la volée des formules de calculs des paramètres	GNU	Tous les OS standards	

5.2 LOGICIELS REUTILISES PREVISIONNELS

COTS	Version	Licence	Usage	Communauté / pérennité	OS supporté	Commentaire
Apache Tomcat	TBD	Apache License V2.0	**Apache Tomcat** est un conteneur de servlets Java 2 Enterprise		Linux, solaris, Windows, Mac OS X	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

COTS	Version	Licence	Usage	Communauté / pérennité	OS supporté	Commentaire
heartbeat	V2	GPL ou LGPL.	Permet d'assurer la haute disponibilité du service AMDA-NG	Démarré en novembre 1997, dernière version sortie en Avril 2010, Projet très vivant, très utilisé dans la haute disponibilité de serveur	Linux	Voir http://www.linux-ha.org/wiki/Main_Page et http://doc.ubuntu-fr.org/tutoriel/mirroring_sur_deux_serveurs#configuration1
Rsync	V3.0.7	GNU GPL.	Permet d'assurer la redondance des données utilisateurs	Démarré en juin 1996, dernière version sortie en décembre 2009, Projet très vivant, très utilisé dans la sauvegarde de données	Linux, solaris, Windows	Voir http://samba.anu.edu.au/rsync/
plplot	V5.9.7	LGPL	Bibliothèque graphique	Très utilisée par la communauté scientifique,	Linux, Unix, Windows, Mac OS	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

COTS	Version	Licence	Usage	Communauté / pérennité	OS supporté	Commentaire
gsoap	V2.8.0	gSOAP Public License 1.3 et GPL	Serveur web service SOAP	Démarré en 2001 Utilisé par de nombreuses compagnies voir http://www.cs.fsu.edu/~engelen/soapstories.html	Linux, Unix, Windows, Mac OS	

6 DOCUMENTS APPLICABLES ET DE RÉFÉRENCE (A/R)

A/R	Référence	Titre
R1	DA-AMDA-001	AMDA architecture document V1.1 édition 10/07/2010
R2	MU-AMDA-001	Dossier des cas d'utilisation d'AMDA V1.0 édition 20/07/2010
R3	CDPP-CD-32500-436-SI	Dossier de conception du noyau d'AMDA
R4	CDPP-ST-32500-417-CES	Spécification technique industrialisation du noyau AMDA
A1	CDPP-IF-32500-438-SI	Dossier de contrôle des interfaces du noyau AMDA-NG
A2	CDPP-CO-32500-416-CNES	Dossier de consultation développement du Noyau AMDA-NG

7 GLOSSAIRE ET ABREVIATIONS

7.1 GLOSSAIRE

Terme	Définition
Acteur	Ensemble cohérent de rôles qu'une entité, humaine ou non, peut jouer lorsqu'elle interagit avec le système étudié. Un acteur peut jouer des rôles différents en fonction du cas d'utilisation auquel il participe. La notion « d'acteur » ne doit pas être confondue avec l'entité physique à laquelle elle est reliée.
Acteur principal	Un acteur est qualifié de principal lorsqu'il déclenche des évènements qui réalisent le

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

	cas d'utilisation.
Acteur secondaire	Un acteur est qualifié de secondaire lorsqu'il ne fait que recevoir des informations suite à la réalisation du cas d'utilisation.
Cas d'utilisation	Il permet de définir le comportement du système. Chaque cas d'utilisation spécifie un scénario, pouvant comporter des variantes, que le système est susceptible de réaliser, en interaction avec ses acteurs.
Composant pris sur étagère	Un composant pris sur étagère (commercial off-the-shelf ou COTS) est un mot d'origine anglo-saxonne utilisé dans le jargon électronique français pour désigner un composant fabriqué en grande série et non pour un projet en particulier.
DD server	serveur permettant l'accès à la base de données des paramètres basiques AMDA
Package	Groupement logique d'éléments du modèle. Un package peut lui-même contenir d'autres packages. Les packages permettent d'organiser le modèle suivant une structure en arbre.
Paramètre basique	c'est le paramètre élémentaire AMDA, l'utilisateur pourra les combiner, les tracer ou récupérer leur données dans un time table défini.
Scénario	Séquence d'échanges d'évènements entre les acteurs et le système dont la finalité est de réaliser un objectif précis.
TimeTable	Objet représentant une liste d'intervalles de temps

7.2 ABREVIATIONS

Abréviation	Nom détaillé
API	**A**pplication **P**rogramming **I**nterface
COTS	**C**ommercial **O**ff-**T**he-Shelf
CRUD	**C**reate, **R**ead, **U**pdate and **D**elete
GNU	**G**NU's **N**ot **U**NIX
IHM	**I**nterface **H**omme **M**achine
PDF	**P**ortable **D**ocument **F**ormat
PNG	**P**ortable **N**etwork **G**raphics
UML	**U**nified **M**odeling **L**anguage
XML	e**X**tensible **M**arkup **L**anguage

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ANNEXE - A. CHOIX DU COTS GRAPHIQUE

Matrice du choix du COTS pour effectuer les tracés

gnuplot	• Tracer à partir d'un fichier de données est une fonction de base • Les deux types de tracé sont des fonctions de base • Large choix de formats en sortie.	• API script
Plplot	• API C • Large choix de format en sortie dont: CGM, GIF, JPEG, LaTeX, PBM, PDF, PNG, PostScript, SVG, Xfig • nombreux exemples de code	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ANNEXE - B. CHOIX DU COTS DE PARSEUR DE FORMULE

Matrice du choix du COTS de parseur de formule

flex/bison	• Plus connu chez les développeurs C	• 2 nouveaux langages: lex et yacc • génération d'un fichier intermédiaire en C (et non C++)
Boost::spirit	• Nouvelle technologie • un seul langage à appréhender C++ • Boost est une technologie reconnue dans la communauté C++ qui devient standard	

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ANNEXE - C. CHOIX DU COTS DE AXIS2/C ET GSOAP

Matrice du choix du COTS de Axis2/C et gsoap

Axis2/C	• Permet de gérer des requêtes REST comme SOAP simplement par configuration	• Documentation légère • le format de sortie est toujours du XML SOAP (il n'est possible de sortie un type mime et des données comme en FullREST) • implémentation lourde, on doit gérer le XML en entrées et en sortie • Un serveur est fournie avec Axis2C qui se configure par des fichiers XML et des librairies dynamiques (.so) contenant l'implémentation des services • Plus difficile à déboguer de par l'utilisation d'un serveur fourni
gsoap	• Bonne documentation • Création de « stub » au niveau objet permettant une implémentation rapide des services par simple implémentation de méthodes retournant des paramètres natif C++ • permet de crée un serveur SOAP indépendant • Déboguable classiquement avec votre débogueur préféré	• Ne gère que le protocole SOAP

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ANNEXE - D. LICENCE BOOST

Cette licence est très similaire aux licences BSD et MIT.
Licence BOOST
Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ANNEXE - E. REST SOAP AXIS

REST SOAP AXIS

http://ws.apache.org/axis2/c/docs/axis2c_manual.html#quick_start

5.2 REST on Server Side

You basically need to add the REST Location, and the REST Method parameters to the services.xml to enable REST in a service operation. The REST location is the template that needs to be matched to find your operation, and the REST Method is the HTTP Method associated with the service. Note that the REST Method is optional for each operation. If no REST Method is specified, POST, will be assumed. Optionally you may specify the default REST Method for all operations at the service level. Then, if you haven't specified a REST Method for your operation, the default REST Method specified will be assumed instead of POST. Please have a look at the `echo` sample service for a complete source code on how to set up REST. Shown below is an example, on how to configure the `locate` operation to work with HTTP GET on REST.

```
<operation name="locate">
 <parameter name="RESTMethod">GET</parameter>
 <parameter name="RESTLocation">location/{lat}/{long}</parameter>
</operation>
```

The corresponding request would look like, `http://www.sample.org/service/location/34N/118W`, which would return Los Angeles, California. In here, the portion `location` is fixed and `lat` and `long` are optional parameters which will be captured to the payload.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0

ANNEXE - F. LICENCE MIT

The MIT License (MIT)

Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

© AKKA Technologies - Tous droits réservés empsModèle DOC-FR 03.0