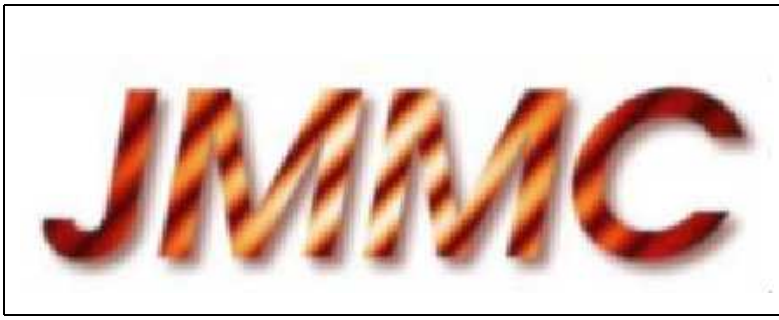

JMMC-SPE-2910-0001

Revision 1.0

Date: 17/05/2004

JMMC

Spécification du logiciel de génération de serveur de fonctions

Authors:
Thierry Stein <Thierry.Stein@obs.ujf-grenoble.fr> — LAOG/JMMC

Author: Thierry Stein Institute: LAOG/JMMC	Signature: Date: 17/05/2004
Approved by: Gérard Zins Institute: LAOG/JMMC	Signature: Date: 17/05/2004
Released by: Guillaume Mella Institute: LAOG/JMMC	Signature: Date: 17/05/2004

Change record

Revision	**Date**	**Authors**	**Sections/Pages affected**
	Remarks		
1.0	17/05/2004	T.Stein	all

Table of contents

Présentation

Le logiciel de génération de serveur de fonctions à pour but d'analyser le code source d'une librairie de fonctions C ou de classes C++, pour en extraire un certain nombre de méthodes. Celles ci sont ensuite rendues accessibles à l'utilisateur, qui pourra fixer la valeur de chaque paramètre, et voir s'afficher chaque sortie.

Le déroulement de ce logiciel peut se décomposer en deux fonctions principales :

1. Extraire depuis les fichiers sources de la librairie les fonctions à rendre disponible
2. Générer le code source du serveur

1 Extraire depuis les fichiers sources de la librairie les fonctions à rendre disponibles

A partir des fichiers spécifiés par l'utilisateur, cette fonction effectue un traitement sur chaque fichier, qui sont supposés compilables. Elle extrait la liste des fonctions qui seront accessibles à l'utilisateur, avec leurs paramètres d'entrées et de sortie. Puis elle extrait la définition des types de donnés utilisés dans ces fonctions. Cette dernière étape s'effectue dans le cas de variables qui ne sont pas d'un type prédéfini dans le langage de programmation (C ou C++).
Cette fonction se décompose en deux grandes étapes :

- Saisir la liste des fichiers spécifiés par l'utilisateur
- Analyser le code

1.1 Saisir la liste des fichiers spécifiés par l'utilisateur

Entrées : L'utilisateur indique le nom et le chemin d'accès des fichiers sources dont il souhaite extraire des méthodes.

Traitement : Cette fonction permet de saisir l'ensemble des fichiers à analyser. Elle vérifie que les fichiers sont bien dans le répertoire spécifié par l'utilisateur. Dans le cas contraire, elle renvoie un message d'erreur, et stoppe l'execution du programme. Elle regarde l'extension dans le nom des fichiers pour vérifier que le type des fichiers est bien celui attendu en entrée (fichier source C, C++, header). Dans le cas contraire, elle renvoie un message d'avertissement, et demande à l'utilisateur si il veut continuer le traitement.

Sortie : La liste des fichiers à analyser,un message d'avertissement ou un message d'erreur.

1.2 Analyser le code

Cette fonction effectue une analyse du code situé dans chacun des fichiers. Elle commence par les ouvrir, afin qu'ils soient lus par le logiciel. Puis elle traite chaque fichier. Ce traitement, détaillé par la suite, se décompose en trois étapes :

- Lister les fonctions qui se trouvent dans les fichiers sources
- Pour chaque fonction, déterminer si elle doit être rendu accessible à l'utilisateur
- Lister les nouveaux types de données utilisés dans les fonctions

1.2.1 Lister les fonctions qui se trouvent dans les fichiers sources

Entrée : L'ensemble des fichers sources à analyser.

Traitement : Cette fonction sort la liste des méthodes définies dans les fichiers sources. Elle effectue sur chaque fonction définie dans le code le traitement suivant : elle détermine le type de retour de la fonction, le nom de la fonction, et ses paramètres d'entrées/sorties. Pour chacun de ces paramètres, elle saisit le type et le nom.

Sortie : La liste des méthodes déclarées dans les fichiers sources.

1.2.2 Pour chaque fonction, déterminer si elle doit être rendu accessible à l'utilisateur

Entrée : La liste des méthodes déclarées dans les fichiers sources.

Traitement : Pour chaque fonction, le logiciel détermine si elle doit être disponible sur le serveur. Le critère de disponibilité est à définir lors de la spécification technique.

Sortie : Une sous liste de celle fournie en entrée, ne contenant que les méthodes qui seront effectivement disponible sur le serveur.

1.2.3 Lister les nouveaux types de données utilisés dans les fonctions

Entrée : Les prototypes des fonctions qui seront disponibles sur le serveur, ainsi que l'ensemble des fichiers sources.

Traitement : Cette fonction extrait la définition des types des variables utilisés dans les fonctions à rendre disponible. Ce travail se fait pour les types qui ne sont pas des types prédéfinis du langage C ou C++. Les nouveaux types peuvent être simples ou structurés.

- Pour les types simples:
  La fonction extrait le nom de ce nouveau type, ainsi que sa nature. Ceci est possible car un nouveau type est défini à partir d'un type prédéfini (par exemple : entier, chaîne de caractère...).
  Exemple : **typedef** *int* Hex.
- Pour les types structurés:
  La fonction extrait là encore le nom du type. Puis elle détermine le nom et la nature de chaque champ composant la structure.

Sortie : La liste de tout les types utilisés en entrée/sortie dans les fonctions à rendre disponible.

2 Générer le code source du serveur

Cette fonction génère un fichier source C++, nommé par défaut “main.cc”, qui, une fois compilé, permet à l'opérateur d'utiliser les fonctions extraites dans la première partie. Cette fonction commence par créer un fichier vide. Puis elle écrit le code source permettant à l'opérateur d'utiliser le serveur.
Cette fonction effectue donc les actions suivantes :

- Interfacer les commandes intégrées
- Interfacer les fonctions utilisateurs

2.1 Interfacer les fonctions utilisateurs

Entrée : Le fichier “main.cc”, la liste des méthodes et des types extraits dans la première partie.

Traitement : Cette fonction écrit le code qui permet à l'utilisateur d'appeler les méthodes extraites dans la première partie. L'utilisateur pourra affecter une valeur aux paramètres d'entrées de la méthode qu'il souhaite, exécuter la fonction, et voir s'afficher le résultat.

2.1.1 Saisir le nom de la fonction

Entrée : L'ensemble des fonctions disponibles, et un nom de fonction spécifié par l'utilisateur.

Traitement : Cette fonction saisit le nom spécifié par l'utilisateur, et regarde si ce nom est correct. Si le nom ne correspond pas à une fonction existante, le serveur renvoie un message d'erreur. Sinon, la fonction spécifiée devient la fonction courante, en attente d'initialisation de ses paramètres. C'est avec elle que s'effectue la suite du traitement.

Sortie : La fonction choisie par l'utilisateur, ou un message d'erreur.

2.1.2 Saisir la liste des paramètres

Entrée : Le nom de la fonction spécifié par l'utilisateur, et les valeurs à affecter à ses paramètres.

Traitement : L'opérateur peut initialiser les paramètres, de type simple ou structuré. Cette initialisation se fait au clavier. Dans le cas ou la variable dispose d'une valeur par défaut, l'utilisateur peut ne rien spécifier. Dans l'autre cas, il est obligé d'affecter une valeur.

Sortie : Les valeurs affectées à chaque variable de la fonction choisie.

2.1.3 Fixer des valeurs de paramètres

Entrée : Les valeurs affectées à chaque variable de la fonction choisie.

Traitement : L'utilisateur peut, s'il le souhaite, fixer des valeurs de paramètres. Ainsi, un paramètre gardera la même valeur durant toute la durée de la session, et ne pourra pas être modifié.

Sortie : Les valeurs affectées à chaque variable de la fonction choisie, en précisant si elles sont fixes ou non.

2.1.4 Donner des valeurs par défaut aux paramètres

L'opérateur peut choisir de fournir au serveur un fichier de ressource. Ce fichier contient des valeurs à affecter aux variables désirées, qui seront considérées comme étant des valeurs pas défaut. Ainsi, au moment de l'appel d'une fonction, l'utilisateur ne sera pas obligé d'affecter une valeur à ces variables.

Entrée : Un fichier script fourni par l'utilisateur

Traitement : Cette fonction analyse le fichier fourni en entrée. Elle extrait les noms des variables à initialiser, et les valeurs à leur affecter. Puis elle identifie les variables correspondantes dans la ou les méthodes existantes. Elle controle la validité des variables à initialiser, cette fonction de contrôle étant décrite plus bas. Si tout est correct, elle procède à l'initialisation. Sinon, elle renvoie un message d'avertissement à l'utilisateur, pour lui dire que la valeur par défaut n'a pas pu être affectée.

Sortie : Les variables initialisées, ou un message d'avertissement.

2.1.5 Vérifier l'intégrité des paramètres

Entrée : Les valeurs affectées à chaque variable de la fonction choisie.

Traitement : Cette fonction vérifie que les valeurs assignées aux paramètres sont correctes. Elle contrôle que le type de la valeur fourni est bien le même que le type attendu pour la variable.

Sortie : Les variables correctement initialisées, ou un message d'erreur.

2.1.6 Appeler la fonction demandée

Entrée : Le nom de la fonction, et les valeurs affectées à ses paramètres.

Traitement : Une fois les paramètres saisies de manière correcte, cette fonction effectue un appel pour exécuter la méthode désirée par l'utilisateur, avec les valeurs qu'il a choisit.

Sortie : Les valeurs affectées aux paramètres de sortie, après l'execution.

2.1.7 Afficher le résultat de l'execution

Entrée : Les valeurs affectées aux paramètres de sortie.

Traitement : Cette fonction affiche les valeurs des variables de sortie. Elle est capable d'afficher des types simples ou structurés.

Sortie : Si l'execution s'est bien passé, le serveur renvoie un message "OK". Dans le cas ou la fonction dispose d'un type de retour, cette fonction affiche le résultat. Si la fonction n'a pas de type de retour, on n'affiche rien.
L'ensemble des fonctions du serveur redevient à nouveau disponible.

2.2 Interfacer les commandes intégrées

Entrée et sortie: Le fichier “main.cc”.

Traitement : Cette fonction écrit le code des commandes intégrées dans le fichier “main.cc”. Ces commandes seront disponibles sur le serveur, quel que soit les fichiers analysés.
L'utilisateur aura à sa disposition les commandes suivantes :

- Afficher l'aide
- Quitter l'application

2.2.1 Afficher l'aide

Entrée : La ligne de commande “Help”, suivi ou non d'un argument.

Traitement : Cette fonction saisie la requête écrite par l'utilisateur, et identifie la commande. Sans argument. elle écrit à l'ecran l'ensemble des fonctions et des commandes disponibles. Si l'utilisateur demande des précisions sur une fonction (“Help nom_fonction”), cette fonction écrit le type de retour de la fonction demandée, ainsi que les noms et les types des paramètres d'entrées.

Sortie : Affiche la liste des fonctions et des commandes disponibles, ou affiche des informations sur une fonction.

2.2.2 Quitter l'application

Entrée : La ligne de commande “Quit”.

Traitement : Cette commande ferme tout les fichiers ouverts lors de l'execution du serveur. Puis elle ferme la connexion avec le serveur.

Sortie : La fermeture du logiciel.