Ecole d'ingénieurs
du Canton de Vaud

Tél. : + 41 24 423 21 11
Fax : + 41 24 425 00 50

mailto:info@eivd.ch
http://www.eivd.ch

Travail de diplôme 2003
Département E+I

Ecole d'ingénieurs du Canton de Vaud
Route de Cheseaux 1, case postale
CH - 1401 Yverdon - les - Bains

Création d'une interface permettant d'accéder à du code écrit en Ada depuis Python et inversement

Description

Le langage *Python* est un langage orienté objet, interprété. Sa syntaxe simple et claire, son haut niveau de type de données dynamique ainsi que son typage dynamique font de lui un langage de développement particulièrement bien adapté au développement rapide d'applications.

Ada est un langage à typage statique offrant une syntaxe inspirée de Pascal. Son typage fort apporte une grande robustesse dans les développements. Il est souvent utilisé dans des systèmes temps-réel nécessitant un haut niveau de fiabilité.
PyAda offre une interface entre ces deux langages ayant une philosophie différente. Cette interface composée d'une librairie faisant le lien entre l'*API Python/C* et le langage *Ada* via l'interface existante entre *Ada* et *C*.

Mandat

Le travail consiste à étendre la librairie avec les nouvelles fonctionnalités proposées par la version 2.3 de *Python*, sur la base du projet déjà existant. De développer l'intégration de code *Ada* dans *Python*, ainsi qu'une documentation utilisateur démontrant l'ensemble de la démarche pour la réalisation des interfaces.

La librairie

La librairie *PyAda,* écrite en *Ada*, redéfinit les types de *API Python/C* et importe ces fonctions à l'aide du *pragma import* permettant l'appel de méthode *C* dans du code *Ada*.

Résultats

L'intégration de code *Ada* dans *Python* et inversement, s'effectue exactement comme l'intégration de code *C* dans *Python*. Seul quelques fonctionnalités n'ont pas été implémentées à cause des conversions de types possibles en *C* mais impossibles en *Ada*.

Exemples

```
-- fonction accessible depuis python
function fonctions_add( self, args: PyObject_Ptr ) return PyObject_Ptr is
    nb1, nb2:   aliased int;
    i:          int;
    types:      chars_ptr;

    package LongPtrs is new System.Address_To_Access_Conversions( int );

begin

    -- parametres : 2 entiers
    types := New_String( "ii" );

    -- la ligne suivante est analysee par le preprocesseur genvadef de pyAda
    -- !FOR FUNCTION PyArg_ParseTuple USE VARARGS OF TYPE System.Address, System.Address

    i := PyArg_ParseTuple( args, types,
                           LongPtrs.To_Address( nb1'Unchecked_ACCESS ),
                           LongPtrs.To_Address( nb2'Unchecked_ACCESS ) );

    Free( types );

    return PyInt_FromLong(long(nb1 + nb2));

end fonctions_add;
```

Intégration de code Ada dans Python

```
-- le tuple des parametres de la fonction
PArgs := PyTuple_New(Int(Argument_Count) - 2);

for I in 0 ..  Argument_Count - 3 loop
-- pour chaque parametre

    -- la valeur du parametre (int)
    PValue := PyInt_FromLong(Long'Value(Ada.Command_Line.Argument(I + 3)));

    if (PValue = Python.Null_Ptr) then
    -- erreur lors de la convertion du parametre
       Py_DECREF(PArgs);
       Py_DECREF(PModule);

        Put("Echec de convertion d'un parametre");
        return;
    end if;

    -- ajout du parametre convertit dans le tuple pArgs
    Erreur := PyTuple_SetItem(PArgs, Int(I), PValue);

end loop;
```

Intégration de code Python dans Ada

Auteur: **Alexandre D'Amico**
Répondant externe: **Can Tuncelli**
Prof. responsable: **Thierry Gagnebin**
Sujet proposé par: **Thierry Gagnebin (EIVD)**

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale