RESTITUTION DES COMPOSANTS STICS EN PROGRAMMATION ORIENTEE OBJET
UN PAQUETAGE JAVA.STICS

Jean-Claude Poupa
INRA, Département SAE2, UMR SMART, 4 allée A.Bobierre, CS 61103, 35011 Rennes Cedex
Jean-Claude.Poupa@rennes.inra.fr

Contexte et enjeux multidisciplinaires

Le modèle STICS est modulaire. Le logiciel éponyme est basé sur une traduction dans le langage Fortran d'équations évaluées au pas de temps quotidien, regroupées en modules, pour une campagne culturale sur une parcelle. L'évaluation économique et environnementale de systèmes de production sur longue période requiert l'exécution d'un grand nombre d'unités de simulation pour enchaîner cultures principales et intermédiaires, et intercultures, avec des itinéraires techniques variables, du choix variétal aux décisions quotidiennes, et un climat aléatoire. Dans ce contexte, la programmation orientée objet va permettre de piloter les simulations au niveau des modules, composants STICS, les unités spatiales s'exécutant en parallèle sur un territoire, avec des processus qui évaluent les variables quotidiennement et pour les campagnes successives.

Le modèle STICS est générique. Les formalismes, publiés dans un ouvrage, s'appuient sur un ensemble de plus d'un millier de noms symboliques des paramètres et variables, utilisés pour décrire les fonctions d'évaluation dans des équations. C'est un modèle numérique ouvert dans lequel de nouveaux traitements peuvent être ajoutés et des couplages avec des modèles externes spécifiés dans cet espace de noms. La programmation orientée objet offre des moyens pour reprendre ces formalismes dans des langages formels utilisables pour intégrer le modèle dans des applications multidisciplinaires, les fonctions numériques et les structures de données étant restituées par traitement du code informatique des chercheurs-développeurs.

Environnements de programmation

Le modèle STICS est traduit et validé par les chercheurs. La pérennité de STICS repose sur un mode de programmation pragmatique partagé par une communauté d'agronomes concepteurs et développeurs. Si ce langage reste efficace pour traduire des calculs numériques, il ne permet pas de produire des composants intégrables dans les systèmes formels des environnements de programmation actuels, avec ici des fonctions mathématiques évaluées dans des structures algorithmiques.

Le compilateur *CStics* produit une forme fonctionnelle en langage C. Le passage à la programmation fonctionnelle structurée en langage C a été réalisé en fabriquant un compilateur, *CStics,* qui traite les unités de programmes *Fortran 77* pour générer des fonctions numériques définies avec les vecteurs de paramètres (généraux, sol, plante, itinéraire technique, climat) et variables (état quotidien de la culture et état final de la parcelle) . Des applications ont ensuite été développées en langage C et dans des langages objets, les fonctions étant accessibles dans des bibliothèques partagées. *CStics* produit aussi le dictionnaire des noms symboliques avec les règles d'allocation dans les vecteurs, structure utilisée pour gérer les formalismes STICS.

Les modules STICS sont utilisables comme composants en programmation objet. Fonctions et dictionnaire étant fabriqués, les développements peuvent s'effectuer dans différents langages objets. Les premières versions C, en l'absence de dictionnaire et avec une vectorisation statique, ont été reprises avec JAVA pour le projet HiSAFE (plate-forme CAPSIS) et C++ pour MELODIE (plate-forme DIESE). La plate-forme du projet RECORD à l'INRA utilise les versions C actuelles.

Programmation orientée objet (POO)

Les composants des programmes sont des entités appelées objets qui appartiennent à des classes, définies avec des attributs qui représentent l'état des objets, ou données, et des méthodes qui effectuent des traitements sur ces données.

L'unité de simulation STICS est un objet dynamique. L'existant STICS est traduit dans trois classes de base. Les vecteurs des paramètres et variables d'une culture sont des attributs de la classe *CultureStics,* qui offre des méthodes pour évaluer les états de la culture et de la parcelle, les modules STICS. Toutes les évaluations, depuis l'acquisition des paramètres jusqu'aux restitutions finales, sont des exécutions des fonctions de bibliothèques partagées, gérées dans une classe *EvalStics* qui charge ces bibliothèques et demande l'exécution des fonctions au système d'exploitation hôte. La classe *SymboleStics g*ère les formalismes STICS avec le dictionnaire et les méthodes pour vectoriser paramètres et variables, et accéder aux valeurs.

Une interface de programmation graphique restitue les formalismes STICS. La construction des fonctions vectorielles s'appuie sur la représentation de l'ensemble des arbres d'évaluation des fonctions issues des unités de programmes Fortran, graphe produit par *Cstics.* La classe *AlgoStics* utilise ce graphe et le dictionnaire pour fabriquer une interface de navigation dans les formalismes STICS, laquelle restitue dans des boîtes de listes l'ensemble des noms symboliques, et précise les vecteurs d'appartenance ainsi que les

tailles pour les tables. La sélection d'un nom fournit les chemins d'appels des fonctions qui évaluent la variable ou initialisent le paramètre, et pour chaque fonction énumère l'ensemble des paramètres et variables utilisés pour ces évaluations. La liste des fonctions est alors affichée et une sélection visualise l'algorithme sous un éditeur en positionnant le curseur sur la première citation de la variable évaluée : c'est un moyen de dérouler les algorithmes, avec le séquencement des évaluations des expressions arithmétiques associées aux équations, l'indentation restituant ces séquences dans des structures conditionnelles ou itératives contrôlées par des expressions logiques.
Cette interface graphique est aussi utilisable pour consulter les valeurs en cours d'exécution.
La dynamique des systèmes de culture est simulée en continu. Les états des parcelles et cultures sont suivis au quotidien dans des objets de la classe *ParcelleStics,* avec une interface pour piloter les rotations pour le climat local, cultures et intercultures étant spécifiées avec les paramètres des plantes standards et des itinéraires techniques standards validés. Les initialisations sont effectuées sur un ensemble de parcelles à une période initiale en partant de l'existant STICS pour fabriquer les vecteurs à partir des fichiers de paramètres, et les archiver dans une bases de données. Les cultures sont ensuite simulées en continu dans le temps calendaire. Les interventions culturales (travail du sol, semis et choix variétal, fertilisation, irrigation...) sont traitées comme évènements quotidiens, auxquels peuvent s'ajouter des évènements perturbateurs (invasions biologiques, accidents climatiques...). Le vecteur des paramètres généraux, pseudo-constantes, permet d'intégrer des évolutions à terme.
Les processus sont parallélisables pour les couplages. Les applications des économistes s'exécutent dans l'espace régional pour simuler de grands ensembles d'itinéraires techniques, ce qui requiert du calcul intensif. Une classe *CampagneStics* est à construire pour gérer la multiprogrammation avec des exécutions en parallèle et une synchronisation en fin de campagne.
Une synchronisation quotidienne permettrait par ailleurs de traiter les transferts entre des parcelles contiguës d'un territoire, en surface et à la base des horizons du sol.

Choix des langages objets et résultats

JAVA et C++ sont utilisables en fonction des environnements de recherches. Les programmes doivent d'abord pouvoir exécuter des fonctions compilées de bibliothèques, ensuite être eux-mêmes compilables pour fabriquer des modules exécutables pour les besoins de calcul intensif : JAVA et C++ répondent à ces contraintes dans le monde du logiciel libre. Les bibliothèques, liées aux systèmes d'exploitation, sont gérées dans des classes, qui vont fournir les méthodes pour mettre en correspondance les vecteurs des représentations objets et procédurales, et exécuter le code, dit natif, des fonctions numériques.
JAVA, pur langage objet largement enseigné, est utilisé de façon classique en mode interprété pour développer les applications, les programmes étant ensuite compilés avec ***gcj*** de la GNU Compiler Collection (GCC) pour fabriquer l'application finale. Les interfaces d'accès aux bibliothèques, appelées JNI (Java Native Interface}, offrent des solutions pour utiliser des logiciels scientifiques, dont GAMS et R. JAVA apporte par ailleurs dans le monde du logiciel libre des interfaces de programmation sous la forme de paquetages : programmation graphique, multiprogrammation, bases de données...
C++, sur-ensemble du langage C, objet et procédural, n'est pas interprété et les applications sont compilées. La plate-forme VLE de RECORD utilise ce langage pour développer des applications, avec des espaces de noms qui donnent accès au logiciel libre de statistique R et à des extensions pour les systèmes d'équations différentielles et d'équations aux différences.
Les syntaxes de ces deux langages ont proches pour la partie objet et les traductions sont faciles.
Les classes STICS traduites en JAVA constituent un paquetage de composants libre. Regroupées dans le paquetage *JAVA.STICS* les classes précédentes pourraient fournir sous forme de contribution INRA dans le monde du logiciel libre une interface de programmation objet pour modèle de culture, utilisable par la communauté scientifique. Outre les simulations de la dynamique des cultures sur un territoire, les mécanismes d'héritage sont utilisables dans différents contextes applicatifs, pour ajouter ou remplacer des modules, coupler des modèles pour piloter des décisions ou forcer des variables, et ceci au moyen des formalismes STICS. Plusieurs bibliothèques peuvent être fabriquées, pour produire différentes versions modulaires ou optimisées pour des contextes applicatifs, en simple ou double précision, *CStics* générant le nouveau code natif et les structures de données liées.

Plates-formes informatiques

La chaîne de traitement réalisée reprend l'existant STICS, avec le langage informatique des équipes de recherches en agronomie et les fichiers de paramètres des cultures standards, pour produire des composants mobilisables pour des travaux dans d'autres disciplines. Le résultat n'est pas un logiciel mais un environnement de programmation en informatique scientifique, base de travail intégrable sur des plates-formes.
Les objets STICS sont des composants génériques pour la plate-forme RECORD. Le projet RECORD a adopté le système formel DEVS pour la modélisation et la simulation des systèmes dynamiques

à évènements discrets. Les composants STICS précédents, avec les données représentées dans des vecteurs et des méthodes qui reprennent formalismes et modules du modèle de culture, sont utilisables sur cette plate-forme. Les évènements peuvent être gérés dans ces vecteurs, solution générique qui réduirait les coûts d'ingénierie liés aux couplages sur mesure pour des projets thématiques d'équipes.

Références bibliographiques

Nadine Brisson, Marie Launay, Bruno Mary, Nicolas Beaudoin, editors,
Conceptual basis, formalisations and parameterization of the STICS crop model, éditions Quae.

Jean-Claude Poupa, *Intégration de modèles de cultures sur des plates-formes de modélisation agronomique et environnementale Application au modèle STICS,* 260 p.

Jean-Claude Poupa, *Programmation orientée objet du modèle de culture STICS.*