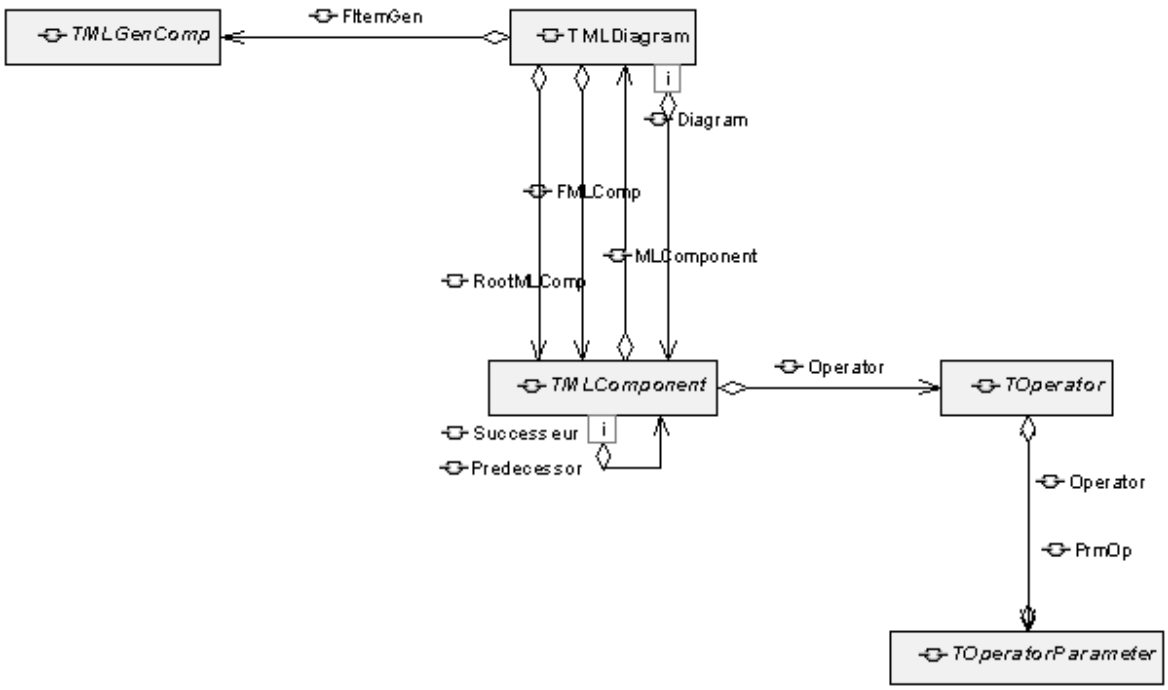
A. Structure d'un diagramme de traitements

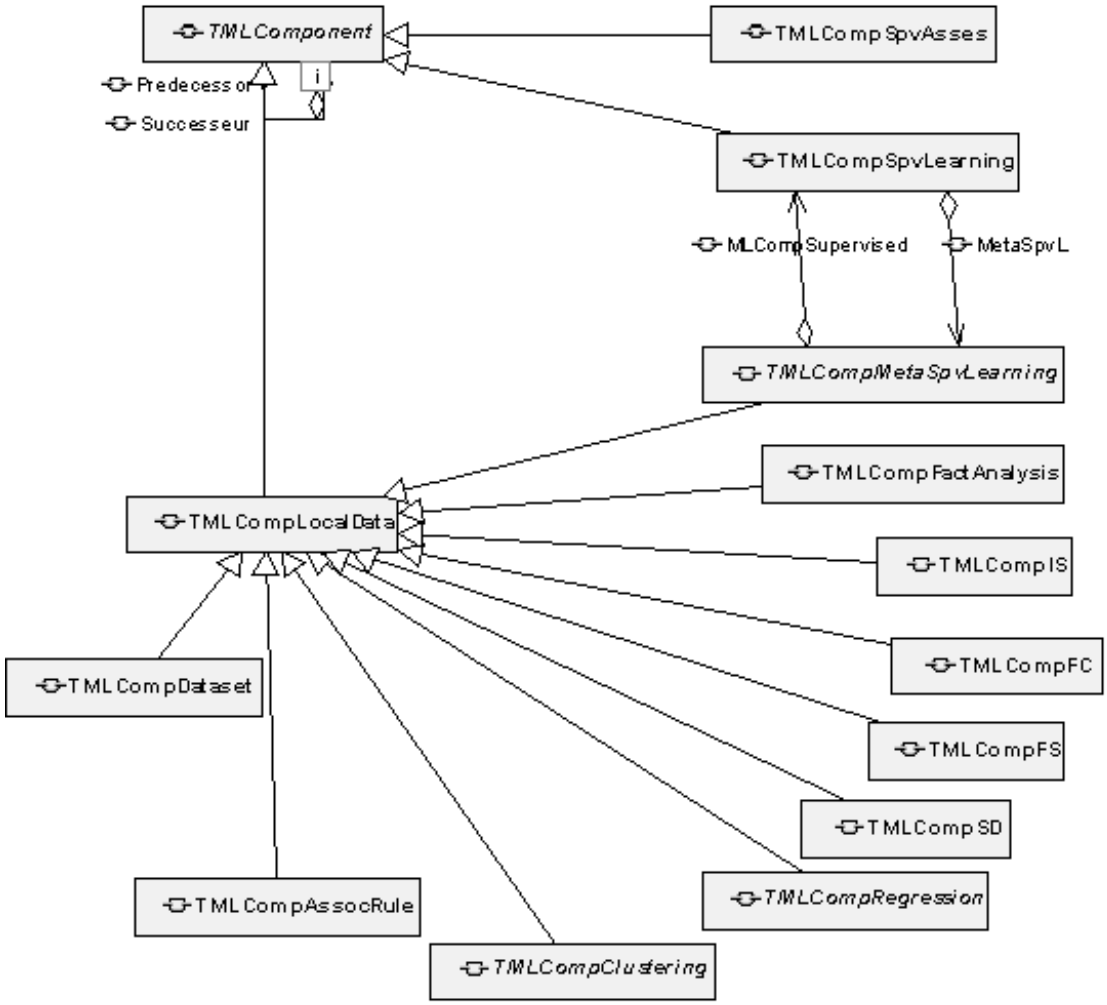
Un diagramme de traitements contient une série de composants visuels. Chaque composant est associé à un opérateur de calcul qui peut être paramétré. Un générateur de composant permet d'ajouter un composant dans le diagramme. Les classes TMLComponent (UcompDefinition.pas), Toperator (UoperatorDefinition.pas) et ToperatorParameter (UoperatorDefinition.pas) sont abstraites.



B. Hiérarchie de classes des composants

Les composants dans un diagramme de traitements

Pour chaque composant, trois classes doivent être implémentées. Pourquoi une telle redondance ? L'idée est de séparer complètement les classes d'interface et les classes dédiées au calcul. Les descendants de TMLComponent se chargent de l'intégration dans le diagramme, visuellement dans le TREEVIEW ; ceux de TOPeratorParameter se chargent de collecter les paramètres introduits par l'utilisateur à travers l'affichage d'une boîte de dialogue. Les descendants de Toperator se chargent du calcul proprement dit. Cette déconnexion permettra, peut-être, plus tard, d'implémenter les classes de calculs dans des DLL ou encore des ActiveX, on pourrait également faire appel à des programmes extérieurs, des exécutables ou des classes JAVA, pour effectuer les calculs. En tous les cas, pour chaque nouveau composant à créer : les héritiers de ces trois classes doivent être implémentés, la plupart du temps, elles sont regroupées dans la même unité DELPHI.



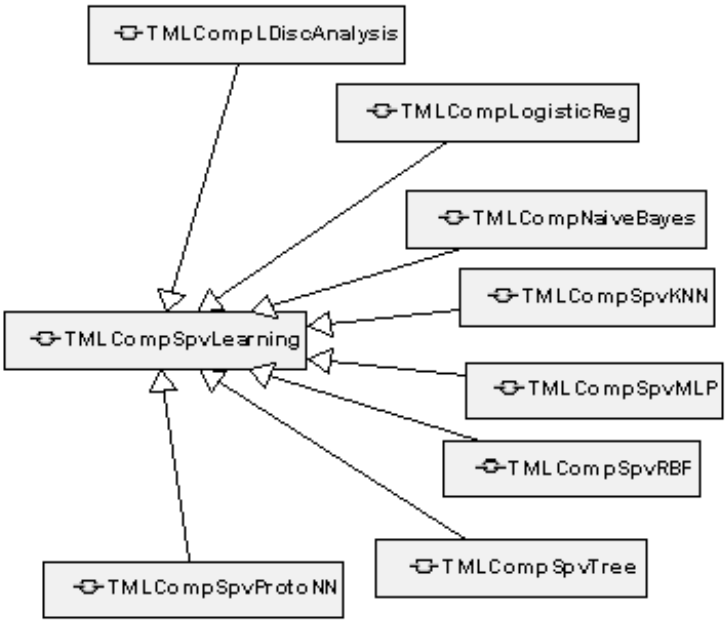
On remarquera que :

1. Une grande partie des classes sont des héritiers de TMLCompLocalData, elle-même dérivée de la classe abstraite TMLComponent, ces classes ont pour point commun la manipulation des données qui leur sont transmises en entrée ;
2. Les classes destinées à l'apprentissage supervisé sont un peu différentes. Seule TMLCompMetaSpvLearning est un descendant manipulant directement les données, pour fonctionner, on doit lui associer un composant apprentissage supervisé, du type TMLCompSpvLearning ;
3. Enfin, la classe d'évaluation de l'apprentissage supervisé ne peut être placée sur un diagramme que s'il succède à un TMLCompMetaSpvLearning, il ne manipule donc pas des données en entrée. En réalité, si l'on entre dans le détail du code, il manipule les individus actifs sur le premier composant de la chaîne de traitements ;
4. A peu de chose près, on aura la même hiérarchie pour les descendants de Toperator.

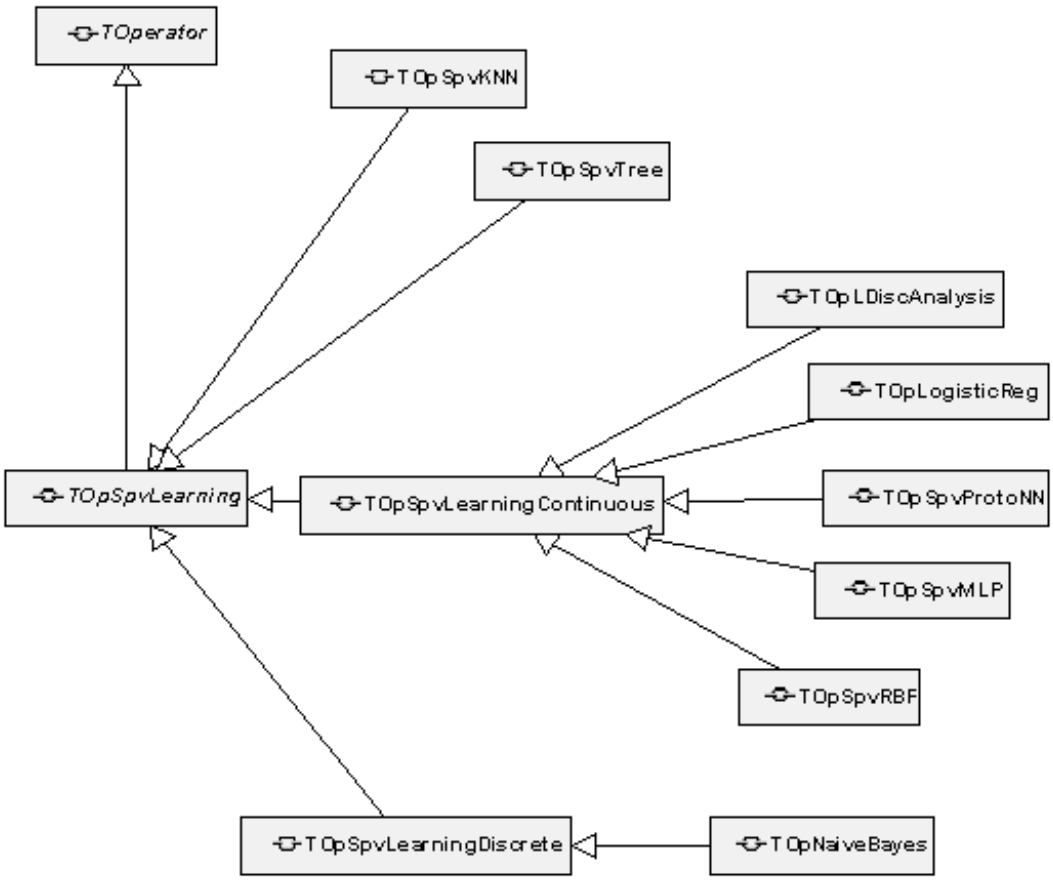
Hiérarchie sous TMLCompSpvLearning

Bien entendu, les classes qui figurent dans le graphique ci-dessus représentent les grandes catégories de composants du logiciel, ils correspondent aux onglets de la palette de composants.

Chacune d'elle se déclinent par la suite en séries de sous-classes qui implémentent réellement des méthodes. Prenons l'exemple des méthodes supervisées.



Et si nous nous penchons sur les classes de calculs, héritiers de Toperator, nous observerons quasiment la même hiérarchie. Quasiment car des classes intermédiaires peuvent être introduites pour des vérifications supplémentaires : une analyse discriminante ne peut fonctionner par exemple que si tous les descripteurs sont continus.

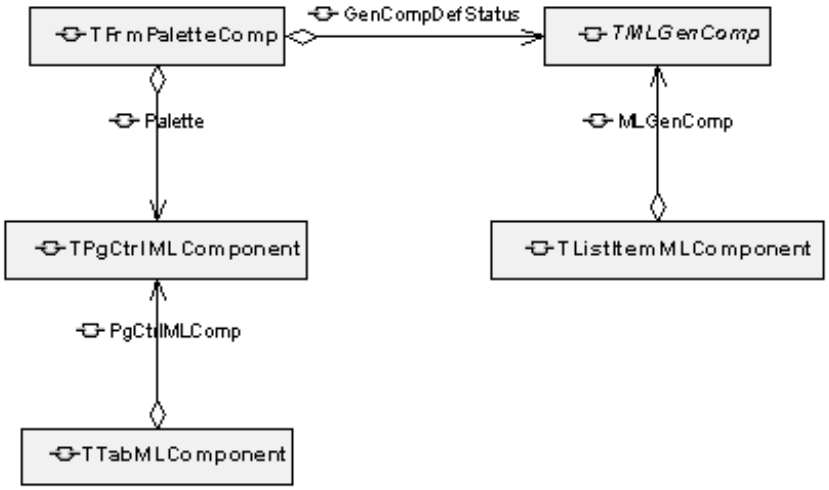


On constate :

1. Les arbres de décision et notre implémentation des plus-proches voisins (distance HVDM) acceptent tous types de descripteurs ;
2. Les autres méthodes, en revanche, n'acceptent que des attributs tous discrets ou tous continus.

C. Gestion des composants de l'application

Si l'on veut vraiment modifier la structure de l'application, il est possible d'accéder à la gestion des composants via les classes suivantes, à savoir leur intégration dans l'interface, leur classement dans les TABSHEET et la classe génératrice de composant. On accède à toutes les classes dans le code source, en revanche les composants réellement disponibles lors de l'exécution de TANAGRA sont lus dans un fichier de configuration au format XML (« tanagra_components.xml »). Les générateurs de composants se distinguent par le nom de classe, ils sont « identifiant » dans le fichier de configuration, les doublons sont impossibles dès lors que le fichier est validé en relation avec sa DTD.


TFrmPaletteComp
GenCompDefStatus
TMLGenComp
Palette
MLGenComp
TPgCtrlMLComponent
TListItemMLComponent
PgCtrlMLComp
TTabMLComponent