

# **Concept d'interopérabilité dans l'architecture d'un serveur thermodynamique COM. Exemple d'intégration dans Microsoft Excel.**

---

**SIMO**  
**24-25 Octobre 2002**  
**Toulouse, France**

---

**Alain Vacher, Philippe Guittard**

**ProSim SA**  
**Stratège Bâtiment A**  
**BP 2738**  
**F-31312 LABEGE Cedex**  
**France**

**Alain.Vacher@prosim.fr, Philippe.Guittard@prosim.fr**

---

## **Résumé**

Depuis leur origine, les outils de simulation ont du répondre à une demande toujours plus exigeante du marché et s'adapter aux matériels et technologies informatiques du moment. Aujourd'hui, ils doivent satisfaire, pour n'en citer que certains, des critères tels que interopérabilité, intégration et réutilisabilité. Notre projet de Recherche et Développement StarDust<sup>®</sup>, axé autour d'une nouvelle architecture logicielle, ambitionne de proposer un tel environnement. Bien que fondé sur une spécification propriétaire, ce projet intègre le standard CAPE-OPEN et s'appuie sur le middleware COM/DCOM de Microsoft<sup>®</sup> largement diffusé au sein de son système d'exploitation Windows<sup>®</sup>. Dans le cadre de StarDust<sup>®</sup>, un soin tout particulier a été apporté dans le développement du serveur thermodynamique, considéré, à juste titre, comme la pierre angulaire de toute application de simulation en génie des procédés. Les concepts de ce serveur permettent de bien appréhender les capacités de la nouvelle architecture qui peuvent être illustrées par un exemple d'intégration dans Microsoft<sup>®</sup> Excel.

---

## **1. Introduction**

Afin de mieux cerner les exigences actuelles, nous commencerons par retracer sommairement l'évolution des outils de simulation. Même si cela peut apparaître comme une évidence, il est important de ne pas oublier que les nouveaux systèmes que nous proposerons demain doivent continuer à satisfaire les attentes d'hier. Nous présenterons ensuite, dans ses grandes lignes, notre projet de Recherche et Développement StarDust<sup>®</sup> qui définit une nouvelle architecture logicielle orientée composants devant être adoptée par l'ensemble de nos produits. Nous expliquerons nos attentes d'un tel système et son positionnement par rapport aux standards tant sur le plan métier, avec l'implémentation des interfaces CAPE-OPEN, que sur le plan purement informatique, avec le choix du middleware COM/DCOM de Microsoft<sup>®</sup> face à CORBA<sup>®</sup>. Nous montrerons les avantages offerts par une telle architecture en détaillant le serveur thermodynamique développé dans le cadre de StarDust<sup>®</sup>. Enfin, nous finirons avec un exemple

d'intégration de ce serveur dans Microsoft® Excel pour montrer les services que peuvent se rendre mutuellement ces deux systèmes logiciels.

## **2. Evolution des outils de simulation**

Tout au long de leur histoire, les outils de simulation des procédés ont subi des périodes de mutation leur imposant de s'adapter aux matériels et technologies informatiques du moment et répondre à une demande toujours plus exigeante du marché. Afin de dresser un succinct rappel de cette évolution, telle que nous pouvons l'appréhender aujourd'hui, nous décrirons trois grandes générations d'environnement qui se sont succédées au cours du temps :

### Années 1970-80 :

Les outils étaient des systèmes monolithiques utilisés sur des mainframe ou mini-ordinateurs. Pour la plupart, écrits dans des langages tels que FORTRAN, ces applications se présentaient sous la forme de traitements par lots (fichiers d'entrée/fichiers de sortie) ou tout au mieux d'interactifs (déroulement séquentiel de questions/réponses). Le souci majeur était alors de disposer d'outils généraux robustes et de maintenir des performances acceptables en temps de calcul.

### Années 1990 :

L'arrivée des stations de travail et des ordinateurs personnels est venue bouleverser les habitudes de travail. Avec la montée en puissance des ordinateurs, le critère de performance, qui reste même encore aujourd'hui un facteur important, n'était plus la préoccupation majeure des utilisateurs et, donc, des développeurs. Les outils devaient être conviviaux et simples d'utilisation. Les interfaces graphiques apportèrent alors une réponse.

### Aujourd'hui :

Les habitudes se trouvent aujourd'hui une nouvelle fois perturbées avec l'avènement des réseaux et du travail coopératif. En plus des critères de performance et de convivialité, les différents outils doivent être maintenant capables de se substituer et de s'échanger des services, ils doivent être interopérables et intégrables. Les nouvelles technologies à base de composants développées avec des langages orientés objets doivent permettre de satisfaire cette nouvelle exigence du marché.

En conclusion de ce rapide historique, nous pouvons également nous demander où se situe l'avenir des environnements de simulation de procédés dans la mouvance Internet. A l'image de ce qui émerge déjà dans d'autres secteurs d'activités, nous pouvons imaginer trouver, au sein d'annuaires spécialisés, des composants métiers ayant des fonctions standardisées et qui auront la faculté d'exposer dynamiquement leur protocole d'échange de services afin de permettre leur intégration dans une application cliente.

## **3. Le projet StarDust®**

Devant le constat fait précédemment, nous avons entrepris de changer l'architecture de nos outils pour répondre à l'attente, non seulement de nos propres clients, mais aussi, de l'ensemble de la communauté des utilisateurs de logiciels de simulation de procédés. C'est l'ambition du projet StarDust®.

Le projet StarDust® est construit autour d'une spécification d'interfaces (ou contrats) que doivent respecter les composants représentant les différents objets (propriétés physico-chimiques, calculs thermodynamiques, réactions chimiques, opérations unitaires, ...) rencontrés dans notre métier. Ces interfaces régissent bien évidemment le fonctionnement des différents composants mais aussi la façon dont ils peuvent communiquer entre eux et avec des entités externes pouvant être, sur un plan structurel, fortement hétérogènes.

Bâtir une nouvelle architecture n'implique pas pour autant une réécriture complète de l'ensemble de nos codes existants. Un des défis de StarDust® est en effet la conservation ou, tout au moins, la modification minimale de ces codes qui, aujourd'hui, possèdent l'énorme qualité d'avoir été testés et validés, au cours de nombreuses années, par de multiples utilisations et dans

des contextes très variés. Les développements effectués dans le cadre de StarDust<sup>®</sup> se traduisent donc, successivement, par une modularisation et une encapsulation des codes existants.

Cette décomposition nous permet ainsi de disposer d'une palette de composants assemblables entre eux et avec des systèmes externes. L'assemblage des composants peut nous permettre ainsi de disposer de nouveaux produits généraux tels qu'un simulateur de procédés continus mais aussi de disposer de systèmes plus légers conçus pour un objectif plus spécifique. Certains composants sont même viables dans un environnement autre que StarDust<sup>®</sup>.

Les composants StarDust<sup>®</sup> doivent répondre à des critères rigoureux de qualité. Leurs performances doivent être comparables à celles que nous pouvons enregistrer aujourd'hui dans des codes classiques. Les composants, en plus des services métiers qui leur sont confiés, doivent répondre à des exigences de convivialité et de persistance de données. En d'autres termes, ils disposent, si cela est nécessaire, d'interfaces homme-machine permettant de manipuler leurs données. Un mécanisme de persistance de données, permettant de mémoriser l'état du composant à un instant donné, est également implémenté. Ainsi, ces deux services dégagent l'application cliente de ces charges et lui permettent de se concentrer sur son propre objectif. Ce genre de services est un des grands apports d'une approche composants qui ne se résume pas à une approche simplement modulaire.

#### **4. StarDust<sup>®</sup> et CAPE-OPEN**

L'architecture imaginée dans StarDust<sup>®</sup>, bien que basée sur une spécification propriétaire, distincte du standard CAPE-OPEN, assure, néanmoins, une complète compatibilité avec les spécifications décrites par celui-ci. Cela se traduit ainsi par une parfaite ouverture aux composants CAPE-OPEN qui peuvent offrir leurs services spécifiques au sein de notre architecture, mais aussi offrir, de par leurs définitions, des services dérivés tels que ceux de sources d'informations requises par les composants StarDust<sup>®</sup>. En plus de cette interopérabilité placée sous l'angle des entrées, les composants StarDust<sup>®</sup>, qui implémentent bien évidemment leur propre jeu d'interfaces, proposent également, là où cela se justifie, une implémentation des interfaces du standard CAPE-OPEN.

En résumé, même si l'architecture conçue dans le cadre du projet StarDust<sup>®</sup> ne se base pas de façon native sur le standard CAPE-OPEN (fait essentiellement lié à la conservation de codes existants), elle propose une interopérabilité quasi complète, entrante et sortante, avec les composants fondés sur ce standard.

#### **5. Technologie COM**

Les composants logiciels qui constituent l'ossature du système StarDust<sup>®</sup> doivent pouvoir être déployés sur un ou plusieurs ordinateurs d'un réseau local et pouvoir, néanmoins, communiquer les uns avec les autres. Cette contrainte requiert donc que les composants soient développés sur la base d'un middleware assurant cette gestion. Au regard des standards de facto du marché, le choix est restreint à COM/DCOM de Microsoft<sup>®</sup> et CORBA<sup>®</sup> de l'OMG qui sont d'ailleurs les deux seuls middleware pour lesquels il existe une spécification du standard CAPE-OPEN. Le principal atout de CORBA<sup>®</sup>, autorisant une gestion hétérogène des plateformes, perd en partie de son intérêt face à l'importance du parc d'ordinateurs fonctionnant sous Microsoft<sup>®</sup> Windows<sup>®</sup> qui demeure notre cible privilégiée. Un autre argument essentiel qui nous a amené à choisir COM/DCOM comme technologie de base du projet StarDust<sup>®</sup>, s'exprime par le fait que ce middleware est natif au sein de Microsoft<sup>®</sup> Windows<sup>®</sup> et ne nécessite donc, pour l'utilisateur final, ni installation, ni coût ou administration spécifiquement liés au middleware.

#### **6. Serveur thermodynamique StarDust<sup>®</sup>**

La connaissance et l'estimation de données physico-chimiques s'avèrent fondamentales et indispensables dans tout outil de simulation de procédés. Aussi, il apparaît naturel que les premiers développements concernent un serveur thermodynamique utilisable, bien évidemment, par tous les futurs composants StarDust<sup>®</sup> mais aussi, par tout autre système externe.

Afin de mieux comprendre les services offerts par le serveur thermodynamique StarDust<sup>®</sup>, nous présentons ci-après un extrait de son architecture :

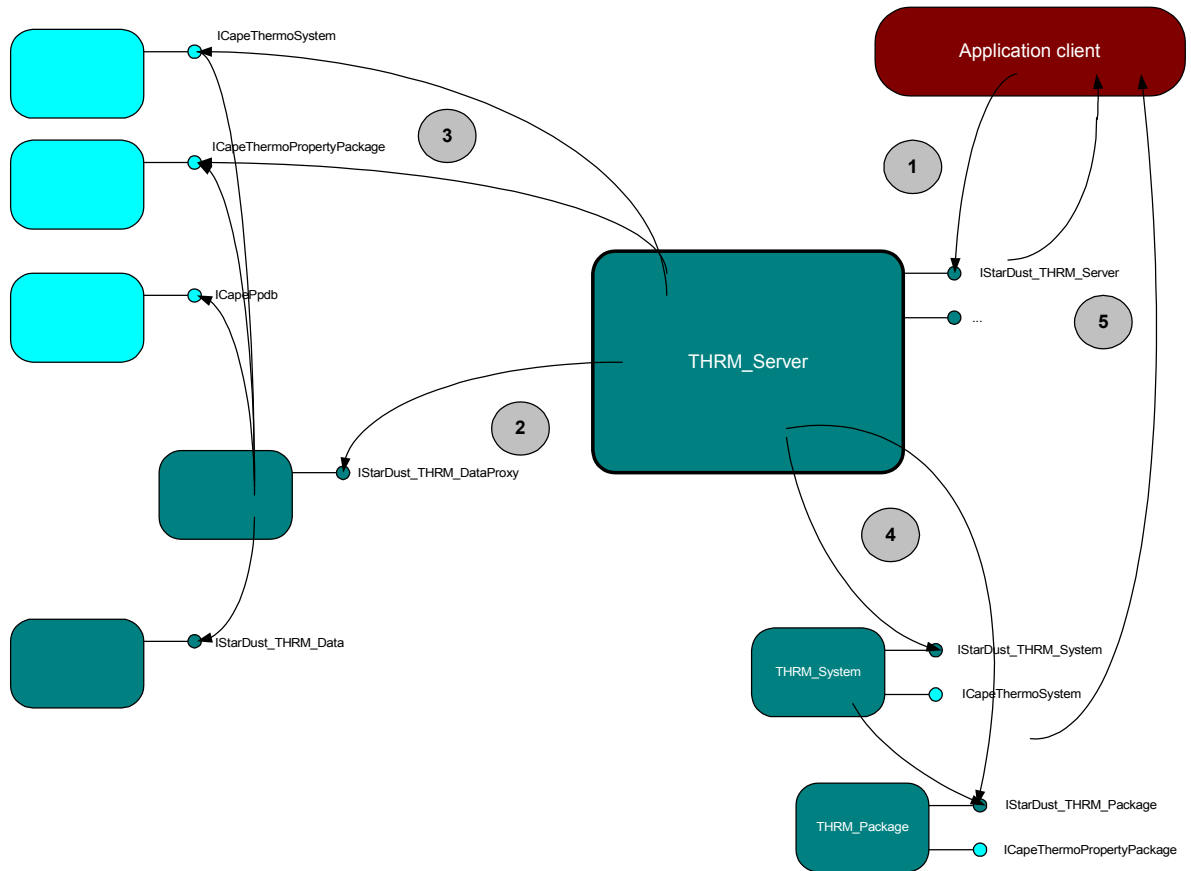


Figure 1 : Architecture du serveur thermodynamique

Un scénario classique d'utilisation du serveur thermodynamique suit la séquence suivante:

1°) L'application cliente obtient une instance du serveur thermodynamique lui autorisant ainsi d'utiliser ses services grâce à l'interface IStarDust\_THRM\_Server.

2°) Tout serveur thermodynamique nécessite la connaissance de différentes données comme, par exemple, les propriétés intrinsèques des corps purs en présence dans le mélange traité. Des interfaces standards (IStarDust\_THRM\_DataProxy) d'accès à ces données ont été créées dans StarDust<sup>®</sup> afin de se connecter à des sources hétérogènes telles que des composants internes à notre architecture (IStarDust\_THRM\_Data) ou externes comme des composants compatibles CAPE-OPEN (ICapePpdb et mêmes ICapeThermoSystem ou ICapeThermoPropertyPackage qui sont également des conteneurs de propriétés). D'autres données nécessaires au serveur thermodynamique ne sont pas décrites ici.

3°) Le serveur thermodynamique propose également ce qu'il est convenu d'appeler une "socket thermodynamique CAPE-OPEN" permettant d'utiliser directement au cœur de notre système des composants externes implémentant les interfaces ICapeThermoSystem ou ICapeThermoPropertyPackage.

4°) Le serveur thermodynamique est capable de créer des composants réutilisables tels qu'un système thermodynamique (interface IStarDust\_THRM\_System) et un package thermodynamique (interface IStarDust\_THRM\_Package). Ces deux derniers concepts sont identiques à ceux de CAPE-OPEN et il est donc logique que ces composants implémentent respectivement les interfaces ICapeThermoSystem et ICapeThermoPropertyPackage.

5°) Enfin, l'application cliente peut obtenir, soit directement du serveur thermodynamique (THRM\_Server), soit des systèmes (THRM\_System) ou packages (THRM\_Package) thermodynamiques, les informations qu'elle attend en final (propriétés de corps pur ou de mélange, équilibre entre phases, ...).

En conclusion, nous pouvons constater que le serveur thermodynamique StarDust® autorise l'application cliente à se connecter de façon transparente et centralisée à plusieurs types de composants issus d'environnements différents. Devant l'éventail des possibilités proposées, l'application cliente peut alors choisir la meilleure option lui permettant d'atteindre son objectif spécifique.

## **7. Exemple d'intégration dans Microsoft® Excel**

A titre d'illustration, nous terminerons par un exemple d'intégration du serveur thermodynamique StarDust® dans Microsoft® Excel. Nous détaillerons tout d'abord un exemple de code Visual Basic (VBA) réalisant un calcul de température de bulle qui doit permettre d'évaluer la simplicité de mise en œuvre et la nature des services offerts :

```

Dim ThrmServer As ThrmStarDust.StarDustThrmServer
Dim nbObject As Long
Dim Composition(1 To NCMAX) As Double
Dim Pressure As Double
Dim BubbleTemperature As Double
Dim TypeOfComposition As Long
Dim IThermodynamic As Long
...
'création du serveur thermodynamique ...
'... permettant un stockage de l'état du serveur thermodynamique dans
'... le fichier XLS
ActiveSheet.OleObjects.Add ClassType:= "ThrmStarDust.StarDustThrmServer" _
    Link:=False, DisplayAsIcon:=False, Left:=0, Top:=0, _
    Width:=0, Height:=0
'Affectation dans une variable
nbObject = ActiveSheet.OleObjects.Count
Set ThrmServer = ActiveSheet.OleObjects(nbObject)
'initialisation du serveur
Call ThrmServer.InitializeSvr
'entrée des paramètres nécessaires au calcul :
'1- sélection des corps purs (ouverture d'un dialogue spécifique)
Call ThrmServer.SelectCompounds
'2- sélection des modèles thermo (ouverture d'un dialogue spécifique)
Call ThrmServer.SelectThermodynamics
'3- affectation des paramètres d'appel
'... modèle thermodynamique
IThermodynamic = 1
'... affectation de la pression
Pressure = 1
'... affectation de la composition (2 corps purs)
Composition(1) = 0.5
Composition(2) = 0.5
'... affectation du type de composition (molaire)
TypeOfComposition = 0
'appel du calcul de la température de bulle
Call ThrmServer.ymTb(IThermodynamic, Pressure, Composition(1), _
    TypeOfComposition, BubbleTemperature)
...

```

**Figure 2 : Code VBA de calcul de température de bulle**

D'autre part, nous avons développé sur cette base, une macro complémentaire Microsoft® Excel appelé BibPhy AddIn qui permet d'effectuer sur une feuille Excel tout calcul de propriétés de corps purs ou de mélanges et d'équilibre entre phases. Un exemple de résultat de calcul d'enveloppe de phase obtenu grace à différentes fonctions disponibles à travers cette macro est exposé dans la figure ci-dessous :

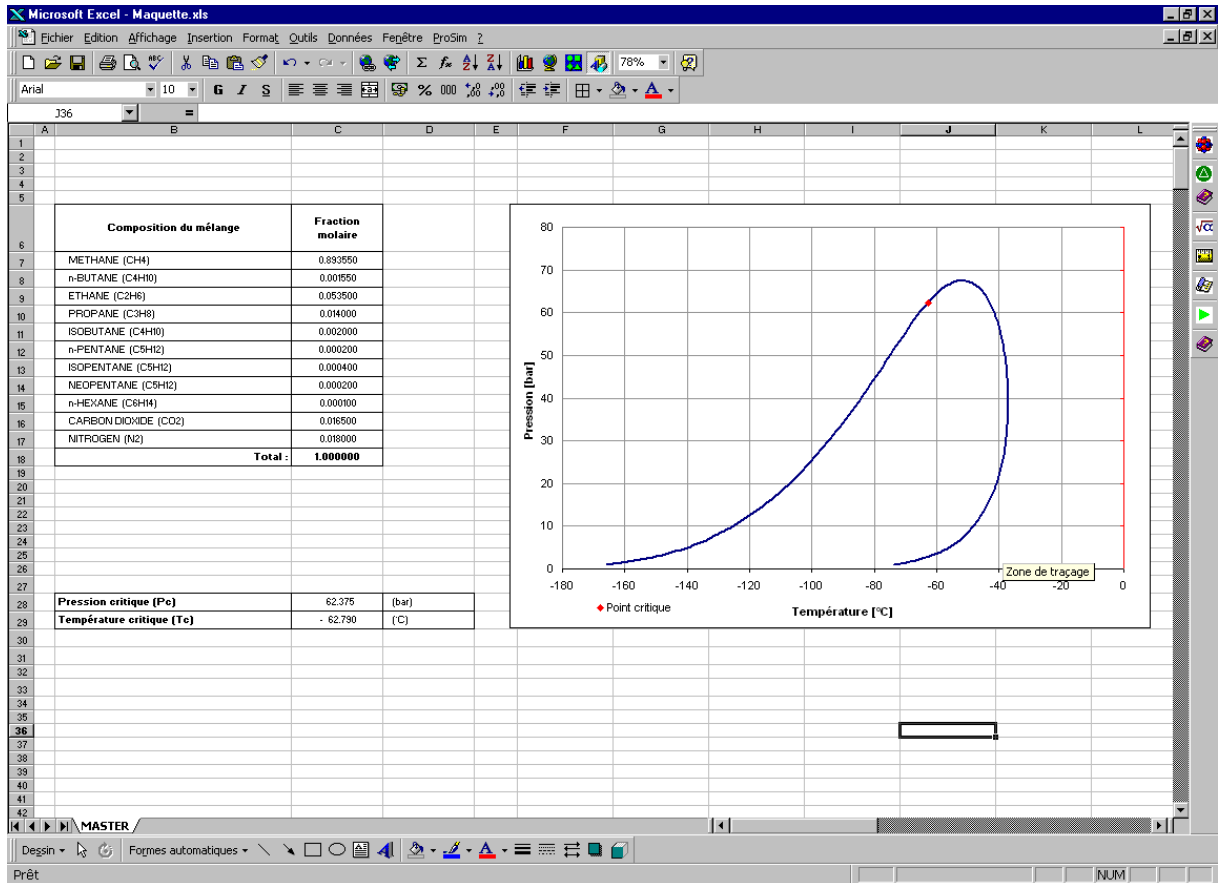


Figure 3 : Enveloppe de phase réalisée avec BibPhy AddIn

## 8. Conclusion

Dès aujourd'hui, les environnements de simulation à base de composants ouvrent de nouveaux horizons dans l'étude des procédés. Ils permettent aux ingénieurs de procédés, par un assemblage judicieux de composants prêts à l'emploi, d'améliorer la qualité de leurs développements tout en réduisant les temps associés. Grâce au standard CAPE-OPEN, l'interchangeabilité de composants devient possible. Cette souplesse permet de choisir le composant jugé le plus performant pour un cas de figure donné. L'approche composant apporte donc incontestablement des gains de productivité aussi bien aux utilisateurs qu'aux développeurs de ces systèmes.

## Sites Web

CO-LaN, CAPE-OPEN Laboratory Network, informations relatives au standard : [www.colan.org](http://www.colan.org)

Microsoft®, informations relatives à COM : [www.microsoft.com/com](http://www.microsoft.com/com)

OMG, Object Management Group, informations relatives à CORBA® : [www.corba.org](http://www.corba.org)

ProSim SA, site internet : [www.prosim.fr](http://www.prosim.fr)