Solution pour l'interopérabilité avec COMETH

Benjamin Haas, Patrick Corrales

Solution pour l'interopérabilité avec COMETH

Benjamin Haas[*1], **Patrick Corrales** [1]

[1] **Centre Scientifique et Technique du Bâtiment**
84, avenue Jean Jaurès BP 02 Champs sur Marne
[*]**benjamin.haas@cstb.fr**

RESUME. Le cœur de calcul COMETH développée par le CSTB se présente sous la forme d'une bibliothèque de calcul compilée. La facilité d'utilisation et d'intégration dans des logiciels du commerce ont guidé les choix de structure, poussant vers une approche par paramétrage plutôt que par câblage de modules (comme TRNSYS, MATLAB-SIMULINK ou MODELICA). A contrario, ceci peut apparaitre comme limitant les possibilités de modélisation. Nous montrons que nous avons palié à ce problème en proposant un système de plug-in, via la définition d'ontologies de systèmes dédiées à COMETH pour permettre le développement collaboratif, le partage et la cosimulation de nouveaux modules. D'ores et déjà, plusieurs systèmes innovants ont été intégrés par des industriels et mis à disposition des bureaux d'études

MOTS-CLÉFS. interopératbilié, COMETH, plug-in, SED

ABSTRACT. COMETH is computing kernel developed at CSTB. It is a compiled library that computes the building's energy consumption at an hourly time step. The library format makes its integration easy in commercial software and many of the patterns used in the development reflect this aspect. In particular, this is why we chose not to use block diagram simulator(à la TRNSYS, MATLAB-SIMULINK or MODELICA) and preferred to focus our work on a ready-to-integrate API. A contrario, such choices may complicate the work of upgrading the tool such as adding new innovative systems, making the initial development team the bottleneck of every update. We show that this problem can be solved using a plug-in approach in our development. We define a COMETH-related ontology for systems in order to allow for collaborative developments to enable co-simulation, while guarantying the overall quality. By now, many systems has been developed under this procedure and coupled to the COMETH platform

KEYWORDS. interoperability, plugins, energy modeling, collaborative development, COMETH

1 Introduction

Les réflexions sur l'interopérabilité sont aussi anciennes que l'existence des outils numériques eux-mêmes. Cette notion dénote initialement un besoin de réutilisabilité du code dans des environnements autres que ceux pour lesquels ils sont initialement imaginés. La problématique technique est essentiellement résolue par l'invention des compilateurs en 1951, permettant tout d'abord la réutilisabilité du code sur plusieurs machines, puis la communication entre différents codes issus de différents langages. Toutefois, faire communiquer du code objet est complexe et

réservé à des experts de l'informatique. Or, le domaine de la physique du bâtiment est par essence multi-disciplinaire. L'utilisateur est un bureau d'étude ou un architecte, un développeur est un physicien ou un ingénieur système dont l'informatique n'est pas le cœur de métier. La problématique est donc ici déplacée vers l'enjeu de partager simplement les développements, à moindres coûts, entre tous ces différents acteurs.

Partant du principe que la notion de compilateur résout les problèmes de compatibilité informatique, la notion d'interopérabilité se déplace tout d'abord vers des notions de compatibilité mathématique et physique.

- Compatibilité mathématique : peut-on (et faut-il ?) adopter un formalisme pour tous les développements (acausal vs. causal ? modèles d'états vs. modèles continus ?)
- Compatibilité physique : peut-on (et faut-il ?) s'accorder sur une ontologie universelle, permettant à tous d'identifier des fonctionnalités (ou des systèmes ?) aux limites bien définies et figées ?

Elle est ensuite très liée à des choix d'architectures de code pour mettre en œuvre les échanges d'objets compilés ou pas, tout en garantissant la cohérence scientifique d'ensemble.

Loin de tenter de répondre à ces questions de manière globale ou d'apporter des solutions figées, nous souhaitons présenter le travail effectué autour de COMETH, outil de simulation énergétique dynamique (SED) développer par le CSTB et, entre autres, support des réglementations thermiques.

COMETH, pour *COre for MOdelling Energy and THermal Comfort*, est un outil développé dans un cadre d'hypothèses que l'on peut tenter ici de résumer (Videau et al., 2013). Les modèles sont dynamiques simplifiés, semi-dynamiques ou statiques. Il n'y pas de solveurs de systèmes d'ODE, chaque modèle se résolvant lui-même. L'avantage d'une telle solution est de pouvoir valoriser rapidement des modélisations "expertes", issues de tests en laboratoire et basées (au moins en partie) sur des points de fonctionnement en lieu et place de modélisations fines particulièrement coûteuses à développer et à valider [1]. Les temps de calculs sont également un gain appréciable. Alors que COMETH fonctionne au pas de temps horaire, qu'il dispose d'un calcul de besoin thermique, d'un modèle aéraulique (ISO, 2008a), de modèles d'éclairage, de gestion des ouvrants et des protections mobiles ainsi que les masques proches et lointains, il calcul en un dixième de seconde sur un machine standard, pour une maison individuelle. L'ajout de systèmes allonge le temps de calcul de manière variable suivant la complexité des cas.

COMETH est développé par le CSTB depuis 2009. Cette bibliothèque de calcul sert de fondation au code réglementaire dans la RT2012 (CSTB, 2011). Il est également utilisé dans les outils diffusés suivants :

- En tant qu'outil d'étude pour l'étude faisabilité exigée par la RT2012 (JORF, 2013) ;
- IdCET permettant l'identification des résultats certifiés EN NF 16147 pour les chauffe-eau thermodynamique avec les entrées de la méthode Th-BCE (CSTB, 2012) ;
- Plusieurs outils en cours de développement interne CSTB pour le compte d'industriels.

COMETH est également utilisé dans les projets ANR Fiabilité, PLUMES (Robert et al., 2013) (qui a contribué au financement des présents travaux), et COSIMPHI. Au niveau européen, COMETH est utilisé pour le projet HOLISTEEC issu de l'appel FP7.

Pour COMETH, les enjeux et contraintes de l'interopératbilié sont les suivants.

1. Valoriser des innovations issues du monde industriel (PME/TPE comprises) en leur faisant profiter d'un environnement dynamique, et ce à moindre coût ;
2. Leur permettre de se confronter aux contraintes normatives et réglementaires le plus amont possible, en particulier des RT françaises, et des normes européennes ;
3. Garantir la propriété intellectuelle des codes sources des industriels ;
4. Garantir la stabilité d'un noyau commun pour une comparaison "toute chose égale par ailleurs" des innovations (ainsi que la compatibilité à la réglementation pour COMETH

1. sans bien sûr nier l'intérêt de tels travaux

configuré pour les réglementations thermiques) ;

5. Intégrer l'ensemble COMETH+innovations dans les logiciels commerciaux.

Dans la suite nous présentons la stratégie adoptée pour répondre à ces éléments. Ensuite, nous montrons un rapide cas exemple, bien qu'en date d'écriture de cet article, nous ne pouvons encore rentrer dans les détails car ces exemples correspondent à des cas réels non encore publiés. Enfin, nous concluons par les perspectives d'une telle approche.

2 COMETH et sa solution pour l'interopérabilité

2.1 Le choix d'une architecture plug-in

Dans le cadre des contraintes précédemment mentionnées, la divulgation de codes source est exclue. Autant la divulgation de COMETH lui-même, du fait de son utilisation a des fins réglementaires, que des développements de nouveaux modules dont les industriels ne souhaitent pas toujours divulguer les fondamentaux. Dans ce contexte, une solution qui ne manipule que du code objet est une contrainte absolue.

D'autre part, pour assurer la traçabilité des développements, il est essentiel de différencier chacun des développements, potentiellement effectués par des acteurs différents, ne serait-ce que pour assurer la maintenance du code par l'organisme ayant également assuré le développement.

Enfin, COMETH est un code complexe. Il est nécessaire d'assurer une solution d'interopérabilité qui ne nécessite pas une connaissance approfondie du fonctionnement de COMETH.

Par conséquent, une solution "plug-in", que nous appelons également « extensions dynamiques »est optimale. Rappelons une définition largement répandue du plug-in (Misc, 2014) :

La plupart du temps, ces programmes sont caractérisés de la façon suivante :

- *ils ne peuvent fonctionner seuls car ils sont uniquement destinés à apporter une fonctionnalité à un ou plusieurs logiciels ;*
- *ils sont mis au point par des personnes n'ayant pas nécessairement de relation avec les auteurs du logiciel principal.*

La notion de plug-in est, à notre connaissance, nouvelle dans le monde du logiciel scientifique. Quand évoquée, elle se résume à l'intégration de blocs dans des approches diagrammatiques, ce qui, à notre sens, ne relève pas du plug-in. La stratégie plug-in contraint à définir des points d'entrée avec le code principal, identifié à COMETH. Pour faciliter les développements, nous avons choisi d'organiser ces points d'entrée en fonctionnalités associées à une ontologie de systèmes. Ce choix d'ontologie est, conformément à l'esprit du plug-in, dédié à COMETH. Toutefois, il n'y aucune verrou technique, dans le domaine du logiciel scientifique, à réutiliser les brisques que constituent les plug-ins dans d'autres environnements. Ceci doit rencontrer une réel besoin, et n'est pas étudié dans le présent article.

2.2 Description de la structure informatique et scientifique

COMETH est développé en .Net (C#). Comme le JAVA, ce langage permet de manipuler la notion de plug-in car le code objet compilé embarque également des informations sémantisées sur ses fonctionnalités, intérogeables par un code tiers. Ceci permet de lier à la volée des bibliothèques a priori indépendantes, et donc de réaliser des architectures plug-in facilement se rapprochant des programmations orientées composant Sametinger (1997). De plus, et de nouveau comme le JAVA, le .Net est nativement compatible avec les systèmes Windows, MAC et Linux, sans recompilation. Il offre donc un environnement de développement multi-plateforme.

Nous montrons Figure 1 une représentation très schématique de COMETH. Celui-ci est constitué d'un ensemble pré-assemblé de systèmes et modèles. Le modèle thermique, basé sur la norme EN NF ISO 13790 (ISO, 2008b) version calcul horaire, est couplé à un modèle aéraulique

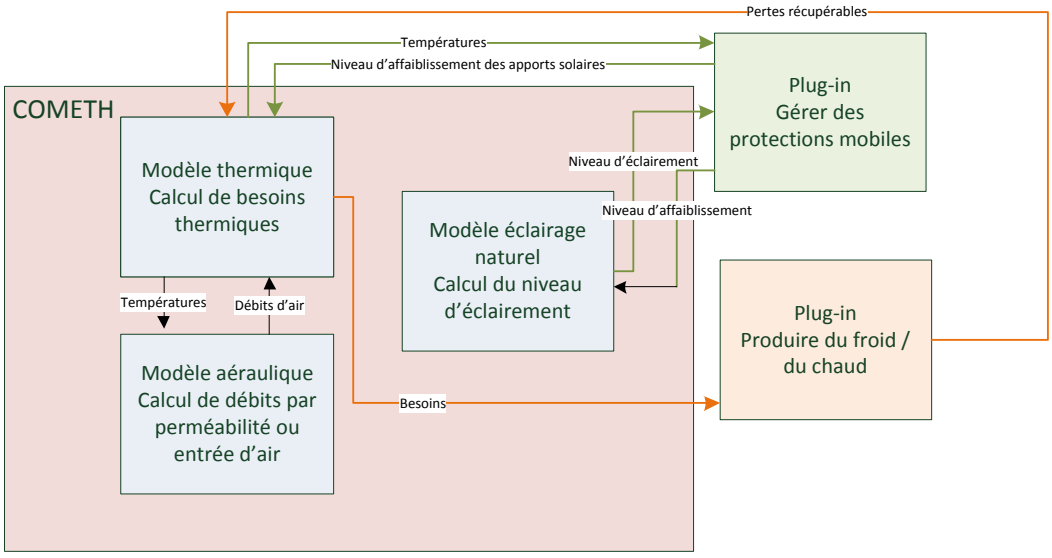

FIGURE 1. Exemple d'interaction entre COMETH et ses plug-ins. COMETH contient les modèles fondamentaux (thermiques, aéraulique et éclairement naturel). Il interagit ici avec deux exemples de plug-ins, pour la gestion des protections mobiles ou pour la génération de chaleur, typiquement une pompe à chaleur ou une chaudière.

mono-zone en pression, multi-zone en température, basé sur un équilibre des débits. Un modèle d'éclairage naturel y est adjoint, et interagit avec le reste via son impact sur les gestions des protections mobiles et les systèmes d'éclairage artificiels générant des pertes thermiques récupérables.

Les plug-ins interviennent au niveau des équipements dans le domaines suivants :

- systèmes CVC,
- éclairage articiciel,
- production d'eau chaud sanitaire,
- production électrique solaire.

Des prototypes, i.e. un modèle comportement et de données, de systèmes se greffent sur cette structure, permettant une interaction dynamique, au pas de temps horaires, entre ces deux aspects du calcul. Figure 1, on trouve deux exemples de plug-ins, simplifiés pour le besoin de la représentation. Un prototype "Produire de chaud et du froid" représente les systèmes de générations de chauffage et de refroidissement. Un exemple concret est un générateur thermodynamique.

Les systèmes actuellement développables par la méthode des extensions dynamiques sont les suivants :

1. Générateurs pour le chauffage et/ou le refroidissement et/ou l'ECS
2. Les modèles de sources amont pour les générateurs thermodynamiques
3. Les espaces non chauffés (dits "tampons")
4. Les réseaux de distribution
5. Les systèmes production d'ECS avec stockage
6. Les ventilations mécaniques

2.3 Le comportement du code en présence d'extensions dynamiques

En Figure 2, nous proposons une représentation schématique de la manière dont un code extension dynamique interagit avec COMETH, et ce dans le cas d'une application aux Titre

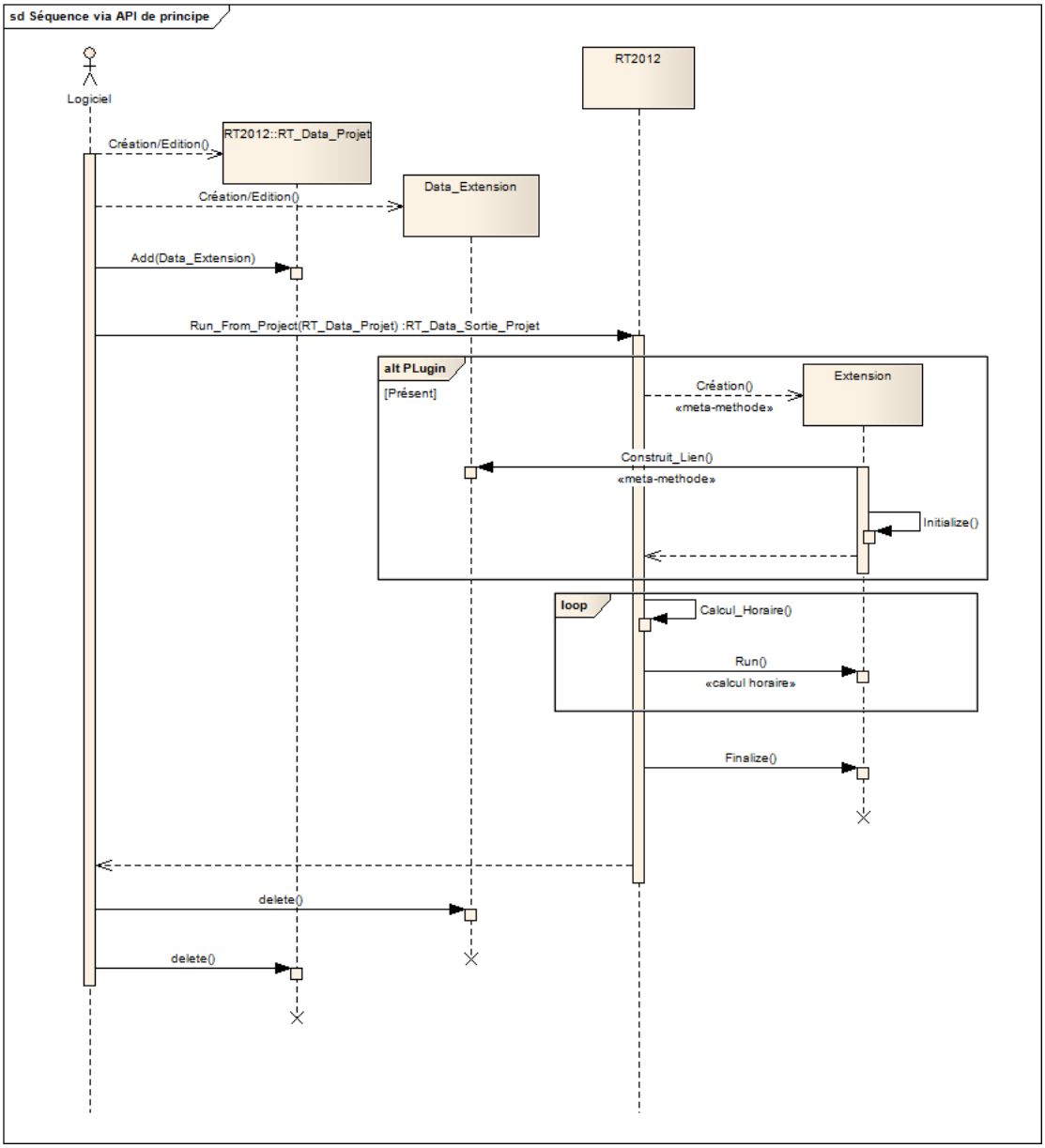

FIGURE 2. Description de l'interaction de COMETH avec les extensions dynamiques dans le cadre d'une application aux Titre V RT2012.

V RT2012. L'utilisateur doit spécifier dans son jeu de données habituel (au format XML par exemple) la présence d'une extension dynamique par son nom, puis déclare les paramètres de dimensionnement. Le moteur identifie alors automatiquement la présence d'une extension et vérifie qu'elle est bien représentée par un fichier DLL référencé dans un dossier standard. COMETH vérifie également que le DLL contient bien tous les éléments pour effectuer le calcul, et "raccroche" ensuite une instance du module à son propre système et le paramètre avec les jeux de données fournis par l'utilisateur. L'utilisateur n'a donc **qu'à spécifier dans son jeu de paramètres la présence d'une extension dynamique**.

D'un point de vue de l'utilisateur, ce système est donc particulièrement simple, toute la complexité étant cachée. A priori, les éditeurs de logiciels commerciaux intègrent totalement les extensions dynamiques au fur et à mesure de leur livraison en les référençant dans leur interface, l'utilisateur ne voyant aucune différence entre des fonctionnalités natives COMETH et les ajouts.

3 Un exemple concret

Pour permettre de saisir ce qui définit un plug-in, ou extension dynamique, nous présentons un exemple concret, un générateur de chaleur.

Comme toute extension dynamique, le générateur de chaleur doit respecter une définition, autrement appelée prototype, patron ou modèle. Cette définition est représentée, en langage orienté objet, par une classe abstraite (C++) ou interface (JAVA ou .Net). Elle contient trois types d'informations

1. les variables d'entrée au pas de temps horaire que peuvent fournir COMETH dans le contexte du système considéré,
2. les variables de sorties au pas de temps horaire nécessaires au fonctionnement de COMETH,
3. les méthodes appelées par COMETH (c'est COMETH qui pilote l'appel de l'extension)

L'extension dynamique doit impérativement implémenter ces trois aspects. Elle peut également en proposer d'autres, mais ceux-ci seront sans impact sur le fonctionnement de COMETH. Nous résumons dans le Tableau 1 ces trois éléments pour le générateurs de chaleur.

Entrées horaire du composant	
Description	Unité
Température ambiant du système	°C
Sorties horaires du composant	
Description	Unité
Service appelé	Chauffage/Refroidissement/ECS
Part de l'énergie demandée n'ayant pas pu être fournie	Wh
Pertes thermiques transmises vers l'ambiance	Wh
Production électrique du générateur	Wh
Consommation des auxililières	Wh
Puissance maximale pouvant être fournie	W
Énergie consommée	Wh
Taux de charge	Wh
Puissance fournie	Wh
Méthode disponible	
Description	Arguments
Appel du générateur	Energie demandée (Wh), Température avale (°C)

Table 1: Définition d'un générateur de chaleur dans le cadre de l'ontologie COMETH. Plusieurs familles sont disponibles et mises à disposition des développeurs de plug-in, ceci en est un exemple.

L'utilisateur a ensuite toute latitude pour définir le comportement du générateur dans les limites des données mises à disposition. A noter que les conditions limites à l'échelle globale du projet (météo et notion de date) sont toujours disponibles à toutes les extensions, au-delà des variables mentionnées dans le tableau.

A noter enfin la possibilité a priori de combiner différentes fonctionnalités. Ainsi, une même extension dynamique peut hériter des fonctionnalités de ventilation mécanique et de génération de chaleur, permettant de développer des machines multi-fonctions. Ceci peut toutefois se heurter à des limites de conception de logiciel principal lui-même : la solution technique informatique ne vient qu'en soutien d'une modélisation physique adaptée.

4 Conclusion et perspectives

Ce travail est aujourd'hui la pierre angulaire de la valorisation de l'innovation dans le monde des réglementations thermiques. On compte actuellement une dizaine de développements terminés, effectués en majorité par des bureaux d'étude spécialisés pour le compte d'industriels [2]

Nous pouvons donc aujourd'hui esquisser un retour critique de cette solution. Après une prise en main qui a demandé un accompagnement fort sur les premiers projets, nombre de développements se font maintenant en autonomie en dehors du CSTB, dans des sociétés de type bureau d'étude. Ceci représente un succès que nous expliquons par le cadre simple (mais contraignant, nous y reviendrons) qui une fois compris peut être facilement utilisé. En effet, il suffit d'identifier les entrées, les sorties et les méthodes, et de développer un code unitaire qui respecte ces contraintes.

Revenons au questionnement de l'introduction. Nous avons fait des choix forts. La modélisation est fondamentalement causale. Il est impossible actuellement de valoriser tel quel des modèles acausaux (sauf à les rendre causaux pour l'application dédiée). A l'inverse, il est impossible de valoriser les extensions dynamiques dans des environnements acausaux, ceux-ci nécessitant une formalisation bien particulière, et la mise à disposition des codes source pour pouvoir valoriser toutes les particularités de l'aspect acausal. Un autre choix et celui de l'ontologie. Nous avons définis des familles de systèmes par des jeux d'entrées/sorties et des méthodes. Nous ne prétendons pas pouvoir exporter cette ontologie vers d'autres outils de manière systématique, car elle dérive d'hypothèses de modélisation propres à chaque outil.

Cette solution ne résout pas la totalité des questions posées par les industriels en termes d'innovation. En effet, elle permet essentiellement de remplacer des fonctionnalités existantes par des nouvelles, mais pas de créer ex-nihilo de nouveaux comportements. Si, par exemple, il n'existe nulle part de mécanisme prévoyant de gestion/régulation couplant des éléments aussi divers que le bâti, la ventilation mécanique et les systèmes de refroidissement, cette solution ne permet pas de l'inventer. Cette solution est donc contraignante, et nous considérons que ceci constitue le prix à payer pour un développement simple et une utilisation simple via des logiciels commerciaux.

Une perspective intéressante de cette approche est le développement actuel d'un outil accompagnant les nouvelles normes européennes sur l'EPBD. Actuellement en refonte, le CSTB est associé au TNO pour montrer la faisabilité d'une approche modulaire des normes, où les états européens peuvent remplacer les calculs pour leurs propres standards. L'approche par extension dynamique parait extrêmement bien adaptée à ce contexte.

5 Remerciments

La partie recherche de ce travail a été en partie financée par le projet ANR PLUMES, sur l'appel à projet HABISOL 2010 (ANR-10-HABI-0009).

Les auteurs tiennent à remercier les partenaires du consortium pour des discussion qui leur ont permis d'aboutir à la vision présentée dans cet article.

2. En date d'écriture de cet article, bien que les développements soient terminés, les décrets correspondant ne sont pas encore publiés. Nous ne pouvons donc mentionner les systèmes correspondant.

6 Bibliographie

Références

CSTB (2011). Méthode Th-BCE.

CSTB (2012). http ://dimn-cstb.fr/idcet.

ISO (2008a). EN NF 15242 : Ventilation for buildings - Calculation methods for the determination of air flow rates in buildings including infiltration.

ISO (2008b). ISO 13790 :2008 : Energy performance of building - Calculation of energy use for space heating and cooling.

JORF (2013). Décret no 2013-979 du 30 octobre 2013 relatif aux études de faisabilité des approvisionnements en énergie des bâtiments nouveaux.

Misc (2014). http ://fr.wikipedia.org/wiki/plugin.

Robert, S., Delinchant, B., Hilaire, B., & Tanguy, Y. (2013). PLUMES : a unified approach to buildings physical modeling, Sylvain Robert. *accepted to IBPSA2013*.

Sametinger, J. (1997). *Software engineering with reusable components*. Springer.

Videau, J.-b., Alessandrini, J.-m., Haas, B., Pelé, C., Millet, J.-r., Jallet, P., Reynier, L., & Fleury, E. (2013). An introduction to the development of the French Energy Regulation indicators and their calculation methods. In *CLIMA 2013*.