Le Framework 'maison' : la pierre angulaire de l'architecte technique.

par Silver Nakache 18/09/2006

1- Framework ! Vous avez dit Framework ?

Ces dernières années ont vu fleurir les frameworks de toutes sortes avec notamment le .Net Framework de Microsoft. Avant d'entamer la discussion, je souhaite recentrer le sujet sur la bonne définition des différents « Frameworks».

1.1- "Application Framework" = "Plateforme de Développement"

Définition Wikipédia d'un « Application Framework » :

> « In computer programming, an application framework is a term usually used to refer to a set of libraries or classes that are used to implement the standard structure of an application for a specific operating system. By bundling a large amount of reusable code into a framework, much time is saved for the developer, since he/she is saved the task of rewriting large amounts of standard code for each new application that is developed. »
> Lien Wiki complet: http://en.wikipedia.org/wiki/Application_framework

Le .Net Framework est en fait un sur ensemble de Software Framework qui permet d'implémenter différentes solutions logicielles au même titre que la plateforme Java J2EE.
Ces plateformes sont qualifiées d'« Application Framework ».

1.2- « Software framework »

Microsoft s'est approprié le terme de 'Framework' pour qualifier une plateforme de développement que l'on ne doit pas confondre avec un « Software Framework ».

Pour clarifier les choses, voici deux définitions d'un « Software framework »:

Définition du Gang of Five [Gov] :

> « Architectural Pattern: Expressing a fundamental structural organization schema for software systems.
> It provides a set of predefined subsystems, specifies their responsibilities, and includes rules and guidelines for organizing the relationships between them. »
> Source : Pattern-Oriented Software Architecture, Volume 1: A System of Patterns
> by Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad, Michael Stal " ISBN 0-471-95869-7

A noter que le GoV exprime ici un « Software Framework » comme « Architectural Pattern » ce qui est, stricto sensu, la même chose.

Définition du Wikipédia:

> «A software framework is a reusable design for a software system (or subsystem). This is expressed as a set of abstract classes and the way their instances collaborate for a specific type of software (Johnson and Foote 1988; Deutsch 1989). All software frameworks are object-oriented designs. »
> Lien Wiki complet: http://en.wikipedia.org/wiki/Software_framework

Un «Software Framework » (Architectural Pattern terminologie GoV) capture une logique, une méthode récurrente et la matérialise sous forme de « Classes abstraites », donnant la possibilité aux développeurs de spécialiser un comportement pour l'intégrer dans son logiciel. Le SF est donc régi par des lois qui caractérisent la méthodologie que l'on souhaite implémenter.

Dans l'équivalence du langage Microsoft nous avons donc « Software Framework » = « Application Block ».

Dans les « Software Framework » les plus connus nous retrouvons « Stouts » par exemple, où le pattern MVC a été adapté pour servir des applications WEB.

Le .Net Framework, comme la plateforme J2EE, sont des « Application Frameworks » qui fournissent un ensemble de classes destinées à augmenter la productivité des développeurs.

En résumé :

Terminologie employée	Exemple	Commentaires
Software Framework	Framework maison, Composite UI Application Block, Cachin Application Block, Stouts	Constitue un axe, une fonctionnalité que l'on ajoute au logiciel.
Application Framework	.Net Framework, J2EE,	Constitue un socle technique

Cette petite mise au point effectuée, je vais maintenant vous parlez de « Software Framework ».

2- Comment utiliser un « Software Framework » maison pour formaliser les développements dans l'entreprise ?

Comment fédérer les développements dans une entreprise ? Gérer les flux d'informations ? Augmenter la qualité des processus métiers ? Faire les meilleurs choix technologiques ? Etc.…

On trouve une réponse à ces questions dans les solutions qu'utilise l'architecte au quotidien : utilisation de logiciels de cartographie applicative, langage UML, méthodes agiles, tests automatisés, usine de développement, mais aussi le 'Framework maison'.

Pour atteindre l'objectif de 'fédérer les développements', il est bienvenu pour l'architecte de mettre ses talents d'abstraction au service des autres développeurs afin de structurer leur développement à l'aide d'un « framework maison ». En mettant en application ses qualités relationnelles, ses capacités de communication et ses aptitudes d'expert technique, il est le mieux placé pour le développer.
L'architecte doit le structurer sous forme de services métiers propre à l'entreprise, de couches d'abstraction visant à simplifier l'élaboration et la construction de ces services métiers et d'un système de communication inter-services efficace.

Par son action, l'architecte :

1- Matérialise son expertise dans le/les 'framework(s)' qui l'aidera/ont à faire évoluer et à maîtriser les futurs développements.
2- Encapsule une méthodologie en tissant des liens comportementaux entre les classes.
3- S'assure que les développements concrets (Classe dérivées) sont régis par un ensemble de lois conforme à la méthodologie.
4- Assure la cohérence des services et composants métiers dans le système d'information.

Quand l'écriture du framework est réussie, son adoption par les développeurs est facilitée par le simple fait que les tâches quotidiennes en deviennent moins répétitives, plus rapides à développer. Les développements des composants métiers deviennent plus faciles à intégrer, les services communiquent facilement entre eux par l'intermédiaire du framework. C'est le cas des actions de monitoring par exemple, chaque service s'intègre au framework, obéissant aux mêmes règles de fonctionnement. L'ajout de nouveaux aspects (cf. : AOP) devient un jeu d'enfant.

3- Le Framework comme 'outil de communication' :

Quand l'architecte a 'réussi son coup', c'est-à-dire que son framework permet de créer des services de manière fédérée pour l'entreprise ; alors le framework agit comme outil de communication au sein des équipes de développement. Le développeur d'un service métier particulier peut être sollicité par d'autre application ayant besoin de produire les mêmes fonctionnalités. L'objectif de mutualisation est alors atteint.
La construction de nouvelles applications devient un assemblage de 'blocs' permettant le jeu de lego applicatif attendu. Ainsi, on comprend mieux l'appellation « Applications blocks » de Microsoft.

Le SF joue donc son rôle de communication à différents niveaux :

- Au sein de l'équipe de développement : entre les développeurs car les développements sont basés sur un socle technique commun.
- Entre l'architecte et l'équipe de développement,
- Entre la demande business et du système d'information.

L'architecte l'utilise aussi pour créer une émulation autour de sa nouvelle plateforme.

4- Le « Framework maison » améliore la lisibilité du Système d'Information:

Pour l'architecte qui réussit l'intégration de 'frameworks maisons' dans son entreprise, c'est un bénéfice direct pour la 'lisibilité' du système d'information.
Il permettra notamment :

1- Aux développeurs d'intégrer et de diffuser à moindre coût de nouvelles fonctions aux autres développeurs.
2- De fournir des abstractions sous forme de « blocs » logiques que l'on verra apparaître dans les schémas d'urbanisation du SI.
3- De faciliter le dialogue avec les fonctionnels qui identifieront clairement les différents aspects sur logiciel.

Ce que je reproche souvent à la méthode UML, c'est de ne pas avoir comblé le « gap » qu'il existe entre un « Cas d'utilisation » et le code produit. Bertrand Meyer, auteur du célèbre langage Eiffel, balaye d'ailleurs d'un coup de crayon (Dans son livre « Conception et Programmation orientées objet» ISBN: 2212091117) les « cas d'utilisation » d'UML en indiquant qu'il peut aboutir à du code radicalement différent de la solution désirée.

Je vois, pour ma part, une partie de ce « gap » comblée par l'utilisation de ces « frameworks » qui aident à combler le fossé.

5- Multiplier les 'frameworks maison' dans l'entreprise : 'Oui mais attention'

Si la recette fonctionne, alors pourquoi ne pas multiplier les 'frameworks' dans l'entreprise ?
La réponse est : 'Oui mais attention', « L'outil framework » repousse les blocages rencontrés au sein d'un ou plusieurs applicatifs. Les SF doivent être conçus de manière à pouvoir fonctionner entre eux pour ne pas retomber dans les mêmes travers, rencontrés à petite échelle. Le « attention » vient aussi et surtout du fait que chaque framework constitue un socle de service commun, donc une fois déployé en entreprise, toucher au socle signifie toucher aux fondations.

L'architecte devra ainsi veiller à ne pas utiliser de technologies trop propriétaires, mais plutôt choisir des technologies pérennes largement éprouvées et qui s'intègreront facilement à d'autres produits du marchés.

Toucher aux fondations peut être risqué en terme d'évolution. Il s'agit de minimiser les risques en ayant un système de tests suffisamment complet, à la Nunit par exemple, pour ne pas se priver d'une réécriture même importante du framework, sans affecter les applications en production. (Cf. : méthodes Agiles)

Dans une logique d'intégration l'inventaire des autres 'frameworks' développés dans l'entreprise est aussi important pour le succès futur du Framework en développement.

En Conclusion :

En clarifiant le terme de Framework dans ses différentes formes, nous avons pu aborder le bénéfice que peut apporter ce type d'outil en terme de mutualisation des développements, de communication et de lisibilité du système d'information.

Repousser les limites ne veut pas dire pour autant perdre de vue le fait que d'autres limites ne se présenteront pas un peu plus loin. C'est pourquoi, ces socles doivent être développés avec le plus grand soin, avec comme possibilité ultime, la réécriture de socle lui-même sans impacter les applications en production : jeux d'interfaces formant un contrat clairement défini. Il est nécessaire d'effectuer des tests suffisamment étoffés (inspirés des méthodes agiles) et d'avoir une bonne vision des technologies de communication. Il est aussi important de pratiquer un audit précis des tentatives réussies et échouées d'autres frameworks dans l'entreprise.

Le choix de la technologie de base, c'est à dire les plateformes logicielles qui serviront à fabriquer le socle, telles que le .Net framework de Microsoft, la plateforme Java ou d'autres bases, est aussi déterminant dans le succès du futur socle.

Le cercle vertueux n'est pourtant pas facile à atteindre. La construction du framework butte souvent sur des problèmes techniques assez difficiles à surmonter et reste réservée aux développeurs ayant de bonnes capacités d'abstraction et aux architectes ayant un recul suffisant vis-à-vis des technologies.