
[doXulting]
faites travailler vos documents

Méthodologie de mise en place de solutions libres en bibliothèques universitaire

Ludovic MECHIN
doXulting

4 juin 2009

[doXulting]
faites travailler vos documents

Sommaire

- Spécificités d'un projet d'implantation d'un logiciel libre ou open source
  - Comment les logiciels libres s'intègrent-ils dans les systèmes d'information des professionnels de la documentation et des bibliothèques ?
  - En quoi un projet d'implantation d'un logiciel libre ou open source diffère de celui d'un logiciel propriétaire ?

- Méthodologie de conduite d'un projet libre
  - La nécessaire analyse préalable : pourquoi l'aspect budgétaire ne peut être le seul critère de décision
  - Comment gérer le temps du projet ? : maquettage et démarche itérative, constitution de l'équipe projet et évaluation des charges de travail.
  - Comment recenser et évaluer les communautés de développement libre ?
  - Postes d'économie et de dépense : maturité du projet libre et retour sur investissement
  - études de cas

Spécificités d'un projet Open source : intégration dans les systèmes d'information / les familles de logiciels libres « documentaires »

SIGB

Koha
PMB
Evergreen

CMS documentaires + ou - orientés portail

Spip(php mysql)
Alfresco(Java)
Nuxeo (J2EE)
Freedom, Drupal

Infrastructure

Lucene
(moteur de recherche)

Utilitaires

Editeurs XML,
gestion de bibliographie,
Gestion d'enquête

Spécificités d'un projet Open source : les 4 têtes de l'utilisateur de logiciels libres

Le « malgré-lui »

Le « sans le savoir »

Le volontaire

Le « faute de mieux »

Spécificités d'un projet Open source : les 4 profils de l'utilisateur volontaire

Bricoleur

Radin

Opportuniste

Idéaliste

Spécificités d'un projet Open source :
typologie des projets open source

Développement collaboratif

On développe les fonctions manquantes que l'on reverse à la communauté
Type de produit : applications métier immatures ou besoins spécifiques

Ressources : Architecture technique (matériel, réseau…), développeur interne (missionné officiellement, compétent et disponible) ou prestataire de service, communauté « intégrante » et dynamique.

Coûts : Forfait de développement. Economie sur le coût de licence

Précautions : en phase « amont » : étude très sérieuse des fonctionnalités, de la vitalité de la communauté et de ses règles de contribution. Savoir estimer le temps de développement et contrôler le respect de cette estimation

Risque de dérive coûts/délai si le volume de développement est sous estimé. Risque d'isolement si la communauté refuse d'intégrer les modifications (fork).

Spécificités d'un projet Open source : typologie des projets open source

« Sur étagère »

le logiciel est accepté tel que sans modifications.

Type de produit : Utilitaires, applications matures ou CMS (si les besoins sont modérés et très classiques

Ressources : Architecture technique (matériel, réseau…), ressources internes + aide de la communauté pour l'installation.

Coût : très modéré, voire nul, sauf si l'on fait appel à une assistance externe pour la mise en oeuvre (paramétrages, formations…)

Précaution : en phase « amont » étude très sérieuse des fonctionnalités

Risque : si un manque fonctionnel apparaît en cours ou à l'issue du projet : dérive budgétaire ou retour arrière (temps perdu)

Spécificités d'un projet Open source :
typologie des projets open source

Intégration

Plusieurs logiciels Open Source sont associés sous une même interface graphique/ergonomique/fonctionnelle

Type projet : Infrastructure, applications ou CMS avec besoin spécifique à couvrir

Ressources :développeur interne (missionné officiellement, compétent et disponible) ou prestataire de service très compétent en open source.

Coût : forfait de développement et d'intégration. économie du coût de licence

Précaution :Très bonne connaissance des logiciels de base, de leur interopérabilité, de la compatibilité de leurs licences. Savoir estimer le temps de développement et contrôler le respect de cette estimation.

Risque de dérive coût/délai si le volume de développement est mal estimé.

Un choix fondamental : s'éloigner ou non du standard

Cas de développements additionnels reversés (et intégrés par la communauté) :

Fonctionnement classique d'une communauté, les modifications sont intégrées au logiciel et seront disponibles dans les nouvelles versions.

Cas de développements additionnels non reversés ou non intégrés par la communauté :

Renoncement aux nouvelles versions. éventuellement création d'un fork.

Ou

Documentation rigoureuse des additifs et réintégration des ajouts à chaque chargement d'une nouvelle version. Coût récurrent

Spécificités d'un projet Open source : synthèse

Spécificités d'un projet open source

identification et évaluation des logiciels

- évaluation de la communauté
- choix du reversement ou non des modifications
- absence de relation client/fournisseur dans le cas d'un travail direct avec la communauté
- si prestataire (SSLL) il ne s'engage que sur ses prestations, et non sur l'évolution du logiciel

Similitudes avec un projet "propriétaire"

- analyse des besoins
- gestion projet des développements
- relation client/fournisseur dans le cas d'un recours à une SSLL

méthode de projet :
relativité du critère budgétaire

Dans un projet open source les critères budgétaires ne sont pas toujours déterminants

- coûts d'investissements moindres
  - mais dichotomie charges internes / charges externes
- coûts de licences nuls ou presque (cas des open source à distribution payante)
  - mais des droits attachés aux licences qui peuvent être complexes
- il faut se méfier des coûts cachés
  - investissement temps humain
  - coût interne ou externalisation (études et développements)
  - des mises en œuvre qui s'allongent dans le temps

méthode de projet : relativité du critère budgétaire

Type de coûts	Solutions propriétaires : en % du total des coûts	Solutions libres : en % du total des coûts	Comparaison des coûts bruts
Coûts d'investissement			
Matériels et système d'exploitation	10 à 20 %	20 à 30 %	Coûts équivalents
Hébergement externalisé de l'application	0 à 2 % par an	0 à 2 % par an	Coûts équivalents, peuvent être légèrement supérieurs pour les solutions propriétaires reposant sur des technologies ou plateformes non standard
Logiciels substrats (SGBDR, Middleware…)	0 à 15 %	0 %	Supérieurs pour les solutions propriétaires
Logiciel métier (SIGB, logiciel documentaire, logiciel d'archives, portail, outils multimédia…)	20 à 30 %	0 %	
Prestations externalisées (reprises, paramétrages, formations…)	50 à 70 %	10 à 80 %	Equivalents, peuvent être supérieurs pour les logiciels libres
Prestations assurées en interne et autres coûts cachés	10 à 50 %	20 à 100 %	Supérieurs pour les logiciels libres
Coûts récurrents			
Maintenance logicielle	2 à 5 % par an	0 à 30 % par an	Peuvent être supérieurs pour les logiciels libres
Veille sur le produit et autres coûts cachés	1 à 5 %	10 à 20 %	

méthode de projet : relativité du critère technique

Les critères techniques ne sont plus déterminants

- Il est très rare que des services informatiques s'opposent aujourd'hui techniquement aux solutions libres, puisqu'ils en utilisent de fait (Linux, Apache, Firefox, MySql, Postgresql)
- Les éditeurs de logiciels propriétaires intègrent des briques libres
- Même le monde Mac s'émancipe avec Mac OS X (implémentation libre d'Unix)
- Enfin la caractéristique principale des logiciels open source est d'être intér-opérables et basés sur des technologies ouvertes

[doXulting]
faites travailler vos documents

méthode de projet : importance du critère organisationnel

Dans un projet open source le critère organisationnel est crucial

- un projet open source ne peut être mené sans équipe
- il n'est plus possible de dériver la responsabilité sur un éditeur (externalisation de la responsabilité)
- même si le retour en arrière est toujours possible : une maquette ne devient pas un prototype si ce dernier n'est pas porté par une équipe convaincue et opérationnelle
- **Conclusion** : le temps passé par les équipes est plus important. On y gagne en adhésion et motivation, on y perd en ressources mobilisées

méthode de projet : analyse préalable

Dans un projet open source on ne peut pas faire l'économie d'une analyse préalable

- même si un projet open source n'entre pas dans les "canons projets" (budget prévisionnel, cahier des charges, mise en concurrence, validation du choix par une commission d'AO)

Cette analyse préalable doit répondre aux questions suivantes :

- l'outil pressenti répond il aux besoins minimaux ?
- les contraintes ont-elles été analysées ?
- le contexte organisationnel a-t-il été pris en compte ?
- si cette étude est externalisée, elle constitue un coût à prendre en compte

- Le temps du projet, moins d'urgence, plus long

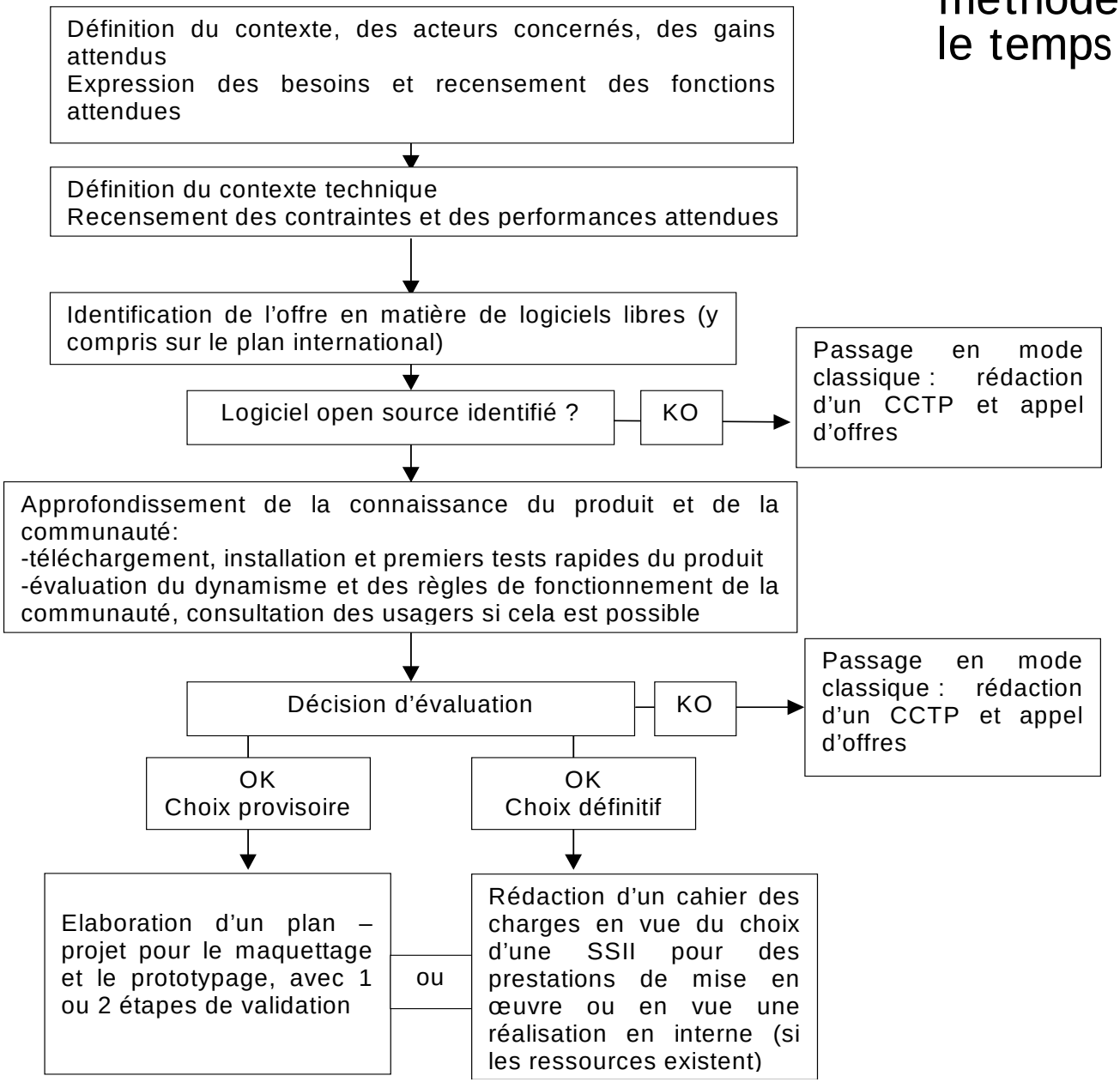
- Maquettage facilité (pas besoin de prêt de logiciel) et démarche itérative : prototypage beaucoup plus facile qu'avec des logiciels propriétaires, part de risques moins importante (benchmarcking, montée en charge)

- Caractériser l'équipe projet : bibliothécaire, informaticiens

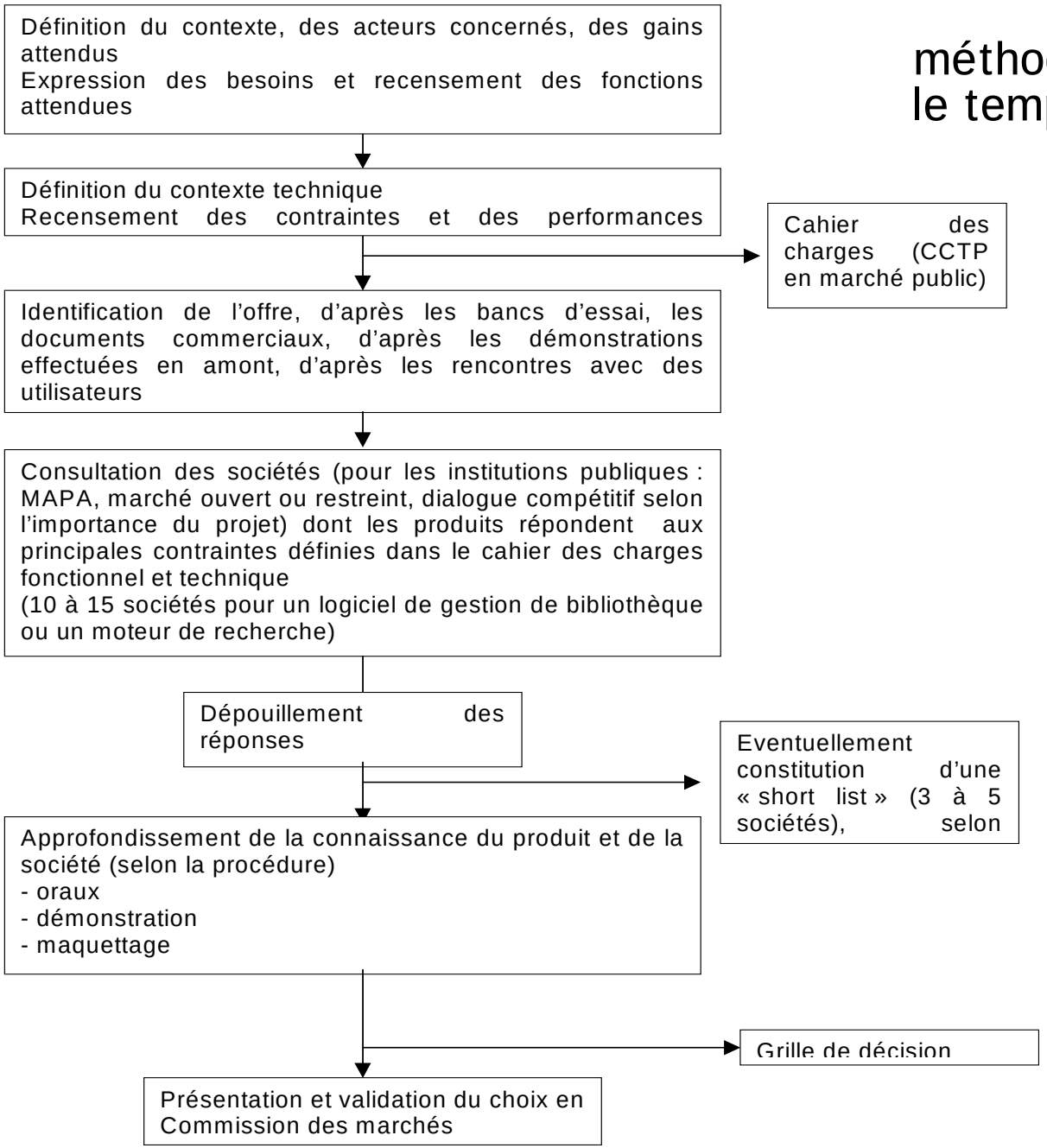
- Evaluer les charges de travail : reprise des données, formation, paramétrage, scénarisation étape par étape

- Impact sur l'appel d'offres (démarche open source pressenti / démarche classique)

méthode de projet : le temps du projet / les jalons



méthode de projet : le temps du projet / les jalons



[doXulting]
faites travailler vos documents

méthode de projet : évaluer les communautés

- Evaluer la communauté
  - site web, forums, listes de diffusion

- Critères d'évaluation
  - Mesurer le nombre et la variété des participants
  - Vérifier la proximité d'intérêt des institutions concernées
  - S'assurer de l'importance et du positionnement des développeurs
  - Valider que les procédures de contributions des uns et des autres est bien formalisée

- Identifier les compétences
  - SSLL
  - consultants
  - équipes informatiques dans des établissements
  - croiser avec des critères thématiques (qui fait quoi en terme de maintenance évolutive, de reprise de données, d'intégration, …)

Tout dépend du degré de maturité du projet

- les projets pionniers ne sont pas forcément plus coûteux en investissement, l'objectif est le retour sur investissement **à long terme, à condition**
  - de drainer de nouveaux entrants sur la solution (pas de fork)
  - de partager les développements (et leurs coûts)

- l'investissement humain est toujours très important, quel que soit le projet et il est rarement chiffré
  - vacataires
  - gens du métier versés dans l'informatique (bibliothécaires,documentalistes)
  - informaticiens de métier

- à long terme, la rigidité des contrat de maintenance, qui est le propre des logiciels propriétaires, permet de consacrer, le cas échéant, des moyens à une véritable maintenance évolutive

- la présence d'une communauté active permet de disposer de modules qui seraient chiffrés auprès d'éditeurs propriétaires

méthode de projet : étude de cas / Ecole des Mines

- Contexte favorable
  - Environnement technique hétérogène
  - Logiciel non maintenu
  - Enveloppe budgétaire limitée
- Décisions affirmées
  - Analyse préalable (l'essentiel, le superflu)
  - L'harmonisation des données
  - L'observation de Koha
- La démarche
  - Installation du logiciel
  - Maquettage itératif, tests, pédagogie
  - Développements, réflexion sur le support
  - Mise en exploitation définitive au bout de 15 mois
- Analyse
  - Une équipe impliquée
  - Un projet assez coûteux en assistance et en développement mais plus économique que s'il y avait eu appel d'offres
  - Un effet d'entraînement notable
  - Un retour sur un investissement qui se profile

méthode de projet :
étude de cas / Universités d'Aix-Marseille

- Contexte
  - Etude préalable pour la gestion commune des bibliothèques
  - 2 options : 1 SIGB / 3 SIGB et une couche fédératrice
  - Choix d'un SIGB unique avec un périmètre précis (couches basses)
- Préparation du cahier des charges
  - Visites de sites
  - Démonstrations produits
  - Intérêt pour le libre (VP mais aussi bibliothécaires)
- Lotissement du cahier des charges
  - Un lot ferme prototype (20 bibliothèques)
  - Un lot optionnel : toutes les bibliothèques
  - Souplesse du Cahier des charges de manière à ce que des prestataires libres puissent répondre
- Un libre en BU
  - Choix de koha…et de BibLibre

méthode de projet : étude de cas / Conseil Général du Jura

- Contexte

- Dès le départ le périmètre fonctionnel est cadré : permettre aux jurassiens de localiser un ouvrage dans le réseau de lecture publique et d'effectuer des demandes de prêts entre bibliothèques en ligne
- Après un tour d'horizon, le choix se porte sur un produit open source (MoCCAM)
- mais nécessité de développer les fonctions manquantes (PEB)

- La démarche

- Appel d'offre pour le choix d'un prestataire pour la mise en œuvre (installation, paramétrage + développement de fonctions manquantes),
- Projet long (difficultés sur l'architecture technique notamment) : 2 ans
- Fort investissement de l'équipe de la BDP (Conseil Général)
- Participation active à la communauté : les développements payés par le Jura sont ou seront reversés à la communauté

- Analyse et bilan financier

- Coût global : 100 000 € (non compris charges internes)
- Coûts récurrents : 1 800 € HT / an
- Principal problème : le logiciel choisi n'était pas encore vraiment utilisé et la communauté était très réduite et peu active en échanges ; tâtonnements et pertes de temps
- Au final, un coût équivalent à celui d'une solution propriétaire

conclusion

- dans une période pionnière, le coût d'un projet open source n'est pas nécessairement déterminant par rapport à une solution « propriétaire »
- le retour sur investissement n'est pas immédiat
- solution non viable si non portée par une équipe
- l'avenir du projet open source (retour sur investissement rapide) réside donc dans la mutualisation (consortium, groupements d'achat..)
- ...ce qui est dans la logique communautaire de l'open source