Logiciel Libre
Cours 9 — Modèles Économiques

Stefano Zacchiroli
zack@pps.univ-paris-diderot.fr

Laboratoire PPS, Université Paris Diderot

2013–2014

Rappel du cours 8

- Comment fonctionne un projet de développement dans le monde du logiciel libre ?
- *La cathédrale et le bazar*
- Le cycle de vie d'un projet du logiciel libre
- L'infrastructure de développement
- Comment contribuer ?

Contenu cours 9

Est-ce que le logiciel libre a un sens économique ?

- Quelques fausses idées.
- Le point de vue producteur/vendeur.
- Le point de vue développeur.
- Le point de vue utilisateur/client.
- Le point de vue macro-économique.

Outline

Outline

Quelques fausses idées (1)

« Logiciel propriétaire = logiciel professionnel »

Réponse :

- un logiciel propriétaire n'est pas forcement professionnel
- un logiciel professionnel peut très bien être libre, et il y a des bonnes raisons pourquoi les professionnels peuvent préférer le logiciel libre - voir plus tard dans ce cours.

Quelques fausses idées (2)

« Logiciel libre = bricolage »

Réponse :

- la question libre ou pas est a priori indépendante de la qualité
- il est vrai qu'il y a beaucoup de petit projets de bricolage dans le logiciel libre
- mais il y a aussi beaucoup de logiciels libre de très bonne qualité, et il y a des bonnes raisons pourquoi le logiciel libre a le potentiel d'être d'une excellente qualité.

Quelques fausses idées (3)

« Le logiciel libre n'est pas utilisable par des professionnels car il n'y a pas de garantie »

Réponse :

- Dans le monde propriétaire il n'y a pas de garantie non plus, sauf si on paye cher.
- On peut acheter un service de maintenance, et cela beaucoup plus facilement dans le cas du logiciel libre.

Quelques fausses idées (4)

« Le logiciel libre est dangereux pour la sécurité car tout le monde peut facilement trouver les failles de sécurité. »

Réponse : voir plus tard dans le cours

Quelques fausses idées (5)

« Si tout le monde ne fait que du logiciel libre, alors tous les informaticiens seront au chômage »

Réponse :

- La production de logiciels vendus sur le marché ne font qu'une petite partie de l'activité informatique (voir plus tard dans le cours).
- Pourtant, on ne deviendra pas l'homme le plus riche de la planète par le logiciel libre.

Outline

Le modèle classique

- Dans le monde des logiciels *propriétaires* :
  Revenus basé sur la vente de *licences d'utilisation*.
- C'est un modèle économique qui découle de l'idée de la *rareté* des biens vendus (vente de biens matériels).
- Production de logiciels : coût élevé de développement, coût de la copie très bas (voir nul).
- C'est un modèle de *rente* qui peut être extrêmement profitable.
- Problème similaire pour la musique, des films, ...

Modèles économiques du logiciel libre

1. Les modèles basés sur le *développement* payant à la place de la *vente*.
2. Les modèle basés sur la vente de logiciels : le *semi-libre*, coexistence de versions libres et de versions propriétaires.
3. Les modèles qui ne sont pas basés sur la vente des logiciels mais sur des services « annexes ».

Modèles économiques du logiciel libre

1. Les modèles basés sur le *développement* payant à la place de la *vente*.
2. Les modèle basés sur la vente de logiciels : le *semi-libre*, coexistence de versions libres et de versions propriétaires.
3. Les modèles qui ne sont pas basés sur la vente des logiciels mais sur des services « annexes ».

En vérité il y a autant de modèles qu'il y a d'acteurs.
Dans la suite plutôt des *éléments* de modèles économiques.

Développement payant

- Un client a besoin d'un certain produit logiciel.
- Il peut embaucher un développeur, ou une entreprise (éditeur de logiciel) qui développe le logiciel.
- Le client (qui est à priori propriétaire du logiciel qu'il a fait développer à ses frais) peut choisir de publier le produit sous licence libre pour des raisons diverses.
- Exemple : Développement du compilateur ADA dans les années 80 sous licence libre.

Les modèles basés sur la vente de logiciels

- On peut changer la licence libre choisie pour la rendre plus restrictive, ou alors séparer clairement les groupes des utilisateurs, pour se garantir une exclusivité d'exploitation commerciale.
- Basés sur la coexistence de versions libres et de versions non libres du logiciel (modèle hybride).
- Ce sont des modèles qui sont parfois dans l'esprit du libre, mais pas toujours.

Le modèle des licences décalées

- Idée : Vendre des licences pour les versions les plus récentes et les plus à jour à des clients qui ont des besoins poussés.
- Après un certain temps, les logiciels tombent dans le domaine du logiciel libre (« licence biodégradable »).
- Les versions libres profitent des avantages de la grande communauté.
- Exemple : L'ancien modèle de *Ghostscript*, un interpréteur du langage Postscript (description de la mise en page, utilisé par exemple par les imprimantes).

L'exclusion d'usage commercial

- Exemple : la licence utilisé par *Trolltech* (version 1)
- Le vendeur essaye de cette façon de profiter de la communauté pour améliorer son produit.
- C'est contre les principes du logiciel libre (et directement exclu par les critères de Debian)

Le modèle double licence

- Publication sous licence libre GPL (qui exige que tous les produits dérivés sont publiés sous licence GPL).
- Vente de licences commerciales qui permettent l'utilisation du logiciel dans des produits qui ne seront pas publiés sous GPL.
- C'est absolument en accord avec les critères du logiciel libre (même si on peut avoir des avis partagés si ça correspond à son esprit).
- Exemple : *MySQL*, une des systèmes de bases de données les plus importants (entreprise rachetée par Sun pour 1.000.000.000 USD, et ensuite par Oracle).

L'exclusion de la concurrence

- La licence permet accès au code source et l'utilisation gratuite, mais interdit de développer un produit concurrent.
- Ce n'est bien sûr *pas* libre.
- Exemple : *Bitkeeper*, un logiciel pour le contrôle de versions (essentiel pour les projets de développement de logiciels). On a pas la droit d'utiliser Bitkeeper si on contribue à des produits concurrents.

Vente de sièges dans un consortium

- *OCaml* : langage de programmation développé par INRIA.
- Publié sous licence libre, mais développement historiquement plutôt dans le mode « cathédrale ».
- Les membres du consortium ont le droit d'utiliser OCaml sous une licence différente.
- Appartenance au consortium donne une influence directe sur le futur développement du système.
- À partir de 2000 EUR par an, actuellement 7 membres (tous des entreprises).

Les modèles basés sur services / produits «annexes»

- C'est souvent la partie la plus importante !
- Distribuer un produit sous licence libre pour mieux vendre un autre produit.
- Les distributeurs.
- Les prestataires.
- Vente de produits connexes.

Mieux vendre un autre produit : Exemple Adobe

- Adobe Systems Incorporated : multimédia et publication assistée par l'ordinateur (desktop publishing)
- Vente de logiciels propriétaires pour la production de contenus (par exemple, Acrobat Acrobat pour le format PDF)
- Les outils nécessaires pour la *lecture* sont gratuits (par exemple, Acrobat Reader), ce qui permet d'assurer que les utilisateurs peuvent lire les documents produits.

La lutte pour les quasi-standards

- *Quasi-standard* : un standard *de fait*, en opposition à un standard *de jure* (porté par un organisme avec le pouvoir de définir des standards).
- Logiciel libre : la grande disponibilité d'une technique peut mener à la formation d'un quasi-standard, ou empêcher un concurrent à établir un quasi-standard basé sur une autre technologie (peut-être propriétaire).
- Exemple : La décision de *Netscape* de convertir son navigateur vers un produit libre (*Mozilla*), lutte contre la menace d'un quasi-monopole de Microsoft (enjeu : le marché des serveurs).

Les distributeurs

- Ne développent pas eux mêmes des logiciels (au moins ce n'est pas leur fond de commerce) mais fournissent des collections de logiciel.
- Parfois des versions de base gratuites et sous licence libre, et des versions « à valeur ajoutée » payantes et sous licence propriétaire.
- Historiquement (avant la vulgarisation d'internet à haut débit) : vente des distributions sur supports (CD plus documentation).
- Les valeurs ajoutées : des logiciels non libres, mais surtout : installation, service de maintenance, formation.
- Exemple : SUSE, Mandriva, RedHat, Canonical.

Exemple distributeur : RedHat

- Fondé en 1994.
- Chiffre d'affaires : 1.130.000.000$ en 2012.
- Résultat net : 199.000.000 $ en 2012.
- Nombre d'employés : 6.100 en 2014.
- Vente de support : entre 350$ et 2.500$ par an et serveur.
- Basée sur une distribution gratuite et libre : *Fedora*
- 82% des revenus (2012) viennent de la vente de contrat de support, le reste de la formation et du conseil.
- 18% des coûts (2012) sont pour le développement Open Source (les résultats reviennent donc à la communauté).

Les prestataires

- Le logiciel est sous licence libre.
- Vente de services autour du logiciel : conseil, maintenance, assemblage et configuration de solution logiciels à mesure formation.
- Arguments de vente : grande expertise, prestation de proximité.
- Intégration de solutions Open Source, ou prestation autour d'un produit spécifique.

Exemple : Sendmail Inc.

- Sendmail : un *serveur* de mail, logiciel qui s'occupe de la *transmission* des messages.
- C'est une technologie très sensible aux problèmes de sécurité (comme les serveurs web).
- Le logiciel est sous licence libre (GPL).
- Configuration très complexe.
- Vente de services aux entreprises : conseil, configuration, formation, aussi des outils propriétaires.

Exemple : Triade

- Solution de gestion d'école : planning, suivi des élèves, forums de discussion, communication avec des parents, ...
- Le logiciel est sous licence libre.
- Vente de service : Hébergement des sites pour des écoles, personnalisation, sauvegarde, ...
- C'est le modèle *SaaS : Software as a Service*

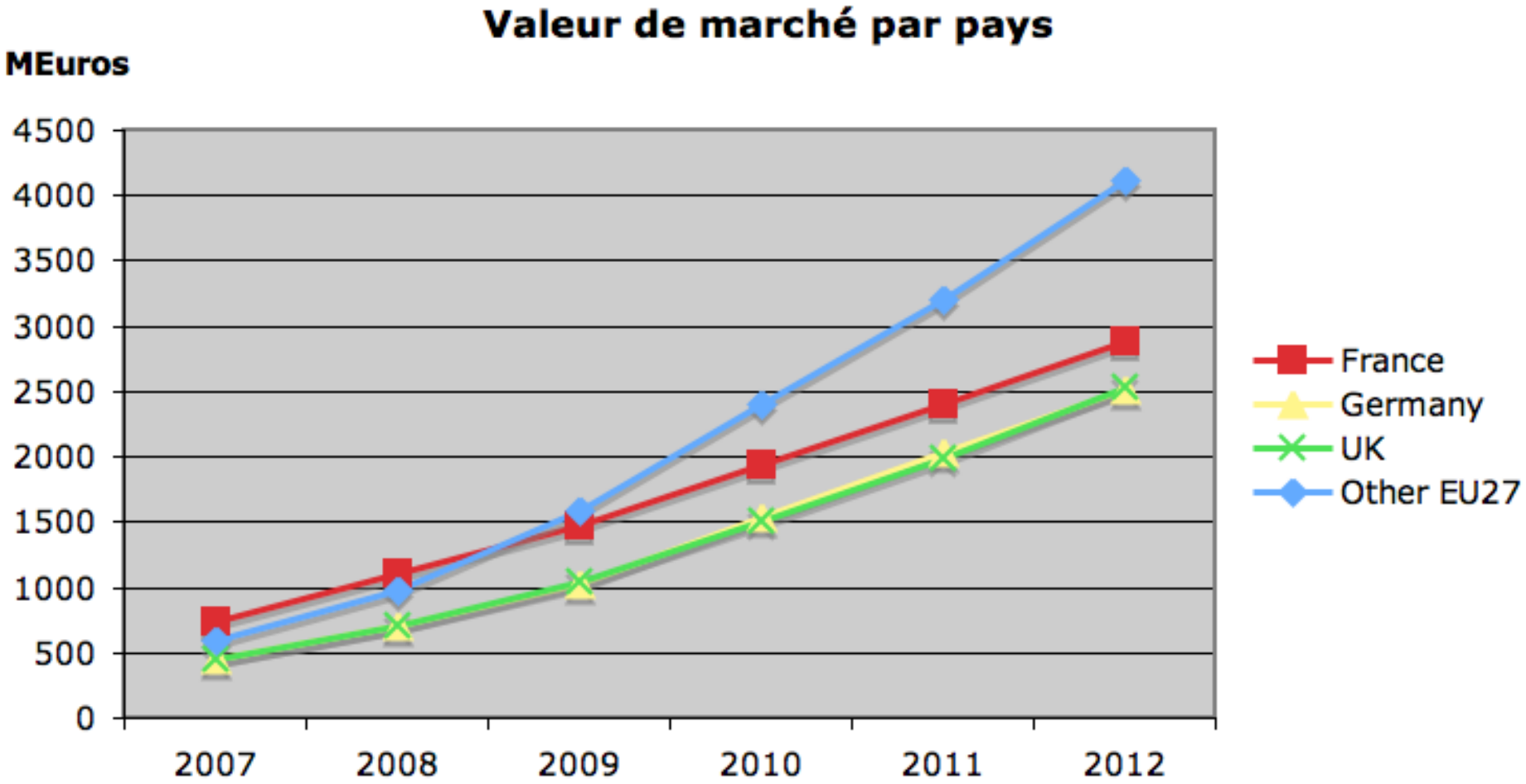
Exemple : Doom

- Jeu créé en 1992, technologie d'animation très innovante.
- Code source publié en 1997, licence GPL en 1999.
- Améliorations du code source et portage vers des nouveaux architectures d'ordinateurs par la communauté.
- Depuis : revenus de l'entreprise *Id Software* basés sur la vente de *scénarios*.

Vente de produits autour du logiciel

- Exemple phare : O'Reilly Media.
- Maison d'édition de documentation de logiciels libre.
- Déteneur des droits d'auteur pour des livres publiés (limité à 28 ans).
- A embauché un certain nombre de personnalités célèbres du logiciel libre.
- Essentiel : des bonnes relations avec la communauté.

Le marché pour le Logiciel Libre



Source : Pierre Audoin Consultants, 2012

Outline

Raisons du dévelop./client qui commande du logiciel

Ici le client va devenir propriétaire du logiciel, quelles peuvent être ses raisons pour le publier sous une licence libre ?

- Profiter de la communauté pour la révision de code, pour la détection et la réparation des erreurs, pour des améliorations.
- Disponibilité pour l'enseignement : formation des étudiants sur une certaine technologie.
- Mutualisation des efforts en Recherche & Développement (voir le groupe de travail *Logiciel Libre* dans le pôle de compétitivité *System@tic*).

Raisons du client contre une licence libre

- Le logiciel est la réalisation d'un secret industriel essentiel pour le client (Exemple : un algorithme très sophistiqué pour résoudre un certain problème).
- Le logiciel utilise des secrets industriels (Exemple : des pilotes pour des périphériques informatiques - souvent pas une bonne raison car les secrets derrière peuvent être découvertes par rétro-ingénierie).
- Empêcher des concurrents de profiter de la même technologie.

Outline

Raisons du client en faveur d'une solution libre

Quelles sont les avantages d'un client pour utiliser une solution logiciel libre au lieu d'acheter une licence pour une solution propriétaire ?

- Profiter de la communauté : inspection du code, contributions, disponibilité dans l'enseignement, établissement d'un quasi-standard (voir au dessus pour le cas d'un client qui commande un logiciel).
- Pas de dépendance d'un vendeur unique (*vendor lock-in*).
- Assurance contre le cas de disparition d'un vendeur.
- Sécurité.

Sécurité ?

Il y a deux questions :

1. Est-ce que le logiciel contient des failles de sécurité qui ont simplement échappé à la vigilance du développeur ?
2. Est-ce que je peux faire confiance au développeur qu'il n'a pas volontairement introduit des fonctionnalités cachées qui vont contre mes intérêts de sécurité ? (par exemples des logiciels espion – *spyware*)

Une fausse idée

« Le fait que le code source est librement disponible permet à tous les voyous de trouver facilement, et d'exploiter, les failles de sécurité. »

- C'est l'idée de la *sécurité par l'obscurité* (on gagne en sécurité quand on cache le fonctionnement).
- Il suffit de regarder le nombre de virus etc. contre un système d'exploitation bien connu dont le code source est censé être un secret . . .

Pourquoi c'est une mauvaise idée

Sécurité par l'obscurité ne fonctionne pas :

- Rétro-ingénierie.
- Espionnage, vente de secret industriels par des initiés.
- Révélation accidentelle de secrets.
- En cryptographie : la sécurité doit seulement reposer sur le secret d'une *clef cryptographique* (Principe de Kerckhoff).

Outline

L'argument de Richard Stallman

Logiciel propriétaire : un peu comme des routes payantes, avec un péage à chaque coin de rue.

- Permet de collecter des fonds, et de financer la maintenance
- Empêche beaucoup d'utilisateurs
- Inefficace (coût important de la collecte elle-même)
- Pour une société entière, il est moins cher de construire des routes aux frais de la collectivité.

Le point de vue macro-économique

- La valeur d'*utilisation* du logiciel est largement supérieure à la valeur de *vente*.
- L'adaptation du modèle du logiciel libre changera radicalement le marché de la *production* de logiciels.
- Les profits pharamineux basés sur les monopoles ne seront plus possibles dans ce modèle.
- Mais tous les utilisateurs, et avec eux l'économie globale, gagneront.