

The Art of UNIX Programming

Stéphane Bortzmeyer

<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 27 Juillet 2006

<http://www.bortzmeyer.org/art-unix-programming.html>

Auteur(s) : Eric Raymond
ISBN n°0-13-142901-9
Éditeur : Addison-Wesley
Publié en

Il existe de très nombreux livres de programmation. Et de très nombreux livres sur les détails pratiques de tel ou tel langage ou bien de tel ou tel environnement de programmation. Mais ce livre est original : il présente plutôt l'essence de la programmation sur Unix, pas les détails des outils mais plutôt comment faire des programmes qui soient bien conformes à l'esprit Unix.

Pas besoin de présenter Eric Raymond. L'un des meilleurs auteurs sur Unix et la programmation explique ici les principales caractéristiques d'Unix, pourquoi les programmes Unix sont comme ils sont et pourquoi il faut continuer. Par exemple, qu'il vaut mieux faire des petits programmes ("*Rule of Modularity*"), connectés par des interfaces simples (et pouvant donc être composés, comme on compose des fonctions), plutôt que des monstres comme OpenOffice, ininstallables et inutilisables sur un vieux PC et qui, surtout, ne permettent pas facilement de coopérer avec d'autres programmes.

Comme toujours avec Raymond, il présente toutes ses règles et aphorismes comme exprimant un consensus, alors que certaines sont plutôt personnelles. Des dix-sept règles qu'il présente, certaines (comme la "*Rule of Modularity*" donnée plus haut ou bien la "*Rule of Silence*", qui dit qu'un programme doit se taire, sauf s'il a quelque chose d'utile à dire) seront en effet approuvées par tous les Unixiens mais ce n'est pas le cas de toutes.

Raymond, après avoir exposés les dix-sept règles, donne ensuite un bon cours de génie logiciel, qui est utilisable dans bien d'autres environnements qu'Unix. Il couvre un très grand nombre de sujets passionnants comme la conception de mini-langages. De nombreuses études de cas ponctuent son exposé.

Il passe ensuite à la mise en œuvre : quel langage de programmation choisir, quel système de contrôle de versions, la documentation, etc.

En résumé, un excellent livre, qui devrait être lu par tous les programmeurs et chefs de projet. Cela leur serait bien plus utile que de passer du temps à faire des diagrammes avec le dernier outil graphique à la mode.