Sébastien **Blondeel**
Daniel **Cartron**
Hermantino **Singodiwirjo**

Débuter sous **Linux**

Avec la contribution de
Juliette **Risi**,
Laurent **Rathle**
et Gaël **Thomas**

ISBN : 2-212-11559-8

EYROLLES

Émulation et interopérabilité

B

Théoriquement, tous les ordinateurs sont équivalents à des machines de Turing. Concrètement, en revanche, ils présentent des incompatibilités. Heureusement, il existe des traducteurs et interpréteurs qui, au prix parfois d'une petite baisse de performances, assurent dans une certaine mesure la compréhension mutuelle.

SOMMAIRE

MOTS-CLÉS

- Émulation
- Compilation
- Interprétation
- Formats de fichiers
- Équivalences fonctionnelles

Cadre

Les ordinateurs, on l'oublie parfois, sont avant tout conçus pour servir l'homme. En particulier, tout élément frustrant et stressant, mais évitable étant donné l'état des technologies, est le signe d'une mauvaise conception. Ils doivent donc proposer une interface intuitive et conviviale à leurs utilisateurs et programmeurs - on parle de « haut niveau ».

Originellement, pour des raisons de recherche et de limites technologiques, les premiers calculateurs électroniques n'étaient capables d'effectuer que des opérations techniques et abstraites, de peu d'intérêt aux yeux du non spécialiste. C'étaient les premiers informaticiens qui faisaient l'effort d'apprentissage et de concentration pour communiquer avec les machines dans ce langage non intuitif.

Peu à peu, l'électronique et les connaissances évoluant, on a pu ajouter des couches de niveaux de plus en plus haut, chacune reposant sur des instructions de la précédente. Ainsi, on pouvait « lire un caractère » dans un fichier sur le disque sans rien connaître du fonctionnement de sa mécanique ou de son système de fichiers, une bibliothèque de « virtualisation » faisant l'interface. Plus tard encore, des fonctions « lire un nombre entier » ou « lire une ligne complète » se sont appuyées sur la fonction « lire un caractère », etc.

Chacune de ces couches de l'échafaudage est une source possible d'incompatibilités, à commencer par les couches les plus basses (micro-code, langage machine, instructions reconnues par le processeur). Il existe pourtant de nombreux outils de traduction ou d'adaptation ; nous allons en détailler quelques- uns.

CULTURE Haut niveau et bas niveau

Il ne s'agit nullement de jugements de valeur, mais d'une graduation sur une échelle virtuelle s'appuyant sur les niveaux de fonctionnement du processeur et du langage machine pour « monter » couche par couche jusqu'à l'interface avec l'humain, utilisateur ou programmeur.

Le « haut niveau » est caractérisé par un vocabulaire riche et étendu, proposant des termes pour une grande variété de concepts couramment manipulés par les hommes. C'est plutôt une description fonctionnelle des choses. Chacun de ces ordres du catalogue se décompose au niveau électronique en de multiples opérations élémentaires. Voici un exemple d'ordre de haut niveau dans la vie courante : « Viens me chercher à la gare à 11 heures ».

Le « bas niveau » est caractérisé par un vocabulaire limité, très organique et peu conceptualisé, et décompose chaque action évoluée en une longue séquence d'ordres élémentaires. C'est de cette manière que fonctionnent les puces électroniques et circuits électriques. Si on devait tenter de traduire en instructions de bas niveau l'ordre donné précédemment en exemple, on pourrait décomposer à l'envi, comme suit : « Alarme à 10h30. Ouvrir la portière de la voiture. S'asseoir sur le siège conducteur. Boucler la ceinture de sécurité. Insérer la clé de contact. Tourner la clé de contact. […] Avancer de 100 mètres. Si le feu est vert, passer et tourner à droite. Si en revanche il est rouge, s'arrêter. En cas d'obstruction ou de déviation… ». Ces instructions semblent toutes évidentes à tout être humain ayant déjà conduit en ville, mais il faut les programmer si l'on souhaite qu'une machine s'en acquitte.

Émulation au niveau processeur

Le processeur est la puce électronique qui joue le rôle de « cerveau » de l'ordinateur - il est plus exact de dire que c'est sa « centrale de calcul », puisque la mémoire et toute autre forme d'« intelligence » d'un ordinateur sont situées dans d'autres composants, respectivement la RAM et les programmes saisis par des hommes.

Il lit successivement les instructions de son « segment de code » et applique les ordres qu'il y trouve sur son « segment de données ». Il n'est capable que d'opérations très simples et limitées en nombre, comme de lire un nombre en mémoire, d'écrire un nombre en mémoire, d'effectuer des opérations arithmétiques simples sur les nombres stockés dans ses registres (sa propre petite mémoire de transit), etc.

Langage d'assemblage

On utilise un « assembleur » pour programmer un processeur en langage d'assemblage. C'est une programmation très rébarbative et difficile à relire et maintenir, qui est de moins en moins connue et pratiquée. Deux processeurs incompatibles ne parlant pas le même langage, un programme écrit pour l'un sera inutilisable sur l'autre.

Dans la même famille, les processeurs proposent souvent une « compatibilité » ascendante : ainsi, un Pentium (586) comprendra tout le jeu d'instructions d'un 386, qu'il suit de plusieurs années, mais proposera de nouvelles instructions. Ainsi, un programme écrit en langage d'assemblage pour Pentium ne fonctionnera pas sur 386.

CULTURE RISC et CISC

Même si les plus répandus au niveau grand public sont les processeurs compatibles Intel de type x86 (386, 486, Pentium, etc.), il existe de nombreuses familles de processeurs - les Macintosh d'Apple utilisent par exemple des Motorola, qui parlent un langage différent.
On peut classer ces familles en deux superfamilles, chacune ayant choisi une option différente. Les « Complex Instruction Set Computers » (CISC) ont opté pour des processeurs au vocabulaire plus étendu, ce qui complique évidemment leur conception au niveau circuits imprimés. On les trouve par exemple dans les Sun, ordinateurs plutôt réservés à un usage professionnel de type serveurs sous Unix. Les « Reduced Instruction Set Computer » (RISC), plus répandus dans le grand public, ont fait le choix inverse. C'est le cas des processeurs de la famille Intel et Motorola.

Langages compilés

Plusieurs approches ont été menées pour résoudre ou contourner ces incompatibilités au niveau du processeur. La première est la mise en place de langages de programmation de plus haut niveau, utilisant un « compilateur » pour transformer (compiler) ce code en langage d'assemblage. Ainsi, un même code source peut être ré-utilisé et recompilé sur différentes architectures, le compilateur se chargeant de l'adaptation au langage d'assemblage spécifique de l'architecture. L'inconvénient évidemment est que malgré les progrès et les efforts investis dans le développement des compilateurs, ces codes ainsi générés automatiquement sont moins efficaces qu'un code écrit directement en langage d'assemblage par un bon spécialiste car le compilateur ne « comprenant » pas ce qu'il fait - il est rare qu'un ordinateur « comprenne » quoi que ce soit à ce qu'il manipule - il ne procède pas à toutes les optimisations possibles. Citons quelques exemples de langages compilés : Fortran, Pascal, C, C++, Java.

Langages interprétés

La puissance sans cesse croissante des ordinateurs a permis le développement de langages « interprétés ». Ils n'existent sur le disque que sous forme de fichiers de code source et c'est un « interpréteur » qui se charge de les comprendre et traduire à la volée en instructions de langage machine. On peut ainsi reprendre un fichier sur un autre système sans devoir le recompiler ni même disposer d'un compilateur. Évidemment, ceci est plus lent à l'exécution que le résultat d'une compilation, et c'est à réserver aux applications n'ayant pas de gros besoins en puissance de calcul. Voici quelques exemples de langages interprétés : Basic, Korn Shell, Perl, Python.

Des ordinateurs puissants

La « loi de Moore » a très tôt prédit une amélioration exponentielle des performances et, empiriquement, on constate depuis plusieurs années que la puissance de calcul double tous les dix-huit mois. Cependant, il est probable qu'on atteigne bientôt des limites physiques dues à la vitesse de la lumière et à la structure atomique de la matière.

Traduction à la volée du langage d'assemblage

Souvent, on ne dispose pas du code source pour espérer recompiler une application sur une nouvelle architecture cible. C'est en particulier le cas pour les logiciels propriétaires classiquement trouvés dans le commerce, mais aussi pour les solutions fournies par des prestataires de services qui disparaissent de la vie économique sans avoir fourni les codes sources à leurs clients.

La vitesse toujours plus grande des processeurs a permis d'envisager de traduire à la volée le langage d'assemblage dans le cadre d'un ordinateur virtuel.

Un ordinateur virtuel est un programme normalement codé dans le cadre de l'architecture utilisée (processeur et système d'exploitation) qui réserve une partie de la mémoire et du disque pour « créer » un deuxième ordinateur sans existence matérielle : son disque dur sera une portion réservée du disque de la machine, son écran sera une fenêtre de l'interface graphique, et son clavier sera le clavier normal de l'ordinateur lorsque cette fenêtre sera activée.

On comprendra que cela n'est possible que si les tailles de disque et de mémoire, voire les résolutions graphiques ont suffisamment évolué entre le moment où l'ordinateur qu'on souhaite émuler était courant, et le moment présent.

De plus, il faut aussi que la puissance de calcul ait progressé : chaque instruction de langage d'assemblage des programmes exécutés sur l'ordinateur virtuel devant évidemment s'exécuter sur un matériel réel, elle sera traduite par le programme d'émulation en instructions de langage d'assemblage de l'ordinateur physique. Ceci étant fait automatiquement, sans aucune tentative d'optimisation ou de compréhension de la sémantique du code ainsi interprété, on observera évidemment des dégradations de performances d'autant plus importantes que les deux architectures sont éloignées. Ce qui s'exécutait en un cycle d'horloge sur l'ordinateur original, pourra prendre en moyenne 5 à 10 cycles d'horloge sur la machine virtuelle.

Si le nouvel ordinateur n'est pas suffisamment puissant, on souffrira donc d'une lenteur crispante de l'exécution de l'application émulée. Dans tous les cas de figure, ce genre de pratique est en général à réserver aux utilisations ponctuelles, dans le cadre de pratiques d'interopérabilité par exemple.

EXEMPLES Émulateurs de plate-forme compatible PC

C'est le cas de figure le plus fréquent, notamment pour exécuter des applications MS-Windows sous système GNU/Linux ou inversement. Pour cela, il faudra commencer par « installer » MS-Windows dans l'ordinateur virtuel, à l'aide d'un CD-Rom d'installation classique, en suivant la méthode habituelle, ce qui ne manquera pas de surprendre la première fois puisque tout s'affichera dans une fenêtre à l'écran. Citons notamment Bochs (logiciel libre) et VMware (logiciel propriétaire) :

- http://bochs.sourceforge.net/
- http://www.vmware.com/

NOTION Système d'exploitation

Comme son nom l'indique, c'est l'ensemble des logiciels techniques qui permettent de tirer parti d'une machine. C'est une couche d'assez bas niveau, surplombant la couche de langage machine et « virtualisant » les interfaces d'accès aux périphériques matériels (cartes réseau, cartes graphiques, disques surs). Ainsi, les applications pourront accéder aux ressources de la machine sans devoir connaître leurs caractéristiques techniques précises et tout changement ou mise à jour de matériel ne devra être prise en compte que par le système.

Un système d'exploitation gravite autour d'un noyau qui se charge d'exécuter les appels système, fonctionnalités utilisées par les applicatifs et qui peuvent consister à ouvrir un fichier sur le disque, clore une connexion réseau, etc. Tous les systèmes ne proposent pas les mêmes appels système, ce qui explique que l'on ne peut pas utiliser de façon transparente une application MS-Windows sur un ordinateur sous GNU/Linux, même s'il fonctionne sur la même architecture matérielle (avec le même langage d'assemblage donc).

Émulation au niveau système

On a parfois besoin d'utiliser un applicatif MS-Windows car il n'a aucun équivalent fonctionnel sous systèmes Unix. On peut bien sûr installer MS-Windows sur un disque dur ou une partition de la machine et redémarrer en choisissant le système d'exploitation souhaité dans le menu de

JEUX VIDÉO **Émulateurs de bornes d'arcade et consoles de jeux**

À partir des années 1980, les jeux vidéo ont envahi les cafés et salles de jeu avec des machines embarquant, dans un caisson de bois, un écran de moniteur, une manette et quelques boutons.

Ces titres ont rapidement été dépassés car la puissance des machines et les efforts consentis en développement de jeux n'ont cessé de croître. Malheureusement, comme souvent dans ce cas de figure, au lieu de « libérer » les éléments du jeu dans le domaine public, les détenteurs de droits les ont souvent enterrés sans plus les exploiter (pratique dite de l'« abandonware ») et certains d'entre eux ont mis la clé sous la porte. Le même phénomène a eu lieu avec les consoles de jeu qui se branchaient sur la télévision.

Les nostalgiques peuvent désormais accéder à des émulateurs de ces machines de jeux, tels que MAME :

- http://www.mame.net

Ces machines n'ayant pas vocation à être des ordinateurs programmables, elles se contentaient de stocker le code compilé du jeu dans leur mémoire permanente (ROM). Si un autre jeu pouvait fonctionner sur la même circuiterie, il suffisait de changer de ROM. Aujourd'hui encore, disposer du seul émulateur ne permettra de jouer à aucun jeu si l'on ne dispose pas de ROM.

Le site web précédemment cité rappelle que « les sites web proposant des ROM sont illégaux » (ces ROM sont protégées par le copyright de leurs ayants-droit, souvent les sociétés de développement des jeux), mais en donne quand même une liste.

démarrage du chargeur d'amorçage. Ceci s'avèrera rapidement rébarbatif et interrompra toute session, connexion ou calcul en cours.

On peut recourir alors à la solution de l'émulation au niveau du processeur, détaillée dans la section précédente. Mais elle ne convient pas toujours, et si l'on dispose déjà d'une partition avec le système d'exploitation que l'on souhaite utiliser complètement installé, le ré-installer dans un ordinateur virtuel (qui ne pourra utiliser cette partition) fait double emploi.

L'émulateur de système d'exploitation le plus répandu est Wine, qui permet d'exécuter de nombreuses applications MS-Windows sur des systèmes Unix, donc notamment sur GNU/Linux.

Il nécessite pour fonctionner correctement un système MS-Windows installé sur une partition, sans lequel il sera limité et ne pourra exécuter qu'un nombre très restreint d'applications.

C'est un produit remarquable, en phase de développement intensif ; il n'est pas exclu que la version sortie récemment corrige certains problèmes ou permette d'exécuter de nouveaux programmes. Évidemment, plus un programme fait de manipulations non classiques (comme intervenir sur le fond de l'écran en écrivant des messages ou des jeux vidéo très gourmands en ressources d'affichage et de calcul), moins il est probable qu'il fonctionne correctement. Wine permet toutefois d'employer des éléments de la suite Office et notamment de lire, modifier et sauvegarder des documents Word.

Wine

- http://www.winehq.com/

On prendra garde à l'installation des polices de caractères sur la machine Unix, car Word fait appel au gestionnaire de polices de MS-Windows pour son affichage, et Wine aura par exemple besoin de polices TrueType pour afficher correctement les applications utilisant des polices évoluées.

Compatibilité de formats de fichiers

Qu'est-ce qu'un format de fichier ?

Un format de fichier est une notion souvent inconnue de l'utilisateur final de suite bureautique, mais elle est cruciale et explique souvent de nombreuses incompatibilités.

Au plus bas niveau, toute donnée manipulée par un ordinateur (images, sons, textes, documents bureautiques, codes sources ou codes compilés), est stockée sous forme de nombres. En particulier, un fichier est une suite d'octets, c'est-à-dire de nombres pouvant prendre 256 valeurs différentes, de 0 à 255 (de même que deux cases carrées permettent d'inscrire 100 nombres différents, de 00 à 99).

L'enjeu crucial est donc celui de la manière dont les données de haut niveau sont codées, et comment les décoder ou visualiser. Tous les ordinateurs utilisent désormais le même jeu de caractères, ASCII, qui consiste à attribuer la place 65 à la lettre « A », 48 au chiffre « 0 », etc. Représenter graphiquement à l'écran tout « 65 » d'un fichier par le glyphe « A » et tout « 48 » par « 0 » permettra donc à un œil humain de prendre connaissance du contenu d'un fichier texte (fichier texte brut ou courrier électronique simple, par exemple).

Toutes les valeurs possibles des octets ne sont pas associées à des caractères alphabétiques ou typographiques : on trouve ainsi des « caractères de contrôle » (0 à 31) et des « caractères étendus » (128 à 159, voire 255 parfois) qui n'ont pas vocation à être interprétés dans le cadre d'un jeu de caractères. On appelle « fichier binaire » un fichier incompréhensible s'il est visualisé en tant que fichier texte.

Le format binaire permet d'économiser de l'espace de stockage en exploitant tout le spectre des valeurs possibles pour les octets : ainsi les images associeront une couleur à chaque valeur. Les formats d'images qui n'utilisent que des caractères ASCII, tels que XPM, sont moins efficaces en termes de stockage et de décodage par l'ordinateur, mais plus lisibles par l'être humain.

Les documents bureautiques, même dépourvus d'insertions de graphiques, ne consistent pas simplement en du texte brut : ce sont des documents structurés et les fichiers doivent bien embarquer une manière de coder les méta-informations, c'est-à-dire par exemple que tel groupe de mots est un sous-

FAUSSE ALERTE **Format de fichiers d'OpenOffice.org**

OpenOffice.org est une suite bureautique libre largement compatible avec les formats de fichiers MS-Office. Cependant, son format de fichiers natif semble binaire lui aussi. À quoi bon donc la recommander, quelles garanties de pérennité de plus offre-t-elle ?

Tout d'abord, le fait qu'en tant que logiciel libre, son code source soit disponible (voir chapitre 18) est la seule et meilleure garantie possible qu'on puisse exiger.

Surtout, ce format n'est qu'apparemment binaire : OpenOffice.org travaille en XML, format texte, mais comprime à la volée les documents pour occuper moins d'espace de stockage. Le format de compression utilisé est ZIP, et le lecteur qui a la curiosité d'extraire ses documents d'une archive OpenOffice.org découvre des fichiers au format texte, sans rupture de ligne et chargés, mais dont il devrait pouvoir se convaincre qu'ils sont potentiellement facilement exploitables par une autre application, a fortiori étant donné que le schéma directeur de cette syntaxe est publié sur le site web d'OpenOffice.org.

- 🕮 *OpenOffice.org 1.1 efficace*, « Accès libre », de Frédéric Labbe, Sophie Gautier, Christian Hardy et Michel Pinquier, présente en détails la suite OpenOffice.org.
- http://www.openoffice.org/
- http://xml.openoffice.org/

titre alors que tel mot est en italique. Une manière « lisible » de procéder est d'utiliser un marquage en balises à la manière de HTML : c'est le cas par exemple des documents écrits en LaTeX ou XML. On y trouvera ainsi des choses comme « (TitreNiveau2)Qu'est-ce qu'un format de fichier ? » pour représenter le titre de la présente sous-section, à charge pour l'ordinateur de représenter cela correctement à l'écran et de l'imprimer en bonne et due forme. Ceci garantit la pérennité du format et l'indépendance de l'utilisateur, qui pourra toujours espérer comprendre et reprendre ses documents même en cas de défaillance de l'éditeur de suites bureautiques.

Malheureusement, la suite MS-Office a opté pour un format binaire qui lui est propre, non ou mal documenté, et qui change beaucoup à chaque nouvelle version. Même entre architectures différentes (PC et Mac), la compatibilité n'est pas toujours assurée. Pour vous rendre compte de ce que cela signifie, ouvrez un document MS-Office dans un éditeur de texte tel que `wordpad` ou `edit` et comparez par exemple avec le source d'une page HTML pas trop chargée (fonction accessible sous le menu `Visualiser` de la plupart des navigateurs).

Évidemment, aucune application ne pourra comprendre un codage de données différent du sien propre, ce qui explique tous ces drames de pertes de documents ou de mises en page. C'est aussi la raison pour laquelle il est impoli de transmettre à un correspondant dont on n'est pas sûr qu'il travaille avec les mêmes outils que nous en temps normal, un document codé dans un format de fichier semi-secret tel que ceux de la suite MS-Office.

AMUSANT **Secrets transmis par vos correspondants à leur insu**

Même s'il est d'apparence absconse, un fichier MS-Word lu en tant que document texte peut être décrypté dans une certaine mesure avec des outils simples : une simple extraction des caractères imprimables (caractères de texte et pas de contrôle) permet de reconstituer une grande partie du texte du document, à l'exception des accents et de la mise en page.

Dans son mode « sauvegarde rapide », MS-Word se contente de transcrire sur le disque les opérations menées au clavier : insertions, destructions, etc. Un destinataire fouineur pourra ainsi lire les versions intermédiaires d'un document qu'on lui fournit, voire accéder à des informations secrètes dont il n'était pas supposé prendre connaissance !

Compatibilité au niveau format de fichiers

La bureautique étant l'un des besoins les plus courants en matière d'informatique grand public, d'importants efforts ont été mobilisés pour que d'autres applications puissent interopérer avec les formats des suites MS-Office.

Ces projets ont désormais abouti à des versions suffisamment compatibles et utilisables par l'utilisateur final : citons notamment AbiWord, KOffice, GNOME Office et OpenOffice.org.

Ces produits permettent de reprendre des documents MS-Office sans perdre de données ou d'informations de mise en page (ou très peu), voire d'en produire de manière à ce qu'ils soient lisibles sous MS-Windows. Ils sont de plus en plus testés avec succès dans des situations professionnelles, dans des entreprises privées ou des administrations.

INTEROPÉRABILITÉ
Suites bureautiques compatibles

AbiWord, KOffice, GNOME Office et OpenOffice.org sont respectivement disponibles aux adresses suivantes :

- http://www.abisource.com/
- http://www.koffice.org/
- http://www.gnome.org/gnome-office/
- http://www.openoffice.org/

En résumé...

L'émulation est un besoin naturel en informatique, suite aux changements d'architecture ou de système. En soi, elle n'est pas difficile à appréhender, mais dans la pratique elle induit un ralentissement du fonctionnement qui devra être compensé par les progrès technologiques survenus entre-temps, notamment en vitesse du processeur. Elle existe à trois niveaux différents (processeur, système, applications), tous trois fort bien pourvus et représentés sur les systèmes GNU/Linux.

Dans le monde MS-Windows, il est nécessaire de réaliser ses mises à jour très régulièrement. Cela signifie parfois de faire une mise à jour d'anti-virus par jour ! Le monde Unix est moins sensible à ce genre de turpitudes, mais il arrive régulièrement que des failles de sécurité soient découvertes et documentées, qui mettent en péril l'intégrité des systèmes informatiques concernés.

Les nouvelles versions de logiciel parues proposent de nouvelles fonctionnalités, prennent en charge de nouveaux matériels, etc. C'est pourquoi il est sain et recommandé de mettre à jour régulièrement son système, d'autant plus que cette opération est automatisée ou semi-automatisée dans la plupart des distributions. Même les émulateurs évoluent régulièrement et rapidement, et chaque nouvelle version apporte des choses nouvelles.