VMS POSIX

Lancés parallèlement, POSIX et UNIX font aujourd'hui figure de frères ennemis.

LE DÉFI DE VMS POSIX À UNIX

Vers le milieu des années 80, tandis que Digital vendait un nombre incroyable de VAX, avec pour slogan "une architecture, un système d'exploitation", UNIX sortait tranquillement des universités et des centres de recherche pour s'implanter dans le monde des entreprises américaines.

Le temps que Digital, aveuglé par le succès, réalise enfin ce qui se passait, le mal était fait. Sa poignée d'ULTRIX et produits connexes ne pouvait rivaliser avec les produits correspondants de Sun Microsystems et de Silicon Graphics sur la côte ouest et d'Apollo Computer sur la côte est. Les VAXstations et les MicroVAX se prirent une bonne "raclée" et les actions DEC plongèrent. Les DECstations et les DECsystems arrivèrent trop tard pour renverser le cours des choses. La partie était terminée.

Mais il se pourrait bien qu'un nouveau match se prépare. Les VAX ont continué à bien se vendre (bien qu'à un rythme plus lent), tandis que VMS continue de défier UNIX en tant que système d'exploitation multi-tâches à usage général. Il existe un parc de quelques 500.000 systèmes VAX/VMS installés pour un nombre d'utilisateurs de VMS qui dépasse les 10 millions. Digital continue à investir sérieusement dans ce système d'exploitation.

Pendant ce temps, UNIX vacille tandis que AT&T - par l'intermédiaire de Sun Microsystems, UNIX Int'l et UNIX Systems Labs - tente de reprendre le contrôle d'un système d'exploitation qu'il a accidentellement perdu il y a bien longtemps. Des produits tels que OSF/1, développé dans une direction diamétralement opposée aux intentions d'AT&T, n'ont décidément plus grand chose à voir avec UNIX. OSF/1 repose plus sur les capacités du noyau Mach que sur celui de l'UNIX générique. Si l'on excepte POSIX, UNIX, en tant que plate-forme prétendument ouverte, s'est révélée être à l'origine de plus de programmes incompatibles et de plus de contentieux entre sociétés informatiques que tout autre produit dans l'histoire de l'industrie informatique.

Bien plus, l'engouement pour POSIX en tant que norme indépendante d'UNIX démontre que ce n'est pas UNIX qui est populaire mais les concepts de portabilité, d'adaptabilité et de couches logicielles. VMS peut s'enorgueillir de mettre en œuvre ces concepts depuis longtemps.

Il apparaît aujourd'hui qu'UNIX pourrait bien entrer dans l'histoire comme une utile "plate-forme de référence", qui a donné naissance à des produits tels que DCE d'Open Software Foundation, XPG de X/Open ou POSIX d'IEEE. N'ayant que peu de rapport avec le noyau UNIX géné-

BRADFORD T. HARRISON

rique, ces derniers se développent donc de leur côté. Qui sait s'ils ne trouveront pas en VMS, plutôt qu'en UNIX, la plate-forme qui fera leur succès, car VMS possède une base installée, des fonctionnalités tels que le clustering, et bénéficie du soutien constant d'une société informatique qui place la recherche technique et le développement en tête de sa liste des priorités. Par une curieuse ironie du sort, UNIX sera peut-être à l'origine d'une renaissance de VMS.

Alpha/POSIX

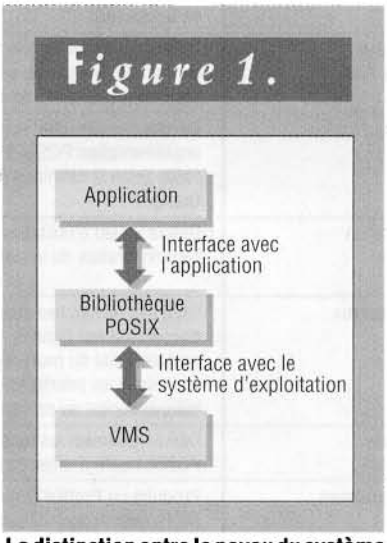
Le renouveau de VMS paraît être en bonne voie. DCE sur VMS est en cours d'élaboration, XPG3 est en période d'essai, et la conformité avec POSIX.1, .2 et .4 est déjà au rendez-vous avec VMS V5.5. De plus, les systèmes Alpha vont bientôt arriver pour fournir la puissance brute requise pour faire tourner les logiciels VMS nécessaires à la mise en œuvre de ces standards en plus des fonctionnalités traditionnelles de VMS. Avec Alpha POSIX, Digital peut directement répondre aux préoccupations des utilisateurs qui avaient permis à RISC/UNIX de s'approprier les parts de marché de VAX/VMS. En effet, RISC/UNIX constituait un "concept" technologique plutôt qu'une "concrétisation" sur un produit spécifique. Hewlett-Packard, fort de la puissance de sa Precision Architecture et de la réputation de HP/UX, son système d'exploitation basé sur UNIX, a largement contribué à diffuser l'idée qu'UNIX tournant sur une architecture RISC représentait le nec plus ultra, tandis que le VMS mono-marque tournant sur un vieux VAX à architecture CISC était un concept préhistorique. Sun, de son côté, s'est également plût à critiquer VMS, et continue d'ailleurs à le faire.

Mais aujourd'hui, Alpha n'a rien à envier à la puissance de ses concurrents RISC, et la conformité de VMS avec POSIX procure une portabilité et un environnement de programmation sophistiqué que HP et SUN brandissaient comme l'apanage d'UNIX. Certaines fonctionnalités de l'architecture d'Alpha la rendent particulièrement adaptée à VMS, comme l'était l'architecture initiale du VAX. Digital soutient actuellement la version 32 bits de VMS pour Alpha, tandis qu'Alpha pour OSF/1 fonctionnera sur 64 bits dès sa première mise en œuvre. DEC entend ainsi faciliter la transition vers Alpha pour les utilisateurs VMS. Digital continuera en outre à améliorer les performances de l'architecture VAX CISC grâce à de nouvelles technologies plus rapides pour les processeurs.

Par ailleurs, Digital accorde dorénavant des licences pour VMS et pour Alpha à quiconque les demande. A n'en pas douter, DEC continuera à surveiller de près l'usage fait de ces technologies ; cette pratique maintiendra toutefois la compétitivité d'Alpha VMS face aux nombreuses solutions RISC/UNIX actuellement disponibles, et par la diver- sification des sources, elle assurera un large approvisionnement en processeurs Alpha.

Plus important encore, VMS est le premier système d'exploitation exclusif à sauter le pas vers l'informatique normalisée. Si l'on en croit une enquête réalisée par X/Open sur les conditions requises pour les systèmes ouverts Xtra World, l'avenir appartient aux interfaces de système d'exploitation Common Applications Environnement (CAE) de X/Open et POSIX. Selon X/Open, VMS est le système d'exploitation non UNIX qui a le plus progressé en direction de la conformité avec XPG3 et CAE. Notons que les deux tiers des personnes interrogées au cours de cette enquête étaient des responsables ou des subordonnés directs du département informatique de leur entreprise.

Le marché s'oriente nettement vers des systèmes d'exploitation dits "mono-marque standard". Certaines sociétés, telles que Mortice Kern Systems (MKS), auteur du code source qui est à l'origine de VMS POSIX.2, se sont glissées dans un nouveau créneau : la fourniture de logiciels qui rendent les systèmes exclusifs conformes aux normes.



La distinction entre le noyau du système d'exploitation et les bibliothèques permet à des systèmes d'exploitation non UNIX d'entrer dans les rangs des systèmes ouverts.

Selon Richard Kittle, Directeur du marketing de VMS, Digital vise trois marchés avec le VMS "mono-marque standard" : la base installée des utilisateurs de VMS, les systèmes d'exploitations non UNIX concurrents et les utilisateurs déjà munis d'installations UNIX.

"Les utilisateurs de VMS veulent garder ce qu'ils ont," affirme M. Kittle. "En premier lieu, nous devons nous pencher sur les besoins de cette base installée. POSIX leur ouvre tout un nouveau champ d'applications, sans qu'ils aient à changer de système d'exploitation." Des applications compatibles avec POSIX peuvent en effet coexister, et même interfonctionner sur le même système avec des applications VMS traditionnelles.

L'immense base installée des systèmes d'exploitations non UNIX concurrents constitue un marché presqu'aussi important. "Ces utilisateurs sont de plus en plus intéressés par une solution composite" explique Richard Kittle. "Ils veulent à la fois l'ouverture et des fonctionnalités jusqu'alors uniquement disponibles sur des systèmes mono-marques. Ils veulent les prix et les performances d'UNIX en même temps que la compatibilité interne de VMS."

Enfin, Richard Kittle estime que VMS POSIX exercera un attrait non négligeable chez les utilisateurs disposant déjà d'installations UNIX. N'oublions pas que POSIX provient à l'origine de l'univers d'UNIX, et que POSIX.2 leur procure donc un environnement de programmation familier, enrichi de toutes les fonctionnalités de VMS disponibles. "Mais l'effort de POSIX dans VMS est complémentaire de celui d'ULTRIX/OSF", ajoute-t-il.

UNIX contre POSIX

La question de l'ouverture est au cœur du débat UNIX contre POSIX. Ayant été

Figure 2.

Comité	Nom	Objectif
POSIX.1	Interface système	Définir l'interface entre des programmes d'application portables et le système d'exploitation basé sur un modèle UNIX.
POSIX.2	Shell et Utilitaires	Définir le langage de commande du shell et les utilitaires auxquels peuvent faire appel des applications portables.
POSIX.3	Tests et vérifications	Développer une méthodologie standard pour tester la conformité d'une implémentation avec les normes POSIX.
POSIX.4	Temps réel	Définir les extensions minima à POSIX.1 pour permettre la portabilité d'applications en temps réel.
POSIX.5	Liaisons avec le langage Ada	Définir comment les fonctionnalités de POSIX.1 seront utilisées depuis un environnement Ada et pour traiter certaines fonctions spécifiques d'Ada.
POSIX.6	Extensions de Sécurité	Développer des interfaces qui permettraient à une implémentation POSIX.1 de devenir un système fiable selon la définition du Orange Book (Livre Orange).
POSIX.7	Administration du système	Définir un jeu d'utilitaires communs nécessaire à l'administration du système.
POSIX.8	Mise en réseaux	Faire des recherches dans les secteurs de l'accès transparent aux fichiers, l'interface réseau indépendante du protocole, les appels de procédure distants et les interfaces d'application OSI indépendantes du protocole.
POSIX.9	Liaisons avec le FORTRAN	Définir comment les fonctionnalités de POSIX.1 seront utilisées depuis le FORTRAN.
POSIX.10	Supercalculateurs	Produire un Profil d'Environnement d'Application (AEP) destiné aux utilisateurs de supercalculateurs.
POSIX.11	Traitement des transactions	Produire un AEP orienté vers le traitement des transactions pour l'utilisateur.
POSIX.12	Accès aux fichiers	Déterminer si l'accès transparent aux fichiers supportera toute la sémantique des accès aux fichiers de POSIX.1.
POSIX.13	Support des applications en temps réel	Définir un AEP pour des applications utilisant les interfaces POSIX.
POSIX.14	Support des applications multiprocesseurs	Définir un AEP pour des environnements d'applications multiprocesseurs.
POSIX.15	Traitements par lots	Fournir un système de file d'attente et de traitement par lots dans un environnement POSIX.
POSIX.16	Liaisons avec le langage C	Fournir une liaison entre le C ISO 9989 et l'API indépendante du langage correspondant exactement à ISO 9945-1 : 1990.
POSIX.17	Services de répertoires	Fournir une interface standard supportant le développement d'applications qui utilisent les services de répertoires, incluant X.500.
POSIX.18	Profil d'environnement de plate-forme	Elaborer un profil d'environnement de plate-forme basé sur les normes liées au travail de l'ISO 9945.
POSIX.19	Liaisons avec le FORTRAN	Fournir un liaison entre le FORTRAN 90 et la Spécification indépendante du langage de l'API système de POSIX.
POSIX.20	Liaisons avec l'Ada pour le temps réel	Fournir une liaison entre l'Ada et les normes POSIX de temps réel.

La série de normes POSIX, déjà assez importante, continuera à s'étendre.

porté sur de nombreuses plates-formes, UNIX est devenu synonyme de "système ouvert". UNIX est relativement facile à porter, dans la mesure où c'est un petit système d'exploitation écrit en C, langage d'assez haut niveau. OSF/1, pour sa part, est considéré comme extrêmement portable, du fait de la séparation très marquée qui existe entre le code dépendant du matériel et le code indépendant.

On accède au noyau UNIX par le biais d'appels système. Ces derniers permettent aux programmeurs d'avoir accès aux fonctions du noyau telles que le fichier système, les mécanismes du multi-tâches, et les fonctions de communication interprocessus. Les bibliothèques systèmes sont constituées de routines qui utilisent (pour la plupart) des appels système pour fournir aux applications une interface avec le système d'exploitation.

A mesure que mûrit le concept de système ouvert, la distinction entre le noyau du système d'exploitation et les bibliothèques permet à des systèmes d'exploitations non UNIX plus anciens d'entrer dans les rangs des systèmes ouverts. Si cela est possible, c'est avant tout grâce à la norme POSIX, comme le montre la Figure 1.

POSIX repose en effet sur des interfaces de UNIX Système V et de UNIX Berkeley, et définit l'interface entre une application et le système d'exploitation. Cette interface est réalisée sous forme de bibliothèque des fonctions offertes à une application, tout en communicant avec le système d'exploitation.

POSIX est en réalité une série de normes (cf. Figure 2) qui devrait continuer à s'étoffer. POSIX.1 est fondé sur une sémantique d'appels en C et incorpore 110 fonctions de la bibliothèque standard du C ANSI, tandis que POSIX.5 est constitué d'un jeu de liaisons pour Ada avec les services de POSIX.1. De façon similaire, POSIX.9 est composé d'un jeu de liaisons pour le langage FORTRAN. POSIX.16, dont l'élaboration a été confiée au groupe de travail auquel on doit POSIX.1, développe pour C une liaison de langage ISO 9989 avec une version de POSIX indépendante du langage, tandis que POSIX.19 fait de même pour le FORTRAN 90. POSIX.20 traite quant à lui de Real-time

Ada. POSIX.1 est à ce jour le seul qui ait dépassé l'état d'ébauche pour devenir une norme officielle.

POSIX fut d'abord homologuée par l'IEEE (IEEE 1003.1) en 1988, puis par ISO/IEC (9945-1) en 1990. Elle fut également homologuée par le gouvernement des Etats-Unis en tant que Federal Information Processing Standard 151.

POSIX.1, cœur de la norme POSIX, est soutenue par tous les grands noms de l'informatique, que ce soit Digital ou HP, IBM ou Sun et Unisys. POSIX est le seul dénominateur commun qui réunisse des consortiums de normes aussi divers que UNIX Int'l, OSF et X/Open. Il recoupe divers systèmes d'exploitation, notamment toutes les variantes d'UNIX, ainsi que VMS, CTOS et MPE.

Par ailleurs, le National Institute for Standards and Technology (NIST) abrite les Laboratoires de tests POSIX accrédités NIST qui valident les produits. POSIX.3 fournit les spécifications des tests et des vérifications mis en œuvre dans les laboratoires accrédités. Les produits Digital ont été validés par Mindcraft et par Datafocus.

Dans un ouvrage intitulé "The POSIX.1 standard: A Programmer's Guide" (paru chez The Benjamin/Cummings Publishing Co. Inc, 1991), Fred Zlotnick, de la firme Mindcraft, explique que POSIX décrit une interface et non une mise en œuvre d'UNIX. "La description, écrit Fred Zlotnick, consiste principalement en une série de spécifications des fonctions pouvant être appelées à partir de programmes écrits en C, des fichiers d'en-tête et des structures de données." Fred Zlotnick signale par ailleurs que, bien que la norme POSIX.1 soit actuellement rédigée en C, les travaux en cours doivent aboutir à une version indépendante du langage, qui sera accompagnée d'interfaces "spécifiques au langage" pour plusieurs langages de programmation.

Aucune distinction n'est faite dans la norme entre les appels système et les fonctions de bibliothèque. "En réalité," explique Fred Zlotnick, "un appel système dans une mise en œuvre donnée pourrait fort bien être une fonction de bibliothèque dans une autre". Selon lui, POSIX constitue une extension d'UNIX qui généralise certains concepts UNIX tels que les fichiers système. "Dans UNIX, un fichier système est uniquement caractérisé par son système de fichiers et son numéro d'identification (i-number) au sein de ce système", écrit-il. "Dans POSIX.1, un système de fichiers est simplement un emplacement pour les noms, dans lequel les numéros de série des fichiers (l'équivalent des i-numbers) sont uniques."

Fred Zlotnick souligne également que l'un des points forts de POSIX.1 est son aptitude à déterminer les limites et les attributs d'un système à un instant donné et à prendre les mesures appropriées. Cela permet à des applications conformes à POSIX de tourner sur de nombreux systèmes différents, en faisant sonder le système par lui-même plutôt qu'en faisant simplement effectuer un dénominateur commun d'appels système.

La conformité avec POSIX est définie en termes de mises en œuvre (implémentation), telles que VMS POSIX, et d'applications spécifiques qui tourneront sur elles. Une mise en œuvre conforme peut, en outre, supporter des conventions d'appels ne faisant pas partie de la norme POSIX, mais qui apportent un compatibilité à une base d'applications existante (notamment des programmes plus anciens en C) ou des fonctionnalités supplémentaires (telles que les appels VMS RMS, le support de MOTIF et celui de SQL). Dans certaines réalisations, plusieurs appels de fonctions peuvent avoir un effet identique mais des syntaxes différentes, dans le seul but de procurer une compatibilité avec diverses applications existantes.

Il existe quatre niveaux de conformité des applications. Une application du premier niveau est dite "strictement conforme" et utilise uniquement les fonctionnalités décrites dans l'ISO/IEC 9945. A ce jour, seules des interfaces avec POSIX.1 pour le langage C ont été normalisées.

Au deuxième niveau, une "application POSIX.1 conforme à ISO/IEC" utilise uniquement les fonctionnalités décrites dans l'ISO/IEC 9945 et des liaisons de langages avec toutes les normes ISO ou IEC. Ce niveau de conformité permet à des applications d'effectuer des appels conformes à d'autres normes reconnues, telles que les normes graphiques GKS ou PHiGS, et d'améliorer les exigences minima spécifiées par la norme POSIX. Il permet notamment d'allonger les noms de fichiers, limités à 14 caractères dans POSIX.1.

Le troisième niveau, "l'application POSIX.1 conforme à une nationale", est identique au précédent niveau, si ce n'est qu'il peut également utiliser des normes adoptées par une seule institution membre de l'ISO/IEC, telle que ANSI.

Enfin, le niveau "Application POSIX.1 conforme utilisant des extensions" permet d'utiliser des fonctions non standards compatibles avec la norme POSIX.1. Il s'agit là d'un type très libre de conformité et, selon Fred Zlotnick, presque tous les programmes en C pourraient s'en prévaloir, avec une documentation adéquate.

POSIX.1 et le C

Selon Donald Lewine, auteur de "POSIX Programmer's guide: writing portable UNIX programs" (aux éditions O'Reilly & Associate Inc., 1991), il est théoriquement possible d'écrire un livre sur POSIX sans utiliser d'exemples écrits dans un langage spécifique, bien que cela ne soit pas encore réalisable dans la pratique. "La norme POSIX est rédigée en langage de programmation C", écrit-il. "POSIX supporte deux environnements de programmation. L'un est constitué du C traditionnel, l'autre du C Standard défini par la norme ANSI X3.159-1989. Le C Standard est plus précis et permet une meilleure portabilité que le C traditionnel."

Etant donné que POSIX.1 a été homologué en 1988 et que le C ANSI ne l'a été qu'un an plus tard, le comité POSIX.1 n'a pas imposé l'utilisation du C ANSI. Toutefois, selon Donald Lewine, nul n'est besoin d'utiliser le C traditionnel si le C ANSI est disponible. Selon lui, il est peu probable qu'un système conforme à la norme POSIX ne fournisse pas un compilateur et des bibliothèques C ANSI.

Mindcraft propose un POSIX Portability Verifier (vérificateur de portabilité POSIX) permettant de s'assurer de la conformité à la norme POSIX.1 des programmes en C. Par ailleurs, les ouvrages tels que ceux de Fred Zlotnick ou de Donald Lewine vous aideront à modifier vos programmes actuels pour les rendre strictement conformes et qu'ils puissent ainsi tourner sur tous les

Figure 3.

POSIX.2 Utilities

!	cd	echo	getconf	ls	readonly	touch
(	chgrp	ed	getopts	mailx	return	tr
{	chmod	egrep	grep	make	rm	trap
[	chowm	env	head	mkdir	rmdir	true
:	cksum	eval	id	mkfifo	sed	tty
.	cmp	exec	if	mv	set	umask
ar	comm	exit	join	nohup	sh	uname
asa	command	export	kill	od	shift	uniq
awk	continue	expr	lex	paste	sleep	unset
basename	cp	false	ln	pathchk	sort	until
bc	cut	fgrep	locale	pax	strip	wait
break	date	find	localedef	pr	stty	wc
c89	dd	fold	logger	printf	tail	while
case	diff	for	logname	pwd	tee	xargs
cat	dirname	fort77	lp	read	test	yacc

POSIX.2a Utilities

alias	csplit	fc	mailx	nm	strings	uudecode
at	ctags	fg	man	passwd	tabs	uuencode
batch	df	file	mesg	patch	talk	vi
bg	du	history	more	ps	tput	wait
compress	ex	jobs	newgrp	renice	unalias	
crontab	expand	kill	nice	split	unexpand	

POSIX.2 définit l'interface de portabilité d'une application et l'interface utilisateur par l'intermédiaire d'un langage de commande du shell et des utilitaires. POSIX.2a, l'extension de portabilité utilisateur, est une annexe.

systèmes d'exploitation conformes, y compris VMS V5.5.

POSIX.2

L'ensemble des plates-formes Digital supportent désormais POSIX.2 et POSIX.4. DEC a acquis sa licence de mise en œuvre POSIX.2 auprès de MKS, qui a également concédé à HP et à Unisys une licence pour la mise en œuvre du code source du shell et des utilitaires InterOpen/POSIX sur leurs systèmes d'exploitation respectifs, MPE et CTOS. Le shell de POSIX.2 ressemble à celui de Korn et possède des fonctionnalités similaires à celles de DCL.

L'ensemble permet aux développeurs de créer des outils et des applications pouvant aisément être portés sur les plates-formes conformes à POSIX. Selon Richard Kittle, de Digital, DEC a exploité tous les éléments standard et optionnels dans sa mise en œuvre de VMS POSIX.2.

POSIX.2 n'est encore qu'une ébauche de norme IEEE et ISO, mais on s'attend à ce qu'elle devienne bientôt une norme à part entière. Elle définit à la fois une interface de portabilité pour l'application et pour l'utilisateur, grâce au langage de programmation du shell et aux utilitaires. Le but de POSIX.2 est la portabilité des applications, par laquelle des programmes d'application peuvent accéder à un jeu standard de services système de haut niveau. POSIX.2 vient compléter POSIX.1 en rendant les fonctions sous-jacentes de cette dernière accessibles aux développeurs, aux testeurs et aux utilisateurs.

POSIX.2a, l'Extension de portabilité utilisateur, est une annexe à la norme POSIX.2 qui y sera plus tard incorporée. POSIX.2a est respectée par VMS POSIX. Elle traite notamment d'une interface d'application normalisée et d'un environnement commun pour les utilisateurs en temps partagé, sur un terminal en mode texte. Cette extension comprend, entre autres, un utilitaire de messagerie, des éditeurs de texte et un interpréteur interactif de langage de commande standard. L'objectif de POSIX.2a est de fournir aux développeurs et aux utilisateurs de POSIX un environnement de travail interactif commun, réduisant ainsi les temps de réapprentissage dans différents systèmes.

POSIX.4

La norme POSIX.4, bien qu'elle aussi à l'état de projet, fournit les bases d'une standardisation de différents services système qui, traditionnellement, étaient mis en œuvre de façons très spécifiques aux constructeurs. Au nombre de ces services, on compte les sémaphores et la mémoire partagée, ainsi que la gestion des threads. Les fonctionnalités telles que les horloges et les extensions de signaux en temps réel, les entrées/sorties asynchrones et synchrones, et la mémoire partagée, sont traitées par POSIX.4.

Digital et les autres constructeurs observent de très près les développements de POSIX.4, car elle fournira une mise en œuvre standard des threads employés dans la plupart des environnements informatiques distribués. POSIX.4a spécifie les threads à la fois au niveau du noyau (dans le noyau) et au niveau des extensions ("au dessus" du noyau). La mise en œuvre des threads DCE est basée sur les spécifications de l'extension de bibliothèque, tandis que Digital utilise DECthreads et l'extension de bibliothèque POSIX.4a dans VMS V5.5. DEC incorporera sans doute la spécification au niveau du noyau lorsque la norme POSIX sera finalisée.

Dans un article présenté à Uniforum en janvier, Bill Gallmeister et Chris Lanier, de Lynx Real-Time Systems, concluaient que, bien que ces normes soient issues d'UNIX, "le travail nécessaire pour supporter POSIX.4 et POSIX.4a sur un système UNIX est conséquent ... Mais même ainsi, les avantages à tirer du soutien et de l'adoption à grande échelle d'une interface de programmation standard pour les systèmes en temps réel peuvent avoir un impact majeur en terme de qualité de logiciels et de productivité."

Il est amusant de voir que le travail nécessaire pour intégrer une norme à un système d'exploitation dont cette norme est issue est "conséquent". UNIX est en passe de devenir un simple noyau, comme le noyau VMS. En fin de compte, la seule contribution d'UNIX à l'ouverture des systèmes aura été de donner naissance aux standards qui uniront enfin tous les systèmes d'exploitations des différents constructeurs. ■