Collection Archéologie informatique

Une histoire de l'informatique

Digital Equipment Corporation

La saga d'une autre informatique

De Maynard à Compaq, 1957–1998

Sébastien Inion

Première édition — 2026

Digital Equipment Corporation — La saga d'une autre informatique

Collection « Archéologie informatique »
Une collection du site *Histoire de l'informatique*
`https://www.histoireinformatique.info` Édition numérique — version 1.0
Publication de cette version : 26 juillet 2026

Table des matières

Avant-propos **ix**

Prologue — Le monde d'IBM **1**

I Deux ingénieurs face aux grands systèmes **7**

1 Deux chemins vers Lincoln **9**
1.1 Ken Olsen, le goût du concret 9
1.2 Harlan Anderson, de la physique à l'informatique . 11
1.3 Un rendez-vous dans l'histoire 12

2 Le laboratoire des possibles **13**
2.1 Du calcul à l'interaction 13
2.2 La leçon de la modularité 15
2.3 Quitter une institution puissante 16

3 Le moulin de Maynard **17**
3.1 Un projet volontairement raisonnable 17
3.2 L'ancien moulin 19
3.3 Deux fondateurs, plusieurs rôles 20

4 Commencer par les modules **23**
4.1 Des briques pour les ingénieurs 23
4.2 Apprendre la série 24
4.3 La tentation de la machine complète 26

II Rapprocher la machine 27

5 Le PDP-1, vendre une idée 29
5.1 La preuve par la démonstration 30
5.2 BBN, le premier client 31
5.3 Une autre manière de compter 32

6 Le cadeau du MIT 35
6.1 Pourquoi donner une machine rare ? 36
6.2 Une machine presque nue 37
6.3 Une politique qui dépasse le MIT 38

7 La machine des hackers 41
7.1 Un week-end pour fabriquer un outil 42
7.2 Spacewar ! 43
7.3 Partager sans perdre l'interaction 44
7.4 Ce que le MIT donne à DEC 45

8 Douze bits pour changer d'échelle 47
8.1 Une architecture volontairement étroite 48
8.2 Une machine que l'on ne voit pas toujours 49
8.3 Le mini-ordinateur devient un marché 50

III Du laboratoire au marché 53

9 Brancher le monde réel 55
9.1 Le télétype, porte d'entrée ordinaire 56
9.2 DECtape, une bande qui se comporte presque comme un disque 57
9.3 Du disque au système de stockage 58
9.4 Voir, mesurer, agir 59

10 Partager la puissance 61
10.1 Une grande machine selon DEC 62

10.2 Le PDP-10 et ses communautés 63
10.3 Une famille brillante mais isolée 64

11 Reconstruire la famille 65
11.1 Une régularité nouvelle 66
11.2 L'Unibus 67
11.3 Une famille qui déborde son projet initial 68

12 UNIX, C et l'autre destin du PDP-11 71
12.1 Après Multics 71
12.2 Le passage au PDP-11 72
12.3 Un langage pour le système 74
12.4 Expliquer et transmettre 75

13 De la machine au système 77
13.1 Une pluralité de logiciels 78
13.2 La force de DECUS 79
13.3 Le service comme promesse 80
13.4 Changer d'échelle sans perdre son identité 81

IV L'empire VAX 83

14 Évoluer sans rompre 85
14.1 Étendre l'adresse 86
14.2 Le VAX-11/780 87
14.3 Une famille plutôt qu'un sommet 88

15 Le logiciel comme héritage 89
15.1 Concevoir avec la machine 90
15.2 Migrer sans tout recommencer 91
15.3 La disponibilité comme argument 91

16 Le réseau devient le système 93
16.1 Une entreprise distribuée 94

16.2 Le terminal devient une norme 95
16.3 VAXcluster et le stockage commun 96
16.4 Un monde cohérent, mais de moins en moins fermé 97

17 Le bureau échappe à DEC 99
17.1 Trois réponses au lieu d'une 100
17.2 La station de travail 101
17.3 Quand l'utilisateur choisit l'écosystème 102

V Le dernier pari 105

18 Une culture devenue empire 107
18.1 Des entrepreneurs à l'intérieur de l'entreprise 108
18.2 Le départ d'Harlan Anderson 109
18.3 Ken et la preuve technique 110
18.4 L'utilisateur au centre, le marché parfois au second plan 111

19 Alpha contre le temps 113
19.1 Une architecture tournée vers l'avenir 114
19.2 La puissance en démonstration 115
19.3 Une architecture sans répit 116

20 La fin de l'indépendance 117
20.1 Le départ de Ken Olsen 118
20.2 Ce que Compaq veut acheter 119
20.3 Une fin plus complexe qu'un échec 120

Épilogue — Ce que DEC a rendu possible 121

A Chronologie 125

B Repères sur les familles 129

C Glossaire 133

Bibliographie **137**

Index général **141**

Avant-propos

Digital Equipment Corporation n'est plus une marque familière au grand public. Son nom ne figure ni sur les téléphones, ni sur les ordinateurs portables, ni sur les services en ligne qui occupent aujourd'hui le devant de la scène. Pourtant, une partie décisive de l'informatique moderne s'est construite sur ses machines, dans les laboratoires qui les ont accueillies et parmi les communautés qui ont appris à les détourner de leur destination première.

Écrire l'histoire de **DEC**, ce n'est donc pas dresser le catalogue d'un constructeur disparu. C'est suivre une transformation plus vaste : celle qui conduit de l'ordinateur central, tenu à distance de ses utilisateurs, vers des machines installées au plus près des expériences, des ingénieurs et des programmeurs. Le PDP-1 donne une forme commerciale à l'informatique interactive. Le PDP-8 élargit l'accès au calcul. Le PDP-10 accompagne les communautés du temps partagé. Le PDP-11 devient l'un des terrains sur lesquels grandissent UNIX et le langage C. VAX, VMS et DECnet font ensuite de l'ordinateur un élément d'un système distribué. Alpha montre enfin qu'une entreprise en difficulté peut encore produire une architecture remarquable.

Les machines occupent une place importante dans ce récit, mais elles n'en sont pas les seuls personnages. Il faut aussi rencontrer *Ken Olsen* et *Harlan Anderson* avant qu'ils ne fondent l'entreprise, comprendre ce qu'ils apprennent au **MIT Lincoln Laboratory**, suivre *Gordon Bell*, *Alan Kotok*, *Dave Cutler* et les équipes qui transforment des idées en produits. Il faut surtout rendre leur place aux utilisateurs : étudiants du MIT, chercheurs, techniciens, membres de DECUS, ingénieurs d'usine et programmeurs de **Bell Laboratories**. Ce sont eux qui donnent aux architectures une vie

que les seules fiches techniques ne peuvent restituer.

L'ouvrage adopte donc la forme d'une saga industrielle et technique. Les scènes et les anecdotes reposent sur des faits documentés ; aucun dialogue ni détail de décor n'est inventé. La technique est expliquée lorsqu'elle éclaire une décision, un usage ou une rupture historique. Les notions secondaires sont parfois précisées en note, mais les idées indispensables restent dans le corps du récit.

Les sources visibles ont été volontairement limitées. Une note bibliographique à chaque paragraphe rendrait la lecture pénible sans rendre le texte plus vrai. En revanche, les dates, chiffres, épisodes structurants et affirmations discutables ont été contrôlés à partir de documents primaires et de travaux indépendants. La bibliographie et la notice méthodologique placées en fin d'ouvrage permettent de retrouver les principaux matériaux utilisés.

Le récit s'arrête en 1998, lorsque **Compaq** achève le rachat de **DEC**. Les technologies, les équipes et les systèmes installés poursuivent leur existence au-delà de cette date, mais l'entreprise indépendante a disparu. Ce choix donne au livre une unité : quarante et une années durant lesquelles une petite société installée dans un ancien moulin devient le deuxième constructeur informatique mondial, avant de perdre la maîtrise d'un marché qu'elle avait contribué à transformer.

Prologue — Le monde d'IBM

En 1957, le mot « ordinateur » n'évoque ni un objet posé sur un bureau, ni un outil que l'on allume pour quelques minutes. Il désigne une installation. Derrière une porte surveillée, des armoires occupent une salle entière, des bandes magnétiques tournent devant des voyants, des lecteurs avalent des paquets de cartes et une imprimante produit les résultats plusieurs heures plus tard. Autour de la machine travaillent opérateurs, programmeurs, techniciens de maintenance et responsables de l'ordonnancement. L'utilisateur ordinaire n'entre pas nécessairement dans la pièce.

Ce monde est dominé par **IBM**. L'entreprise vient des machines à cartes perforées et de l'organisation mécanographique des grandes administrations. Elle maîtrise le commerce, le service, la location des équipements et la relation avec les directions générales. Ses ordinateurs scientifiques de la série 700 et ses machines de gestion donnent une forme industrielle à un secteur encore jeune. Pour un nouveau constructeur, la difficulté n'est donc pas seulement de concevoir une bonne électronique : il faut affronter une organisation mondiale, des équipes de vente, des techniciens présents chez les clients et une réputation déjà installée.

Le contraste entre les deux mondes ne doit toutefois pas être simplifié. **IBM** ne représente pas seulement la lourdeur bureaucratique que les jeunes ingénieurs aiment dénoncer. L'entreprise fournit des machines fiables, une documentation abondante et un service que ses clients peuvent appeler dans plusieurs pays. Elle a compris très tôt qu'un ordinateur n'est pas vendu une seule fois : il faut installer les périphériques, former les opérateurs, remplacer les

pièces et garantir la continuité des programmes. Ce savoir industriel constitue une barrière au moins aussi haute que la technique.

Pour les directions d'entreprise, cette organisation rassure. Confier la paie, la facturation ou les stocks à une machine oblige à croire que le constructeur existera encore plusieurs années plus tard. La domination d'IBM vient donc d'un ensemble : architecture, location, cartes perforées, imprimantes, équipes commerciales et maintenance. Un nouvel entrant ne peut espérer l'imiter avec quelques bons circuits. Il lui faut choisir un autre terrain.

En dehors des entreprises, les universités et les organismes publics dépendent eux aussi de centres de calcul. Les chercheurs réservent du temps, remettent leurs programmes et récupèrent les résultats. Les machines sont parfois remarquablement rapides, mais leur organisation tient l'expérimentation à distance. Cette situation crée une frustration particulière chez ceux qui ont connu les premiers systèmes interactifs du MIT : ils savent qu'une autre relation avec l'ordinateur est techniquement possible, tout en constatant qu'elle reste presque absente du marché.

La machine avant l'utilisateur

Dans les centres de calcul, le traitement par lots s'impose comme une réponse rationnelle à la rareté du matériel. Les programmes et leurs données sont préparés sur cartes ou sur bande, regroupés, puis exécutés les uns après les autres. Cette méthode évite de laisser un équipement coûteux attendre qu'un humain réfléchisse devant sa console. Elle optimise le temps de la machine, mais allonge celui du programmeur. Une erreur de frappe peut n'apparaître qu'après le passage du lot ; il faut alors corriger le paquet, le soumettre de nouveau et attendre une autre exécution.

Le décalage ne choque pas tous les utilisateurs. Pour calculer une paie, trier un fichier de clients ou résoudre une série d'équations préparées à l'avance, le modèle fonctionne. Il devient plus gênant lorsqu'un scientifique veut modifier immédiatement un paramètre, lorsqu'un ingénieur souhaite relier un capteur au calculateur ou lorsqu'un programmeur cherche une erreur en observant pas à pas

le comportement de son code. Dans ces situations, le temps humain reprend de la valeur.

La puissance reste également inséparable de ses périphériques. Un processeur isolé ne sert à rien sans lecteurs, imprimantes, bandes et moyens de stockage. Le coût réel d'un système comprend donc des équipements nombreux, des locaux adaptés et une maintenance permanente. Les gros ordinateurs se comptent en centaines de milliers de dollars, parfois en millions. Ils sont achetés ou loués par des organisations capables d'assurer un taux d'utilisation élevé.

Le lot possède sa propre géographie. Le programmeur travaille dans un bureau ou une salle de perforation ; l'opérateur, derrière une autre porte, manipule les supports et surveille la console. Entre eux circule un paquet de cartes accompagné d'instructions. Cette séparation protège la machine contre les erreurs et permet d'enchaîner les travaux, mais elle retire au programmeur la possibilité de regarder le système réagir.

Un cycle de mise au point peut ainsi occuper une journée pour une erreur qui serait corrigée en quelques secondes devant une console. Les programmeurs apprennent à relire soigneusement leur code et à préparer des sorties de diagnostic. Cette rigueur n'est pas inutile, mais elle décourage les essais spontanés. Elle favorise les tâches bien définies, moins les recherches où le problème se précise à mesure que l'on obtient des résultats.

Le traitement par lots n'est donc pas un archaïsme irrationnel. Il est la réponse économique à un ordinateur rare. La rupture viendra lorsqu'un constructeur acceptera de valoriser autrement le temps : non plus seulement le temps de la machine, mais celui de l'ingénieur, du scientifique ou de l'étudiant qui attend devant elle.

Un marché qui s'élargit sans se rapprocher

IBM comprend elle-même que le marché ne peut rester limité aux plus grandes installations. Annoncé en 1959, l'IBM 1401 est proposé à partir d'environ 2 500 dollars par mois, bien moins que les grandes configurations scientifiques. Son succès est immense : au

milieu des années 1960, la famille 1401 représente, selon IBM, plus de la moitié des ordinateurs installés dans le monde. La machine rapproche l'automatisation des entreprises moyennes, mais elle demeure un système de traitement de données organisé autour des cartes, des bandes et de l'impression. Elle ne place pas encore le programmeur au centre de l'expérience. [1]

En 1964, le System/360 pousse plus loin la logique industrielle. IBM remplace plusieurs gammes incompatibles par une famille commune, capable de couvrir les besoins scientifiques et commerciaux. L'investissement est colossal, mais la promesse est simple : un client pourra changer de puissance sans abandonner ses programmes ni ses périphériques. La compatibilité devient un avantage stratégique aussi important que la vitesse. Ce principe marquera toute l'industrie et, quelques années plus tard, posera à **DEC** une question difficile : comment préserver l'agilité des petites machines tout en offrant la continuité attendue d'un grand constructeur ? [2]

Le succès du 1401 confirme qu'il existe un immense marché sous les plus gros calculateurs. Mais il confirme aussi la puissance de la logique IBM : une machine plus accessible reste intégrée à une chaîne de cartes, de bandes, d'impression et de service. Quelques années plus tard, le System/360 réunira sous une architecture commune des modèles de puissances très différentes. Pour la première fois à cette échelle, un client peut espérer grandir sans récrire tous ses programmes. Cette continuité deviendra une référence que même les constructeurs de petites machines devront affronter.

À l'autre extrémité, des ingénieurs commencent à raisonner non en volumes de dossiers traités, mais en secondes de réponse. Un radar, un simulateur ou un appareil de mesure impose son propre rythme. Lorsque l'événement physique change, le calculateur doit le suivre. Le dialogue avec la machine naît moins d'un désir de confort que d'une nécessité opérationnelle. C'est cette nécessité qui donnera à *Ken Olsen* et *Harlan Anderson* une expérience que la plupart des futurs entrepreneurs de l'informatique ne possèdent

1. IBM. *The IBM 1401*. IBM Heritage ; Paul E. Ceruzzi. *A History of Modern Computing*. 2e éd. Cambridge, Massachusetts : MIT Press, 2003. ISBN : 978-0-262-53203-7.

2. IBM. *The IBM System/360*. IBM Heritage.

pas.

Une brèche dans le modèle

Le grand système n'est pourtant pas le seul avenir possible. Les besoins militaires en temps réel, les expériences de radar, les simulateurs et les travaux sur l'interaction graphique imposent une autre relation entre l'homme et la machine. Dans ces environnements, recevoir le résultat le lendemain n'a aucun sens. L'ordinateur doit réagir pendant que l'événement se produit, afficher une situation et accepter une commande immédiate. [3]

C'est dans cette brèche que se forme l'idée de *Ken Olsen*. Au **MIT Lincoln Laboratory**, il rencontre des ordinateurs conçus non pour absorber silencieusement des lots, mais pour être observés, commandés et modifiés par des ingénieurs. À ses côtés, un jeune physicien nommé *Harlan Anderson* découvre le même univers. Ils ne cherchent pas encore à détrôner IBM. Leur ambition est plus précise et, à court terme, plus prudente : construire des éléments électroniques que des techniciens pourront employer directement sur leur établi.

Cette prudence donnera naissance à une entreprise qui, pendant quatre décennies, travaillera à déplacer la frontière entre la salle informatique et l'utilisateur. Son histoire commence avant le premier PDP, avant même que ses fondateurs osent inscrire le mot « ordinateur » dans leur projet.

3. Le *temps réel* ne signifie pas nécessairement que le calcul est très rapide. Il signifie que le système doit répondre avant une échéance imposée par le phénomène qu'il observe ou contrôle.

Première partie

Deux ingénieurs face aux grands systèmes

Chapitre 1

Deux chemins vers Lincoln

Avant de devenir les fondateurs de **Digital Equipment Corporation**, *Ken Olsen* et *Harlan Anderson* suivent deux parcours qui ne semblent pas devoir se rejoindre. L'un grandit dans le Connecticut, répare des radios et travaille l'été dans un atelier mécanique. L'autre étudie la physique dans l'Illinois et découvre l'informatique au moment où elle se construit encore à partir de schémas, de tubes et de cours donnés par ceux qui inventent la discipline.

Leur rencontre ne doit rien à un projet entrepreneurial mûri dans une école de commerce. Elle se produit dans un laboratoire chargé de résoudre un problème militaire considérable. Cette origine explique une grande partie de ce que deviendra **DEC** : le goût des systèmes concrets, la confiance accordée aux ingénieurs et l'idée qu'un ordinateur peut être un instrument avec lequel on travaille, non une autorité lointaine à laquelle on remet des cartes.

1.1 Ken Olsen, le goût du concret

Kenneth H. Olsen naît en 1926 dans le Connecticut. Adolescent, il démonte et répare des postes de radio, activité qui oblige à relier un symptôme visible à un composant caché. Il travaille également dans un atelier mécanique. Ce mélange d'électronique et de fabrication restera au cœur de sa manière de penser : une idée n'acquiert de valeur que lorsqu'elle peut être construite, testée et rendue fiable.

Après un service dans l'US Navy entre 1944 et 1946, il entre au **Massachusetts Institute of Technology**. Il obtient un diplôme de premier cycle en génie électrique en 1950, puis un master en 1952. Le MIT de l'après-guerre n'est pas seulement une université prestigieuse. Les recherches sur le radar, la mémoire à tores et les calculateurs numériques ont créé une continuité entre sciences, défense et industrie. Olsen se trouve au milieu d'une génération pour laquelle l'électronique n'est plus un ensemble de composants isolés : elle devient l'infrastructure de systèmes capables de percevoir, calculer et agir. [1]

Son tempérament est déjà celui d'un constructeur. Il s'intéresse à la fiabilité, aux méthodes de test et aux architectures modulaires. Au lieu d'accepter qu'un ordinateur soit une pièce unique dont chaque panne exige une enquête interminable, il cherche des moyens de séparer les fonctions, d'observer les signaux et de remplacer rapidement ce qui ne fonctionne pas.

La trajectoire d'Olsen n'est pas celle d'un théoricien qui découvrirait tardivement la fabrication. Il apprend au contraire à juger une conception par ses pannes. Les mémoires à tores, par exemple, promettent une mémoire rapide et non volatile, mais leur mise au point exige des procédés précis et des dispositifs de test capables de détecter des défauts intermittents. Dans ce travail, la frontière entre invention et production s'efface : une technologie n'est réellement utile que si elle peut être reproduite.

Cette expérience explique aussi son rapport futur aux utilisateurs. Un ingénieur de terrain qui décrit une panne ne doit pas être considéré comme le dernier maillon d'une hiérarchie. Il apporte une information sur le produit. Olsen conservera longtemps l'habitude de regarder les machines, d'interroger ceux qui les réparent et de défendre des décisions à partir d'une preuve matérielle. Cette attention donnera à **DEC** une exigence de qualité, mais elle rendra parfois difficiles les arbitrages qui ne peuvent être tranchés sur un établi.

1. Computer History Museum. *Ken Olsen* ; MIT News. *Kenneth Olsen, Computing Pioneer and Founder of DEC, Dies at 84*. 2011.

1.2 Harlan Anderson, de la physique à l'informatique

Harlan E. Anderson naît en 1929 à Freeport, dans l'Illinois. À l'université de l'Illinois, où il étudie la physique, il suit en 1950 des cours liés à l'ILLIAC I, alors en construction. L'ordinateur n'est pas encore un produit que l'on apprend à utiliser dans un manuel standard. Les étudiants approchent une machine conçue localement, inspirée des principes de l'architecture à programme enregistré. Anderson découvre ainsi l'informatique par la programmation et par la proximité avec ceux qui bâtissent le matériel.

Il obtient un bachelor en 1951 puis un master en physique en 1952. Cette même année, il rejoint le Lincoln Laboratory. Son premier responsable est Ken Olsen. Le lien hiérarchique précède donc de plusieurs années l'association entre les deux hommes. Anderson apporte une formation scientifique et un intérêt marqué pour les usages du calcul ; Olsen possède déjà une expérience plus affirmée de l'électronique et de l'organisation des équipes. [2]

Leurs caractères ne se confondent pas. Les témoignages disponibles présentent Olsen comme un dirigeant fortement attaché au produit, capable de défendre une intuition technique avec obstination. Anderson joue un rôle important dans la création de l'entreprise, son financement et ses premières relations professionnelles. Cette complémentarité sera précieuse au démarrage, même si leurs trajectoires finiront par diverger.

Anderson découvre de son côté une discipline encore dépourvue de frontières professionnelles stables. Un physicien peut devenir programmeur, participer au câblage d'une interface puis discuter d'un usage scientifique. Cette mobilité compte : elle forme des ingénieurs qui ne séparent pas nettement le matériel, le logiciel et l'application. Dans la future entreprise, un produit sera souvent conçu à partir du problème d'un client plutôt qu'à partir d'une catégorie établie par le marché.

Il apporte également un tempérament plus extérieur. Là où Olsen

2. Computer History Museum. *Harlan Anderson* ; Computer History Museum. *Harlan Anderson*.

aime la profondeur technique et le contrôle du produit, Anderson se montre attentif aux réseaux professionnels, au financement et à la manière dont une jeune société peut se présenter. Le contraste ne doit pas être transformé en opposition romanesque : les deux hommes sont d'abord des ingénieurs et partagent la même confiance dans les petites équipes. Mais leurs priorités ne sont pas identiques, et cette différence réapparaîtra lorsque **DEC** cessera d'être une aventure de quelques dizaines de personnes.

1.3 Un rendez-vous dans l'histoire

En 1952, aucun des deux hommes ne peut prévoir qu'ils fonderont cinq ans plus tard l'un des grands constructeurs informatiques. Leur horizon immédiat est le Lincoln Laboratory, créé par le MIT pour répondre à la menace des bombardiers soviétiques. Le laboratoire rassemble des mathématiciens, des électroniciens, des spécialistes du radar et des programmeurs autour du projet SAGE.

Ce contexte compte davantage que la simple ligne biographique. Les futurs fondateurs de **DEC** apprennent dans un environnement où l'ordinateur doit communiquer avec des radars, afficher des informations et recevoir des ordres en quelques secondes. Ils voient aussi combien les systèmes deviennent difficiles à construire et à maintenir lorsqu'ils sont conçus comme des ensembles monolithiques.

Leur rencontre possède enfin une dimension générationnelle. Ils arrivent à l'informatique au moment où aucune carrière standard n'existe encore. Les écoles ne forment pas des architectes de processeurs au sens moderne ; les langages et les systèmes d'exploitation ne constituent pas des industries séparées. Les personnes qui construisent les machines définissent en même temps les méthodes de travail. Cette liberté permet des choix audacieux, mais elle impose de tout apprendre : électronique, programmation, fabrication et relation avec les utilisateurs.

Leur projet d'entreprise naîtra de cette double expérience : l'enthousiasme devant une informatique interactive et l'impatience face au poids des grandes organisations. Mais avant Maynard, il faut comprendre l'école qui leur donne cette vision.

Chapitre 2

Le laboratoire des possibles

Le **MIT Lincoln Laboratory** naît en 1951 dans l'urgence de la guerre froide. Les États-Unis veulent détecter assez tôt des bombardiers susceptibles d'arriver par le nord et coordonner une réponse sur un territoire immense. Le système SAGE doit relier radars, centres de calcul, écrans et communications. Aucun ordinateur existant ne peut remplir seul cette mission. Il faut inventer une grande partie de l'infrastructure en même temps que le système.

Pour *Ken Olsen* et *Harlan Anderson*, le laboratoire est une école incomparable. Ils y voient la technique sortir des limites du calcul numérique traditionnel. Un écran n'est plus seulement un périphérique de démonstration : il représente une situation tactique. Un opérateur ne se contente pas d'attendre une impression : il désigne une information et reçoit une réponse. Le logiciel devient assez vaste pour nécessiter une organisation nouvelle. La fiabilité cesse d'être une qualité souhaitable : elle devient une condition de fonctionnement.

2.1 Du calcul à l'interaction

Le projet Whirlwind, lancé au MIT avant la création du Lincoln Laboratory, avait déjà rompu avec l'ordinateur calculateur de tables. Conçu à l'origine pour un simulateur de vol, il devait réagir aux actions d'un pilote. Cette exigence conduisit à privilégier un fonctionnement rapide et régulier, une mémoire adaptée au

temps réel et des moyens d'affichage. SAGE reprend et étend cette orientation à une échelle nationale.

Le TX-0, construit au milieu des années 1950, pousse plus loin l'usage des transistors et de la mémoire à tores. Il sert d'abord à tester des technologies destinées au TX-2, puis rejoint le campus du MIT. Son écran et son crayon lumineux rendent possible une relation directe avec le programme. Le calculateur n'est plus seulement un service auquel on soumet une tâche ; il devient un objet que l'on explore. Le Lincoln Laboratory présente aujourd'hui le TX-0 comme l'un des premiers ordinateurs entièrement transistorisés et comme une étape majeure de l'informatique interactive. [1]

Olsen travaille notamment sur des équipements de test de mémoire et sur des éléments des TX-0 et TX-2. Anderson participe au projet TX-0 après avoir été recruté par Olsen. Ils observent une idée dont l'industrie ne mesure pas encore toute la portée : lorsqu'un utilisateur obtient rapidement une réponse, il modifie sa manière de penser. Il essaie, corrige, recommence. Le programme devient moins un objet livré à la machine qu'une conversation conduite avec elle.

Dans les salles de contrôle de SAGE, l'écran et le dispositif de pointage annoncent un futur qui n'est pas encore celui du bureau. L'opérateur voit des symboles, sélectionne une trace et transmet une décision. La machine n'attend pas une pile de cartes ; elle maintient un état du monde et le met à jour. Cette relation transforme la notion même de programme. Le logiciel n'est plus seulement une suite de calculs qui se termine par un résultat, mais un système qui doit rester actif, recevoir des événements et continuer à répondre.

Pour les ingénieurs, cette permanence introduit une nouvelle discipline. Une panne qui interromprait un calcul scientifique peut faire perdre quelques heures. Dans un dispositif de défense, elle ouvre une période d'aveuglement. Les techniques de test, de redondance et de diagnostic prennent donc une importance que les brochures commerciales résument mal. Olsen apprend que la disponibilité se construit dès l'architecture, dans la manière de découper

1. MIT Lincoln Laboratory. *Lincoln Laboratory History*. Massachusetts Institute of Technology ; MIT Lincoln Laboratory. *Interactive Supercomputing : From Whirlwind to TX-2*. Massachusetts Institute of Technology.

et d'observer la machine.

2.2 La leçon de la modularité

SAGE est aussi une leçon par l'excès. Une installation composée de milliers de composants ne peut être maintenue uniquement par des spécialistes capables de comprendre l'ensemble. Il faut découper le système en fonctions testables, standardiser les connexions et prévoir le remplacement des éléments défaillants. La modularité n'est donc pas un choix esthétique : elle réduit le temps de diagnostic et permet à plusieurs équipes de travailler en parallèle.

Olsen développe au Lincoln Laboratory des solutions de test et s'intéresse à des blocs électroniques réutilisables. Cette démarche prépare directement les premiers produits de **DEC**. Un ingénieur qui veut construire un compteur, un additionneur ou un contrôleur ne devrait pas recommencer à chaque fois par un schéma complet. Il devrait pouvoir assembler des fonctions connues, comme on combine des instruments sur un établi.

Le laboratoire forme également ses ingénieurs à une culture du prototype. Une idée doit être câblée, mesurée et confrontée à l'usage. La démonstration possède une force particulière : elle permet de trancher des débats que les notes administratives prolongeraient sans fin. Cette confiance dans la preuve matérielle deviendra l'une des caractéristiques de l'entreprise de Maynard.

Le TX-0 ajoute une autre leçon, plus intime. Lorsqu'il rejoint le campus du MIT, la machine est utilisée par des chercheurs et des étudiants qui peuvent approcher sa console. Ils ne la considèrent pas seulement comme un équipement coûteux à protéger. Ils examinent ses réactions, écrivent des outils et modifient leurs programmes au fil des essais. Ce comportement n'est pas encore nommé comme une culture, mais il prépare celle que le PDP-1 rencontrera quelques années plus tard.

Le laboratoire offre aussi un réseau humain. Des ingénieurs passent de projet en projet, connaissent les fournisseurs et apprennent à reconnaître ceux qui savent transformer un schéma en matériel fiable. Les futurs fondateurs ne quitteront donc pas le MIT

avec une idée isolée. Ils emporteront des méthodes, des contacts et la conviction que la région de Boston peut soutenir une industrie électronique nouvelle.

2.3 Quitter une institution puissante

Le Lincoln Laboratory offre des moyens considérables, mais sa mission et sa taille limitent l'autonomie individuelle. À mesure que les projets grossissent, le délai entre une idée et sa réalisation peut s'allonger. Olsen souhaite fabriquer des équipements plus petits, destinés aux ingénieurs et aux laboratoires civils. Anderson partage la conviction qu'un marché existe pour des modules et, à terme, pour des systèmes moins coûteux que les grands ordinateurs.

La décision de partir mûrit dans un contexte où le mot « ordinateur » effraie les investisseurs. Les machines connues demandent des millions, des équipes nombreuses et des années de développement. Olsen et Anderson ne peuvent pas présenter leur projet comme la construction d'un concurrent direct des grands systèmes. Ils doivent réduire l'ambition visible pour préserver l'ambition réelle : rendre disponibles les briques d'une informatique plus proche des ingénieurs.

Partir reste pourtant risqué. Les sociétés informatiques connues disposent de capitaux, de brevets, d'usines et d'équipes commerciales. Les deux hommes n'ont pas l'intention de lever immédiatement les fonds nécessaires à un ordinateur complet. Leur stratégie consiste à commencer par ce qu'ils savent construire, ce que les laboratoires savent déjà acheter et ce que leur expérience rend crédible : des modules logiques.

Cette prudence n'est pas une absence d'ambition. Elle est la première expression d'une méthode qui reviendra souvent chez **DEC** : entrer par un besoin précis, gagner la confiance des utilisateurs techniques, puis élargir progressivement le système.

Chapitre 3

Le moulin de Maynard

Au printemps 1957, *Ken Olsen* et *Harlan Anderson* ne disposent ni d'une usine, ni d'un produit terminé, ni d'une fortune personnelle capable de financer plusieurs années de développement. Ils possèdent en revanche une expérience rare des circuits transistorisés, une idée claire du public qu'ils veulent servir et un projet assez prudent pour être défendable devant des investisseurs.

L'entreprise ne naît pas dans un garage devenu légendaire, mais dans le cadre encore nouveau du capital-risque américain. [1] Leur interlocuteur principal est le général *Georges Doriot*, professeur à Harvard et dirigeant de l'**American Research and Development Corporation**. ARD cherche à transformer des recherches issues de la guerre et des universités en entreprises industrielles.

3.1 Un projet volontairement raisonnable

La proposition remise à ARD ne promet pas de renverser IBM. Olsen et Anderson veulent vendre des modules électroniques aux laboratoires, aux fabricants d'équipements et aux ingénieurs qui construisent des systèmes de contrôle. Le marché paraît assez étroit

1. Le *capital-risque* finance une entreprise jeune en échange d'une part de son capital. L'investisseur accepte une forte probabilité d'échec parce qu'une réussite peut prendre une valeur très élevée.

pour ne pas effrayer les grands constructeurs, mais assez large pour soutenir une petite société. La stratégie évite également le coût d'un ordinateur complet, qui exigerait immédiatement logiciel, périphériques, service et force de vente.

ARD investit 70 000 dollars et reçoit une part majoritaire du capital. Le chiffre est modeste à l'échelle de l'industrie informatique, mais il donne aux fondateurs le temps d'acheter du matériel, de recruter et de produire leurs premières cartes. Le financement ne les libère pas de toute contrainte : le conseil d'administration reste attentif aux risques, notamment lorsque l'idée d'un ordinateur apparaîtra. [2]

Le nom choisi est lui-même révélateur. **Digital Equipment Corporation** contient le mot « équipement », non le mot « ordinateur ». Il décrit une entreprise d'électronique numérique capable de vendre des éléments à d'autres constructeurs. Ce détour protège le projet contre l'image d'un marché réservé à IBM, **Sperry Rand** ou **RCA**. Il laisse aussi ouverte la possibilité de fabriquer un jour des systèmes complets.

Le financement par **American Research and Development** n'est pas un détail administratif. La société de *Georges Doriot* appartient aux premières expériences américaines de capital-risque organisé. Elle investit l'argent de plusieurs institutions dans des entreprises dont la valeur repose moins sur des actifs existants que sur une technologie et une équipe. Pour Olsen et Anderson, cette forme de financement évite de dépendre entièrement d'un industriel qui pourrait absorber leur projet.

Doriot n'accorde pas un chèque sans conditions. Il exige un plan, un produit rapidement vendable et une discipline de gestion. L'idée des modules répond exactement à cette attente : ils peuvent être dessinés, fabriqués et facturés sans attendre l'achèvement d'un ordinateur complet. Le futur reste ouvert, mais la survie se joue sur les commandes du mois.

Le partage du capital, très favorable à ARD au départ, rappelle la fragilité des fondateurs. Leur richesse éventuelle dépendra de

2. Computer History Museum, *Ken Olsen*, cf. note 1 ; Computer History Museum, *Harlan Anderson*, cf. note 2 ; Computer History Museum. *Digital Equipment Corporation Founded*. 1957.

la croissance de la société, non de la conservation immédiate du contrôle financier. Cette situation contribue à leur obsession du chiffre d'affaires et du produit livrable : chaque série de modules vendue rapproche **DEC** d'une autonomie que le contrat initial ne lui donne pas encore.

3.2 L'ancien moulin

La société s'installe à Maynard, dans un vaste ensemble industriel du Massachusetts construit au XIX^e^ siècle. Les bâtiments avaient abrité une production textile. Ils offrent des surfaces disponibles à un coût compatible avec les moyens de la jeune entreprise. Les premières installations occupent un peu plus de 8 000 pieds carrés. Le contraste est saisissant : une technologie associée aux laboratoires militaires et aux calculateurs les plus modernes prend place derrière les murs d'un ancien moulin. [3]

Le lieu ne reste pas un simple décor. Maynard devient progressivement une ville d'entreprise. Les ateliers, les bureaux et les couloirs du moulin accueillent des équipes de plus en plus nombreuses. Les bâtiments s'étendent sans que la société perde complètement l'impression de proximité de ses débuts. Des ingénieurs peuvent traverser un étage pour parler à la fabrication, observer un prototype ou discuter avec le service de terrain.

Cette proximité nourrit une culture particulière. Le produit circule physiquement entre ceux qui le conçoivent, ceux qui le montent et ceux qui le réparent. Le dirigeant n'est pas installé loin de la machine. Olsen restera longtemps attaché à ce contact direct, parfois au prix d'une organisation difficile à formaliser lorsque l'entreprise comptera des dizaines de milliers d'employés.

Maynard se situe à distance de Cambridge, mais reste dans l'orbite de la Route 128, où se concentrent laboratoires, fournisseurs et jeunes entreprises électroniques. Le moulin offre davantage qu'un loyer modéré. Ses volumes permettent de déplacer des cloisons,

3. Computer History Museum. *Digital Equipment Corporation* ; Digital Equipment Corporation, Information Research Services. *DIGITAL Computing History Timeline, 1957–1997*. 1998.

d'installer de nouveaux ateliers et d'ajouter des équipes sans quitter immédiatement le site. À mesure que la société grandit, l'ancienne usine devient un assemblage presque organique de bâtiments, de passerelles et de couloirs.

Cette géographie influence la mémoire de l'entreprise. Les anciens employés se souviendront moins d'un siège abstrait que d'un lieu où l'on pouvait croiser un prototype, un technicien de terrain ou le président dans le même bâtiment. La proximité donne de la vitesse aux débuts. Elle nourrit aussi le sentiment qu'une grande entreprise peut continuer à fonctionner comme un atelier, conviction de plus en plus difficile à tenir lorsque les effectifs se comptent en dizaines de milliers.

3.3 Deux fondateurs, plusieurs rôles

Olsen dirige l'orientation technique et industrielle. Il veille à la qualité, au coût et à la possibilité de fabriquer en série. Anderson participe à la gestion, aux relations extérieures et au développement de l'entreprise. Autour d'eux, une petite équipe doit accomplir des tâches que les grandes sociétés répartissent entre plusieurs services : dessiner les circuits, acheter les composants, assembler les cartes, écrire la documentation, chercher des clients et répondre aux premières pannes.

La petite taille impose une discipline. Un produit mal conçu ne peut pas être sauvé par une campagne publicitaire massive. Il faut qu'il fonctionne chez des utilisateurs capables de comprendre ce qu'ils achètent. Ces premiers clients sont souvent des ingénieurs ; ils lisent les schémas, mesurent les signaux et demandent des modifications. La relation commerciale prend donc la forme d'une conversation technique.

À la fin du premier exercice, **DEC** a vendu pour environ 94 000 dollars de modules et compte déjà une soixantaine d'employés. La société reste minuscule face à IBM, mais le modèle est validé. Elle sait concevoir, produire, documenter et vendre. Elle possède surtout un catalogue de fonctions logiques qui, assemblées autrement, pourraient former une machine complète.

La première année confirme pourtant que le projet ne repose pas seulement sur l'enthousiasme de deux fondateurs. Les commandes obligent à tenir des délais, les retours clients révèlent les faiblesses des cartes et les nouveaux employés apportent leurs propres méthodes. **DEC** apprend vite parce que le produit est simple à comparer : un module fonctionne ou ne fonctionne pas, respecte ses niveaux électriques ou les dépasse. Cette clarté réduit les débats et installe une culture de la mesure.

Le conseil d'administration n'est pas encore prêt à financer une aventure aussi risquée. Les fondateurs devront d'abord montrer que les modules ne sont pas une étape provisoire, mais une activité solide. C'est en apprenant à fabriquer ces briques que **DEC** acquiert les moyens de construire son premier ordinateur.

Chapitre 4

Commencer par les modules

Les premiers produits de **DEC** ne ressemblent pas à des ordinateurs miniatures. Ce sont des boîtiers et des cartes qui accomplissent une fonction logique précise : amplifier un signal, compter des impulsions, mémoriser un état ou réaliser une opération booléenne. Un chercheur peut les poser sur un établi, les relier par des cordons et construire rapidement un dispositif adapté à son expérience.

Cette apparente modestie donne à la jeune entreprise trois avantages. Les modules correspondent à un besoin réel, ils peuvent être vendus sans développer un environnement logiciel et leur fabrication enseigne à **DEC** la discipline industrielle qui manque souvent aux prototypes de laboratoire. Chaque carte doit fonctionner comme la précédente, résister au transport et être comprise à partir d'une documentation claire.

4.1 Des briques pour les ingénieurs

Les *Laboratory Modules* sont conçus pour l'expérimentation et le contrôle. Leur présentation rend les connexions visibles. L'utilisateur peut modifier le câblage, observer le résultat et réemployer les mêmes éléments dans un autre projet. Le produit prolonge directement la culture du Lincoln Laboratory : décomposer une fonction complexe en blocs identifiables, testables et remplaçables.

Les *System Modules*, commercialisés ensuite, augmentent la

densité et utilisent un câblage de fond de panier plus fixe. Ils conviennent à des ensembles destinés à fonctionner durablement : testeurs de mémoire, contrôleurs, équipements industriels et systèmes logiques plus complexes. **DEC** les présente comme des « Digital Building Blocks », expression qui transforme la modularité en promesse commerciale. [1]

Le catalogue joue un rôle essentiel. Il ne se contente pas de donner un prix et une référence. Il explique les niveaux électriques, les délais, les connexions et les combinaisons possibles. Pour un ingénieur, lire ce document revient presque à suivre un cours de conception numérique. Cette transparence attire un public qui ne veut pas acheter une boîte fermée, mais comprendre comment intégrer le produit à son propre système.

Le catalogue transforme aussi la relation commerciale. Un représentant n'a pas besoin de promettre une application complète ; il peut demander à l'ingénieur ce qu'il cherche à compter, mémoriser ou commander, puis sélectionner les blocs correspondants. La vente commence par le schéma du client. Cette manière de travailler prépare le modèle OEM et l'attention durable de **DEC** aux interfaces.

Les utilisateurs ne restent pas passifs. Certains associent les cartes à des relais, des instruments analogiques ou des mémoires d'autres constructeurs. Ils signalent des besoins qui conduisent à de nouveaux modules. Le catalogue devient ainsi le résultat d'une conversation entre Maynard et les laboratoires. Il révèle les usages avant même que l'entreprise sache les classer en marchés.

4.2 Apprendre la série

La production de cartes oblige **DEC** à résoudre des problèmes moins spectaculaires que l'architecture d'un ordinateur, mais tout aussi déterminants : choix des composants, approvisionnement, contrôle qualité, méthodes de soudure et organisation des tests. Une différence minime entre deux séries peut provoquer une panne

1. Digital Equipment Corporation, Information Research Services, cf. note 3.

difficile à reproduire chez le client. L'entreprise développe donc des bancs de test et des procédures qui rapprochent la fabrication de la conception.

Cette expérience prépare les futurs ordinateurs. Un PDP ne sera pas un prototype entièrement nouveau, mais un assemblage organisé de modules dont les caractéristiques sont connues. L'inventaire de l'entreprise devient une bibliothèque matérielle. Selon une remarque attribuée avec humour à *Ben Gurley*, le PDP-1 aurait été conçu en partie pour être réalisé avec ce qui se trouvait déjà en stock. L'anecdote résume une contrainte réelle : la beauté de l'architecture doit composer avec le prix, les délais et la fabrication. [2]

Les modules permettent également de vendre avant que le marché des petits ordinateurs soit clairement identifié. Ils entrent chez des clients qui deviendront ensuite des acheteurs naturels de systèmes complets. Un laboratoire satisfait d'un compteur ou d'un contrôleur **DEC** connaît déjà la qualité des produits et les ingénieurs de Maynard.

L'apprentissage de la série modifie également le travail des concepteurs. Au laboratoire, un circuit peut être ajusté jusqu'à ce qu'un exemplaire fonctionne. Dans une usine, la conception doit tolérer les variations des composants et être testée par une personne qui n'a pas participé au dessin. La documentation, les points de mesure et les limites électriques deviennent des parties du produit.

Cette contrainte expliquera la robustesse des premiers PDP. Leur logique ne repose pas sur une accumulation d'astuces impossibles à reproduire. Elle utilise des fonctions déjà caractérisées. Le processeur est nouveau comme organisation, mais beaucoup de ses éléments appartiennent à une famille industrielle éprouvée. La modularité réduit ainsi le risque du passage à l'ordinateur complet.

2. Computer History Museum. *Origins of the PDP-1*.

4.3 La tentation de la machine complète

À mesure que les commandes progressent, la frontière entre module et ordinateur devient moins nette. Certaines installations réunissent mémoire, logique de contrôle et interfaces en quantité suffisante pour ressembler à une petite machine spécialisée. Des clients demandent des ensembles plus intégrés. Les ingénieurs de **DEC** savent, par leur expérience du TX-0, qu'une architecture interactive peut être construite avec cette technologie.

Le conseil d'administration reste prudent. Les entreprises informatiques ont besoin d'un financement important et d'un service durable. Un ordinateur vendu sans logiciel ni périphériques peut devenir un coût plutôt qu'un succès. Olsen et Anderson doivent donc convaincre qu'il ne s'agit pas d'imiter IBM en plus petit, mais de servir un marché que les grands systèmes couvrent mal.

Le vocabulaire fournit une solution. La future machine ne sera pas présentée comme un *computer*, terme chargé de risques financiers et commerciaux, mais comme un *Programmed Data Processor*. Le nom décrit un équipement de traitement programmable, proche des instruments déjà achetés par les laboratoires. L'acronyme PDP deviendra pourtant l'un des plus célèbres de l'histoire informatique.

Le débat au conseil ne porte donc pas seulement sur la faisabilité. Il porte sur l'identité de l'entreprise. Vendre un ordinateur signifie promettre un support pendant des années, fournir des logiciels et accepter que la panne d'un seul système puisse mobiliser plusieurs ingénieurs. Olsen soutient que le marché existe précisément parce que les grands constructeurs ne servent pas bien ces utilisateurs. Les modules ont créé la technologie et les relations nécessaires ; refuser la machine complète reviendrait à laisser d'autres entreprises exploiter le terrain préparé par **DEC**.

En 1959, la société possède les circuits, l'expérience de production et les premiers clients. Il lui manque un architecte capable de transformer ces ressources en une machine cohérente. *Ben Gurley*, recruté après avoir travaillé sur des systèmes informatiques et des circuits rapides, va donner une forme concrète à l'idée.

Deuxième partie

Rapprocher la machine

Chapitre 5

Le PDP-1, vendre une idée

Le premier ordinateur de **DEC** ne naît pas d'une étude de marché démontrant l'existence d'une nouvelle catégorie de produits. Il naît de la conviction qu'une machine inspirée des TX du Lincoln Laboratory peut intéresser des laboratoires qui n'ont ni les moyens ni l'envie d'exploiter un grand centre de calcul. Il faut néanmoins transformer cette conviction en un système assez fiable pour être vendu et assez différent pour ne pas se retrouver dans une comparaison frontale avec IBM.

Ben Gurley joue ici le rôle décisif. Ingénieur rapide et inventif, il conçoit l'architecture du PDP-1 en quelques mois à partir des modules et des techniques maîtrisées par la société. La machine utilise des mots de 18 bits, une mémoire à tores de 4 096 mots dans sa configuration de base et un jeu d'instructions volontairement compact. Elle n'est pas construite pour battre les plus gros calculateurs sur tous les terrains. Elle doit offrir une puissance suffisante à un prix et dans un volume accessibles à une équipe de recherche.

Le choix des 18 bits rattache la machine à une tradition scientifique et aux travaux du TX-0, sans tenter de reproduire les formats des ordinateurs de gestion. La mémoire de base paraît minuscule, mais elle suffit pour des programmes interactifs à condition d'écrire avec économie. Surtout, l'architecture prévoit que le processeur soit interrompu par un événement extérieur et qu'il commande des périphériques variés. Cette ouverture est plus importante pour le marché visé qu'une longue liste d'opérations arithmétiques.

Le panneau avant donne une forme visible à cette philosophie. Les rangées de lampes ne sont pas une décoration destinée aux photographies : elles montrent l'état des registres et les étapes du programme. Les commutateurs permettent d'introduire une instruction, de déposer une valeur en mémoire ou de lancer l'exécution. Un ingénieur peut donc démarrer la machine sans attendre qu'un système d'exploitation prenne le contrôle. Cette proximité exige des connaissances, mais elle crée aussi une confiance : le calculateur n'est pas une boîte interdite.

Gurley dessine une machine suffisamment simple pour être comprise par une petite équipe et suffisamment ouverte pour que chaque client l'adapte. L'économie de moyens devient une signature. Là où une grande architecture cherche à couvrir toutes les applications par avance, le PDP-1 fournit un noyau autour duquel l'utilisateur construit son propre environnement.

5.1 La preuve par la démonstration

Le prototype est montré à la fin de 1959 lors de l'Eastern Joint Computer Conference de Boston. Dans un salon rempli de machines et d'annonces, le PDP-1 se distingue moins par sa taille que par son comportement. L'utilisateur peut se tenir devant la console, lancer un programme, observer les registres et recevoir une réponse immédiate. Avec l'affichage à tube cathodique Type 30, la machine trace des points et ouvre la possibilité d'une interaction graphique.

Cette présence change la vente. Un grand système est souvent présenté par ses performances théoriques, ses débits de bandes ou le volume de fichiers qu'il peut traiter. Le PDP-1 peut être démontré comme un instrument. Il devient possible de montrer un programme, de modifier un paramètre et de recommencer devant le client. L'ordinateur cesse d'être seulement une infrastructure et acquiert quelque chose de la machine de laboratoire.

Le prix de base, autour de 120 000 dollars, reste considérable. Il n'annonce pas encore l'ordinateur individuel. Mais il se situe très en dessous des installations scientifiques de plusieurs millions de dollars et ne réclame ni plancher renforcé ni climatisation spécialisée

dans les mêmes proportions. Le coût et la simplicité d'installation permettent à un département ou à un laboratoire d'envisager sa propre machine.

La présentation publique compte d'autant plus que **DEC** ne possède pas encore une force de vente comparable à celles des constructeurs établis. Les visiteurs doivent saisir la différence en quelques minutes. Une courbe qui se déplace à l'écran, une commande donnée depuis la console ou un calcul relancé immédiatement rendent l'argument sensible. Le prospect ne voit pas seulement ce que la machine calcule ; il voit comment il pourrait travailler avec elle.

Cette manière de démontrer le produit rapproche les ingénieurs des commerciaux. Le programme présenté doit fonctionner, l'écran doit être réglé et la console doit résister aux manipulations. La vente devient une épreuve technique en public. Elle expose les défauts, mais elle donne aussi à la jeune société une crédibilité que les slogans ne suffiraient pas à créer.

5.2 BBN, le premier client

Le premier PDP-1 de production est vendu à **Bolt Beranek and Newman**, société de recherche et de conseil de Cambridge connue notamment pour ses travaux en acoustique. *Ed Fredkin*, qui avait vu le prototype, recommande l'achat pour soutenir les recherches de *J. C. R. Licklider* sur la psychoacoustique et l'interaction homme-machine. Le système est livré en 1960.

BBN n'achète pas une boîte fermée. Ses ingénieurs modifient la machine, écrivent des outils et discutent directement avec **DEC**. Fredkin produit notamment l'assembleur FRAP. Cette coopération préfigure une relation qui deviendra caractéristique du constructeur : le client technique n'est pas seulement un utilisateur à former, mais une source d'idées et parfois un coconcepteur. [1]

L'ouverture des entrées-sorties est fondamentale. Un laboratoire peut connecter des convertisseurs, des dispositifs expérimentaux

1. Computer History Museum. *Ed Fredkin* ; Computer History Museum. *Scientific Applications on the PDP-1*.

ou des commandes particulières sans demander au constructeur de redessiner l'ordinateur entier. La machine gagne ainsi une diversité d'usages que sa seule puissance de calcul n'aurait pas assurée.

Chez BBN, l'ordinateur entre dans un milieu où la frontière entre recherche fondamentale, contrat public et application industrielle reste mobile. Les travaux de *J. C. R. Licklider* sur la communication homme–machine trouvent dans le PDP-1 un instrument plus souple qu'un centre de calcul traditionnel. Le client n'attend pas que **DEC** fournisse toutes les réponses ; il veut pouvoir modifier la machine et écrire les logiciels qui manquent.

Cette relation oblige Maynard à écouter. Une amélioration conçue pour un client peut devenir une option, une routine peut circuler vers une autre installation et un ingénieur de terrain peut rapporter un usage auquel les concepteurs n'avaient pas pensé. Le PDP-1 installe ainsi un cycle que **DEC** reproduira longtemps : vendre une plate-forme incomplète mais intelligible, laisser les utilisateurs l'étendre, puis intégrer une partie de leurs idées dans la gamme suivante.

La faiblesse des volumes masque donc une réussite stratégique. Chaque installation fonctionne comme un laboratoire de marché. Les usages graphiques, le temps partagé, la musique, la simulation et le contrôle ne sont pas encore des secteurs clairement séparés. Le PDP-1 permet de les observer avant qu'ils ne deviennent des catégories commerciales.

5.3 Une autre manière de compter

Le PDP-1 n'est vendu qu'à quelques dizaines d'exemplaires. Mesuré avec les critères de la production de masse, le chiffre paraît modeste. Pour **DEC**, son importance est pourtant immense. Il prouve que la société sait livrer un ordinateur complet, soutenir ses utilisateurs et attirer des ingénieurs par une philosophie différente de celle des grands systèmes.

La machine donne aussi une identité au nom PDP. Les modèles suivants ne seront pas simplement des versions plus rapides. **DEC** explorera plusieurs largeurs de mots, plusieurs marchés et plusieurs

technologies. Ce manque d'uniformité pourra devenir une difficulté, mais il exprime d'abord une liberté : chaque architecture doit répondre à un besoin précis.

Le succès le plus durable du PDP-1 ne se produit cependant ni chez BBN ni dans les chiffres de vente. Il commence lorsque **DEC** décide de ne pas vendre son deuxième exemplaire. La société l'offre au département de génie électrique du MIT, là même où ses fondateurs ont appris à considérer l'ordinateur comme un outil interactif. Le geste semble généreux. Il est aussi stratégiquement remarquable.

Chapitre 6

Le cadeau du MIT

En 1961, une jeune entreprise qui n'a encore vendu qu'un seul exemplaire de son ordinateur décide d'en offrir un autre. Le bénéficiaire n'est pas choisi au hasard. Le PDP-1 rejoint le Research Laboratory for Electronics du MIT, institution située au croisement de la recherche, de l'enseignement et des premiers travaux sur l'informatique interactive.

Le don du deuxième PDP-1 de production inaugure, selon les archives du Computer History Museum, une politique de soutien aux établissements qui préparent les futurs professionnels de l'informatique. Il ne s'agit donc pas d'une opération philanthropique isolée. **DEC** place une machine entre les mains d'étudiants et de chercheurs qui pourront l'utiliser intensément, développer des logiciels et emporter plus tard cette expérience dans les entreprises où ils travailleront. [1]

La cérémonie de remise a aussi une valeur symbolique. **DEC** revient vers l'institution dont sont issus ses fondateurs, mais elle n'y retourne plus comme sous-traitant ou groupe de chercheurs. Elle apporte un produit commercial, fabriqué à Maynard et destiné à vivre hors de l'entreprise. Pour une société âgée de quatre ans, voir sa machine installée au MIT constitue une forme de reconnaissance.

Le calcul économique ne peut pas être établi comme une cam-

1. Computer History Museum. *Invitation to a Presentation of a PDP Computer to Massachusetts Institute of Technology.* 1961 ; Computer History Museum. *Invitation to the Presentation of a PDP-1 to MIT.* 1961.

pagne moderne de marketing. Personne ne sait combien de contrats futurs seront directement liés au don. Mais l'intuition est solide : un étudiant formé sur une machine emporte avec lui des habitudes, des programmes et une préférence. Lorsque ces étudiants rejoindront des laboratoires ou des entreprises, ils sauront expliquer ce qu'un PDP permet et parleront le même langage que les ingénieurs de **DEC**.

Le geste est également cohérent avec le marché. Une machine interactive ne se vend pas seulement par sa puissance ; elle se vend par les pratiques qu'elle rend possibles. Offrir le matériel à une communauté capable d'inventer ces pratiques revient à financer indirectement le logiciel, la démonstration et la formation.

6.1 Pourquoi donner une machine rare ?

À première vue, le choix paraît coûteux. Un PDP-1 représente du matériel, des semaines de fabrication et une vente possible. Mais **DEC** ne dispose pas des budgets publicitaires ni des forces commerciales d'IBM. Une machine active dans une université prestigieuse peut accomplir davantage qu'une brochure. Elle forme des utilisateurs, produit des démonstrations et donne à la société une légitimité scientifique.

La décision correspond aussi aux convictions de *Ken Olsen* et *Harlan Anderson*. Ils ont eux-mêmes bénéficié d'un environnement où l'on pouvait approcher le matériel et comprendre son fonctionnement. Les grands ordinateurs IBM présents au MIT restent soumis aux règles d'un centre de calcul. Le PDP-1, lui, peut être confié à une communauté plus restreinte, avec une liberté d'accès inhabituelle.

Jack Dennis, professeur et ancien membre du Tech Model Railroad Club, joue un rôle essentiel. Il supervise l'accès au TX-0 puis au PDP-1 et accepte que des étudiants talentueux utilisent la machine en dehors d'un programme de recherche strictement défini. Cette confiance ne signifie pas l'absence de règles. Le temps machine reste précieux, mais l'initiative individuelle reçoit une place

que le traitement par lots refuse presque toujours.

L'accès n'est pas uniformément libre. Le PDP-1 appartient à un laboratoire, les projets officiels demeurent prioritaires et les utilisateurs doivent prouver qu'ils savent respecter la machine. Cette contrainte donne du relief au rôle de Jack Dennis. Il ne distribue pas simplement des heures ; il reconnaît la compétence et accepte qu'un travail sans objectif contractuel immédiat puisse produire quelque chose d'important.

Cette confiance crée une responsabilité. Les étudiants apprennent à restaurer l'état du système après leurs essais, à partager les supports et à laisser des outils utilisables par les suivants. La communauté se donne des règles informelles parce qu'elle veut conserver l'accès. La liberté n'est pas l'absence d'organisation : elle est une organisation construite par ceux qui dépendent les uns des autres.

6.2 Une machine presque nue

Lorsque le PDP-1 arrive, il ne possède pas encore l'environnement logiciel qu'un lecteur contemporain associerait à un ordinateur. Il faut disposer d'un assembleur, d'un programme de débogage, de routines d'entrée-sortie et d'outils pour exploiter l'affichage. Chez un grand constructeur, ces éléments auraient été attendus du fabricant. Au MIT, leur absence devient une invitation.

Cette situation crée une boucle féconde. Les étudiants développent les programmes dont ils ont besoin. Les outils circulent, sont améliorés, puis rejoignent parfois la bibliothèque de **DEC** ou les installations d'autres clients. La société reçoit ainsi des logiciels, des retours d'usage et une réputation que son équipe interne n'aurait pas pu produire seule.

Le don transforme également l'image de l'ordinateur. Une machine offerte à un département universitaire n'est plus seulement un équipement de production. Elle devient un lieu d'apprentissage. Les étudiants peuvent observer les lampes du panneau, modifier un mot mémoire, écrire une routine et mesurer immédiatement l'effet de leur décision. L'architecture cesse d'être abstraite.

La pauvreté initiale du logiciel a un effet paradoxal. Elle impose du travail, mais elle rend chaque contribution visible. Un assembleur, un débogueur ou une routine d'affichage change immédiatement le quotidien de tous les utilisateurs. La gratification est directe : un programme écrit la nuit peut devenir l'outil du laboratoire le lendemain.

Pour **DEC**, cette production extérieure réduit une faiblesse structurelle. La jeune entreprise ne peut pas employer assez de programmeurs pour couvrir tous les usages. En ouvrant la machine à des communautés compétentes, elle obtient une bibliothèque plus diverse que celle qu'aurait conçue une équipe centralisée. Cette méthode ne garantit ni l'homogénéité ni le support, mais elle accélère l'exploration.

Le don révèle ainsi une stratégie qui dépassera les universités. **DEC** tend à considérer que la valeur se construit avec l'utilisateur. Cette conviction nourrit plus tard DECUS, les interfaces ouvertes et les marchés OEM. Elle distingue le constructeur, mais elle lui fera aussi sous-estimer parfois l'avantage d'un logiciel fortement standardisé et contrôlé.

6.3 Une politique qui dépasse le MIT

Le MIT constitue le cas le plus célèbre, mais le principe est plus large. Soutenir les universités revient à créer un réseau de compétences autour des produits. Lorsqu'un diplômé rejoint un laboratoire ou une entreprise, il connaît déjà les conventions, les outils et la manière de travailler de **DEC**. Le constructeur ne vend pas seulement une machine : il aide à former le marché qui saura l'utiliser.

Cette stratégie comporte un risque. En donnant beaucoup de liberté, **DEC** ne contrôle pas entièrement ce qui sera fait de son ordinateur. Les étudiants peuvent employer le PDP-1 à des activités éloignées des applications scientifiques mises en avant dans les brochures. Mais c'est précisément cette liberté qui révèle la puissance culturelle de la machine.

Dans les locaux du MIT, le PDP-1 rencontre un groupe d'étu-

diants habitués à démonter les systèmes, à comprendre les relais téléphoniques et à considérer l'élégance technique comme une valeur. Ils se nomment eux-mêmes *hackers*. Leur relation avec la machine donnera au don de 1961 une portée que personne ne pouvait inscrire dans un plan commercial.

Chapitre 7

La machine des hackers

Au MIT, le mot *hacker* ne désigne pas encore un intrus informatique. Il décrit un étudiant capable de comprendre un système complexe, d'en découvrir les possibilités cachées et de produire une solution aussi ingénieuse qu'inattendue. Le Tech Model Railroad Club, en particulier son comité *Signals and Power*, fournit un terrain idéal à cet état d'esprit. Les membres du club travaillent sur des relais, des circuits de commande et des équipements téléphoniques récupérés. Ils apprennent à lire un système par ses comportements, à suivre un signal et à modifier une architecture sans en demander la permission à chaque étape.

Le PDP-1 leur semble conçu pour cette manière de travailler. Contrairement à la grande machine du centre de calcul, il possède une console visible, un écran et des dispositifs que l'on peut connecter. Il répond immédiatement. L'utilisateur ne prépare pas seulement un programme : il observe une machine en action.

Le local du TMRC a préparé ces étudiants à une forme particulière d'attention. Les relais du réseau ferroviaire miniature produisent des comportements qu'il faut suivre d'un contact à l'autre. Un circuit bien conçu se reconnaît à son élégance et à l'économie de ses détours. Lorsqu'ils passent du sous-sol du club à la console du TX-0 puis du PDP-1, ils retrouvent la même joie de comprendre un système de l'intérieur.

Le mot *hack* porte alors une idée de performance inventive. Un bon hack résout un problème, mais il le fait avec une surprise,

une justesse ou une audace qui mérite d'être racontée. La valeur n'est pas mesurée par le nombre d'heures facturées. Elle vient de la compréhension acquise et de l'admiration des pairs. Cette morale informelle explique la vitesse avec laquelle les outils circulent : garder pour soi un programme utile priverait la communauté de la possibilité de l'améliorer.

Le PDP-1 convient à cette culture parce qu'il rend ses mécanismes accessibles. Les lampes permettent de voir une boucle qui ne se termine pas, l'écran révèle immédiatement une erreur de calcul et le ruban perforé peut passer d'une main à l'autre. La machine reste coûteuse, mais son usage possède quelque chose de personnel. Pendant le temps qui lui est accordé, le programmeur dispose réellement du processeur.

7.1 Un week-end pour fabriquer un outil

Alan Kotok, étudiant en génie électrique et membre du TMRC, devient rapidement l'une des figures de cette communauté. Le PDP-1 dispose de peu de logiciels. *Jack Dennis* envisage d'employer l'assembleur FRAP écrit chez BBN par *Ed Fredkin*. Kotok propose plutôt d'adapter et d'améliorer un outil connu des utilisateurs du TX-0.

Le défi prend la forme d'un pari. Six programmeurs se réunissent pendant un week-end et consacrent environ 250 heures de travail cumulées à l'écriture d'un assembleur capable de s'assembler lui-même. Ils alternent le codage, la vérification et l'occupation de la console. Le lundi, l'outil fonctionne. L'épisode, raconté notamment par Kotok et repris par *Steven Levy*, ne vaut pas seulement pour sa performance. Il montre comment une communauté partage un objectif sans attendre une organisation formelle. [1]

L'assembleur réduit la distance entre l'idée et l'exécution. Kotok développe également des outils de débogage associés à DDT, nom

1. Computer History Museum. *Alan Kotok* ; Steven Levy. *Hackers : Heroes of the Computer Revolution*. Garden City, New York : Anchor Press/Doubleday, 1984.

qui joue avec l'image du produit chimique destiné à éliminer les insectes. Le programmeur peut examiner la mémoire, placer des points d'arrêt et reprendre l'exécution. Le débogage devient une activité interactive plutôt qu'une comparaison tardive entre un listing et une impression.

Le week-end de l'assembleur montre également la différence entre une équipe prescrite et une communauté mobilisée. Aucun service de personnel n'a défini les rôles. Certains écrivent, d'autres relisent, d'autres encore attendent leur tour à la console. La pression vient du défi et de la promesse d'un outil commun. L'épisode ne doit pas être idéalisé : cette intensité repose sur un accès privilégié à une ressource rare et sur la disponibilité d'étudiants capables d'y consacrer leur temps. Mais il révèle une forme de production logicielle qui réapparaîtra dans de nombreuses communautés.

DDT modifie ensuite la relation à l'erreur. Au lieu de considérer un programme fautif comme un paquet à renvoyer au centre, le hacker l'arrête, examine un registre, change une valeur et reprend. Le programme devient un objet vivant dont on observe les états. Cette manière de déboguer s'imposera bien au-delà du PDP-1.

7.2 Spacewar !

À l'automne 1961, *Steve Russell*, *Martin Graetz* et *Wayne Wiitanen* imaginent un programme qui exploiterait réellement l'affichage du PDP-1. Un simple tracé décoratif ne suffirait pas : il faut une action continue, des règles compréhensibles et une démonstration que l'on ait envie de regarder. Le projet devient *Spacewar !*.

Deux vaisseaux s'affrontent autour d'une étoile dont la gravité modifie leur trajectoire. Les joueurs contrôlent la rotation, la poussée et le tir. Le programme est amélioré collectivement. *Peter Samson* ajoute un ciel étoilé fondé sur des données astronomiques. *Dan Edwards* affine la gravitation. Kotok et *Bob Saunders* construisent des boîtiers de commande afin que les joueurs ne maltraitent plus les commutateurs de la console.

La force historique de *Spacewar !* tient autant à sa circulation qu'à son statut de jeu précoce. Le programme voyage vers d'autres

installations PDP-1, est modifié et sert de démonstration. **DEC** comprend qu'un jeu peut révéler la vitesse, l'affichage et les entrées-sorties mieux qu'une longue présentation commerciale. La brochure publiée ensuite montre que la société accepte d'associer une machine scientifique à une activité ludique, choix encore inhabituel au début des années 1960. [2]

L'épisode donne également une leçon plus profonde. Les usages les plus importants d'une technologie ne sont pas toujours ceux que le constructeur avait prévus. En ouvrant l'accès et les interfaces, **DEC** laisse aux utilisateurs la possibilité d'inventer une culture autour du produit.

Le jeu n'est pas une parenthèse étrangère à la recherche. Il oblige à résoudre des problèmes de temps de réponse, de calcul de trajectoire, d'affichage et de commande simultanée. Il met la machine sous une charge continue et rend visibles les défauts d'interface. Les boîtiers de commande fabriqués par Kotok et Saunders apparaissent parce que le programme révèle une insuffisance matérielle : les commutateurs de la console ne sont ni confortables ni conçus pour être actionnés frénétiquement.

La démonstration possède aussi une dimension sociale. Autour de l'écran, des spectateurs comprennent immédiatement ce qui se passe. Ils n'ont pas besoin de lire un tableau de performances. Ils voient deux personnes agir, la machine répondre et un monde simulé conserver ses règles. Pour **DEC**, cette évidence vaut une campagne de publicité. *Spacewar !* montre le produit sans effacer l'utilisateur.

7.3 Partager sans perdre l'interaction

La liberté offerte par une machine personnelle à l'échelle d'un laboratoire se heurte rapidement à son propre succès. Plusieurs personnes veulent utiliser le PDP-1. Le temps partagé cherche à leur donner l'impression que chacune dispose de l'ordinateur, alors

2. Computer History Museum. *Spacewar !*

que le processeur passe rapidement d'un travail à l'autre. [3]

Chez BBN, *Ed Fredkin* et *Jack Dennis* développent un système qui utilise une mémoire à tambour pour échanger les programmes des utilisateurs. Une démonstration a lieu en septembre 1962 sur le premier PDP-1 livré à BBN. Des expériences proches sont menées au MIT et à Stanford. Le modèle du centre de calcul est ainsi renversé : il ne s'agit plus de grouper des tâches non interactives, mais de partager une machine tout en préservant le dialogue. [4]

Le temps partagé prolonge cette culture tout en la modifiant. Lorsque plusieurs personnes se connectent, l'ordinateur devient un espace commun dont les règles sont inscrites dans le système d'exploitation. Les fichiers doivent être protégés, les ressources réparties et les erreurs d'un utilisateur empêchées de détruire le travail des autres. Une question de performance devient une question de communauté.

Les machines 36 bits de **DEC** reprendront cette ambition à une autre échelle. Les étudiants du MIT qui ont appris à attendre quelques minutes pour leur tour au PDP-1 voudront bientôt rester connectés depuis un terminal. Le passage n'est pas une rupture de thème : il transforme l'interaction individuelle en conversation collective.

7.4 Ce que le MIT donne à DEC

Le don de 1961 revient vers Maynard sous plusieurs formes. Il fournit des logiciels, des démonstrations, des idées d'interface et un vivier de recrutement. Kotok rejoint **DEC** en 1962 et participera ensuite à l'architecture des machines 36 bits. D'autres étudiants, chercheurs et visiteurs emportent avec eux une familiarité avec les produits de l'entreprise.

Surtout, le MIT donne au PDP-1 une histoire que les chiffres de vente n'auraient jamais produite. La machine devient liée à l'émer-

3. Le *temps partagé* découpe l'usage du processeur entre plusieurs sessions interactives. Lorsque l'un des utilisateurs attend une saisie ou une opération lente, le système exécute le travail d'un autre.

4. Computer History Museum. *Timesharing on the PDP-1*.

gence de la culture hacker, à *Spacewar !*, aux outils interactifs et au temps partagé. **DEC** acquiert une réputation de constructeur qui comprend les programmeurs. Cette réputation attirera longtemps des utilisateurs techniques exigeants.

Mais le PDP-1 reste coûteux et construit en petit nombre. Pour transformer une philosophie en marché, **DEC** doit réduire davantage la machine. L'étape suivante ne consistera pas à perfectionner indéfiniment les 18 bits. Elle passera par une architecture plus étroite, plus simple et beaucoup moins chère.

Chapitre 8

Douze bits pour changer d'échelle

Le PDP-1 prouve qu'un ordinateur interactif peut trouver des clients. Il ne résout pas encore la question du volume. À plus de 100 000 dollars, la machine reste réservée à des institutions importantes. Pour équiper des laboratoires plus modestes, des ateliers ou des fabricants d'instruments, **DEC** doit réduire le coût de manière radicale, même si cette économie impose des contraintes aux programmeurs.

La voie des 12 bits commence avec le PDP-5. *Gordon Bell* définit l'architecture générale avant de quitter temporairement l'entreprise pour l'université Carnegie Tech. *Edson de Castro* poursuit le développement. Le système remplace certaines connexions spécialisées par un bus d'entrée-sortie plus général, ce qui permet d'ajouter des périphériques sans redessiner la machine entière.

Le passage aux 12 bits ne constitue pas une simple réduction du PDP-1. Il oblige à repenser ce qui est indispensable. Les instructions doivent tenir dans un mot court, les adresses ne peuvent désigner qu'une partie de la mémoire et certaines opérations sont composées à partir de micro-instructions. Le programmeur paie cette économie par davantage de discipline. En échange, le client achète moins de transistors, moins de mémoire et une machine plus facile à intégrer.

Cette relation entre prix et logiciel est centrale dans l'histoire des petits ordinateurs. Une architecture riche peut simplifier un

programme, mais chaque fonction matérielle doit être câblée, testée et entretenue. À l'inverse, une architecture minimale reporte des opérations vers des sous-programmes. Le PDP-8 accepte ce compromis avec une franchise presque pédagogique : ses limites sont visibles, documentées et contournables.

8.1 Une architecture volontairement étroite

Douze bits suffisent pour représenter des mesures, commander des équipements et réaliser de nombreux calculs scientifiques, mais ils limitent la taille immédiate des nombres et des adresses. Le jeu d'instructions reste très réduit. La programmation demande des astuces, notamment pour accéder à une mémoire plus grande que l'adresse contenue dans une instruction.

Cette simplicité ne doit pas être confondue avec une conception primitive. Chaque bit supprimé réduit le nombre de circuits nécessaires dans les registres, les chemins de données et la mémoire. L'architecture est pensée en fonction du coût de la technologie disponible. Elle accepte de transférer une partie de la complexité vers le logiciel afin de rendre l'ensemble abordable.

Le PDP-5 prépare le terrain, mais le PDP-8 transforme réellement l'échelle. Présenté en 1965, il utilise les modules Flip Chip et une organisation plus compacte. Sa configuration de base, comprenant 4 096 mots de mémoire, un télétype et des logiciels, est annoncée à 18 000 dollars. Le chiffre reste élevé pour un particulier, mais il place l'ordinateur dans le budget d'un laboratoire, d'un service d'ingénierie ou d'un constructeur d'équipements. [1]

Le prix annoncé comprend plus qu'un processeur nu. Le télétype, la mémoire et un ensemble de logiciels permettent de commencer réellement à travailler. Cette présentation est essentielle : un laboratoire ne cherche pas à collectionner des cartes logiques, mais à lire une mesure, modifier un programme et obtenir un résultat. En proposant une configuration cohérente, **DEC** transforme une

1. Digital Equipment Corporation, Information Research Services, cf. note 3 ; Computer History Museum. *The PDP-8*.

architecture économique en outil utilisable.

Le panneau du PDP-8 garde pourtant le lien avec l'établi. Les commutateurs permettent de charger un petit programme de démarrage, souvent appelé *bootstrap*, qui lira ensuite un ruban ou un autre support. Quelques mots saisis à la main ouvrent ainsi l'accès à un environnement beaucoup plus vaste. Cette séquence donne à plusieurs générations d'étudiants une compréhension concrète de la frontière entre matériel et logiciel.

8.2 Une machine que l'on ne voit pas toujours

Le PDP-8 est souvent intégré à un système plus vaste. Il analyse les signaux d'un instrument, pilote une expérience, contrôle une machine industrielle ou traite les données d'un appareil médical. L'utilisateur final peut ne jamais voir le panneau avant. Pour **DEC**, cette invisibilité constitue une réussite commerciale : l'ordinateur devient un composant d'un produit fabriqué par une autre entreprise.

Ce marché des OEM est essentiel. [2] Il multiplie les applications sans obliger **DEC** à connaître chaque domaine scientifique ou industriel. Le fabricant de l'instrument apporte son expertise métier ; le PDP-8 fournit une plate-forme programmable, documentée et suffisamment fiable pour être reproduite.

La famille évolue pendant de nombreuses années : PDP-8/S plus lent mais économique, PDP-8/I à circuits intégrés, PDP-8/L compact, puis PDP-8/E et ses variantes autour de l'OMNIBUS. L'architecture survit à plusieurs générations de technologie. Le client conserve ses programmes et ses connaissances tandis que le matériel devient plus petit et moins coûteux.

L'intégration change aussi la nature de la vente. Un constructeur d'analyseurs médicaux ou de machines-outils ne souhaite pas expliquer à son client qu'un ordinateur se trouve à l'intérieur. Il

2. Un *OEM*, ou fabricant d'équipement d'origine, intègre le matériel d'un autre constructeur dans son propre produit, qu'il vend ensuite sous forme de système complet.

veut garantir une fonction. Le PDP-8 devient alors une intelligence discrète, enfermée dans un équipement qui porte un autre nom. La marque **DEC** peut disparaître de la façade tout en gagnant un marché.

Les variantes répondent à des compromis précis. Le PDP-8/S réduit le prix en exécutant les opérations de manière série, donc plus lentement. Le PDP-8/I adopte les circuits intégrés. Le PDP-8/E rassemble processeur, mémoire et périphériques autour de l'OMNIBUS, qui simplifie l'extension du système. La permanence de l'architecture protège les programmes tandis que la technologie change. Le client n'achète plus un modèle isolé : il entre dans une lignée.

Cette continuité favorise l'enseignement. Les limites de la machine rendent l'architecture lisible, les périphériques montrent comment un ordinateur communique et les petits systèmes d'exploitation restent assez compacts pour être explorés. Le PDP-8 devient à la fois un produit industriel et une école de l'informatique.

8.3 Le mini-ordinateur devient un marché

Le terme « mini-ordinateur » s'impose pour désigner cette catégorie de machines moins coûteuses que les grands systèmes, souvent installées au niveau d'un département ou intégrées à un équipement. Le PDP-8 n'est pas le seul petit ordinateur de son époque, mais il devient le modèle emblématique du marché. Plus de dix mille exemplaires et de nombreuses variantes sont produits sur une période d'environ vingt-cinq ans selon le Computer History Museum.

Le succès attire des concurrents. Certains sont fondés par d'anciens ingénieurs de **DEC**. *Edson de Castro* quittera ainsi l'entreprise et participera à la création de **Data General**, dont le Nova mettra directement en cause la capacité de **DEC** à renouveler son offre. Cette concurrence pousse l'entreprise à chercher une architecture plus régulière et plus générale.

Avant d'arriver au PDP-11, le succès du PDP-8 révèle une autre

vérité : le processeur seul ne suffit plus. Les clients demandent des bandes, des disques, des terminaux, des interfaces analogiques et des logiciels adaptés à leur travail. Pour comprendre comment **DEC** passe du mini-ordinateur à l'entreprise de systèmes, il faut suivre les câbles qui relient la machine au monde réel.

Troisième partie

Du laboratoire au marché

Chapitre 9

Brancher le monde réel

Un ordinateur n'entre pas dans un laboratoire uniquement par son processeur. Il doit recevoir des mesures, commander des appareils, conserver des programmes et restituer des résultats. Dans l'histoire de **DEC**, les périphériques ne sont donc pas des accessoires ajoutés après la conception. Ils déterminent les lieux où la machine peut travailler et les personnes qui peuvent l'utiliser.

Le PDP-1 avait déjà montré l'importance d'une structure d'entrée-sortie ouverte. Le PDP-8 généralise cette relation au monde extérieur. Un fabricant d'instruments peut relier des convertisseurs analogiques, des compteurs, des relais ou un dispositif d'affichage. Le manuel d'interface devient presque aussi important que le manuel de programmation : il explique comment faire entrer un phénomène physique dans la logique de la machine.

L'ouverture matérielle n'est utile que si elle est documentée. Les manuels d'interface de **DEC** décrivent les niveaux de tension, les temporisations, les signaux de commande et les procédures d'interruption. Pour un ingénieur en instrumentation, ces pages comptent davantage qu'une présentation générale du processeur. Elles lui permettent de savoir si son appareil pourra imposer son rythme à la machine et comment éviter de perdre une mesure.

Cette documentation encourage une industrie périphérique. Des entreprises spécialisées proposent des convertisseurs, des multiplexeurs, des contrôleurs et des châssis. Elles enrichissent les systèmes sans que **DEC** doive devenir experte de chaque discipline.

Le bus ou l'interface agit alors comme un contrat technique : chacun peut innover de son côté à condition de respecter les signaux communs.

9.1 Le télétype, porte d'entrée ordinaire

Le télétype ASR-33 accompagne de nombreuses petites installations. Il associe un clavier, une imprimante et souvent un lecteur-perforateur de bande papier. Sa vitesse paraît dérisoire selon les critères modernes, mais il réunit plusieurs fonctions dans un appareil standard et relativement abordable. L'utilisateur tape une commande, voit la réponse s'imprimer et peut conserver un programme sous la forme d'un ruban perforé.

Le bruit mécanique fait partie de l'expérience. Les caractères apparaissent ligne après ligne, les rubans s'accumulent, une faute peut obliger à reperforer une portion du programme. Pourtant, cette interface rapproche déjà l'ordinateur du travail quotidien. On n'attend plus nécessairement le passage d'un lot : la machine répond depuis le même local.

Les terminaux vidéo remplaceront progressivement le papier pour l'interaction courante, mais le principe reste identique : un dispositif relativement simple permet à plusieurs familles d'ordinateurs d'être utilisées sans ouvrir l'armoire du processeur. Le périphérique devient un point de stabilité au milieu de générations matérielles changeantes.

Le télétype façonne aussi le rythme du travail. L'imprimante mécanique oblige à choisir ce que le programme affichera ; une sortie trop bavarde immobilise la session. Les programmeurs développent des abréviations, des éditeurs adaptés à la ligne et des commandes concises. Une interface lente n'empêche pas l'interaction, mais elle lui donne une grammaire.

Le ruban perforé conserve une valeur pratique. Il ne nécessite pas de dispositif complexe, se copie avec un lecteur-perforateur et permet d'emporter un programme dans une enveloppe. Il est également fragile : une déchirure, une inversion ou une boucle mal

préparée suffit à interrompre le chargement. Le soin apporté aux bandes fait partie du métier quotidien autant que le code lui-même.

9.2 DECtape, une bande qui se comporte presque comme un disque

La bande magnétique traditionnelle est efficace pour lire ou écrire de longues séquences, mais elle oblige souvent à faire défiler une grande partie du support pour retrouver une information. **DEC** développe avec DECtape un petit ruban organisé en blocs adressables. Le système peut localiser un bloc, lire dans les deux directions et résister à certaines erreurs grâce à une organisation redondante des pistes.

DECtape apparaît commercialement au milieu des années 1960 et accompagne notamment le PDP-7, le PDP-8 et d'autres familles. Les bobines sont assez compactes pour être transportées par un utilisateur. Un ruban peut contenir un système, des programmes et des données. Il joue un rôle intermédiaire entre la bande perforée et le disque, à une époque où ce dernier reste coûteux. [1]

Cette souplesse favorise l'interactivité. Un programme peut être chargé sans manipuler une pile de cartes, un fichier peut être remplacé sans réécrire tout le support et une petite installation dispose d'un stockage réutilisable. Les mouvements des lecteurs TU55 ou TU56, visibles derrière leur façade, deviennent l'une des images familières des systèmes **DEC**.

La particularité de DECtape se voit autant qu'elle se décrit. Les deux bobines accélèrent, s'arrêtent et repartent dans l'autre sens pendant que le système recherche un bloc. L'utilisateur peut associer ce mouvement à l'activité de la machine. Le support est assez robuste pour servir de disque système sur de petites configurations et assez transportable pour circuler entre laboratoires.

Cette présence physique nourrit la mémoire des utilisateurs. Une session informatique est faite de sons, de délais et de gestes : monter

1. Digital Equipment Corporation, Information Research Services, cf. note 3.

un ruban, vérifier son sens, attendre le chargement. Ces contraintes ne sont pas de simples curiosités historiques. Elles déterminent la taille des programmes, l'organisation des fichiers et la manière dont un système reprend après une panne.

9.3 Du disque au système de stockage

À mesure que les logiciels grandissent, le disque devient indispensable. Les familles RK, RL, RP puis RA couvrent des besoins allant du petit disque amovible aux ensembles de grande capacité. Le disque ne sert pas seulement à conserver davantage de données. Son accès direct transforme la conception des systèmes d'exploitation : répertoires, fichiers interactifs, échanges entre utilisateurs et bases de données deviennent plus pratiques.

Les supports amovibles permettent aussi de séparer les environnements. Un laboratoire peut conserver ses propres plateaux, un service changer de jeu de données et un technicien démarrer un système de diagnostic. Mais cette flexibilité a un prix : têtes de lecture, alignement, poussière et contrôleurs ajoutent de nouveaux points de panne. Le service de terrain de **DEC** devient alors une partie essentielle du produit.

Dans les années 1980, les contrôleurs HSC et les VAXclusters feront du stockage un service partagé entre plusieurs ordinateurs. Cette évolution sera développée au chapitre 16. Elle commence pourtant avec une idée simple : le support de données doit être pensé avec la machine, non acheté comme une réflexion secondaire.

Le disque change l'échelle des logiciels. Un moniteur compact chargé depuis ruban peut se contenter de quelques commandes. Avec un accès direct plus rapide et une capacité croissante, le système peut maintenir des répertoires, conserver plusieurs versions et servir plusieurs utilisateurs. Le périphérique crée donc de nouvelles attentes qui reviennent vers le processeur : davantage de mémoire, de protection et de débit.

Les contrôleurs deviennent des ordinateurs spécialisés. Ils gèrent le déplacement des têtes, la détection des erreurs et les transferts de blocs. Cette intelligence déportée soulage le processeur principal,

mais complexifie le diagnostic. Lorsque les données ne sont pas lues, la panne peut se trouver dans le support, l'électronique du lecteur, le câble, le contrôleur, le bus ou le logiciel. Le technicien de terrain doit comprendre toute la chaîne.

9.4 Voir, mesurer, agir

Les écrans graphiques donnent aux PDP une présence particulière. Le Type 30 du PDP-1 trace des points commandés directement par le programme. Les systèmes suivants ajoutent des possibilités de dessin, de conversion et d'interaction au crayon lumineux. Dans les laboratoires, l'affichage permet de suivre une expérience, de visualiser un signal ou de manipuler une représentation avant que le terme « interface graphique » ne devienne courant.

Les convertisseurs analogique-numérique accomplissent le trajet inverse : ils transforment une tension issue d'un capteur en valeur exploitable par le programme. Des convertisseurs numérique-analogique, des relais et des lignes de commande permettent ensuite d'agir sur l'expérience. Le mini-ordinateur devient ainsi le centre d'une boucle : mesurer, calculer, afficher, corriger.

Cette capacité explique la présence des PDP dans la médecine, la physique, la chimie, l'automatisation et le contrôle industriel. Elle explique aussi l'importance des fabricants tiers. Grâce aux bus et aux interfaces documentées, d'autres sociétés proposent des cartes et des périphériques spécialisés. L'écosystème augmente la valeur de la machine sans que **DEC** doive tout concevoir.

Dans une expérience scientifique, la valeur du système se juge souvent à sa capacité à ne pas manquer l'événement. Un convertisseur peut produire des mesures plus vite que le programme ne les traite. Il faut alors des interruptions, des mémoires tampons ou des transferts directs. Ces mécanismes, que nous retrouverons avec l'Unibus du PDP-11, montrent comment un besoin physique pousse l'architecture à évoluer.

Le périphérique rapproche aussi **DEC** de métiers éloignés de l'informatique. Un médecin parle de signaux biologiques, un chimiste de spectres, un industriel de cycles de machine. L'ingénieur

d'application doit traduire ces phénomènes en débits, résolutions et délais. L'entreprise grandit donc en apprenant le vocabulaire de ses clients.

Le succès des périphériques pose cependant une nouvelle question. Lorsqu'un ordinateur possède des disques, plusieurs terminaux et de nombreux utilisateurs, il ne peut plus être seulement un instrument individuel. Il devient un lieu partagé. Les machines 36 bits vont explorer cette autre forme de proximité : non plus un utilisateur devant un processeur, mais une communauté entière connectée au même système.

Chapitre 10

Partager la puissance

Le PDP-8 rend l'ordinateur plus accessible en réduisant son prix et sa taille. Le PDP-6 suit une direction presque opposée. Annoncé en 1964, il utilise des mots de 36 bits, vise le calcul scientifique avancé et doit servir plusieurs utilisateurs à la fois. Pour **DEC**, entrer sur ce marché signifie affronter des machines plus puissantes tout en conservant l'interactivité qui distingue la société.

Seulement vingt-six PDP-6 sont installés selon la chronologie interne de **DEC**. Le modèle ne devient pas un succès commercial comparable au PDP-8. Il fournit cependant l'architecture et l'expérience qui conduiront au PDP-10, l'une des machines les plus influentes des universités, des centres de recherche et des premiers réseaux.

La largeur de 36 bits n'est pas choisie pour impressionner. Elle convient aux calculs scientifiques en précision élevée et à la représentation de pointeurs ou de caractères groupés dans un mot. Elle rapproche également la machine des habitudes de certaines grandes installations universitaires. **DEC** veut montrer qu'elle peut fournir la puissance attendue sans renoncer à une console interactive et à une architecture compréhensible.

Le projet dépasse cependant les moyens habituels de la société. Les circuits, la mémoire et les logiciels coûtent cher, tandis que la clientèle reste plus étroite que celle du PDP-8. Le PDP-6 devient une expérience exigeante : techniquement féconde, commercialement fragile. Son importance vient précisément du fait que l'équipe

ne l'abandonne pas après les faibles volumes. Elle transforme l'architecture en une famille mieux positionnée.

10.1 Une grande machine selon DEC

Le PDP-6 n'est pas un mini-ordinateur au sens habituel. Sa mémoire, ses périphériques et son coût le placent dans une autre catégorie. Mais son usage diffère du grand système de traitement par lots. Il est conçu pour le temps partagé, avec des terminaux à partir desquels les utilisateurs peuvent éditer, compiler et exécuter leurs programmes.

Gordon Bell dirige l'architecture initiale. *Alan Kotok*, arrivé du MIT, joue ensuite un rôle majeur dans la lignée. Les concepteurs connaissent directement les attentes des communautés interactives : réponse rapide, fichiers personnels, outils de débogage et possibilité de faire cohabiter calcul, édition et communications.

Une démonstration de 1965 illustre cette ambition avec une forme d'humour involontaire. Depuis Boston, une équipe commande par télex un PDP-6 installé à Perth, en Australie. La liaison parcourt environ 12 000 miles et utilise un code à cinq trous peu adapté à certaines commandes. Au cours de la session, un participant australien demande aux Américains s'ils pourraient laisser « un peu de mémoire » à leurs collègues éloignés. L'épisode révèle à la fois les limites des télécommunications et l'idée nouvelle qu'un ordinateur peut être utilisé à distance comme une ressource partagée. [1]

Le temps partagé modifie la perception de la puissance. Un utilisateur ne demande pas que le processeur lui appartienne en permanence ; il demande que le système réponde assez vite pour maintenir son attention. Une machine peut donc servir des dizaines de personnes si le moniteur répartit correctement le temps, protège les fichiers et déplace les travaux longs vers les moments disponibles.

Cette organisation produit un nouvel espace de travail. Le terminal n'affiche pas seulement un résultat : il donne accès à un répertoire, un éditeur, un compilateur et parfois aux messages des

1. Digital Equipment Corporation. *36-bit Systems Timeline*. Digital Computing History Timeline. 1997.

autres utilisateurs. La communauté se forme à l'intérieur du système. Les noms de fichiers, les conventions de commandes et les programmes partagés deviennent une culture locale.

Les machines 36 bits accueillent ainsi des pratiques dont l'influence dépassera leurs ventes : intelligence artificielle, traitement symbolique, édition interactive, courrier électronique et premiers jeux multiutilisateurs. Elles sont suffisamment puissantes pour faire fonctionner ces environnements et suffisamment ouvertes pour que les universités les transforment.

10.2 Le PDP-10 et ses communautés

Le PDP-10, lancé en 1967, reprend l'architecture 36 bits avec une technologie plus rapide et un meilleur positionnement commercial. Il est compatible avec les programmes du PDP-6, ce qui protège le travail déjà réalisé. Les systèmes TOPS-10 puis TOPS-20 fournissent des environnements multiutilisateurs complets, avec fichiers, langages, outils et communications.

Dans les universités, la machine devient un espace social. Les utilisateurs ne travaillent pas seulement côte à côte : ils partagent des fichiers, envoient des messages, construisent des éditeurs et développent des langages. Les PDP-10 et leurs proches parents accueillent des travaux en intelligence artificielle, en programmation symbolique et en réseau. Le MACLISP du MIT, les premiers environnements d'Emacs et de nombreux jeux textuels appartiennent à cette culture 36 bits.

Le PDP-10 est également présent sur l'ARPANET. Sa capacité à servir de nombreux utilisateurs et la richesse de ses logiciels en font un hôte naturel du réseau naissant. Des communautés éloignées commencent à partager des ressources au-delà de leur propre établissement. L'informatique interactive, née devant une console locale, devient progressivement une activité en réseau.

TOPS-10 représente le monde hérité du PDP-6 et du PDP-10. TOPS-20, issu du système TENEX développé à l'extérieur de **DEC**, apporte une mémoire virtuelle et un environnement apprécié pour son confort interactif. L'adoption d'une technologie née chez un

autre acteur montre que la société sait parfois reconnaître la valeur produite par ses utilisateurs et ses partenaires. Elle montre aussi que plusieurs cultures logicielles coexistent dans la même famille.

Cette richesse fait du PDP-10 une machine attachante et difficile à remplacer. Un centre universitaire ne possède pas seulement un processeur ; il possède des comptes, des fichiers, des programmes locaux et une mémoire collective. Lorsque la gamme s'arrête, la perte ressentie ressemble à la fermeture d'un lieu. Les utilisateurs ne demandent pas uniquement une vitesse équivalente : ils veulent retrouver leur monde.

10.3 Une famille brillante mais isolée

Le succès intellectuel des machines 36 bits ne garantit pas leur avenir industriel. Leur architecture, leurs systèmes et leur clientèle forment un monde spécifique. Les coûts de développement sont élevés, tandis que la croissance la plus rapide de **DEC** se produit autour des familles plus petites. Le DECsystem-10 puis le DECSYSTEM-20 prolongent la gamme, mais celle-ci finit par entrer en concurrence pour les ressources avec VAX.

Lorsque **DEC** décide d'abandonner la lignée, la réaction des utilisateurs montre la profondeur de leur attachement. Les logiciels et les habitudes ne se déplacent pas aussi facilement qu'un équipement. Une communauté de temps partagé constitue un patrimoine collectif : programmes, fichiers, pratiques et relations.

Cette leçon pèsera sur les choix futurs de l'entreprise. Il ne suffit plus de créer une meilleure machine. Il faut préserver l'investissement logiciel et offrir une trajectoire de migration. Le problème se pose bientôt à une échelle plus vaste avec le PDP-11, famille dont la réussite dépassera toutes les attentes et dont les limites deviendront, à leur tour, difficiles à abandonner.

Chapitre 11

Reconstruire la famille

À la fin des années 1960, **DEC** possède plusieurs familles qui répondent à des marchés différents, mais cette richesse devient difficile à soutenir. Les machines 12 bits sont économiques, les 18 bits servent les laboratoires et les 36 bits le temps partagé. Les logiciels, les périphériques et les équipes se dispersent entre des architectures incompatibles.

La concurrence accentue l'urgence. *Edson de Castro*, après avoir quitté **DEC**, participe à la fondation de **Data General**. Son Nova, annoncé en 1969, propose une machine 16 bits compacte et compétitive. **DEC** ne peut pas se contenter d'améliorer le PDP-8. Il lui faut une nouvelle architecture capable de couvrir davantage d'usages et de devenir une véritable famille.

Le projet naît dans une atmosphère de concurrence interne et externe. **DEC** a déjà tenté plusieurs architectures, tandis que le Nova de **Data General** montre qu'une jeune société peut attaquer le marché des mini-ordinateurs avec une machine simple et rapide. L'entreprise doit répondre sans multiplier encore une lignée isolée. Le PDP-11 doit être assez économique pour commencer bas, mais assez extensible pour monter vers des systèmes plus ambitieux.

La conception associe plusieurs ingénieurs, dont *Harold McFarland*, *Roger Cady* et *Gordon Bell*. Le résultat ne suit pas entièrement les idées d'une seule personne. Il naît de compromis entre coût, technologie et expérience des générations précédentes. Cette origine collective explique la régularité de l'architecture : elle doit fournir

un langage commun à plusieurs équipes et à plusieurs modèles.

11.1 Une régularité nouvelle

Le PDP-11 est annoncé en 1970. Son architecture utilise des mots de 16 bits et traite également les octets, choix important pour les caractères et les applications commerciales. Les registres généraux peuvent servir aux calculs comme aux adresses. La plupart des instructions acceptent les mêmes modes d'adressage pour leurs deux opérandes. Cette régularité réduit le nombre de cas particuliers que doit apprendre le programmeur.

Le principe paraît simple, mais il modifie profondément l'écriture du code. Sur des architectures plus anciennes, certains registres possèdent des rôles fixes et certaines opérations exigent des séquences particulières. Le PDP-11 permet de combiner plus librement registres, mémoire et calcul d'adresse. L'assembleur devient plus expressif, et les compilateurs disposent d'une cible plus régulière.

Gordon Bell et l'équipe décrivent la machine dans un article de 1970 qui insiste sur les choix d'architecture, le coût et les compromis. Le PDP-11 n'est pas présenté comme une perfection abstraite : il est le résultat de contraintes de marché, de technologie et de compatibilité avec les périphériques. [1]

Le traitement de l'octet répond à un déplacement du marché. Les premiers PDP excellent dans le calcul et le contrôle, mais les applications commerciales manipulent des caractères, des chaînes et des fichiers. Avec ses opérations sur huit bits, le PDP-11 peut servir aussi bien un instrument qu'un système de gestion. Il élargit la clientèle sans devenir deux machines différentes.

Les modes d'adressage donnent au programmeur un moyen compact de parcourir tableaux, piles et structures. Un registre peut pointer vers une donnée, être incrémenté automatiquement ou servir de pile. Cette souplesse facilite la construction des compilateurs et des systèmes d'exploitation. Elle contribue directement à l'attrait

1. C. Gordon Bell et al. « A New Architecture for Mini-Computers : The DEC PDP-11 ». In : *Proceedings of the Spring Joint Computer Conference*. 1970, p. 657-675. DOI : 10.1145/1476936.1477037.

du PDP-11 pour les chercheurs des Bell Laboratories.

11.2 L'Unibus

L'élément le plus visible de cette cohérence est l'Unibus. Processeur, mémoire et périphériques partagent un bus asynchrone bidirectionnel. [2] Les appareils apparaissent dans un espace d'adressage commun et peuvent échanger des données sans multiplier les interfaces propres à chaque modèle.

Pour les fabricants tiers, l'Unibus ouvre un marché. Une carte d'acquisition, un contrôleur de disque ou une interface de communication peut être conçue pour la famille entière. Pour le client, la machine devient extensible. Pour **DEC**, cette ouverture augmente les ventes de processeurs, même lorsque le périphérique n'est pas fabriqué à Maynard.

Le bus possède aussi une dimension organisationnelle. Il permet à plusieurs équipes de développer en parallèle des éléments qui se rejoindront dans un système. L'architecture matérielle reflète ainsi la culture modulaire de l'entreprise, depuis les premiers *Digital Building Blocks* jusqu'aux ordinateurs complets.

L'Unibus transforme le châssis en plate-forme. Un périphérique peut prendre le contrôle du bus pour transférer des données, interrompre le processeur et être adressé comme une zone de mémoire. Cette uniformité réduit le nombre de conventions particulières. Elle permet également à des fabricants tiers de proposer des cartes d'instrumentation, de communication ou de stockage. Plusieurs centaines de types de périphériques seront reliés à la famille, selon les analyses rétrospectives de ses concepteurs.

L'ouverture n'est pas gratuite. Un bus partagé possède une capacité limitée, et la croissance des performances finira par exiger d'autres interconnexions. Mais au début des années 1970, l'Unibus offre un équilibre remarquable entre simplicité et extension. Il fait de la périphérie un marché, non une liste fermée d'options.

2. Un *bus* est un ensemble de lignes de communication partagées par plusieurs composants. Il transporte les adresses, les données et les signaux de contrôle.

11.3 Une famille qui déborde son projet initial

Le PDP-11/20 est suivi de nombreux modèles : systèmes plus rapides, versions économiques, processeurs intégrés et machines destinées au contrôle, au calcul ou à l'entreprise. Les systèmes d'exploitation se multiplient : RT-11 pour le temps réel et les petites configurations, RSX-11 pour les applications industrielles et multiutilisateurs, RSTS/E pour le temps partagé, sans oublier UNIX et de nombreux logiciels tiers.

Cette diversité fait du PDP-11 l'une des familles les plus durables de l'histoire informatique. Elle permet à un client de commencer avec une petite configuration puis de changer de processeur tout en conservant une partie de ses périphériques et de ses programmes. Le modèle de la plate-forme, si puissant chez IBM, trouve ici une expression adaptée aux mini-ordinateurs.

La famille se décline à une vitesse qui donne parfois le vertige. Des machines de table côtoient des systèmes en armoires, les microprocesseurs LSI-11 prolongent l'architecture sur une carte, et les systèmes d'exploitation ciblent des usages différents. Cette diversité augmente la base installée, mais elle oblige **DEC** à garantir que les programmes et périphériques ne seront pas abandonnés à chaque nouveau modèle.

Le PDP-11 devient ainsi un contrat implicite. Le client accepte d'investir dans la plate-forme parce qu'il croit à sa durée. Ce contrat fera la fortune de **DEC** ; il rendra ensuite le passage à 32 bits beaucoup plus délicat.

Le succès révèle toutefois une limite inscrite dans le dessin d'origine. L'espace d'adressage de 16 bits devient trop étroit pour les logiciels et les données en croissance. Des mécanismes d'extension repoussent le problème sans le supprimer. Cette contrainte conduira directement au VAX.

Avant cette transition, le PDP-11 connaît un destin que **DEC** n'avait pas organisé. Dans les laboratoires de la compagnie téléphonique américaine, il devient le support d'un système d'exploitation et d'un langage appelés à dépasser largement la machine qui les a

vus mûrir.

Chapitre 12

UNIX, C et l'autre destin du PDP-11

L'une des contributions les plus durables du PDP-11 à l'histoire de l'informatique n'est pas conçue à Maynard. Elle prend forme dans les **Bell Laboratories**, au sein d'un groupe qui ne cherche pas d'abord à mettre au point un produit commercial. Il veut retrouver le plaisir et l'efficacité d'un environnement interactif après l'abandon d'un projet beaucoup plus vaste.

Cette histoire rappelle celle de **DEC** par un trait essentiel : une ambition nouvelle avance sous la forme d'un besoin plus modeste. Les fondateurs de **DEC** avaient commencé par des modules avant d'oser construire un ordinateur. *Ken Thompson* et *Dennis Ritchie* obtiennent, eux, une nouvelle machine en promettant un système de traitement de texte. Dans les deux cas, le détour pratique ouvre un chemin que personne n'avait entièrement planifié.

12.1 Après Multics

Les Bell Laboratories ont participé avec le MIT et **General Electric** au développement de Multics, système de temps partagé très ambitieux. Le projet veut servir de nombreux utilisateurs, offrir une protection élaborée et rendre les ressources informatiques disponibles comme un service. Sa complexité, ses délais et ses coûts

conduisent toutefois les Bell Labs à se retirer en 1969.

Ken Thompson ne renonce pas à l'idée d'un environnement interactif. Il souhaite un système plus petit, qu'une équipe réduite puisse comprendre et modifier. Un PDP-7 peu utilisé lui fournit le terrain nécessaire. Avec *Dennis Ritchie* et quelques collègues, il construit progressivement un système de fichiers, des processus, un interpréteur de commandes et des outils simples.

La faiblesse de la machine impose l'économie. Il n'est pas possible de reproduire Multics en miniature. Les fonctions doivent être séparées, les programmes rester courts et les fichiers servir d'interface commune. Cette contrainte contribue à une philosophie qui deviendra célèbre : accomplir une tâche précise avec un outil compréhensible, puis combiner les outils plutôt que bâtir pour chaque besoin un programme gigantesque.

Le nom UNIX, d'abord écrit *Unics*, joue avec celui de Multics. Derrière le trait d'esprit se trouve un changement d'échelle. Le système n'est pas défini par un grand plan institutionnel, mais par les usages quotidiens de ceux qui l'écrivent. Ils emploient UNIX pour développer UNIX. L'éditeur sert à modifier le compilateur, le compilateur produit les commandes, et le système de fichiers organise les sources comme les documents.

Le PDP-7 devient rapidement trop limité. Le groupe souhaite un PDP-11, machine alors récente dont l'architecture traite naturellement les octets et offre des registres plus réguliers. Pour justifier l'achat, l'équipe propose de fournir un environnement de préparation de textes au service des brevets. La demande est acceptée. Le processeur arrive en 1970, avant certains périphériques nécessaires, et les premiers outils doivent encore être préparés à partir de l'ancienne machine.

12.2 Le passage au PDP-11

Lorsque le stockage devient disponible, UNIX migre réellement vers le PDP-11. Le nouveau processeur n'est pas seulement plus rapide. Il correspond mieux au type de logiciel que l'équipe veut écrire. Les octets adressables facilitent les textes et les fichiers, tan-

dis que les modes d'adressage conviennent aux piles, aux tableaux et aux pointeurs. La machine reste assez petite pour qu'un groupe restreint en comprenne l'ensemble.

En 1971, le système assure un travail régulier de traitement documentaire. Cette utilisation apporte au projet une légitimité interne. UNIX n'est plus un exercice mené sur une machine délaissée : il rend un service et permet à ses auteurs d'obtenir du matériel plus puissant. La même année, la première édition du manuel des programmeurs commence à fixer les commandes et les conventions.

Le PDP-11 favorise une relation étroite entre matériel et logiciel. Lorsqu'une fonction manque, les chercheurs peuvent examiner les registres, les interruptions et les pilotes. Ils ne dépendent pas d'une couche opaque fournie par un constructeur lointain. La documentation de **DEC** et la régularité de l'architecture rendent cette exploration possible.

Cette proximité ne signifie pas qu'UNIX soit un produit de **DEC**. Les Bell Laboratories définissent leurs propres priorités et n'ont pas pour mission de promouvoir le PDP-11. Le constructeur fournit pourtant un environnement matériel particulièrement propice à cette expérimentation. En retour, UNIX donne à la machine une réputation académique et technique qui dépasse largement la publicité officielle.

Les petits programmes du système peuvent être reliés par des fichiers puis, bientôt, par des tubes de communication. Le résultat d'une commande devient l'entrée d'une autre. Cette composition fait du terminal un atelier logiciel : le programmeur construit une solution en assemblant des outils, comme les ingénieurs de Maynard avaient assemblé les premiers modules logiques.

Les travaux de Thompson et Ritchie montrent ainsi qu'une petite machine interactive peut accueillir un environnement complet, multiutilisateur et productif. Le PDP-11 n'est plus seulement un contrôleur ou un calculateur de laboratoire. Il devient le support d'une manière de programmer appelée à se diffuser dans les universités.

12.3 Un langage pour le système

Les premières versions d'UNIX sont largement écrites en assembleur. Cette situation donne un code efficace, mais attache le système à l'architecture. Thompson avait déjà conçu le langage B à partir de BCPL. Ritchie le fait évoluer pour mieux représenter les données et exploiter le PDP-11. Le nouveau langage reçoit le nom de C.

Dennis Ritchie n'essaie pas de masquer le matériel. C permet de manipuler des adresses, des bits et la disposition des données en mémoire. Il introduit cependant des types, des structures de contrôle et des fonctions qui donnent au programme une organisation plus claire que l'assembleur. Il se place dans une zone intermédiaire : assez proche de la machine pour écrire un noyau, assez abstrait pour qu'une grande partie du code survive à un changement de processeur.

Le PDP-11 influence ce développement. Ses octets, ses registres et ses modes d'adressage donnent une forme efficace à certaines expressions du langage. C ne constitue toutefois pas une transcription du jeu d'instructions. Son intérêt apparaît précisément lorsqu'il permet de décrire les structures du système sans répéter tous les détails du processeur.

En 1973, le noyau d'UNIX est réécrit en grande partie en C. Le système conserve nécessairement des portions dépendantes du matériel, notamment pour le démarrage et les entrées-sorties, mais la majorité du code peut être comprise et adaptée sans reprendre chaque instruction. Les portages réalisés ensuite démontrent qu'un système d'exploitation peut franchir la frontière d'une architecture avec un effort raisonnable. Cette évolution est documentée par Ritchie et par l'article que Thompson et lui publient en 1974. [1]

La portabilité change progressivement les rapports de force. Dans le modèle traditionnel, le système d'exploitation accompagne une

1. Dennis M. Ritchie. *The Evolution of the Unix Time-Sharing System*. 1984 ; Dennis M. Ritchie. *The Development of the C Language*. 1993 ; Dennis M. Ritchie et Ken Thompson. « The UNIX Time-Sharing System ». In : *Communications of the ACM* 17.7 (1974), p. 365-375. doi : 10.1145/361011.361061.

gamme de machines et disparaît avec elle. Avec UNIX et C, le logiciel peut devenir une continuité plus durable que le processeur. Les compétences du programmeur se déplacent avec le langage, les outils et les conventions.

Cette possibilité ne produit pas immédiatement un marché ouvert. Les portages restent complexes, les licences sont contrôlées et les matériels diffèrent profondément. Mais l'idée est installée : le code source peut porter une culture au-delà d'une machine particulière. Pour un constructeur intégré comme **DEC**, cette idée sera à la fois un avantage et une menace.

12.4 Expliquer et transmettre

Brian Kernighan occupe dans cette histoire une place distincte. Il ne crée pas le langage C et n'est pas l'auteur principal du noyau. Il participe cependant à l'environnement des Bell Laboratories, contribue à nommer et présenter plusieurs outils, et possède un talent particulier pour l'explication.

Avec Ritchie, il publie en 1978 *The C Programming Language*. Le livre est bref, dense et construit autour de programmes concrets. Il ne cherche pas à remplacer la pratique par une théorie exhaustive. Il donne au lecteur assez de règles et d'exemples pour commencer à écrire, puis l'amène vers les pointeurs, les structures et les entrées-sorties.

Cette clarté joue un rôle considérable. Un langage ne se diffuse pas seulement parce que son compilateur existe. Il faut des mots communs, un style et une manière de transmettre les bons usages. L'ouvrage de Kernighan et Ritchie fournit cette forme. Des générations de programmeurs apprennent C à travers des exemples dont l'arrière-plan matériel remonte au PDP-11, même lorsqu'ils travaillent sur une autre architecture.

La combinaison d'UNIX et de C donne également aux universités un objet pédagogique exceptionnel. Le système est assez compact pour être étudié, son code source circule sous licence et le langage permet de relier les concepts de haut niveau aux mécanismes de la machine. Les étudiants peuvent lire un noyau réel au lieu d'une

maquette construite uniquement pour l'enseignement.

Pour **DEC**, cette culture extérieure ajoute de la valeur au PDP-11. Les universités achètent ou conservent la machine parce qu'elle exécute UNIX, des chercheurs développent des logiciels qui circulent entre établissements, et les diplômés emportent leur expérience vers l'industrie. Le constructeur bénéficie donc d'un écosystème qu'il ne dirige pas.

Le paradoxe devient visible à long terme. UNIX et C donnent au PDP-11 ses lettres de noblesse logicielles, mais ils apprennent aussi aux utilisateurs à séparer le système de la machine. Lorsque les stations de travail et les microprocesseurs se multiplieront, le même environnement pourra passer d'un constructeur à l'autre. L'une des plus belles réussites associées à une machine **DEC** prépare ainsi un monde dans lequel le matériel propriétaire ne suffit plus à retenir le client.

Avant que cette conséquence n'apparaisse, le PDP-11 porte **DEC** vers une autre dimension. Les systèmes d'exploitation, les langages, les périphériques, la documentation et le service deviennent aussi importants que le processeur. La société cesse d'être seulement un fabricant de machines ; elle devient un fournisseur de mondes informatiques.

Chapitre 13

De la machine au système

Une architecture réussie attire des utilisateurs. Une entreprise durable doit ensuite les accompagner pendant des années. À mesure que le PDP-8, le PDP-10 et le PDP-11 s'installent dans les laboratoires et les entreprises, **DEC** découvre que le produit réel dépasse largement le processeur. Il comprend les périphériques, les systèmes d'exploitation, la documentation, la formation, les pièces de rechange et les personnes capables d'intervenir lorsqu'une machine s'arrête.

Cette extension transforme l'organisation. La société de modules, qui pouvait discuter directement avec chacun de ses premiers clients, devient un constructeur mondial. Le défi consiste à conserver la proximité technique tout en créant des procédures assez solides pour soutenir des milliers d'installations.

La diversité des systèmes ne relève pas d'une incapacité à choisir. Elle répond à des contraintes réelles. Un laboratoire temps réel veut une réponse déterministe et une petite empreinte mémoire. Une université privilégie le partage entre étudiants. Une entreprise demande fichiers, langages commerciaux et procédures d'exploitation. Au début, aucun système unique ne peut satisfaire ces priorités sans devenir trop lourd pour les petites configurations.

RT-11, RSX-11, RSTS/E et les autres environnements incarnent donc des compromis. Ils partagent une architecture matérielle, mais proposent des expériences différentes. Cette pluralité attire des marchés variés et donne aux équipes internes une grande autonomie.

Elle crée aussi une dette : documentation, corrections et évolutions doivent être assurées pour plusieurs mondes en parallèle.

13.1 Une pluralité de logiciels

DEC ne cherche pas immédiatement à imposer un système d'exploitation unique. Les petites configurations du PDP-8 utilisent des moniteurs, puis OS/8. Le PDP-11 accueille RT-11, RSX-11, RSTS/E et d'autres environnements. Les machines 36 bits disposent de TOPS-10 et TOPS-20. Cette pluralité répond à des besoins différents : contrôle en temps réel, enseignement, calcul interactif ou traitement d'entreprise.

Elle donne aux clients une grande liberté, mais augmente le coût de maintenance. Chaque système possède ses outils, ses pilotes et ses habitudes. Les langages doivent être adaptés à plusieurs architectures. Les corrections circulent entre équipes. Lorsque le matériel change, le logiciel devient un capital qu'il faut protéger.

Les manuels jouent ici un rôle stratégique. Les *handbooks* décrivent les registres, les interfaces et les exemples de programmation. Les guides de maintenance permettent de suivre un signal jusqu'à la carte défaillante. La documentation n'est pas seulement un accompagnement commercial : elle autorise l'utilisateur à comprendre et parfois à modifier son système.

Le logiciel devient progressivement une condition de la vente. Un processeur plus rapide ne suffit pas si le client doit récrire son application ou ne trouve pas le langage dont il a besoin. **DEC** investit dans les compilateurs, les éditeurs, les bibliothèques et les diagnostics. Les catalogues s'épaississent. La société ne vend plus seulement une architecture : elle promet un environnement de travail.

Cette promesse rapproche les équipes de logiciel des ingénieurs de terrain. Une fonction qui paraît secondaire au concepteur peut être décisive pour une installation. Les retours des clients alimentent les versions suivantes, mais ils rendent également le système plus complexe. Chaque demande satisfaite devient une compatibilité à préserver.

13.2 La force de DECUS

Le **Digital Equipment Computer Users Society**, créé au début des années 1960, organise l'échange de programmes et d'expérience entre utilisateurs. Des bibliothèques de logiciels circulent sur bandes et rubans. Les réunions rassemblent chercheurs, programmeurs, responsables de sites et ingénieurs de **DEC**. Un participant peut arriver avec une question précise et repartir avec un utilitaire écrit dans une autre université.

Un badge resté célèbre invite à « voler à ses amis ». La formule ne célèbre pas le pillage, mais la circulation des bonnes solutions dans un monde où chaque site ne peut pas tout développer. DECUS augmente la valeur des machines sans que le constructeur écrive lui-même tous les programmes. Il offre également à **DEC** un observatoire privilégié des besoins réels. [1]

Cette proximité peut être exigeante. Les utilisateurs techniques comparent les performances, signalent les défauts et demandent des améliorations. Mais elle crée une fidélité rare. Acheter une machine **DEC**, c'est entrer dans une communauté de pratiques, pas seulement signer un contrat de matériel.

Les réunions DECUS donnent à cette relation un visage. Les utilisateurs présentent leurs programmes, échangent des rubans ou des listings et critiquent directement les choix du constructeur. Les discussions ne sont pas toujours confortables pour **DEC**, mais elles fournissent une information qu'une enquête commerciale classique capterait mal. Les participants connaissent les détails du système et décrivent les conséquences concrètes d'une décision.

La bibliothèque de programmes réduit la duplication. Un laboratoire peut adapter une routine écrite ailleurs au lieu de recommencer. Cette circulation rappelle le MIT, mais elle est désormais organisée à l'échelle d'une clientèle internationale. DECUS devient une institution relativement autonome, suffisamment proche du constructeur pour influencer ses produits et suffisamment indépendante pour défendre les utilisateurs.

---

1. Computer History Museum. *"Steal from Your Friends" DECUS Button*. 1965.

13.3 Le service comme promesse

Une installation industrielle ne peut attendre plusieurs semaines une pièce ou un technicien. **DEC** développe donc un réseau de service de terrain. Les ingénieurs disposent de diagnostics, de pièces de rechange et d'une documentation détaillée. Ils doivent comprendre le système dans son contexte : un arrêt peut interrompre une expérience, une production ou un service administratif.

La relation avec le client se prolonge sur toute la durée de vie de la machine. Les mises à jour de logiciel, l'ajout d'un disque ou le remplacement d'un processeur deviennent des occasions de vente, mais aussi des responsabilités. La qualité du service renforce la réputation de **DEC** et justifie le choix d'une plate-forme cohérente.

Les fabricants OEM complètent cet ensemble. Ils intègrent les processeurs et les bus à des produits spécialisés, tandis que des sociétés tierces proposent mémoires, contrôleurs et interfaces. L'ouverture crée un marché autour de **DEC**, parfois au prix d'une concurrence avec ses propres périphériques.

Le service de terrain est le lieu où l'intégration devient réelle. Le technicien ne peut pas déclarer que le processeur fonctionne si le client ne parvient pas à lire son disque ou à démarrer son application. Il doit diagnostiquer l'ensemble, conserver des pièces et parfois improviser une solution avant l'arrivée d'une correction officielle.

Cette présence coûte cher, mais elle construit la confiance. Les systèmes contrôlent des expériences, des lignes industrielles ou des dossiers administratifs ; leur arrêt a des conséquences immédiates. Le contrat de maintenance transforme donc la vente initiale en relation durable. Il donne aussi à **DEC** un accès constant aux problèmes de ses clients.

L'entreprise développe des diagnostics qui testent les cartes, les mémoires et les périphériques. Ces programmes, souvent moins visibles que les applications, participent à la qualité de la plate-forme. Ils prolongent l'obsession de Ken Olsen pour les systèmes observables et réparables.

13.4 Changer d'échelle sans perdre son identité

À la fin des années 1970, **DEC** n'est plus une petite société de Maynard. Elle possède des usines, des filiales, des centres de formation et une présence internationale. Des rédacteurs, formateurs, techniciens et commerciaux rejoignent les architectes et les programmeurs. **Digital Press** publie des ouvrages qui dépassent la documentation immédiate des produits, notamment *Computer Engineering*, où *Gordon Bell*, *Craig Mudge* et *John McNamara* analysent les machines comme les étapes d'une évolution.

Cette croissance renforce la capacité de l'entreprise à fournir des systèmes complets. Elle rend aussi chaque rupture plus coûteuse. Un nouveau processeur doit préserver les logiciels, les périphériques et les compétences des clients. Le PDP-11 a construit une base immense, mais son espace d'adressage devient insuffisant. Abandonner cette base serait dangereux ; la prolonger sans limite paraît impossible.

La solution prendra le nom de VAX. Elle devra accomplir un exercice que **DEC** n'avait encore jamais mené à cette échelle : changer profondément l'architecture tout en donnant au client le sentiment d'une continuité.

Quatrième partie

L'empire VAX

Chapitre 14

Évoluer sans rompre

Au milieu des années 1970, le PDP-11 est à la fois la plus grande réussite de **DEC** et l'un de ses principaux problèmes. Les utilisateurs ont investi dans des programmes, des périphériques et des équipes formées. Pourtant, l'espace d'adressage de 16 bits limite les applications les plus ambitieuses. Les extensions ajoutées aux modèles supérieurs compliquent la programmation sans supprimer la contrainte fondamentale.

Une rupture complète ferait perdre l'avantage de la base installée. Une simple amélioration ne suffirait pas. Le comité réuni en 1975 doit donc concevoir une architecture nouvelle qui paraisse familière aux utilisateurs du PDP-11. **DEC** emploiera l'expression « compatibilité culturelle » : mêmes habitudes générales, mêmes types de données, outils proches et chemin de migration crédible, même lorsque le matériel change profondément.

Les extensions du PDP-11 montrent les limites d'une croissance par ajouts. Des espaces séparés pour les instructions et les données, des mécanismes de gestion mémoire et des modèles supérieurs permettent de dépasser certaines barrières, mais le programmeur doit connaître des règles de plus en plus complexes. L'élégance initiale risque de disparaître sous les solutions de transition.

Le comité VAX examine donc le problème comme une continuité culturelle plutôt que comme une compatibilité instruction par instruction. Le client doit reconnaître les tailles de données, les conventions, les outils et la manière d'administrer le système. Cette

notion est moins rigide qu'une reproduction exacte du PDP-11, mais plus exigeante qu'une simple passerelle de fichiers. Elle oblige les équipes à penser la migration dès la conception.

14.1 Étendre l'adresse

William Strecker joue un rôle majeur dans la définition de la nouvelle architecture. Le nom VAX signifie *Virtual Address eXtension*. Le passage à des adresses de 32 bits ouvre un espace théorique de quatre milliards d'octets, sans exiger que toute cette mémoire soit physiquement installée. Le système d'exploitation peut donner à chaque processus un espace d'adresses cohérent et déplacer les pages entre mémoire et disque selon les besoins. [1]

L'architecture augmente également le nombre de registres et généralise les opérations sur plusieurs tailles de données. Son jeu d'instructions est abondant. Cette complexité reflète la conviction de l'époque qu'une machine peut prendre en charge des opérations de haut niveau et simplifier le travail des compilateurs.

Le projet porte le nom de code *Star*. Il ne consiste pas à construire seulement un processeur. Matériel et système d'exploitation sont développés ensemble. Cette coordination représente un changement d'échelle pour **DEC** : le logiciel n'attendra plus la fin de la conception matérielle pour s'adapter.

La mémoire virtuelle ne sert pas seulement à exécuter de gros programmes. Elle isole les utilisateurs, donne au système un moyen de déplacer les pages et simplifie la représentation d'espaces de travail distincts. Pour un environnement multiutilisateur, cette protection est fondamentale. Une erreur ne doit pas permettre à un programme d'écraser le noyau ou les données d'un voisin.

Le jeu d'instructions reflète également les ambitions de l'époque. Certaines opérations complexes peuvent manipuler des chaînes ou faciliter les appels de procédures. Les concepteurs espèrent réduire

1. La *mémoire virtuelle* sépare les adresses utilisées par un programme de leur emplacement physique. Elle facilite l'isolation des processus et permet d'exécuter des programmes dont l'espace logique dépasse la mémoire réellement présente.

l'écart entre les langages de haut niveau et le matériel. Cette richesse sera plus tard critiquée par les partisans des architectures RISC, mais elle correspond au contexte technologique et logiciel du milieu des années 1970.

14.2 Le VAX-11/780

Le premier modèle, le VAX-11/780, est présenté en 1977. Son nom conserve explicitement la filiation avec le PDP-11. La machine peut exécuter certains programmes PDP-11 dans un mode de compatibilité, mais l'objectif principal est de faciliter une migration progressive vers le nouveau monde 32 bits.

Le 11/780 occupe une position inhabituelle. Il offre des capacités comparées à celles de systèmes beaucoup plus coûteux, tout en restant dans la culture du mini-ordinateur : interaction, périphériques variés, logiciels scientifiques et possibilité d'installer le système au niveau d'un département. Le Computer History Museum souligne que le succès de la famille VAX transforme **DEC** en deuxième constructeur informatique mondial. [2]

La machine devient également une unité de comparaison informelle. Le terme VUP, pour *VAX Unit of Performance*, mesure les systèmes par rapport au 11/780. Cette habitude montre combien le modèle s'impose comme référence, même si elle simplifie des performances qui varient selon les applications.

Le lancement du 11/780 met en scène la continuité. Le nom, la documentation et le mode de compatibilité rassurent les utilisateurs du PDP-11, tandis que les démonstrations soulignent l'espace mémoire et la puissance. **DEC** ne demande pas aux clients d'oublier leur histoire ; elle leur propose de la prolonger.

Le succès transforme le centre de gravité de l'entreprise. Les petites machines restent importantes, mais VAX attire les applications dont dépendent des services entiers. Les contrats grossissent, les cycles de décision s'allongent et la comparaison avec IBM de-

2. William D. Strecker. « VAX-11/780 : A Virtual Address Extension to the DEC PDP-11 Family ». In : *AFIPS Conference Proceedings* 47 (1978), p. 967-980 ; Computer History Museum. *VAX-11/780*. 1978.

vient plus directe. **DEC** n'est plus seulement le constructeur qui contourne le grand système : il commence à occuper une partie de son territoire.

14.3 Une famille plutôt qu'un sommet

Le véritable pari n'est pas de vendre un seul grand modèle. **DEC** veut décliner VAX du petit système au calculateur puissant. Le VAX-11/750 puis le 11/730 réduisent le coût. Les MicroVAX installent l'architecture dans des boîtiers plus compacts. Les VAX 8000 et 6000 montent vers des configurations proches du grand système. Les VAXstations l'emmènent sur le bureau des ingénieurs.

Cette largeur de gamme répond à la stratégie de plate-forme qui avait fait la force du System/360, mais avec une différence importante : **DEC** cherche à relier des machines distribuées plutôt qu'à tout concentrer dans un centre unique. Un client peut placer des VAX dans plusieurs services, les connecter et conserver un environnement logiciel commun.

L'architecture apporte ainsi une continuité au matériel. Pour qu'elle devienne un empire, il faut encore un système d'exploitation capable de protéger les programmes, les données et les habitudes. Cette tâche revient à une équipe dirigée par un ingénieur réputé pour son exigence et son indépendance : *Dave Cutler*.

Chapitre 15

Le logiciel comme héritage

Dave Cutler rejoint **DEC** au début des années 1970 et travaille sur les systèmes temps réel du PDP-11. Il dirige notamment le développement de RSX-11M, environnement compact et rigoureux destiné à des applications qui ne peuvent accepter une réponse imprévisible. Cette expérience le prépare à une tâche beaucoup plus vaste : construire le système d'exploitation de VAX pendant que l'architecture matérielle elle-même prend forme.

Le nouveau système reçoit le nom de VMS, pour *Virtual Memory System*. Sa première version est livrée en 1978. VMS doit exploiter la mémoire virtuelle, servir plusieurs utilisateurs, protéger leurs travaux et offrir les langages attendus par les clients scientifiques comme commerciaux. Il doit surtout donner l'impression que VAX prolonge l'investissement réalisé autour du PDP-11 au lieu de l'annuler.

Cutler vient d'une culture où le système doit respecter des délais et où un défaut ne peut être masqué par une interface élégante. Son équipe travaille avec une forte identité, parfois en tension avec d'autres groupes de **DEC**. Cette indépendance favorise la cohérence de VMS. Elle nourrit aussi le style de management que Cutler emportera plus tard chez **Microsoft**.

Le développement simultané exige des simulateurs, des prototypes et des accords constants sur les interfaces. Le système doit être écrit avant que le matériel définitif existe en quantité. Les erreurs découvertes dans l'un peuvent obliger à modifier l'autre.

Cette interdépendance augmente le risque, mais elle évite de livrer un processeur sans environnement exploitable.

15.1 Concevoir avec la machine

Dans de nombreux projets, le logiciel arrive après le matériel et doit s'accommoder de décisions déjà prises. Pour VAX, les équipes échangent en permanence. Les besoins du système influencent les mécanismes de protection, les interruptions et la gestion de mémoire. Les possibilités du processeur, en retour, déterminent l'organisation du noyau et des compilateurs.

Cette coopération n'élimine pas les tensions. Les architectes veulent une machine élégante et complète ; les développeurs du système cherchent des mécanismes fiables, testables et livrables. Cutler est connu pour défendre avec vigueur les besoins de son équipe. Le projet réussit parce que le conflit reste orienté vers un même objectif : faire fonctionner un ensemble, pas seulement réussir une démonstration de processeur.

VMS fournit des processus isolés, un système de fichiers structuré, des mécanismes de sécurité, des files d'impression, des outils de développement et le langage de commandes DCL. Il accueille FORTRAN, BASIC, COBOL, Pascal, PL/I et d'autres langages. La plate-forme s'adresse ainsi aussi bien aux laboratoires qu'aux services de gestion.

DCL offre une interface de commandes structurée, avec des conventions stables pour lancer les programmes, manipuler les fichiers et administrer les travaux. Elle n'imite pas UNIX ; elle reflète une conception plus intégrée du système. Les noms, les privilèges et les procédures sont pensés pour des installations où plusieurs services partagent une machine administrée professionnellement.

Le système de fichiers et les mécanismes de versions répondent au travail quotidien. Conserver plusieurs générations d'un fichier réduit le risque d'une modification malheureuse. Les files d'impression, les comptes et les journaux transforment le processeur en service collectif. La puissance de VAX devient visible à travers une organisation logicielle plutôt qu'à travers le panneau avant.

15.2 Migrer sans tout recommencer

La compatibilité ne se réduit pas à l'exécution de quelques instructions anciennes. **DEC** fournit des outils, des conventions et des guides de migration. Les clients peuvent transférer progressivement leurs données et réécrire les composants qui exigent les nouvelles capacités. Les programmeurs retrouvent des noms, des concepts et une documentation familière.

Cette continuité possède une valeur économique considérable. Un programme scientifique ou industriel représente parfois plusieurs années de travail. Sa disparition coûterait davantage que le processeur. En protégeant le logiciel, **DEC** rend acceptable l'achat d'une architecture nouvelle.

Le succès produit toutefois un attachement croissant. Plus VMS devient complet, plus il devient difficile de le remplacer. Chaque fonction supplémentaire augmente la valeur de la plate-forme, mais aussi le poids de sa maintenance. Le système n'est plus un accompagnement du matériel : il constitue un patrimoine autonome.

La migration est aussi humaine. Les opérateurs et programmeurs formés au PDP-11 ne doivent pas se sentir étrangers. Les manuels, les cours et les outils de conversion accompagnent le matériel. Le coût de cette transition montre que la compatibilité est un produit en soi : elle se conçoit, se documente et se maintient.

VMS renforce ainsi le lien entre client et constructeur. Plus l'installation utilise ses mécanismes de sécurité, de fichiers et de réseau, plus un changement de plate-forme devient coûteux. Cette fidélité assure des revenus durables, mais elle peut se transformer en dépendance mutuelle. **DEC** doit continuer à faire évoluer le système sans rompre les applications anciennes.

15.3 La disponibilité comme argument

Les clients confient aux VAX des activités qui ne doivent pas s'interrompre. VMS développe donc des mécanismes de journalisation, de sauvegarde, de contrôle des accès et de reprise. La stabilité devient un argument commercial aussi important que la vitesse.

Cette orientation prépare les VAXclusters, où plusieurs machines pourront partager des ressources et continuer à servir les utilisateurs malgré certaines pannes. Le système d'exploitation doit alors coordonner des processeurs distincts, gérer les accès concurrents aux fichiers et présenter l'ensemble comme un environnement cohérent.

VMS devient plus tard OpenVMS, signe de l'effort d'ouverture et de l'évolution des standards. Mais son identité demeure liée à la conception intégrée de VAX. Lorsque *Dave Cutler* quitte **DEC** en 1988 pour rejoindre **Microsoft** et diriger le développement de Windows NT, il emporte une expérience des systèmes robustes et portables acquise à Maynard. L'influence de **DEC** se diffuse ainsi jusque dans un concurrent qui dominera le logiciel du PC.

La plate-forme VAX/VMS atteint sa pleine force lorsque les machines cessent d'être des îlots. Réseau, terminaux et stockage vont former un ensemble distribué que les clients perçoivent comme un seul système de travail.

Chapitre 16

Le réseau devient le système

Le mini-ordinateur avait rapproché le calcul d'un laboratoire ou d'un département. Le réseau permet désormais de rapprocher les machines entre elles. Pour **DEC**, cette évolution n'est pas un complément tardif. Dès le milieu des années 1970, l'entreprise conçoit une architecture de communication destinée à relier ses différentes familles et leurs systèmes d'exploitation.

DECnet est annoncé en 1975. Ses premières versions répondent surtout aux besoins des systèmes RSX, puis l'architecture s'étend à VMS, TOPS, RSTS et d'autres environnements. Un utilisateur peut copier un fichier, accéder à une ressource distante ou lancer un travail sur une autre machine. Le réseau ne sert plus seulement à connecter des terminaux à un processeur central : il relie des ordinateurs capables d'agir chacun de manière autonome.

La mise en réseau prolonge une idée ancienne de **DEC**. Le mini-ordinateur avait donné de l'autonomie à un laboratoire en lui évitant de dépendre entièrement du centre de calcul. Il serait contradictoire de reprendre cette autonomie au nom du partage. Le réseau doit donc relier des machines qui restent capables de travailler seules.

Cette conception diffère du simple branchement de terminaux sur un grand ordinateur. Chaque nœud possède son système, ses disques et ses utilisateurs. La communication ajoute des possibilités

sans supprimer l'identité locale. Une équipe peut conserver son VAX ou son PDP-11 tout en échangeant avec les autres services.

Le défi est autant logiciel que matériel. Les systèmes d'exploitation de **DEC** n'ont pas été créés au même moment ni pour les mêmes usages. Leur faire partager des fichiers et des noms oblige à définir des protocoles communs. DECnet devient ainsi une architecture transversale qui tente de réunir la diversité accumulée par l'entreprise.

16.1 Une entreprise distribuée

Cette vision correspond à la structure des clients de **DEC**. Les laboratoires, les usines et les services possèdent souvent leurs propres machines. Les remplacer par un centre unique ferait perdre la proximité acquise. DECnet propose une autre solution : conserver le calcul local tout en partageant données et services.

La phase III, annoncée en 1980, permet de construire des réseaux de plus de deux cents nœuds et fonctionne sur plusieurs systèmes d'exploitation. La complexité est importante, mais l'utilisateur voit progressivement apparaître un espace commun. Les noms de machines et de fichiers distants entrent dans les habitudes quotidiennes. [1]

La même année, **DEC**, **Intel** et **Xerox** coopèrent à la normalisation commerciale d'Ethernet. Le réseau local à paquets offre un moyen commun de connecter processeurs, serveurs de terminaux et périphériques. **DEC** doit désormais faire coexister ses protocoles avec une technologie destinée à devenir un standard multiconstructeur.

Pour l'administrateur, le réseau apporte de nouvelles responsabilités. Une panne n'est plus nécessairement située dans la machine devant laquelle se trouve l'utilisateur. Elle peut venir d'une ligne, d'un routeur, d'un nom mal configuré ou d'un service distant. **DEC** doit former des spécialistes capables de raisonner à l'échelle d'un ensemble distribué.

1. Digital Equipment Corporation. *Networks Timeline*. Digital Computing History Timeline. 1997.

Pour l'utilisateur, en revanche, le succès se mesure à la disparition de cette complexité. Copier un fichier vers une autre machine ou ouvrir une session distante doit devenir une opération ordinaire. Le réseau cesse d'être un projet expérimental lorsqu'il entre dans la syntaxe quotidienne des commandes.

16.2 Le terminal devient une norme

Le VT100, lancé en 1978, montre comment un périphérique peut dépasser la gamme qui l'a produit. Le terminal respecte les séquences de contrôle normalisées par l'ANSI et utilise un microprocesseur Intel 8080 pour gérer plusieurs jeux de caractères, l'affichage en gras et les fonctions de l'écran. Son succès pousse d'autres fabricants à reproduire son comportement. [2]

Pour l'utilisateur, le VT100 est le visage du système. Il affiche l'éditeur, le courrier, les commandes et les résultats, qu'ils proviennent d'un PDP-11, d'un VAX ou d'une machine distante. Le processeur peut changer derrière le réseau ; le clavier et l'écran maintiennent une continuité sensible.

Cette standardisation contient déjà une ambiguïté. Elle facilite la vente de systèmes **DEC**, mais rend aussi le terminal utilisable avec des ordinateurs concurrents. Une interface devenue norme échappe en partie à son créateur.

Le VT100 connaît un succès qui dépasse la simple vente de terminaux. Ses séquences de contrôle deviennent un langage partagé entre logiciels et écrans. Un éditeur peut déplacer le curseur, effacer une zone et mettre en évidence un texte sans connaître le détail du matériel. Des décennies plus tard, les émulateurs de terminaux continueront à reproduire ce comportement.

Cette postérité illustre une loi paradoxale. Lorsqu'un constructeur réussit à imposer une interface, celle-ci devient utile à ses concurrents. Le standard augmente le marché, mais réduit l'exclusivité. **DEC** gagne avec le VT100 une influence durable et, en même

2. Digital Equipment Corporation. *VT100 User Guide*. Digital Equipment Corporation. 1978 ; Computer History Museum. *VT100 Terminal*. 1978.

temps, contribue à un monde où le terminal peut être séparé de la machine qui l'a popularisé.

16.3 VAXcluster et le stockage commun

Annoncé en 1983, VAXcluster relie plusieurs VAX exécutant chacun VMS et leur permet de partager des fichiers et des périphériques de stockage. Un gestionnaire de verrous distribués coordonne les accès concurrents. Les contrôleurs HSC administrent des ensembles de disques et prennent en charge certaines fonctions sans mobiliser le processeur principal.

Le résultat ne correspond ni à un ordinateur unique ni à un simple réseau de postes indépendants. Plusieurs machines coopèrent et peuvent offrir une disponibilité supérieure. Lorsqu'un processeur est arrêté pour maintenance, une partie des services peut continuer ailleurs. Le Computer History Museum considère le HSC comme l'un des produits de stockage les plus réussis de **DEC**. [3]

Les clusters renforcent la fidélité à VMS et aux périphériques du constructeur. Ils donnent également une forme concrète à l'idée d'informatique distribuée : la puissance et les données ne résident plus nécessairement dans une seule armoire.

VAXcluster répond à une demande de continuité de service. Les clients ne veulent plus seulement une machine fiable ; ils veulent que l'application reste disponible pendant la maintenance ou malgré certaines défaillances. Le cluster déplace donc la notion de fiabilité du composant vers l'ensemble. Une machine peut s'arrêter sans que le service entier disparaisse.

Cette idée change l'organisation des salles informatiques. Les disques sont placés dans des ressources communes, plusieurs processeurs coopèrent et les administrateurs planifient les arrêts. La puissance peut être augmentée par étapes. Le client ne remplace plus nécessairement une machine par une plus grande : il ajoute

3. Computer History Museum. *Networked Storage Systems Commercialized*. 1983.

un nœud à un environnement existant.

La réussite repose sur l'intégration de VMS, de DECnet et du stockage. Aucun de ces éléments, pris isolément, n'explique le résultat. Le cluster est l'expression la plus aboutie du modèle **DEC** : une architecture de système où matériel et logiciel ont été pensés ensemble.

16.4 Un monde cohérent, mais de moins en moins fermé

À son apogée, **DEC** peut fournir le processeur, le système, le terminal, les disques, le réseau et le service. Pour un client, cette cohérence réduit les risques. Un interlocuteur unique prend en charge l'ensemble et garantit que les composants ont été conçus pour fonctionner ensemble.

Mais Ethernet, TCP/IP, UNIX et les terminaux standardisés modifient le marché. Les clients veulent relier des équipements de marques différentes. L'ouverture devient une exigence, pas seulement une option. **DEC** tente d'intégrer les standards à DECnet et de proposer des stratégies multiconstructeurs, mais la valeur se déplace progressivement de l'ensemble propriétaire vers les composants interchangeables.

Le changement devient évident lorsque le terminal passif cède la place au micro-ordinateur et à la station de travail. L'utilisateur ne veut plus seulement accéder à la puissance du réseau : il veut posséder de la puissance sur son bureau. C'est un déplacement que **DEC**, malgré toute son expérience des petites machines, aura du mal à transformer en avantage.

Chapitre 17

Le bureau échappe à DEC

L'histoire de **DEC** pourrait laisser croire que l'entreprise était naturellement préparée au micro-ordinateur. Elle avait construit des machines plus petites et moins coûteuses que les grands systèmes, placé le calcul dans les laboratoires et défendu l'interaction directe. Pourtant, le passage au poste personnel ne consiste pas seulement à réduire encore un PDP ou un VAX. Il change la manière dont le matériel est vendu, renouvelé et standardisé.

Au début des années 1980, le bureau devient un marché de masse. Les utilisateurs attendent des logiciels courants, une distribution rapide et des prix comparables d'un constructeur à l'autre. L'IBM PC, lancé en 1981, s'appuie sur des composants largement disponibles et sur un système d'exploitation fourni par **Microsoft**. Les compatibles reproduisent rapidement son architecture. La valeur ne réside plus uniquement dans l'intégration maîtrisée par un constructeur.

Le déplacement vers le bureau semble pourtant correspondre à toute l'histoire de l'entreprise. Depuis le PDP-1, **DEC** a rapproché le calcul de l'utilisateur. Mais le micro-ordinateur introduit une rupture économique. Il est vendu en grand nombre, par des réseaux de distribution, à des clients qui ne demandent pas une étude d'installation. Le logiciel standard et les accessoires disponibles comptent autant que la conception du processeur.

La fabrication change elle aussi. Les volumes abaissent les prix et accélèrent les renouvellements. Un constructeur habitué à maintenir

des systèmes pendant de longues années doit accepter que certains produits deviennent rapidement obsolètes. Cette logique heurte la culture de continuité et de service qui a fidélisé les clients de **DEC**.

17.1 Trois réponses au lieu d'une

DEC propose plusieurs machines personnelles presque simultanément. Le DECmate prolonge la tradition du traitement de texte et l'architecture 12 bits héritée du PDP-8. Le Professional 300 reprend des éléments du PDP-11 dans un système de bureau. Le Rainbow 100 cherche à combiner les mondes CP/M et MS-DOS grâce à deux microprocesseurs.

Chacune possède une logique technique défendable. Ensemble, elles donnent au marché une impression de dispersion. Les logiciels et périphériques ne sont pas toujours compatibles avec le standard IBM PC. Les prix restent élevés et les circuits de vente de **DEC** sont habitués à des contrats de systèmes, non à la distribution rapide d'ordinateurs personnels.

Le problème n'est pas l'absence d'ingénieurs compétents. Il tient à la difficulté de choisir une plate-forme qui réduirait le contrôle de l'entreprise sur le matériel et les logiciels. **DEC** sait vendre une solution complète à une organisation. Le marché du PC récompense au contraire la compatibilité, la disponibilité d'applications tierces et la baisse continue des marges.

Le DECmate, le Professional et le Rainbow révèlent trois héritages différents. Le premier protège le monde 12 bits et le traitement de texte, le deuxième cherche à transporter le PDP-11 sur le bureau, le troisième tente de rejoindre les logiciels des micro-ordinateurs. Aucun n'est absurde. Mais le client voit surtout qu'il doit choisir entre des offres de la même entreprise qui ne partagent pas complètement leurs programmes et leurs périphériques.

Cette dispersion ne vient pas uniquement d'une erreur de marketing. Elle est le produit de l'organisation par lignes, où plusieurs groupes défendent leur marché et leur technologie. Ce qui avait permis à **DEC** d'explorer rapidement les mini-ordinateurs rend plus difficile une réponse unique lorsqu'un standard externe s'impose.

17.2 La station de travail

Un autre déplacement se produit dans les laboratoires et les bureaux d'études. Des sociétés comme **Sun Microsystems** et **Apollo Computer** proposent des stations graphiques reliées en réseau, souvent fondées sur UNIX. Elles offrent à un ingénieur une machine personnelle assez puissante pour le développement, la simulation ou la conception assistée.

DEC répond avec les VAXstations et le système ULTRIX, sa version d'UNIX. Les stations bénéficient de l'écosystème VAX et de la puissance du réseau. Mais la concurrence progresse rapidement grâce aux microprocesseurs RISC et aux standards ouverts. Les clients peuvent acheter le matériel d'un fournisseur, le système d'un autre et les applications d'un troisième.

L'entreprise se trouve alors prise entre plusieurs fidélités. VMS représente un patrimoine immense et rentable. UNIX attire les universités et les ingénieurs qui recherchent la portabilité. Les PC imposent MS-DOS puis Windows dans les bureaux. Soutenir chaque monde coûte cher, mais en abandonner un signifie céder des clients.

La station de travail attire précisément les utilisateurs qui avaient fait la force de **DEC** : ingénieurs, scientifiques et programmeurs. Mais ces clients valorisent désormais les réseaux Ethernet, les fenêtres graphiques et UNIX, environnement qu'ils peuvent retrouver chez plusieurs constructeurs. **Sun Microsystems** comprend que la station vaut par le réseau et par un ensemble de standards, non par une compatibilité avec une famille plus ancienne.

Les VAXstations restent solides et bénéficient de VMS, mais elles doivent soutenir deux identités. Avec VMS, elles prolongent l'empire VAX. Avec ULTRIX, elles entrent dans un marché UNIX où l'architecture matérielle est plus facile à comparer et à remplacer. La même ouverture qui attire de nouveaux clients réduit la fidélité automatique à la plate-forme.

17.3 Quand l'utilisateur choisit l'écosystème

Dans le modèle traditionnel de **DEC**, le constructeur définit l'architecture, fournit le système et organise les périphériques. Dans le nouveau marché, l'utilisateur choisit d'abord un ensemble de logiciels et de formats devenus communs. Le matériel doit s'y conformer. La décision se déplace de l'ingénieur système vers le responsable de parc, le revendeur ou parfois l'utilisateur individuel.

Cette transformation affaiblit l'avantage de l'excellence intégrée. Un VAX peut être techniquement remarquable, mais un réseau de stations moins coûteuses et compatibles avec des logiciels standard peut répondre suffisamment bien au besoin. Le client n'achète plus nécessairement la meilleure machine isolée ; il choisit l'écosystème qui lui donne le plus de liberté.

DEC comprend progressivement cette évolution et annonce des stratégies d'ouverture, intègre TCP/IP et développe des produits compatibles avec plusieurs systèmes. Mais la transition exige des décisions rapides et des renoncements. Or l'entreprise a grandi dans une culture où plusieurs équipes peuvent poursuivre leur propre solution et où la qualité technique justifie souvent la patience.

Le PC modifie enfin le pouvoir dans l'entreprise cliente. Un département peut acheter des dizaines de machines sans mener le long processus associé à un système central. Les utilisateurs installent des applications choisies localement et échangent des fichiers selon des formats devenus courants. Le service informatique ne disparaît pas, mais il ne contrôle plus chaque décision.

Pour **DEC**, cette décentralisation aurait dû être familière. Elle se produit toutefois sur une plate-forme définie en grande partie par d'autres : processeurs Intel, systèmes Microsoft, périphériques compatibles et logiciels indépendants. Accepter ce monde revient à renoncer à une part du contrôle technique qui avait donné sa cohérence aux PDP et à VAX.

Cette culture avait permis l'invention des PDP et de VAX. À la fin des années 1980, elle s'est étendue à une organisation de plus de cent mille personnes, avec de nombreuses gammes et des intérêts

parfois contradictoires. Pour comprendre la crise, il ne suffit donc pas d'opposer un bon produit à un mauvais marché. Il faut regarder comment une manière de travailler, devenue un empire, transforme ses propres forces en inertie.

Cinquième partie

Le dernier pari

Chapitre 18

Une culture devenue empire

La culture de **DEC** ne se résume pas à quelques habitudes sympathiques d'une jeune entreprise. Elle forme un système de décision. Une démonstration technique, un prototype qui fonctionne ou l'avis d'un client ingénieur peuvent peser davantage qu'une longue étude de marché. Les équipes reçoivent une autonomie importante et les responsables de produits sont encouragés à se comporter comme des entrepreneurs au sein de la société.

Cette méthode produit des résultats remarquables lorsqu'un marché nouveau reste à inventer. Elle devient plus difficile lorsque l'entreprise doit coordonner des dizaines de gammes, protéger une base installée mondiale et choisir rapidement les produits qu'elle acceptera d'abandonner.

Dans les premières années, l'autonomie ne constitue pas une doctrine écrite. Elle résulte de la petite taille et du manque de ressources. Un ingénieur qui propose une amélioration doit souvent la construire lui-même, trouver des composants et convaincre un client. La croissance transforme cette nécessité en modèle de management. Les lignes de produits disposent de responsables qui pensent leur activité comme une entreprise presque complète.

Cette structure explique la variété de **DEC**. Une équipe peut poursuivre les 36 bits pendant qu'une autre réduit le PDP-8 ou prépare le PDP-11. Les marchés ne sont pas obligés d'attendre qu'une direction centrale comprenne toutes leurs particularités. La concurrence interne produit des idées et attire des personnalités

capables de défendre une vision.

Mais elle crée une ambiguïté sur la décision finale. Lorsqu'un projet fonctionne, l'autonomie paraît géniale. Lorsqu'il faut fermer une gamme ou choisir un standard commun, chaque groupe présente des clients, des revenus et une feuille de route. L'organisation excelle à commencer ; elle souffre davantage lorsqu'il faut arrêter.

18.1 Des entrepreneurs à l'intérieur de l'entreprise

Au milieu des années 1960, la croissance révèle les limites de l'organisation informelle des débuts. Les équipes d'ingénierie, la fabrication et les ventes ne se coordonnent pas toujours. *Ken Olsen* met en place une organisation par lignes de produits. Chaque responsable défend son projet, négocie l'accès aux ressources communes et répond de ses résultats devant des comités internes.

Le système évite les frontières rigides de divisions entièrement séparées. Un groupe peut utiliser les mêmes usines, les mêmes vendeurs et les mêmes services qu'un autre. Il favorise l'initiative et permet à plusieurs marchés d'être explorés en parallèle. Le PDP-8, les machines 36 bits, les modules et les systèmes spécialisés peuvent progresser sans attendre une stratégie unique parfaitement ordonnée.

La contrepartie apparaît peu à peu. Les projets entrent en concurrence pour les mêmes ingénieurs et les mêmes budgets. Une équipe peut continuer à défendre une architecture parce qu'elle possède des clients fidèles, même lorsque l'entreprise aurait besoin d'une concentration plus nette. Le débat technique, source de qualité, devient parfois une difficulté à décider.

La culture quotidienne repose aussi sur une forme de franchise. Les réunions peuvent être rudes, les démonstrations servent à contester une décision et le rang hiérarchique ne suffit pas toujours à clore un débat technique. Pour de jeunes ingénieurs, cette atmosphère est stimulante. Elle donne l'impression que la qualité de l'idée compte réellement.

Cette liberté n'est pourtant pas également répartie. Les per-

sonnes capables de s'exprimer dans ce langage de confrontation prennent davantage de place, et les décisions restent fortement dépendantes de l'attention des dirigeants. À mesure que la société s'internationalise, un modèle né dans les couloirs de Maynard devient plus difficile à transmettre.

18.2 Le départ d'Harlan Anderson

Harlan Anderson quitte **DEC** en 1966. Les sources s'accordent sur la date, mais attribuent des poids différents aux désaccords personnels, à la réorganisation et aux difficultés de la ligne PDP-6. Il serait réducteur de transformer cette séparation en scène unique. Elle marque néanmoins la fin de la direction à deux qui avait lancé l'entreprise.

Anderson devient ensuite conseiller technologique de **Time Inc.**, puis investisseur dans de jeunes sociétés. Son parcours prolonge l'un des aspects de son rôle initial : relier la technique au financement et repérer des domaines émergents. Olsen, lui, reste à la tête de **DEC** et imprime de plus en plus directement sa personnalité à l'organisation. [1]

Cette divergence rappelle que les entreprises ne sont pas seulement conduites par des idées. Elles dépendent de relations, de responsabilités et de la manière dont le pouvoir se distribue lorsque la petite équipe devient une institution.

Le départ d'Anderson prive l'entreprise d'une voix qui avait accompagné le financement et les relations extérieures des débuts. Olsen demeure entouré de dirigeants expérimentés, mais le récit de **DEC** se confond de plus en plus avec le sien. Cette personnalisation renforce l'identité de la société ; elle rend aussi la succession difficile à penser.

Anderson ne disparaît pas du monde technologique. Son activité de conseil et d'investissement lui permet d'observer d'autres

1. Grainger College of Engineering. *Harlan E. Anderson.* University of Illinois Urbana-Champaign ; Edgar H. Schein et al. *DEC Is Dead, Long Live DEC : The Lasting Legacy of Digital Equipment Corporation.* San Francisco : Berrett-Koehler, 2004. isbn : 978-1-57675-305-7.

entreprises, notamment au moment où les semi-conducteurs et les logiciels créent de nouveaux marchés. Sa trajectoire forme un contrepoint à celle d'Olsen : l'un reste au centre d'un constructeur intégré, l'autre circule entre plusieurs projets.

18.3 Ken et la preuve technique

Les employés appellent souvent le président simplement « Ken ». La familiarité ne diminue pas son autorité. Olsen circule, pose des questions et ramène volontiers une discussion au besoin réel du client. Il se méfie des effets de mode et valorise les solutions utiles, réparables et économiquement justifiables.

Ce style peut libérer une équipe. Un ingénieur convaincu obtient parfois la possibilité de construire un prototype plutôt que de remplir une longue chaîne d'autorisations. Mais il peut également rendre les décisions imprévisibles. Un projet reçoit un soutien parce qu'il séduit le dirigeant, tandis qu'un autre est interrompu malgré des mois de travail. À mesure que **DEC** grandit, la culture de la discussion directe devient difficile à reproduire dans toutes les filiales.

La politique longtemps associée à l'absence de licenciements renforce la loyauté. Elle encourage les salariés à envisager une carrière entière dans l'entreprise et permet de conserver un savoir technique profond. Lorsque les pertes obligent finalement à supprimer des emplois, le choc est donc culturel autant que financier.

Olsen peut consacrer une attention impressionnante à un détail de produit. Cette qualité protège **DEC** contre certaines décisions purement financières et rappelle aux équipes que le client rencontrera une machine réelle. Mais le dirigeant doit désormais arbitrer des questions qui ne se démontrent pas par un prototype : évolution des canaux de vente, pouvoir des éditeurs de logiciels, effet des standards et vitesse de diffusion d'un écosystème.

La preuve technique reste nécessaire, mais elle n'est plus suffisante. Un produit peut fonctionner admirablement et perdre parce que ses applications arrivent trop tard ou que son prix ne bénéficie pas des mêmes volumes. La culture de l'ingénieur doit apprendre à

reconnaître ces mécanismes sans les réduire à des modes.

18.4 L'utilisateur au centre, le marché parfois au second plan

Les ingénieurs de **DEC** parlent avec des utilisateurs capables de lire un schéma ou de décrire précisément un problème logiciel. Cette proximité produit les interfaces ouvertes, les handbooks et les bibliothèques DECUS. Elle donne aussi l'habitude de servir un client qui choisit sa machine pour ses qualités techniques.

Le marché du PC et des standards ouverts fonctionne autrement. L'acheteur peut privilégier le prix, la disponibilité d'une application ou la conformité à une norme, même si la machine est moins élégante. La vente se décide parfois loin de l'ingénieur. La culture de **DEC** interprète trop souvent cette évolution comme une mode passagère ou comme un problème que de meilleurs produits finiront par résoudre.

Au sommet, vers la fin des années 1980, l'entreprise atteint près de 14 milliards de dollars de chiffre d'affaires et plus de 120 000 employés. Elle est devenue, derrière IBM, l'un des plus grands constructeurs du monde. [2] Sa taille rend désormais toute réorientation extrêmement coûteuse.

La prochaine architecture devra donc accomplir plus qu'une prouesse technique. Elle devra remplacer VAX, préserver plusieurs systèmes d'exploitation, convaincre les développeurs et redonner un avenir à une organisation en crise. Le projet Alpha répond avec éclat à la première partie du problème. Le temps lui manquera pour résoudre les autres.

2. Sara Lott. *What the DEC ? ! ? Records of Minicomputer Giant Digital Equipment Corporation Open for Research at CHM*. 2017.

Chapitre 19

Alpha contre le temps

Au milieu des années 1980, les ingénieurs de **DEC** voient arriver les architectures RISC. [1] Les stations de travail fondées sur ces processeurs progressent plus vite que les grands jeux d'instructions complexes. VAX, admirablement compatible, porte aussi le poids de décisions prises dans les années 1970.

Plusieurs projets internes cherchent une nouvelle voie. PRISM explore une architecture RISC, mais il est annulé avant d'aboutir à un produit. Les idées et les personnes ne disparaissent pas. Elles nourrissent Alpha, projet lancé avec une ambition exceptionnelle : définir une architecture 64 bits capable de durer plusieurs générations et d'accueillir les principaux systèmes d'exploitation de l'entreprise.

L'annulation de PRISM laisse une blessure et un réservoir d'idées. Des ingénieurs ont travaillé sur les principes RISC, les compilateurs et les systèmes nécessaires. Lorsque le projet Alpha démarre, ils ne repartent pas de rien. Ils disposent aussi d'une leçon : une architecture nouvelle ne survivra pas si elle reste isolée dans une équipe et si l'entreprise ne prépare pas simultanément les logiciels et les produits.

Alpha est donc conçu avec un horizon inhabituellement long.

1. Une architecture *RISC* privilégie un ensemble d'instructions relativement simples, conçues pour être exécutées rapidement et pour faciliter le travail du pipeline du processeur. Le terme ne signifie pas qu'un système complet soit simple.

Les architectes cherchent à éviter les décisions qui bloqueraient la fréquence ou l'extension future. Certaines fonctions complexes sont laissées au logiciel pour que les processeurs puissent progresser avec les technologies de fabrication. Cette sobriété contraste avec VAX, dont le jeu d'instructions avait accumulé des opérations riches.

19.1 Une architecture tournée vers l'avenir

Alpha élimine une grande partie de la complexité visible de VAX. Les instructions sont de longueur régulière, les opérations mémoire sont séparées des calculs et le dessin favorise des fréquences élevées. Le passage à 64 bits donne un espace d'adressage immense selon les besoins de l'époque et prépare des applications scientifiques, des serveurs et de grandes bases de données.

Le premier microprocesseur, le 21064, est présenté en 1992. Ses performances placent **DEC** parmi les concepteurs de processeurs les plus avancés. L'architecture ne se limite pas à une puce : elle doit évoluer par familles, avec des implémentations plus rapides et des systèmes allant de la station au serveur multiprocesseur. [2]

Le choix technique est clair. La difficulté se trouve dans la migration. Les clients VAX possèdent des programmes parfois anciens et indispensables. **DEC** développe des outils de traduction binaire et porte OpenVMS vers Alpha. Digital UNIX exploite également l'architecture, tandis que **Microsoft** propose Windows NT sur certains systèmes Alpha. Une même puce doit donc soutenir plusieurs écosystèmes qui ne poursuivent pas toujours les mêmes objectifs.

La migration de VAX vers Alpha constitue un exploit moins visible que la fréquence du processeur. Les applications OpenVMS doivent continuer à fonctionner, parfois sans disposer immédiatement d'une version recompilée. Les outils de traduction binaire analysent l'ancien code et produisent une forme exécutable sur la

2. Alpha Architecture Committee. *Alpha Architecture Reference Manual*. Digital Press, 1992. isbn : 978-1-55558-098-8. doi : 10.1016/C2009-0-27261-8 ; Computer History Museum. *DEC Announces Alpha Chip Architecture*. 1992.

nouvelle architecture. Le résultat ne remplace pas toujours une recompilation, mais il protège les clients pendant la transition.

Cette continuité donne à Alpha un accès immédiat à une base logicielle, mais elle lui impose aussi le poids de plusieurs mondes. OpenVMS défend les installations historiques, Digital UNIX cherche les marchés ouverts et Windows NT doit attirer les applications du PC et du serveur. La même architecture sert des stratégies qui ne se renforcent pas toujours mutuellement.

19.2 La puissance en démonstration

Les stations et serveurs Alpha montrent que **DEC** sait encore transformer une architecture en produits de haut niveau. Les systèmes sont employés dans le calcul scientifique, les bases de données et les services Internet. Lancé en 1995, le moteur de recherche AltaVista fonctionne sur des machines Alpha et sert de vitrine à leur capacité à indexer rapidement un volume considérable de pages.

Cette démonstration rappelle les débuts du PDP-1 : un programme spectaculaire rend une architecture visible. Mais le contexte a changé. *Spacewar !* aidait à définir un marché qui n'existait pas encore. AltaVista apparaît dans un secteur où l'avantage se déplace rapidement vers les services en ligne, les logiciels et les volumes d'utilisateurs. La machine puissante ne garantit plus que son constructeur conservera la valeur créée au-dessus d'elle.

AltaVista donne à Alpha une scène publique. Le moteur répond vite, indexe un Web en croissance et démontre les capacités des serveurs. Comme *Spacewar !* autrefois, le programme rend la puissance visible. Mais **DEC** ne transforme pas cette démonstration en domination durable des services Internet. La valeur d'AltaVista se déplace vers la marque, les utilisateurs et la publicité, domaines éloignés du métier historique du constructeur.

Le contraste résume la difficulté des années 1990. **DEC** sait encore fabriquer l'infrastructure d'un futur, mais ce futur est de moins en moins contrôlé par celui qui construit le processeur. Les plates-formes se gagnent par les développeurs, les partenariats et

les volumes. Alpha arrive avec des performances remarquables dans une entreprise qui dispose de moins en moins de temps pour bâtir tout l'écosystème nécessaire.

19.3 Une architecture sans répit

Alpha doit combattre sur plusieurs fronts. Les processeurs x86 progressent en performances et bénéficient d'un écosystème immense. Les architectures RISC de **Sun**, **IBM**, **Hewlett-Packard** et d'autres constructeurs occupent déjà les stations et serveurs. VAX reste rentable chez les clients installés, ce qui ralentit parfois la transition vers son successeur.

Le projet exige en outre des investissements de fabrication considérables. Concevoir un processeur ne suffit pas : il faut suivre la course aux procédés de gravure, produire en volume et attirer les éditeurs. Pour une entreprise dont les revenus se contractent, le calendrier devient aussi important que la qualité de l'architecture.

Alpha prouve donc deux choses apparemment contradictoires. **DEC** conserve une capacité d'innovation exceptionnelle. Mais l'industrie n'est plus structurée de manière à récompenser automatiquement un constructeur qui possède la meilleure architecture propriétaire. La valeur dépend du logiciel, des partenariats, des volumes et de la vitesse avec laquelle un standard se diffuse.

À mesure que les pertes s'accumulent, la société vend des activités et conclut des accords destinés à financer la transition. L'architecture qui devait assurer plusieurs décennies d'avenir devient l'un des actifs que d'autres entreprises chercheront à acquérir.

Chapitre 20

La fin de l'indépendance

La crise de **DEC** ne commence pas le jour où un produit échoue. Elle résulte d'un déplacement général du marché. Les microprocesseurs standard gagnent en puissance, les stations UNIX réduisent le coût du calcul technique, les PC occupent les bureaux et les réseaux ouverts permettent d'assembler des systèmes de marques différentes. La société continue de vendre des VAX, des services et du stockage, mais la croissance qui avait rendu supportable la diversité interne disparaît.

Au début des années 1990, les retards de produits, les investissements dans de grands systèmes et la pression sur les prix provoquent les premières pertes annuelles de l'histoire de l'entreprise. La tradition d'emploi stable cède devant les suppressions de postes. Pour des salariés qui avaient connu une croissance presque continue, la rupture est profonde.

La crise est ressentie jusque dans les lieux qui symbolisaient la croissance. Des bâtiments ferment, des équipes sont regroupées et la politique d'emploi stable ne peut plus être maintenue. Les licenciements ne représentent pas seulement une réduction de coûts : ils rompent la promesse implicite selon laquelle la compétence accumulée trouverait toujours une place dans l'entreprise.

Les clients observent ces changements avec inquiétude. Leur confiance dépend de la durée du support. Une entreprise qui réduit ses effectifs doit continuer à maintenir des machines installées depuis des années tout en finançant Alpha et les nouveaux services.

Chaque économie risque d'affaiblir ce qui reste rentable.

20.1 Le départ de Ken Olsen

Ken Olsen quitte la présidence en 1992 après trente-cinq années à la tête de la société. Son départ ne peut être réduit au jugement selon lequel un fondateur n'aurait pas compris le PC. Il avait construit une organisation capable de déplacer l'informatique des grands centres vers les départements et les réseaux. Mais la structure qu'il dirigeait peinait à choisir entre la protection de VAX/VMS, l'ouverture vers UNIX, le marché du PC et le pari Alpha.

Robert Palmer prend la direction et engage une restructuration plus dure. Les activités sont réorganisées, des sites ferment et des actifs sont vendus. L'objectif est de concentrer l'entreprise sur les serveurs, les services, le stockage et Alpha. Les décisions améliorent certains résultats, mais réduisent aussi l'intégration qui avait fait l'identité de **DEC**.

La société conclut notamment en 1997 un accord avec **Intel** après un conflit de brevets. Intel reprend des activités de fabrication de semi-conducteurs, tandis que **DEC** obtient des ressources et un partenaire industriel. Alpha continue, mais l'entreprise ne maîtrise plus seule toute sa chaîne technologique.

Le départ d'Olsen marque la fin d'une époque, mais pas un redressement immédiat. *Robert Palmer*, issu de la technologie des semi-conducteurs, doit simplifier une organisation dont les gammes et les responsabilités se chevauchent. Il ferme, vend et regroupe. Ces décisions auraient été presque impensables pendant la croissance, lorsque chaque activité pouvait espérer trouver son marché.

La restructuration améliore certaines performances et recentre **DEC** sur les serveurs, le stockage et les services. Elle réduit en même temps la capacité de raconter l'entreprise comme un ensemble unique. Les activités cédées emportent des personnes, des technologies et parfois des relations avec les clients. Le constructeur intégré devient progressivement un portefeuille d'actifs.

20.2 Ce que Compaq veut acheter

Compaq Computer Corporation s'est imposée dans le PC compatible, puis cherche à devenir un fournisseur complet de systèmes d'entreprise. Elle a besoin de services mondiaux, de stockage, de serveurs puissants et de relations avec les grands comptes. **DEC** possède précisément ces éléments, même si son image publique reste associée à une société en difficulté.

L'accord annoncé en janvier 1998 valorise l'opération à environ 9,6 milliards de dollars en numéraire et en actions. Il s'agit alors de la plus importante acquisition de l'industrie informatique. Compaq met en avant le réseau de services de **DEC**, Alpha, OpenVMS, Digital UNIX, les systèmes Windows NT et le stockage. [1]

Le rachat est achevé en juin 1998. Quarante et un ans après la fondation de Maynard, **DEC** cesse d'exister comme entreprise indépendante. La marque, les produits et les équipes ne disparaissent pas immédiatement. Ils entrent dans une organisation qui devra elle-même fusionner avec **Hewlett-Packard** en 2002.

Compaq ne rachète donc pas seulement une marque affaiblie. Elle acquiert une présence mondiale dans les services, une technologie de stockage, des serveurs, des systèmes d'exploitation et des ingénieurs capables de travailler avec les grands comptes. L'opération doit transformer un champion du PC en fournisseur d'infrastructure.

Les cultures sont très différentes. Compaq a grandi avec des cycles rapides, des composants standard et une discipline de coûts liée au marché du compatible. **DEC** porte des engagements de support, des architectures propriétaires et une tradition d'ingénierie de système. Les réunir ne consiste pas à ajouter leurs chiffres d'affaires. Il faut décider quelles technologies survivront et comment les clients seront accompagnés.

1. Elizabeth Corcoran. *Computer Industry Has a New Giant : Compaq to Buy Digital*. The Washington Post. 27 jan. 1998 ; Compaq Computer Corporation. *Quarterly Report : Acquisition and Divestiture Disclosures*. U.S. Securities et Exchange Commission, 1999.

20.3 Une fin plus complexe qu'un échec

Il serait tentant de raconter la disparition de **DEC** comme la punition d'une entreprise trop attachée à ses mini-ordinateurs. Cette explication contient une part de vérité, mais elle oublie l'ampleur de la transformation industrielle. Les composants standard, les logiciels indépendants et les réseaux ouverts déplacent la valeur hors du constructeur intégré. IBM elle-même doit se réorganiser profondément. D'autres fabricants de mini-ordinateurs disparaissent ou sont absorbés.

DEC commet des erreurs : gammes dispersées, choix tardifs, coûts élevés et difficulté à transformer l'excellence technique en standard de masse. Elle porte aussi le poids de son succès. Des milliers de clients attendent la continuité de systèmes essentiels. Une jeune entreprise peut abandonner un produit ; un constructeur mondial doit fournir des migrations, du support et des pièces pendant des années.

Le rachat clôt le récit institutionnel, mais pas celui des produits. Des machines continuent à fonctionner, des contrats restent actifs et les équipes passent sous de nouveaux noms. Alpha, OpenVMS et les services connaîtront encore plusieurs changements de propriétaires. Cette continuité fragmentée explique pourquoi la disparition de la société ne se traduit pas par une date unique dans la vie des utilisateurs.

La fin de l'indépendance ne supprime donc pas l'héritage. Elle le disperse. Des ingénieurs rejoignent d'autres sociétés. OpenVMS poursuit sa vie. Les concepts de Windows NT portent l'expérience de *Dave Cutler*. Les terminaux, les réseaux, UNIX, C et les architectures issues des PDP continuent à influencer les systèmes. Même le mot « Digital », devenu banal, garde quelque chose de l'époque où une petite entreprise choisit ce nom pour éviter d'écrire « computer » sur son premier projet.

Épilogue — Ce que DEC a rendu possible

L'histoire de **DEC** commence par une réduction : réduire la taille, le prix et la distance qui séparent l'utilisateur de l'ordinateur. Elle ne s'achève pas lorsque cette réduction atteint le micro-ordinateur. Entre 1957 et 1998, l'entreprise a contribué à modifier plusieurs fois la place de la machine.

Le PDP-1 montre qu'un ordinateur commercial peut être interactif. Le PDP-8 fait du calculateur un composant de laboratoire ou d'équipement. Le PDP-10 donne une forme sociale au temps partagé. Le PDP-11 fournit une architecture régulière autour de laquelle grandissent systèmes, périphériques et langages. VAX et VMS protègent le logiciel au moment où les machines deviennent des plates-formes. DECnet et VAXcluster transforment plusieurs ordinateurs en environnement commun. Alpha rappelle enfin que l'innovation technique peut survivre plus longtemps que le modèle économique qui la porte.

Mais l'héritage le plus profond se trouve peut-être dans les pratiques. **DEC** a souvent considéré l'utilisateur comme une personne techniquement compétente, capable de lire un manuel, d'écrire une interface et de proposer une amélioration. Cette confiance favorise les communautés, les bibliothèques DECUS et les usages imprévus. Elle explique pourquoi des machines produites à quelques dizaines d'exemplaires ont laissé une trace disproportionnée.

La trajectoire des personnes prolonge celle des machines. *Gordon Bell* apporte son expérience de l'architecture à d'autres entreprises et à la recherche. *Dave Cutler* dirige le développement de Windows NT. Les programmeurs formés sur UNIX et C bâtissent des

systèmes sur des processeurs qui n'existaient pas lorsque le PDP-11 fut dessiné. Les idées ne restent pas enfermées dans les produits qui les ont fait naître.

Il en va de même des interfaces. Les séquences du VT100 vivent dans les émulateurs de terminaux, les principes des bus réapparaissent sous d'autres formes et les clusters annoncent des services distribués où la panne d'une machine ne doit pas interrompre l'ensemble. Ces héritages ne sont pas des copies directes. Ils montrent comment une solution historique peut devenir une habitude de pensée.

Les systèmes de **DEC** restent aujourd'hui accessibles par les collections, les restaurations et les simulateurs. Le projet SIMH, initié par *Bob Supnik*, permet d'exécuter de nombreux environnements historiques sur du matériel contemporain. Les répliques physiques et les restaurations du Computer History Museum redonnent aux panneaux, aux rubans et aux terminaux une présence que les photographies ne peuvent restituer. [2]

Cette conservation ne relève pas seulement de la nostalgie. Elle permet de comprendre les contraintes qui ont façonné les architectures et les logiciels. Programmer avec quelques milliers de mots, attendre un télétype ou charger un système depuis DECtape oblige à regarder autrement les abstractions modernes. L'histoire technique devient une expérience.

La conservation restitue aussi la lenteur et la matérialité. Un manuel décrit une commande, mais il ne reproduit pas le bruit du télétype, l'attente d'un ruban ou l'attention portée aux lampes du panneau. Les restaurations et les répliques ne remplacent pas les archives ; elles ajoutent l'expérience des contraintes.

Pour l'historien, cette expérience évite une illusion fréquente : juger les anciennes machines à partir des possibilités modernes. Une mémoire de quelques milliers de mots n'est pas seulement une quantité faible. Elle structure le programme, impose des choix et rend certaines solutions élégantes. Comprendre ces limites permet de comprendre les personnes qui ont travaillé avec elles.

2. Open SIMH Project. *Open SIMH* ; Computer History Museum. *PDP-1 Restoration Project*.

DEC n'a pas inventé seule l'informatique interactive, le mini-ordinateur, UNIX, le réseau ou la station de travail. Aucune entreprise ne possède une telle histoire à elle seule. Son rôle fut de relier des idées, de les transformer en produits et de les placer entre les mains de communautés qui leur donnèrent une portée nouvelle.

C'est pourquoi l'ordre de ce récit compte. Les modules conduisent au PDP-1 ; le PDP-1 au MIT ; le MIT révèle la puissance d'une communauté ; le PDP-8 élargit le marché ; les périphériques transforment les usages ; le PDP-11 rend possible un écosystème ; UNIX et C dépassent cet écosystème ; VAX tente de préserver la continuité ; les réseaux ouvrent le système ; cette ouverture finit par rendre plus difficile le modèle intégré de **DEC**.

L'entreprise disparaît, mais la question qui l'avait fondée demeure : comment rapprocher la puissance de calcul des personnes qui en ont besoin ? Chaque génération d'informatique apporte une réponse nouvelle. Pendant quatre décennies, **DEC** fut l'un des lieux où cette réponse s'inventa.

Annexe A

Chronologie

Cette chronologie rassemble les principaux jalons du récit. Elle ne cherche pas à recenser chaque modèle ni chaque version logicielle. Les dates correspondent aux annonces ou aux premières livraisons lorsqu'elles sont clairement établies par les archives de **DEC** et les documents techniques.

Date	Événement
1951	Création du MIT Lincoln Laboratory. Les travaux sur la défense aérienne, le temps réel et les calculateurs interactifs forment l'environnement dans lequel travailleront *Ken Olsen* et *Harlan Anderson*.
1952	Anderson rejoint le Lincoln Laboratory, où Olsen devient son premier responsable.
1957	Fondation de **Digital Equipment Corporation** à Maynard, dans un ancien moulin. **American Research and Development** apporte le financement initial.
1957	Commercialisation des premiers *Laboratory Modules*, destinés aux ingénieurs, aux laboratoires et aux systèmes de contrôle.
1958	Introduction des *System Modules*, plus denses et adaptés à des ensembles permanents.
1959	Présentation publique du prototype PDP-1 à l'Eastern Joint Computer Conference.

Date	Événement
1960	Livraison du premier PDP-1 de production à **Bolt Beranek and Newman**.
1960	*Gordon Bell* rejoint **DEC** et participe aux extensions du PDP-1, puis aux architectures suivantes.
1961	Don du deuxième PDP-1 de production au Research Laboratory for Electronics du MIT.
1962	Développement et diffusion de *Spacewar !* sur le PDP-1. Des expériences de temps partagé sont menées au MIT et chez BBN.
1963	Introduction du PDP-5, première architecture 12 bits de la société destinée à un marché économique de laboratoire et de contrôle.
1964	Annonce du PDP-6, machine 36 bits conçue pour le calcul scientifique et le temps partagé.
1965	Le PDP-8 élargit le marché du mini-ordinateur grâce à un prix de base annoncé à 18 000 dollars pour une configuration exploitable.
1965	Création de DECUS, association d'utilisateurs favorisant l'échange de programmes et d'expériences.
1966	Départ de *Harlan Anderson*. *Ken Olsen* demeure à la tête de l'entreprise.
1967	Lancement du PDP-10, compatible avec le PDP-6 et mieux adapté à la diffusion commerciale du temps partagé.
1968	Création de **Data General** par plusieurs anciens de **DEC**, dont *Edson de Castro*.
1970	Livraison du PDP-11/20, premier modèle d'une famille 16 bits caractérisée par une architecture régulière et l'Unibus.
1971	UNIX devient opérationnel sur PDP-11 aux Bell Laboratories et commence à être utilisé pour le traitement de documents.
1973	Le noyau d'UNIX est réécrit en grande partie en langage C, renforçant sa portabilité.
1975	Annonce des premières phases de DECnet, architecture destinée à relier plusieurs systèmes **DEC**.

Date	Événement
1975	Début des travaux du comité VAX sur une extension 32 bits culturellement compatible avec le PDP-11.
1977	Présentation du VAX-11/780 et de l'architecture VAX.
1978	Livraison de VMS 1.0. Introduction du terminal VT100, dont les séquences de contrôle deviendront une référence durable.
1978	Publication de *The C Programming Language* par *Brian Kernighan* et *Dennis Ritchie*.
1980	Coopération entre **DEC**, **Intel** et **Xerox** autour de la spécification Ethernet commerciale.
1981	Lancement de l'IBM PC, qui accélère la formation d'un standard de bureau fondé sur des composants et logiciels indépendants.
1982–1983	**DEC** propose plusieurs réponses au poste personnel, notamment le DECmate, le Professional 300 et le Rainbow 100.
1983	Annonce des VAXclusters, qui associent plusieurs VAX et un stockage partagé sous VMS.
1984	Introduction d'ULTRIX-32, version UNIX de **DEC** pour la famille VAX.
1988	*Dave Cutler* quitte **DEC** et rejoint **Microsoft**, où il dirigera le développement de Windows NT.
1990	L'entreprise connaît la première perte annuelle de son histoire après plus de trois décennies de croissance.
1992	Départ de *Ken Olsen*. *Robert Palmer* prend la direction de la société.
1992	Présentation du microprocesseur Alpha 21064 et des premiers systèmes fondés sur l'architecture 64 bits Alpha.
1995	Lancement d'AltaVista sur des serveurs Alpha, démonstration publique de leur capacité à traiter rapidement un Web en forte croissance.
1997	Accord entre **DEC** et **Intel** à la suite d'un conflit de brevets ; Intel reprend notamment des activités de fabrication de semi-conducteurs.

Date	Événement
1998	Annonce puis achèvement du rachat de **DEC** par **Compaq**. L'entreprise cesse d'exister comme constructeur indépendant.
2002	Rachat de **Compaq** par **Hewlett-Packard**. Plusieurs technologies et équipes issues de **DEC** changent à nouveau de cadre industriel.

Annexe B

Repères sur les familles

Le nom PDP ne désigne pas une architecture unique. **DEC** l'emploie pour plusieurs lignées incompatibles, définies selon les marchés et les technologies disponibles. Le tableau suivant fournit des repères, sans remplacer les explications du récit.

Famille	**Repères**	**Orientation principale**
PDP-1	18 bits 1960	Informatique interactive, laboratoires, affichage graphique, recherche et premiers usages en temps partagé.
PDP-4, PDP-7, PDP-9, PDP-15	18 bits 1962	Prolongement plus économique et plus industrialisé de la lignée 18 bits pour le calcul scientifique, le contrôle et l'instrumentation.
PDP-5, PDP-8, PDP-12	12 bits 1963	Systèmes économiques de laboratoire, d'enseignement, de contrôle et d'intégration OEM. Le PDP-12 associe notamment les traditions du LINC et du PDP-8.
PDP-6, PDP-10, DECsystem-10, DECSYSTEM-20	36 bits 1964	Calcul scientifique, intelligence artificielle, universités, centres de recherche et temps partagé multiutilisateur.
PDP-11	16 bits 1970	Famille générale allant du contrôle embarqué aux systèmes multiutilisateurs, avec une architecture régulière et un vaste écosystème de périphériques.

Famille	Repères	Orientation principale
VAX	32 bits 1977	Extension de l'héritage PDP-11 vers la mémoire virtuelle, les applications plus importantes et une gamme allant du MicroVAX aux systèmes d'entreprise.
Alpha	64 bits 1992	Architecture RISC à hautes performances pour stations, serveurs, OpenVMS, Digital UNIX et certains systèmes Windows NT.

Quelques modèles structurants

PDP-1 Premier ordinateur complet commercialisé par **DEC**. Sa portée historique vient de l'interaction directe, de l'affichage et de l'environnement créé autour de lui au MIT.

PDP-8 Machine emblématique de la réduction des coûts. Ses nombreuses variantes maintiennent une architecture 12 bits pendant plusieurs générations technologiques.

PDP-10 Expression commerciale la plus durable de la lignée 36 bits. Il accueille des communautés de temps partagé et plusieurs travaux importants en intelligence artificielle et en réseau.

PDP-11/20 Premier modèle 16 bits de la famille PDP-11. Il introduit l'Unibus et une architecture dont la régularité favorisera les compilateurs et les systèmes d'exploitation.

VAX-11/780 Première réalisation de l'architecture VAX. Son nom manifeste la continuité avec le PDP-11, tandis que ses adresses 32 bits et VMS ouvrent une nouvelle génération de systèmes.

MicroVAX Déclinaison plus compacte et plus accessible de l'environnement VAX, destinée aux départements, aux laboratoires et aux réseaux distribués.

Alpha 21064 Première implémentation commerciale d'Alpha. Elle démontre que **DEC** reste capable de concevoir un microprocesseur parmi les plus performants de son époque.

Les dates et catégories doivent être lues comme des repères. Une annonce peut précéder la livraison, et un même modèle peut servir des usages très différents selon ses périphériques et son système d'exploitation.

Annexe C

Glossaire

Adresse Nombre qui désigne une position de mémoire ou, dans certaines architectures, un registre de périphérique. La largeur des adresses limite l'espace directement accessible par un programme.

Architecture Ensemble des caractéristiques visibles par le programmeur : instructions, registres, types de données, adressage et organisation des interruptions. Deux machines physiquement différentes peuvent partager la même architecture et exécuter les mêmes programmes.

Assembleur Langage qui représente les instructions de la machine par des noms symboliques. Le même terme désigne aussi le programme qui traduit ce texte en code exécutable.

Bus Ensemble de lignes et de règles permettant à plusieurs composants d'échanger des adresses, des données et des signaux de contrôle. L'Unibus du PDP-11 relie processeur, mémoire et périphériques selon un modèle commun.

Cluster Groupe de machines qui coopèrent pour fournir des services ou partager des ressources. Dans un VAXcluster, plusieurs systèmes VMS peuvent notamment accéder à un stockage commun et coordonner leurs fichiers.

Compatibilité Capacité à préserver des programmes, des données, des périphériques ou des habitudes entre plusieurs modèles. La compatibilité peut être binaire, logicielle, matérielle ou seulement « culturelle », comme lors du passage du PDP-11

à VAX.

DECnet Architecture de réseau développée par **DEC** à partir du milieu des années 1970. Elle relie plusieurs familles de machines et de systèmes d'exploitation avant de devoir coexister avec Ethernet, TCP/IP et les normes multiconstructeurs.

DECtape Support magnétique à petites bobines organisé en blocs adressables. Il offre sur plusieurs systèmes **DEC** un usage plus souple que la bande séquentielle traditionnelle.

DECUS Association d'utilisateurs de **DEC**, créée au milieu des années 1960. Elle organise l'échange de logiciels, les réunions techniques et la représentation des besoins des utilisateurs.

Entrée-sortie Ensemble des opérations par lesquelles l'ordinateur reçoit des données ou agit sur l'extérieur : clavier, télétype, disque, écran, capteur, réseau ou relais industriel.

Interruptions Mécanisme permettant à un événement matériel ou logiciel de suspendre momentanément le programme courant afin d'exécuter une routine particulière. Il évite au processeur de vérifier continuellement l'état de chaque périphérique.

Logiciel système Programme qui organise les ressources de la machine et fournit un environnement aux applications : système d'exploitation, assembleur, compilateur, éditeur, chargeur ou diagnostic.

Mémoire à tores Mémoire utilisant de petits anneaux magnétiques traversés par des fils. Elle a dominé de nombreux ordinateurs des années 1950 aux années 1970 et conserve généralement son contenu après la coupure de l'alimentation.

Mémoire virtuelle Technique qui sépare les adresses utilisées par un programme des emplacements physiques de la mémoire. Le système peut protéger les processus et déplacer des pages entre mémoire et stockage.

Mini-ordinateur Catégorie historique de machines plus petites et moins coûteuses que les grands systèmes, souvent installées dans un laboratoire, un département ou un équipement industriel. Le terme décrit un marché et un mode d'usage autant qu'une taille physique.

OEM *Original Equipment Manufacturer*. Entreprise qui intègre le processeur ou le système d'un fournisseur dans son propre

équipement, par exemple un instrument médical ou une machine industrielle.

Périphérique Équipement connecté au processeur : terminal, lecteur de bande, disque, imprimante, écran, interface réseau ou dispositif d'acquisition. Le périphérique détermine souvent l'usage réel de la machine.

Plate-forme Ensemble relativement stable formé par une architecture, un système, des outils, des périphériques et des conventions. Le terme souligne que la valeur ne réside pas uniquement dans le processeur.

Portabilité Possibilité d'adapter un programme à une autre architecture avec un effort limité. L'écriture d'UNIX en C a montré qu'une grande partie d'un système pouvait survivre au remplacement du processeur.

Processeur Partie de l'ordinateur qui exécute les instructions. Dans les premiers PDP, il occupe plusieurs cartes ou armoires ; dans les générations ultérieures, il peut être intégré dans quelques circuits puis dans un microprocesseur.

RISC Famille de principes d'architecture privilégiant des instructions régulières et relativement simples, adaptées à l'exécution en pipeline. Alpha appartient à cette tradition.

Système d'exploitation Logiciel qui gère la mémoire, les fichiers, les périphériques, les utilisateurs et l'exécution des programmes. OS/8, TOPS-10, RSX-11, UNIX et VMS correspondent à des objectifs et des machines différents.

Temps partagé Organisation dans laquelle le processeur passe rapidement d'une session à l'autre afin de donner à plusieurs utilisateurs une interaction presque simultanée.

Temps réel Fonctionnement dans lequel le système doit répondre avant une échéance imposée par le phénomène observé ou commandé. Le terme ne signifie pas simplement « très rapide ».

Terminal Dispositif de saisie et d'affichage relié à un ordinateur. Les premiers terminaux sont souvent des télétypes mécaniques ; les modèles vidéo comme le VT100 utilisent un écran et des séquences de contrôle.

VMS *Virtual Memory System*, système d'exploitation développé

avec VAX. Il devient plus tard OpenVMS et se distingue par son intégration, sa sécurité, ses outils et ses mécanismes de disponibilité.

Bibliographie

Alpha Architecture Committee. *Alpha Architecture Reference Manual*. Digital Press, 1992. isbn : 978-1-55558-098-8. doi : 10.1016/C2009-0-27261-8.

Bell, C. Gordon et al. « A New Architecture for Mini-Computers : The DEC PDP-11 ». In : *Proceedings of the Spring Joint Computer Conference*. 1970, p. 657-675. doi : 10.1145/1476936.1477037.

Ceruzzi, Paul E. *A History of Modern Computing*. 2e éd. Cambridge, Massachusetts : MIT Press, 2003. isbn : 978-0-262-53203-7.

Compaq Computer Corporation. *Quarterly Report : Acquisition and Divestiture Disclosures*. U.S. Securities et Exchange Commission, 1999. url : https://www.sec.gov/Archives/edgar/data/714154/000101540299001323/0001015402-99-001323-d1.html.

Computer History Museum. *Digital Equipment Corporation Founded*. 1957. url : https://www.computerhistory.org/timeline/1957/.

— *Invitation to a Presentation of a PDP Computer to Massachusetts Institute of Technology*. 1961. url : https://www.computerhistory.org/pdp-1/e282ffc5d7330dc5747327a33f06afaa/.

— *Invitation to the Presentation of a PDP-1 to MIT*. 1961. url : https://www.computerhistory.org/pdp-1/e282ffc5d7330dc5747327a33f06afaa/.

— *"Steal from Your Friends" DECUS Button*. 1965. url : https://www.computerhistory.org/revolution/mainframe-computers/7/172/697.

— *VAX-11/780*. 1978. url : https://www.computerhistory.org/timeline/1978/.

Computer History Museum. *VT100 Terminal*. 1978. URL : https://www.computerhistory.org/revolution/input-output/14/349/1697.
— *Networked Storage Systems Commercialized*. 1983. URL : https://www.computerhistory.org/storageengine/networked-storage-systems-commercialized/.
— *DEC Announces Alpha Chip Architecture*. 1992. URL : https://www.computerhistory.org/timeline/1992/.
— *Alan Kotok*. URL : https://computerhistory.org/profile/alan-kotok/.
— *Digital Equipment Corporation*. URL : https://www.computerhistory.org/revolution/minicomputers/11/335.
— *Ed Fredkin*. URL : https://www.computerhistory.org/pdp-1/ed-fredkin/.
— *Harlan Anderson*. URL : https://www.computerhistory.org/pdp-1/harlan-anderson/.
— *Harlan Anderson*. URL : https://computerhistory.org/profile/harlan-anderson/.
— *Ken Olsen*. URL : https://www.computerhistory.org/pdp-1/ken-olsen/.
— *Origins of the PDP-1*. URL : https://www.computerhistory.org/pdp-1/origins-of-the-pdp-1/.
— *PDP-1 Restoration Project*. URL : https://www.computerhistory.org/pdp-1/.
— *Scientific Applications on the PDP-1*. URL : https://www.computerhistory.org/pdp-1/.
— *Spacewar !* URL : https://www.computerhistory.org/pdp-1/spacewar/.
— *The PDP-8*. URL : https://www.computerhistory.org/revolution/minicomputers/11/331.
— *Timesharing on the PDP-1*. URL : https://www.computerhistory.org/pdp-1/timesharing/.
Corcoran, Elizabeth. *Computer Industry Has a New Giant : Compaq to Buy Digital*. The Washington Post. 27 jan. 1998. URL : https://www.washingtonpost.com/archive/politics/1998/01/27/computer-industry-has-a-new-giant/1dc0f99b-c7ba-4cbf-a87d-a609910269fd/.
Digital Equipment Corporation. *VT100 User Guide*. Digital Equipment Corporation. 1978. URL :

https://bitsavers.org/pdf/dec/terminal/vt100/EK-VT100-UG-001_VT100_User_Guide_Aug78.pdf.

— *36-bit Systems Timeline*. Digital Computing History Timeline. 1997. URL : https://archive.computerhistory.org/resources/text/DEC/dec.digital_%28DEC%29_timeline_1957-1997.102630354/36-bit.htm.

— *Networks Timeline*. Digital Computing History Timeline. 1997. URL : https://archive.computerhistory.org/resources/text/DEC/dec.digital_%28DEC%29_timeline_1957-1997.102630354/networks.htm.

Digital Equipment Corporation, Information Research Services. *DIGITAL Computing History Timeline, 1957–1997*. 1998. URL : https://archive.computerhistory.org/resources/text/DEC/dec.digital_%28DEC%29_timeline_1957-1997.102630354/timelinetext.htm.

Grainger College of Engineering. *Harlan E. Anderson*. University of Illinois Urbana-Champaign. URL : https://grainger.illinois.edu/alumni/hall-of-fame/harlan-anderson.

IBM. *The IBM 1401*. IBM Heritage. URL : https://www.ibm.com/history/1401.

— *The IBM System/360*. IBM Heritage. URL : https://www.ibm.com/history/system-360.

Levy, Steven. *Hackers : Heroes of the Computer Revolution*. Garden City, New York : Anchor Press/Doubleday, 1984.

Lott, Sara. *What the DEC ? ! ? Records of Minicomputer Giant Digital Equipment Corporation Open for Research at CHM*. 2017. URL : https://computerhistory.org/blog/what-the-dec-records-of-minicomputer-giant-digital-equipment-corporation-open-for-research-at-chm/.

MIT Lincoln Laboratory. *Interactive Supercomputing : From Whirlwind to TX-2*. Massachusetts Institute of Technology. URL : https://www.ll.mit.edu/about/history/interactive-supercomputing.

— *Lincoln Laboratory History*. Massachusetts Institute of Technology. URL : https://www.ll.mit.edu/about/history.

MIT News. *Kenneth Olsen, Computing Pioneer and Founder of DEC, Dies at 84*. 2011. URL : https://news.mit.edu/2011/obit-olsen.

Open SIMH Project. *Open SIMH*. URL : https://opensimh.org/.

Ritchie, Dennis M. *The Evolution of the Unix Time-Sharing System*. 1984. URL : https://www.bell-labs.com/usr/dmr/www/hist.html.

— *The Development of the C Language*. 1993. URL : https://www.bell-labs.com/usr/dmr/www/chist.html.

Ritchie, Dennis M. et Ken Thompson. « The UNIX Time-Sharing System ». In : *Communications of the ACM* 17.7 (1974), p. 365-375. DOI : 10.1145/361011.361061.

Schein, Edgar H. et al. *DEC Is Dead, Long Live DEC : The Lasting Legacy of Digital Equipment Corporation*. San Francisco : Berrett-Koehler, 2004. ISBN : 978-1-57675-305-7.

Strecker, William D. « VAX-11/780 : A Virtual Address Extension to the DEC PDP-11 Family ». In : *AFIPS Conference Proceedings* 47 (1978), p. 967-980. URL : https://gordonbell.azurewebsites.net/Computer_Engineering/00000431.htm.

Index général

12 bits, 47
16 bits, 65
32 bits, 85
64 bits, 113

21064, 113

adressage, 65
affichage graphique, 55
Alpha, 113
    famille, 129
AltaVista, 113
American Research and Development Corporation, 17
Harlan Anderson, ix, 9
    départ, 107
ARPANET, 61
assembleur, 41

bande magnétique, 55
BBN, *voir* Bolt Beranek and Newman
Bell Laboratories, 71
Gordon Bell, 47, 65
Bolt Beranek and Newman, 29, 31
byte, 65

capital-risque, 17
cartes perforées, 1
Compaq Computer Corporation, 117, 119
compatibilité, 85
compatibilité logicielle, 89
conservation, 121
convertisseur analogique-numérique, 55
crise de Digital Equipment Corporation, 117
Dave Cutler, 89

Data General, 65
DCL, 89
DDT, 41
Edson de Castro, 47
DECmate, 99
DECnet, 93
DECsystem-10, 61
DECSYSTEM-20, 61
DECtape, 55
Digital Equipment Computer Users Society, 77, 79
Jack Dennis, 35, 36
Digital Building Blocks, 23
Digital Equipment Corporation, ix

chronologie, 125
culture, 107
fondation, 17
héritage, 121
universités, 35
Digital Press, 77
Digital UNIX, 113
disque magnétique, 55
documentation technique, 23, 77
Georges Doriot, 17

enseignement, 35
Ethernet, 93

Ed Fredkin, 29, 31

glossaire, 133
Ben Gurley, 29

hacker, 41
HSC50, 93

IBM, 1
IBM PC, 99
ILLIAC I, 9
informatique interactive, 13, 29
ingénieurs, 107
instrumentation, 55
Intel, 117
intelligence artificielle, 61

Brian Kernighan, 71, 75
Alan Kotok, 41, 42, 61

Laboratory Modules, 23
langage C, 71
licenciements, 117
logiciel, 77

Maynard, 17
mini-ordinateur, 47
MIT, 9
Research Laboratory for Electronics, 35
MIT Lincoln Laboratory, 13
modules logiques, 23
Multics, 71
mémoire virtuelle, 85
mémoire à tores, 13

Nova, 65

OEM, 47, 77
Ken Olsen, ix, 9
départ, 117
management, 107
OpenVMS, 89, 113
ordinateur central, 1
ordinateur personnel, 99
organisation matricielle, 107
OS/8, 77

Robert Palmer, 117, 118
PDP
familles, 129
PDP-1, 29
don au MIT, 35
PDP-10, 61
PDP-11, 65
UNIX, 71
PDP-5, 47
PDP-6, 61
PDP-7, 71
PDP-8, 47
portabilité, 71
PRISM, 113
product line manager, 107
production, 23

Professional 300, 99
périphériques, 55

rachat de Digital Equipment Corporation, 117
Rainbow 100, 99
Research Laboratory for Electronics, 35
restructuration, 117
RISC, 113
Dennis Ritchie, 71, 72
RSTS/E, 77
RSX-11, 77
RSX-11M, 89
RT-11, 77
Steve Russell, 41, 43
réseau local, 93

SAGE, 13
Peter Samson, 41, 43
service de terrain, 77
SIMH, 121
simulation, 121
Spacewar, 41
Star, 85
station de travail, 99
stockage partagé, 93
William Strecker, 85, 86
System Modules, 23
système d'exploitation, 89

Tech Model Railroad Club, 41
temps partagé, 41, 61
terminal, 93
Ken Thompson, 71, 72
TMRC, *voir* Tech Model Railroad Club
TOPS-10, 61
TOPS-20, 61
traitement par lots, 1
TX-0, 13
TX-2, 13
Type 30, 29
télétype, 55

ULTRIX, 99
Unibus, 65
University of Illinois, 9
UNIX, 71, 99
utilisateurs, 107

VAX, 85
  famille, 129
VAX-11/780, 85
VAXcluster, 93
VAXstation, 99
VMS, 89
VT100, 93

Whirlwind, 13
Windows NT, 113
Works Committee, 107