UN LANGAGE POUR TERMINAL GRAPHIQUE INTELLIGENT

J. RAYMOND, D.K. BANERJI ET J.C. GAVREL

DÉPARTEMENT D'INFORMATIQUE
UNIVERSITÉ D'OTTAWA

ABRÉGÉ

Les terminaux graphiques bon marché, ont une réponse assez faible à cause de leur manque de capacité de traitement et de la vitesse lente du bien avec l'ordinateur central. Chaque modification à l'image faite par l'ordinateur est envoyée sur la ligne et parfois même toute l'image dans le cas des tubes à mémoire. L'utilisation d'un microprocesseur local lié à l'ordinateur central par la ligne habituelle et relié au terminal par une interface rapide offre une solution à ce problème. Il peut contenir une copie de l'image pour le traitement de beaucoup de modifications en local. A l'université d'Ottawa, le microprocesseur communique avec l'ordinateur central dans un langage qui simule les instructions d'un terminal graphique très evolué. Cet artice decrit les détails de ce langage.

A LANGUAGE FOR AN INTELLIGENT GRAPHIC TERMINAL

J. RAYMOND, D.K. BANERJI AND J.C. GAVREL

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF OTTAWA

ABSTRACT

Low cost graphic terminals have minimal processing power and the common telephone link to the host computer results in relatively poor response time as every modification of the image must be done by the host computer and then sent to the terminal. In case of storage tube type displays, the entire image must be redrawn. One solution to this problem is to use a local processor linked to the host computer via the telephone link and connected to the terminal via a high speed interface. The local processor can contain a copy of the image for local processing of most modifications. Such a configuration is being used at the University of Ottawa. The host machine communicates with the local processor using a language which simulates a very versatile graphic terminal instruction set; this paper describes the details of this language.

Un langage pour terminal graphique programmable

1. INTRODUCTION

La liaison d'un terminal graphique avec un ordinateur central est souvent réalisée par une ligne téléphonique à basse vitesse. La quantité d'information nécessaire pour les tracés graphiques étant considérable, il est avantageux d'utiliser un terminal programmable, ou encore d'utiliser un mini-ordinateur comme préprocesseur entre l'ordinateur central et le terminal. La liaison entre le terminal et le mini-ordinateur se fait alors à grande vitesse (Figure 1).

Les instructions reçues de l'ordinateur central sont écrites dans un langage machine qui est interpreté par le terminal ou le mini-ordinateur. Ce langage permet de faciliter la communication car il contient des instructions simples réalisant des tracés complexes (cercles, transformations) afin de minimiser le transfert d'informations sur la ligne à basse vitesse.

Le système graphique ainsi réalisé est géré par un logiciel graphique dont certains composants sont dans l'ordinateur central et d'autres dans la mémoire du mini-ordinateur. La figure 2 donne la liste de ces composants ainsi que leur place dans le système. Les routines résidentes dans le mini-ordinateur sont chargées soit au début de session à l'aide d'un programme amorce, soit en cours de traitement si nécessaire par des instructions spéciales du langage. Les instructions sont composées d'octets dont le nombre est variable et la notation utilisée est l'hexadécimal.

2. LA STRUCTURE DU LANGAGE

Les instructions se classent dans deux catégories principales: les instructions de tracés d'image (codes 80 à FF) et les instructions systèmes (codes 00 à 7F). Dans chacune de ces catégories on distingue les fonctions suivantes:

1.	Fonctions système	(0x et 8x)
2.	Fonctions de segmentation	(1x et 9x)
3.	Fonctions d'entrée	(2x)
4.	Fonctions de sortie	(3x)
5.	Primitives graphiques	(4x et Cx)
6.	Transformations	(5x et Dx)
7.	Modifications	(6x et Ex)
8.	Codes spéciaux	(7x et Fx)

La figure 3 liste les codes opérations avec les différentes fonctions qui sont supportées. Le terminal peut implémenter toutes ou parties de ces fonctions selon la complexité désirée.

Dans tous les cas, deux registres sont simulés par l'implémentation:

(1) Le Registre Masque d'interruptions (RMI) qui indique quelles unités d'entrées sorties sont actives à un instant donné (c'est-à-dire qui peuvent générer une interruption). Ce registre est chargé par l'instruction X'61'.

(2) Le Registre d'intensité et de Mode (RIM) qui indique jusqu'à 8 niveaux d'intensité lumineuse et les modes de tracés continu, interrompu, semi-interrompu et pointillé. Ce registre indique également le clignotement. Il est chargé par l'instruction X'E1'.

3. FONCTIONS SYSTEME

Les fonctions systèmes permettent la gestion du programme Image (PIF) dont une copie est gardée en mémoire pour pseudo régération (dans le cas des tubes à mémoire). Les fonctions classiques sont: segmentation de l'image en sous images et sous programmes, création d'images parallèles, envoi de programmes système, gestion des entrées-sorties et communication avec le programme image.

De nouvelles fonctions ont été ajoutées comme par exemple:

- Tracé de message (par contraste avec tracé de texte)
- Création de fonctions très synthétiques: encrage, trainage ligne élastique, potentiomètre lumineux, menu, etc.
- Interruption de l'ordinateur central si une entrée non prévue par le programme est faite par l'opérateur.

4. FONCTIONS IMAGE

Les fonctions image rassemblent les primitives classiques que l'on retrouve sur presque tous les terminaux comme les tracés de droites, points et textes, en absolu et un relatif, les instructions de branchement et de sous-programmation, la synchronisation avec l'unité centrale, les concaténations de matrices pour les transformations. Certaines fonctions moins répandues sont incluses comme les tracés de cercles par exemple.

Chaque fonction est modulaire et peut ou non être implémentée. De plus deux séries de codes opérations sont prévus pour des fonctions spéciales propres à une implémentation particulière.

5. CONCLUSIONS

Ce langage a été implémenté dans un premier cas sur un micro-ordinateur (Intel 8008) relié à un Tektronix 4013 l'ensemble est lié à l'ordinateur central (IBM 360) par une ligne téléphonique (300 bauds). La taille de la mémoire du micro-ordinateur ne permet pas

une implémentation complète et un disque souple est à l'étude pour faciliter cette tâche. Une autre implémentation en cours consiste à utiliser le terminal GT40 de DEC où il semble possible de réaliser un système tridimensionnel. Ce langage de communication est suffisament souple pour couvrir tous les terminaux programmables actuels. Un projet à l'université d'Ottawa consistera dès l'an prochain à microprogrammer l'interpreteur de façon à étudier une fabrication possible d'un terminal ayant ces caractéristiques.

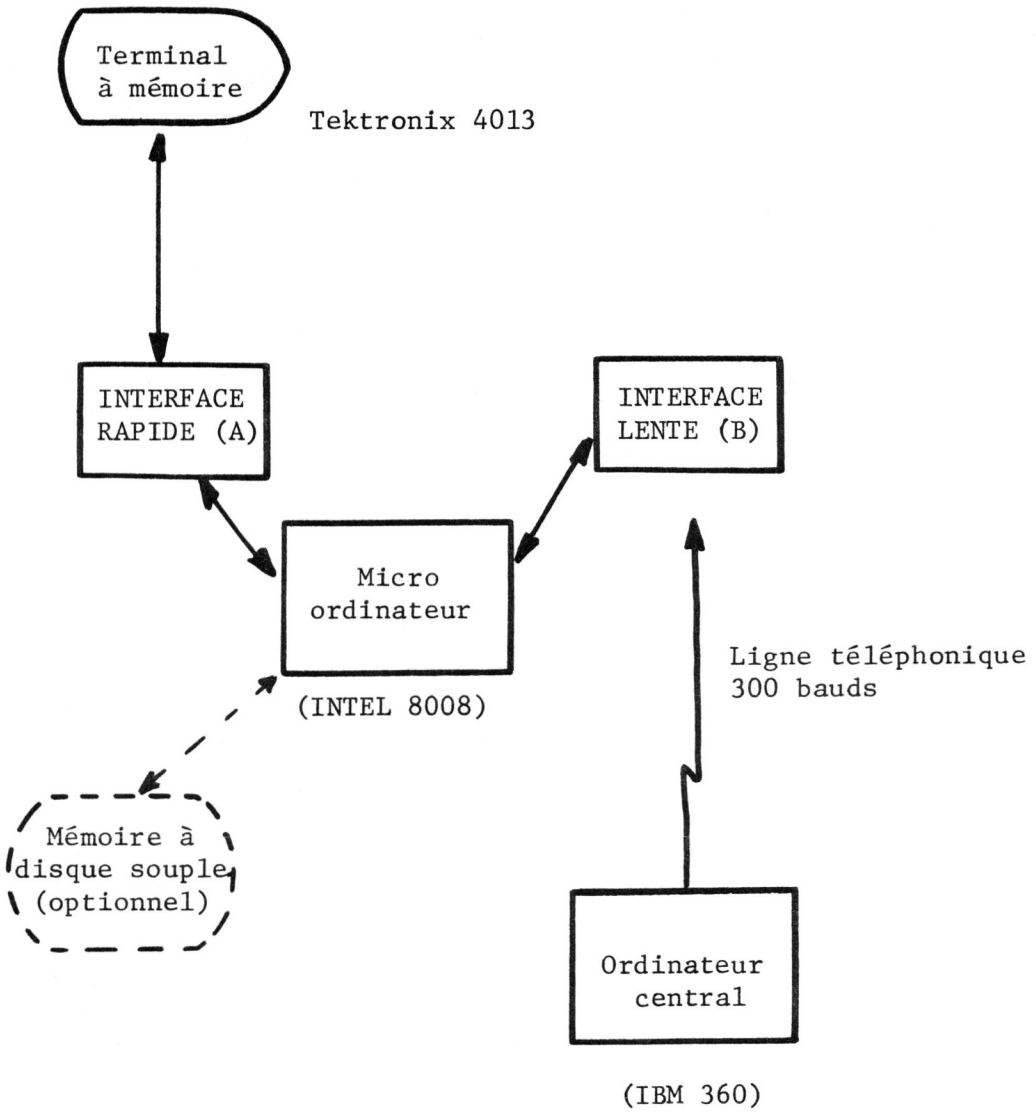

Figure 1 Structure du terminal

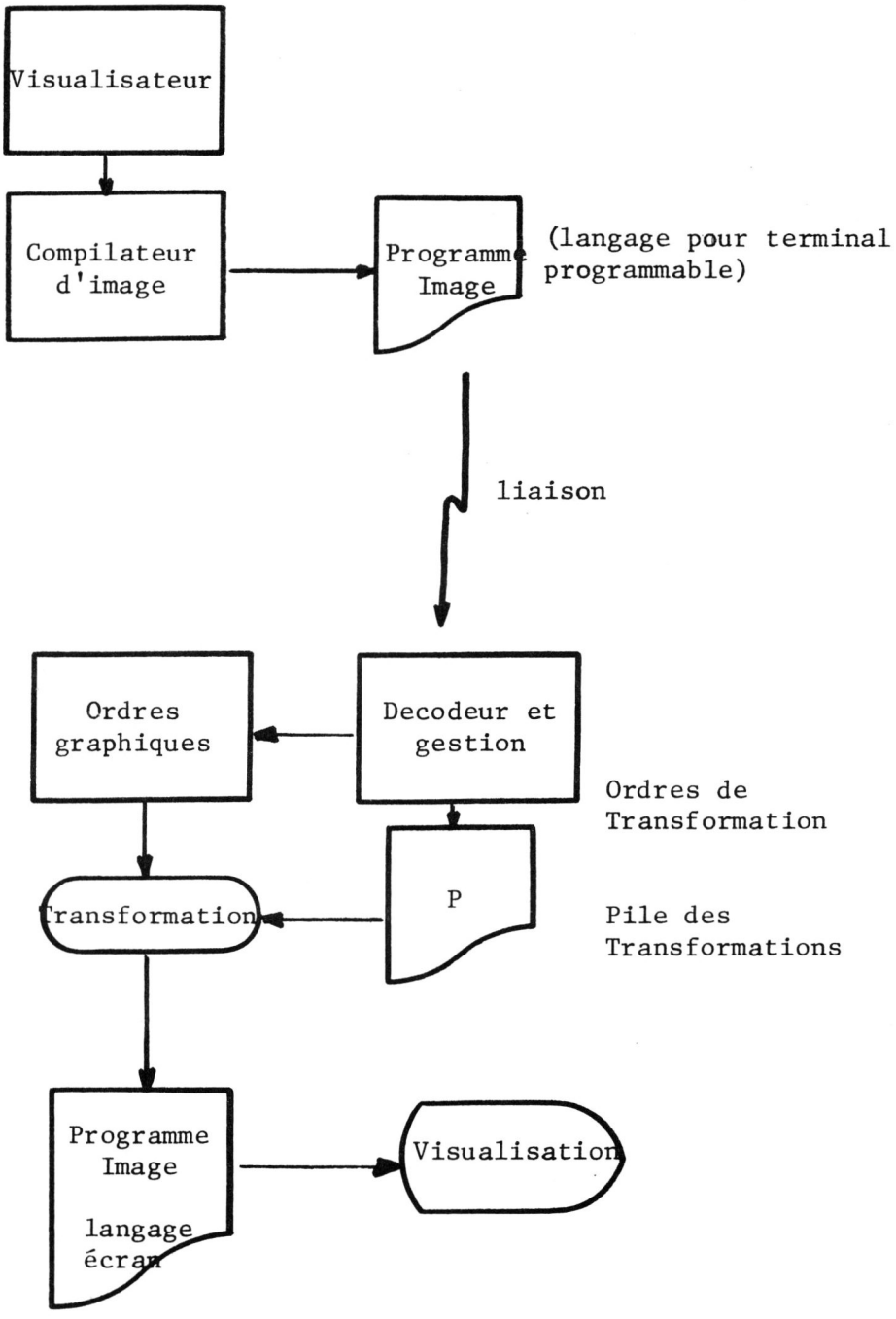

Figure 2 Le logiciel graphique

Code	Fonction
00	NOOP
01	Début de session
02	Début message
03	Fin message
04	Fin de session
07	Signal auditif
0C	Efface (Reinit)
0E	Début de programme
0F	Fin de programme
12	Début image
13	Fin image
14	Début bloc
15	Fin bloc
16	Début sous programme image
17	Fin SP image
21	Lecture clavier de fonctions
22	clavier alphanumérique
23	photostyle
24	Curseur graphique
25	Curseur alphanumérique
26	Position tablette
29-2B	Lire les registres X Y Z
31-3F	Interruptions d'éléments d'entrée
41	Routine d'encrage
42	Trainage
43	Elastique
44	Dessine une grille
45	Potentiomètre lumineux
46	Vernier lumineux
51	Charge les dimensions viseur et cadre
52	Transforme le point X Y Z
61	Charge le RMI
62	Trace le Programme Image
63	Efface le Programme Image
64	Trace un bloc
65	Efface un bloc
66	Ajoute un bloc
67	Enlève un bloc
80	Interrompt l'unité centrale du mini
81	Synchronisation avec la ligne

Figure 3 Codes opérations

Code	Fonctions
91	Noop graphique
92	Branchement
96	Appel SP
97	Retour SP
C0	Fin d'instruction graphique
C1	Position absolue
C2	Point absolue
C3	Droite absolue
C4	Cercle absolue
C7	Caractère
C9	Position relatif
CA	Point relatif
CB	Droite relatif
CC	Cercle relatif (Centre)
CF	Caractère (2ème alphabet)
D0	Transformation identique
D1	Translation
D2	Echelle
D3	Rotation
D7	Restauration de la transformation précédente
E1	Charge le RMI

Figure 3 Codes opérations
(continué)