

UN SYSTEME INTERACTIF POUR L'EDITION DE SIGNAUX NUMERIQUES

J. M. BENEDI (*), F. CASACUBERTA (**), E. VIDAL (**)

UNIVERSIDAD DE VALENCIA

Dans le traitement de signaux numériques il est nécessaire d'avoir la possibilité de manipuler interactivement des données numériques par ordinateur, ce qui n'est pas convenablement supporté par les systèmes d'exploitation et/ou les environnements de programmation actuels.

Dans ce travail, on expose le dessin et implementation d'un système informatique general pour son application à l'édition interactive de signaux (EDS). L'EDS est composé par deux étapes, une première de déclaration d'objets et une deuxième de spécification d'actions sur les objets. Les actions peuvent être: acquérir, entendre, représenter graphiquement, etc. des signaux numériques; ainsi que les opérations d'effacer, échanger, insérer, etc. sur zones spécifiques du signal.

Nous présentons l'état actuel du EDS ainsi que ses nouvelles améliorations, pour son utilisation dans le traitement des signaux numériques.

Keywords: SPEECH PROCESSING, NUMERICAL SIGNAL PROCESSING, INTERACTIVE MODE.

1. INTRODUCTION.

Dans les expériences de Reconnaissance et de Synthèse de la Parole, ainsi comme dans quelques domaines de la Phonétique, de la Psycho-acoustique, de la Medicine, etc...., on a besoin d'une phase dans laquelle, le signal -- soit transformé en signal électrique, digitalisé et stocké sur support magnétique pour son traitement par ordinateur ("acquisition") /9/.

Il est fréquent dans toutes ces cas, de sentir le besoin d'observer graphiquement et/ou acoustiquement le signal acquis, ainsi que d'effectuer des traitements manuels sur ceux-ci, dont: la séparation en items plus petits, l'introduction de marques spéciales pour indiquer instants spécifiques du signal, etc...

En sus des applications indiquées, on peut mentionner, dans le domaine de la Reconnaissance et de la Synthèse de la Parole, quelques procédures où les tâches d'édition sont indispensables:

- Création de dictionnaires d'éléments -- acoustiques (synthèse).

- Segmentation et/ou marquage manuel pour la comparaison avec les résultats automatiques (Reconnaissance).
- Vérification des résultats en systèmes de Synthèse.
- Vérification des résultats en segmentation et dans d'autres phases de reconnaissance.

Les systèmes d'exploitation et/ou les environnements de programmation actuels sont dotés de nombreux outils pour la manipulation interactive de données de types précis, tels que des textes alphanumériques, des sentences de langages de programmation, etc... Cependant, dans un environnement de traitement digital des signaux il est toujours nécessaire de réaliser des manipulations, comme celles déjà mentionnées, sur un type particulier de données (signaux) qui ne sont pas convenablement supportées par les outils standard. Ainsi, pour réaliser ces tâches d'édition, on construit usuellement des systèmes interactifs "ad-hoc" pour chaque observation ou traitement,

- J.M. Benedí (*) Universidad de Valencia - Dep. de Electrónica e Informática - Fac. de Físicas.
E. Vidal (**)
- F. Casacuberta (**) Centro de Informática de la Univ. de Valencia
Dr. Moliner, s/n. - Valencia
- Article rebut 1^o octobre de 1985.

ce qui conduit finalement a un grand nombre de systèmes essentiellement redondants.

Récemment, un système général d'édition interactif de signal (E.D.S. /3/) a été développé. Il utilise des techniques informatiques pour résoudre les problèmes fondamentaux que pose l'édition et dont le but est de fournir un outil de prestations analogues à celles trouvées normalement dans les "éditeurs de texte" informatiques.

L'Editeur de Signal (EDS) à été conçu autour d'un noyau avec capacité de croissance moduler, ce qui permettra une adaptation aisée dès l'apparition de nouvelles nécessités.

Actuellement, le prototype et la première version du EDS ont été complètement développés, vérifiés et utilisés, ce qui a conduit à la nécessité des nouvelles prestations qui ont, en partie, conduit à une mise au jour du EDS et qui sont la base d'une nouvelle version, substantiellement améliorée.

Dans ce travail, sont exposés les principaux problèmes que l'on prétend résoudre avec le système propose, conjointement avec les conséquences que leur solution suppose dans le système implementé.

2. DESCRIPTION DU SYSTEME D'EDITION DE SIGNAL (E.D.S.).

2.1. CONCEPTS BASIQUES DU EDS.

Les données (objects), sur lesquels on applique les traitements définis dans le langage d'édition, sont des représentations du signal. La représentation la plus directe est celle du domaine temporel, où le signal est représenté par une séquence numérique obtenue parmi un échantillonnage à intervalles de temps réguliers. L'EDS permet, aussi, quelques représentations paramétriques du signal /6/. Dans la première version de ce système, on a introduit l'enveloppe et la mélodie, qui sont orientés vers la synthèse de la parole.

La traile de l'information à manipuler exige l'utilisation de la mémoire centrale et de l'auxiliaire. L'EDS permet ou interdit quelques traitements selon l'ubiquité du --

signal; alors, on introduit deux types de données: REGISTRE et FICHIER.

Les objets du langage d'édition ont été définis comme des séquences numériques d'échantillons, celles-ci peuvent être: SIGNAL, ENVELOPPE et MELODIE (dans la version présente).

Pour repérer les différents éléments de chaque type susceptibles de manipulation à travers des instructions d'édition, on a introduit le concept classique de variable.

Dans beaucoup d'opérations d'édition du signal l'accès à certaines zones du signal peut être nécessaire, dans une variable spécifiée. Les concepts introduits pour repérer ces zones sont: POSITION et SEGMENT.

SEGMENT (X n, m) Où X est une variable du type REGISTRE ou FICHIER et n, m sont deux paramètres qui déterminent le principe et la fin du segment.

POSITION (X n) Où X est une variable du type REGISTRE ou FICHIER et n est un paramètre qui détermine la position.

Pour spécifier les pointeurs de segment et de position, ainsi que pour quelques autres éléments du langage d'édition, l'EDS permet l'usage des expressions arithmétiques, d'une façon toute classique, comme on peut voir sur la figure 1.

2.2. LE TRADUCTEUR.

Pour arriver a l'implementation du EDS, deux phases préalables ont été nécessaires:

- La définition d'un langage d'édition.
- La construction d'un traducteur pour le langage.

L'ensemble des instructions du langage d'édition est présenté grâce à la grammaire de contexte libre G(SEANCE), en notation BNF, dans la figure 1.

Il est clair que la caractéristique interactive de L'EDS suggère un interprète pour le traducteur. Un interprète exécute les procédures associés aux instructions, spécifiées par une codification interne obtenue par une

```

<Séance> ::= <Déclaration> t <Séance> / <Commande> t <Séance> / FIN
<Déclaration> ::= REGISTRE <Qualificateur> ident <Type>
                FICHER <Qualificateur> ident <Type>
<Qualificateur> ::= <Exp> / SL / CF / WA / WC / WG / ^
<Type> ::= SIGNAL <Rtype> / ENVELOPPE <Rtype> / MELODIE <Rtype>
<Rtype> ::= <Exp> / ^
<Commande> ::= EFFACER <Seg> / TRANSPOSER <Seg> / ACQUERIR <Seg> /
                ENTENDRE <Paramètre> <Seg> / ETAT <Paramètre> /
                SECOURS <Paramètre> / SUPERPOSER <Segpos> /
                ECHANGER <Segpos> / INSERER <Segpos> /
                INICIALISER <Rinicialiser> /
                GRAFIQUE const <Exp> <Seg>
<Paramètre> ::= const / ^
<Rinicialiser> ::= const <Exp> / ident
<Seg> ::= ident <Exp> , <Exp>
<Segpos> ::= ident <Exp> , <Exp> ident <Exp>
<Exp> ::= <Opéré> <Rexp>
<Rexp> ::= <Opérateur> <Exp> / ^
<Opérateur> ::= + / - / * / /
<Opéré> ::= const / P / F

```

Fig 1.- La grammaire de contexte libre G(SEANCE), en notation BNF.

```

<Symbole> ::= l <Rlettre> / d <Rdigit> / s / t
<Rlettre> ::= l <Rlettre> / d <Rlettre>
<Rdigit> ::= d <Rdigit> / . <Rdréel>
<Rdréel> ::= d <Rdréel>

```

Fig 2.- La grammaire régulière G(SYMBOLE) de spécification des symboles du langage.

```

ANALYSE (X)
si X ∈ T alors
    si SIM = X alors PRENDRE LE SYMBOLE SUIVANT
        sinon ERREUR
    sinon CHOISIR D'UNE REGLE X ::= A1 ... An ET
        ANALYSE (A1); ...; ANALYSE (An)

```

Fig 3.- Algorithme d'analyse syntaxique.

étape de traduction des instructions d'édition. L'étape de traduction se compose d'un préprocès et d'une analyse syntaxique. Le préprocès accepte les phrases d'un langage source et construit les unités syntaxiques, d'après lesquelles ces symboles sont codifiés. De façon toute classique, le préprocès a été implémenté comme une sousroutine appelée par l'analyseur chaque fois que celui-ci a besoin d'un symbole /5/.

Les symboles du langage sont décrits par la grammaire régulière G(SYMBOL) figure 2. Où l'étiquette l représente tous les éléments de l'ensemble des lettres { A,B,...,Z, @ ,.,",=,- }. Les étiquettes d, s, t, représentent, respectivement, les ensembles des chiffres {0,1,...,9} , séparateurs {,,(,),+,-,*,/,} et les finalisateurs {;,NL,FF,RT} .

L'analyse syntaxique est une des étapes les plus importantes d'un traducteur. L'objectif principal de l'analyseur est de reconnaître si une chaîne source appartient ou non au langage généré par une grammaire déterminée, tout en reconstruisant le chemin qui conduit à sa génération /1/. L'analyse Syntaxique produit une codification qui est acceptée par l'étape suivante de l'interprète: les modules de traitements.

La technique d'analyse utilisée est descendante de gauche à droite, déterministe avec lecture d'un seul symbole en avant (LL(1)). L'analyse descendante est fondée sur l'algorithme qui, depuis l'axiome, génère la phrase source, figure 3. Il est intéressant de présenter la syntaxe du langage au moyen d'un graphe syntaxique /8/, figure 4.

La méthode d'analyse descendante est sujette à deux problèmes importants: la possible non finition de cet algorithme et le possible indétérminisme de la grammaire. On peut démontrer que la grammaire décrite G(SEANCE) ne présente pas de recursivité à gauche, ce qui garanti la non existence de calculs infinis dans l'algorithme d'analyse, ainsi que l'absence de problèmes d'indétérminisme /3/.

3. LE LANGAGE D'ÉDITION.

Les instructions du langage d'édition font autant référence, à la définition des objets, qu'à la spécification des actions sur ces objets. Par conséquent, les instructions d'édition sont classées en déclaratives et exécutives.

3.1. INSTRUCTIONS DÉCLARATIVES.

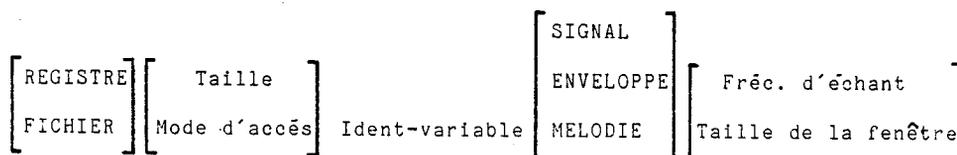
La description d'un langage comportera toujours la définition des objets. Tout objet doit être déclaré une fois et une seule. Une déclaration a toujours un double but: indiquer qu'un nouvel objet apparaît, et donner les caractéristiques de cet objet et notamment celles que nous avons déjà citées, mais aussi l'ensemble fonctionnel de l'objet /10/.

De manière général, la définition d'un objet comportera: un nom ou identificateur et un type. Les types définis sont: REGISTRE et FICHER. Associé à eux a été défini le paramètre QUALIFICATEUR; qui pour les variables REGISTRE, il indique la taille initiale et pour les variables FICHER, il représente le mode d'accès. Les objets du langage d'édition ont été définis comme des séquences numériques d'échantillons, où les éléments de ces séquences peuvent être: SIGNAL, ENVELOPPE et MELODIE. Associé à eux a été défini la constante PARAMETRE; qui pour les séquences numériques du type SIGNAL, elle indique la fréquence d'échantillonnage et pour ENVELOPPE et MELODIE, elle représente la taille de la fenêtre d'analyse.

On pourrait définir le type séquence numérique comme un produit cartésien entre la réunion des types:

Séquence numérique = (REGISTRE U FICHER) X (SIGNAL U ENVELOPPE U MELODIE).

La syntaxe des instructions déclaratives est la suivante (les crochets indiquent optionnalité):



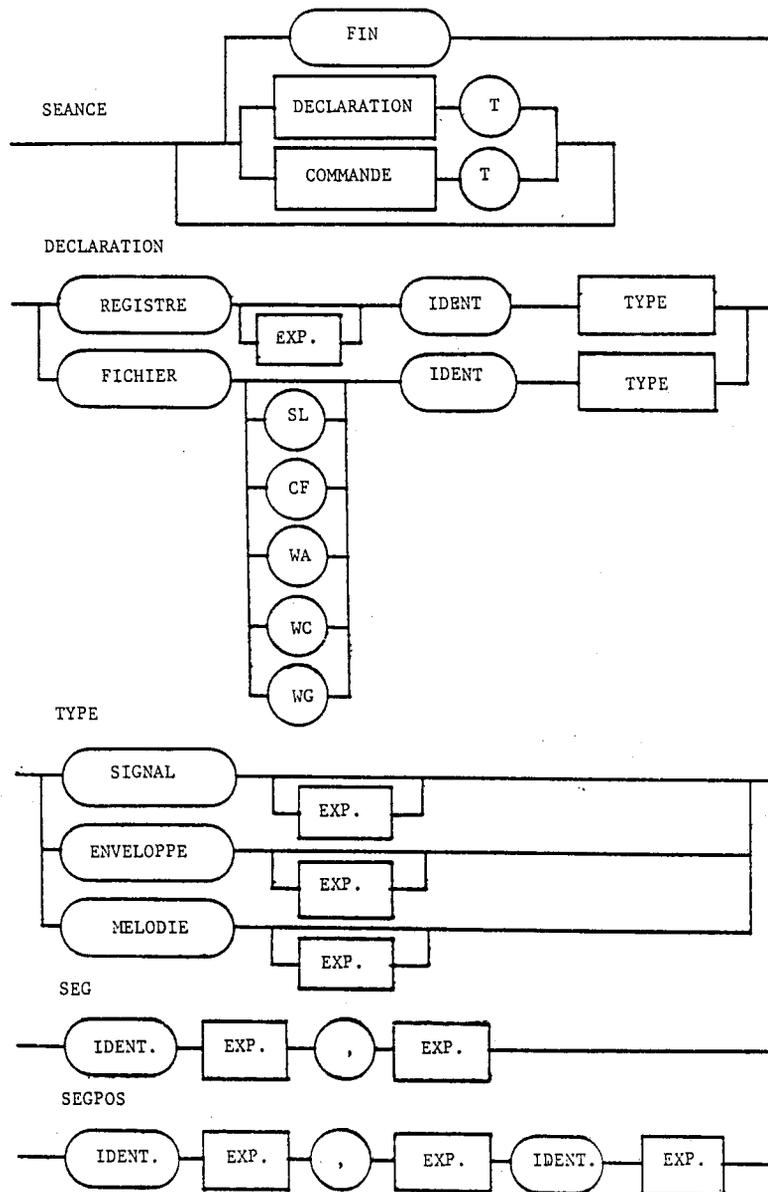


Fig 4.a.- Graphe syntaxique associé au langage. Phase Déclarative.

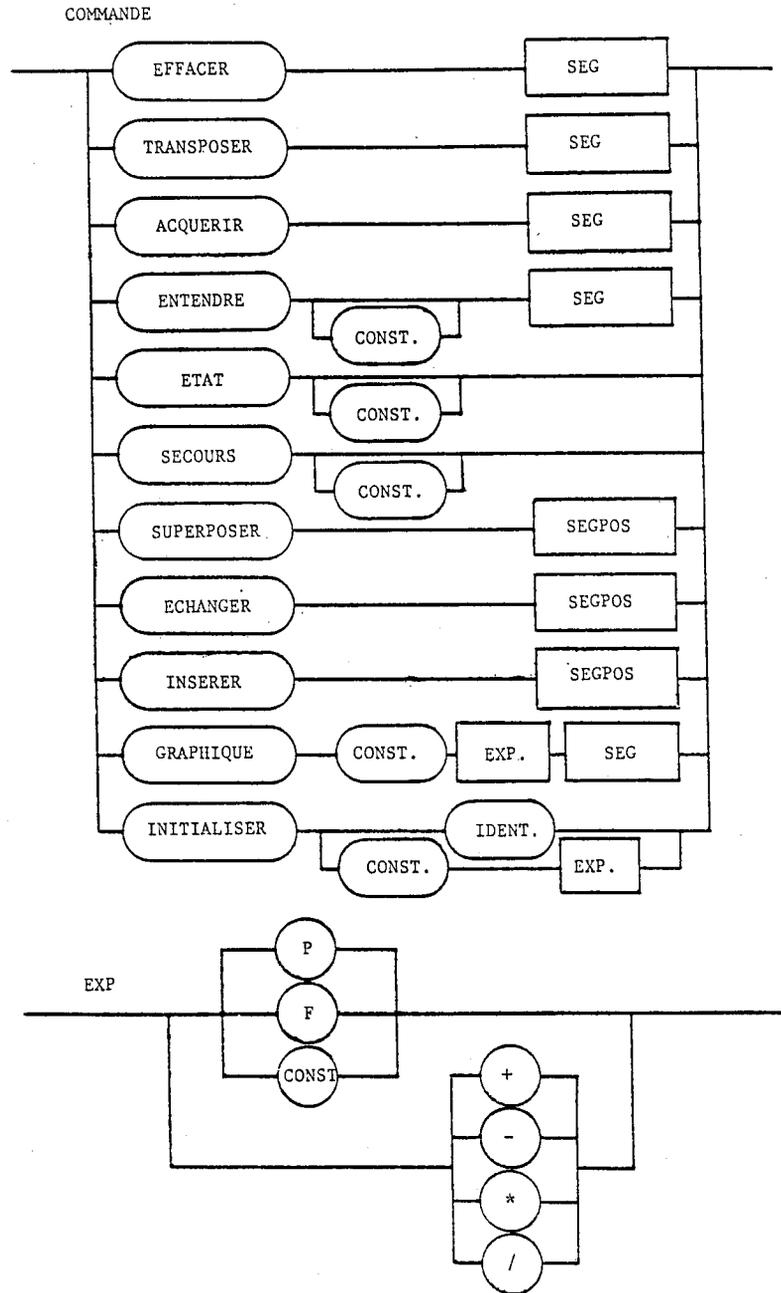


Fig 4.b.- Graphe syntaxique associ  au langage. Phase Ex cutive.

Exemple.

REGISTRE 1000 X SIGNAL 12800.

On fournit une d6notation en pseudo-PASCAL (autoexplicatif) que por l'exemple serait:

```

type  Echant : SIGNAL 1280
var   X      : REGISTRE 1000 Echant 6 s6-
           quence num6rique
      1X, X1 : Echant et
      1X:= 1; X1 := 1000;
    
```

Associ6 6 cette phase, une s6rie d'actions s'ex6cutent selon le type de la variable. - Pour variables FICHIER, les fichiers sont - ouverts ou ferm6s selon le param6tre QUALIFICATEUR. Et pour variables REGISTRE, on r6serve de la m6moire selon la taille d6clar6e dans la constante PARAMETRE. Int6rieurement, une instruction d6clarative conduit 6 une -- consultation et 6 une actualisation du tableau de variables.

3.2. INSTRUCTIONS EX6CUTIVES.

Tous les traitements et actions qu'on r6alise sur les objets sont formalis6s sous la forme d'instructions ex6cutives qui appartiennent au langage d'6dition. Cet ensemble d'instructions a 6t6 divis6 en trois sous-ensembles: de Manipulation, de Repr6sentation Acquisition et d'Information.

Il est n6cessaire, dans les instructions de manipulation et de repr6sentation acquisition, de poss6der des outils qui permettent rep6rer des s6quences num6riques plus localis6es `a l'int6rieur des objets d6clar6s. C'est pour cela, qu'ont 6t6 d6finis les op6rateurs: segment et position.

Segment.- (X n, m) O6 X est une variable du type REGISTRE ou FICHIER et n, m sont deux param6tres qui d6terminent le principe et la fin du segment; qui se d6finit comme tout -- 6l6ment de la s6quence num6rique nomm6e par X, contenus entre les pointeurs initial et final n, m:

X_i $i : n, n+1, \dots, m$
 Segment : S6quence num6rique \rightarrow S6quence num6rique

Exemple.

X 1, 100

Signifie ou d6note:

$$\begin{aligned} \uparrow X (1,100) &= 1 \\ X (1,100)\uparrow &= 100 \end{aligned}$$

Position.- (X r) O6 X est une variable du type REGISTRE ou FICHIER et r est un param6tre qui d6termine la position, qui se d6finit comme l'6l6ment de X indique par le pointeur de position r:

$$X_i \quad i = r$$

Position: S6quence num6rique \rightarrow S6quence num6rique

Exemple.

X 17

Signifie ou d6note:

$$\uparrow X (17) = X (17)\uparrow = 17$$

3.2.1. INSTRUCTIONS DE MANIPULATION.

La syntaxe g6n6rale de ces instructions est:

EFFACER	Ident-variable Segment [(Position)]
TRANSPOSER	
SUPERPOSER	
ECHANGER	
INSERER	

- EFFACER.- Efface un segment d'une taille donn6e.

EFFACER X n, m.

Signifie ou d6note.

$$\uparrow X (n, m) = X (n, m)\uparrow = \emptyset$$

Note.- L'objet vide est repr6sent6 par

$$\uparrow X = X\uparrow = \emptyset.$$

- TRANSPOSER.- Transpose un segment d'une taille donn6e.

TRANSPOSER X n, m.

Signifie ou d6note.

$X_i := X_{(m-i)+1}$ $i:n, n+1, \dots, m$ DIV n. (O6 DIV est l'op6rateur division entier, et :=: est l'op6rateur d'6change mutuel).

Questi6 - V. 9, n.° 3 (setembre 1985)

- SUPERPOSER.- Superpose un segment d'une --
taille donnée à partir d'une position.

SUPERPOSER X n, m Y r

Signifie ou dénote.

$Y_i = X_j$ $i : r, r+1, \dots, r+(m-n)+1$
 $j : n, n+1, \dots, m$

- ECHANGER.- Echange un segment de taille
donnée par un autre, de la même taille,
à partir d'une position.

ECHANGER X n, m Y r

Signifie ou dénote.

$Y_i := X_j$ $i : r, r+1, \dots, r+(m-n)+1$
 $j : n, n+1, \dots, m$

- INSERER.- Insère un segment d'une taille
donnée à partir d'une position.

INSERER X n, m Y r

Signifie ou dénote.

$(Y_i = X_j$ $i:r+(m-n), r+(m-n)+1, \dots, r+2(m-n)+1;$
 $j:r, r+1, \dots, r+(m-n)+1$ ^

$(Y_k = X_s$ $k:r, r+1, \dots, r+(m-n)+1;$
 $s:n, n+1, \dots, m)$

Une conséquence de cette instruction est un déplacement de l'information égal à la taille du segment à insérer, comme on peut observer dans la figure 5.

3.2.2. INSTRUCTIONS DE REPRÉSENTATION ACQUISITION.

Les instructions de ce groupe sont:

- ENTENDRE.- Entendre un segment un nombre
optionel de fois, déterminé par la constante (E).

ENTENDRE [(E)] segment.

- ACQUERIR.- Acquérir un segment d'échantillons déterminé par la taille du (segment).

ACQUERIR segment.

- INITIALISER.- Initialise le dispositif graphique "SCOPE" à un nombre de traces -- (C) et à un nombre de points par trace (E).

INITIALISER C E.

Ou obtenir une copie permanente ("save") sur le fichier indiqué par (Nom-fichier).

INITIALISER Nom-fichier.

- GRAPHIQUE. Représente un segment dans la trace indiquée par (C), avec une constante d'échelle donnée par (E).

GRAPHIQUE C E Segment.

3.2.3. INSTRUCTIONS D'INFORMATION.

Les instructions de ce groupe sont:

- ETAT.- Indique l'état d'occupation des variables.

ETAT

- SECOURS.- Fourni toute l'information générale nécessaire pour l'utilisation de l'EDS. Pour obtenir des informations -- plus détaillées on peut utiliser la constante (N).

SECOURS [(N)].

4. REALISATION.

L'Editeur du Signal a été écrit basiquement en PASCAL, et pour quelques unes des procédures de l'ensemble d'instructions de représentation acquisition on a employé le langage Assembleur. Le système est composé de cinq "modules-overlay" à compilation indépendante.

4.1. TRAITEMENT DES ERREURS.

L'EDS détecte les erreurs dans la construction de la phrase ainsi que pendant leur exécution. Le traitement d'erreurs syntaxiques est spécialement important, car ses effets ne sont pas locaux et ils effectent les phases suivantes de l'interprète. La détection d'erreurs, associée à cette version du EDS, vérifie les prémisses suivantes:

- Aucune instruction d'entrée ne doit faire perdre le contrôle au système.

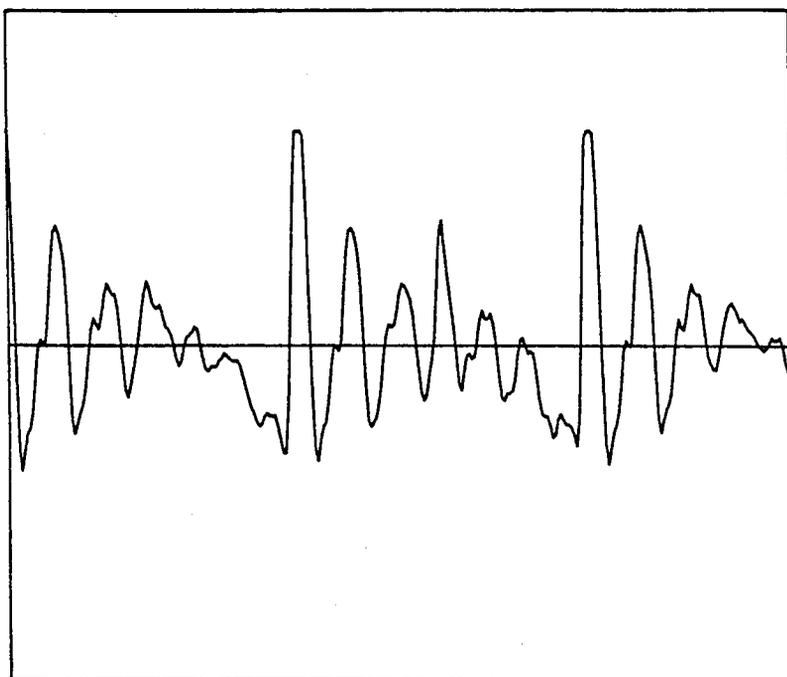


Fig 5.a.- 330 échantillons de /a/.

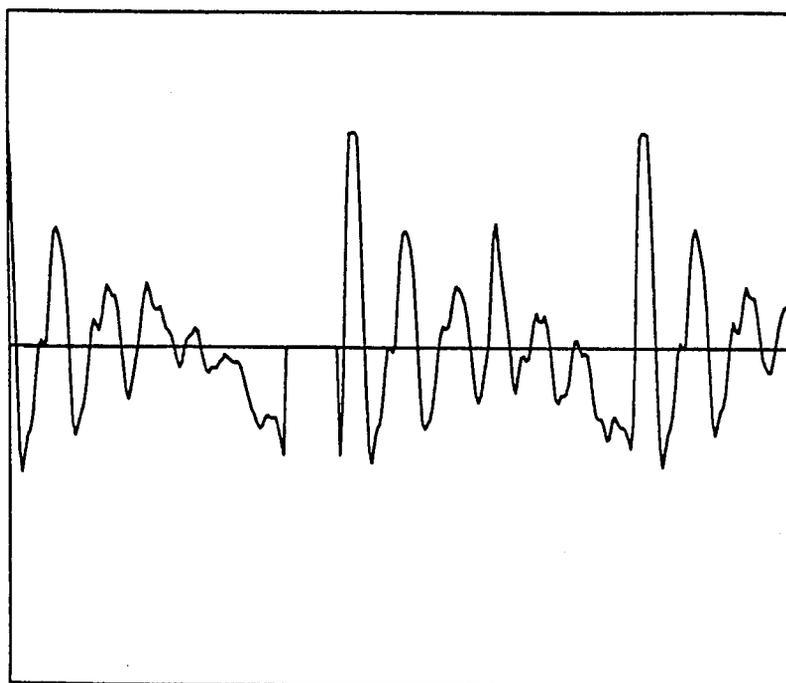


Fig 5.b.- 330 échantillons. Résultat d'insérer un segment de zeros dans /a/.

- Toute erreur détectée doit être marquée.
- Les erreurs les plus fréquentes doivent être diagnostiquées de façon précise.

Selon le secteur affecté par l'erreur, le traitement des erreurs en exécution a les effets suivants:

- Indication de l'erreur et fin de l'exécution, ou.
- Réalisation de toutes les actions associées à l'instruction, et indication que le traitement a été incorrect.

4.2. LA GESTION DE LA MÉMOIRE.

La gestion de mémoire est dynamique par rapport à la taille des variables REGISTRE et le signal réside en positions contiguës de mémoire. La raison de ce type de gestion est de permettre fondamentalement l'exécution de quelques instructions de l'ensemble de représentation acquisition en temps réel. Les temps obtenus pour les instructions de manipulation, qui sont les plus affectées par ce type de gestion de mémoire, sont totalement acceptables.

4.3. SUPPORT MATÉRIEL ET LOGICIEL.

L'EDS fonctionne sur une partition de 64 kilooctets d'un miniordinateur ECLIPSE C/350 de propos général, avec le système d'exploitation multiutilisateur "AOS". La figure 6 montre le matériel et le logiciel dont l'EDS a besoin.

- Matériel: Un convertisseur D/A, A/D. Un (pre/post) processeur analogique. Un système SCOPE pour la représentation graphique /7/.
- Logiciel: Le module ADACH pour l'utilisation du convertisseur. Le module SCREEN pour la manipulation d'un écran alphanumérique. Et le logiciel de manipulation du matériel SCOPE.

5. CONCLUSIONS.

L'EDS a été conçu comme un système d'édition interactif du signal, avec des mécanismes basiques et généraux pour un domaine quelconque de travail en Traitement du Signal Digital, afin qu'il soit le noyau central à partir duquel soit possible la progression vers des systèmes plus évolués.

La première version de l'EDS /2/ a été utilisée dans des projets de synthèse de la parole, pour l'édition de diphonèmes et en décodification acoustique-phonétique, pour l'étude des segments non-stationnaires du signal acoustique. Les résultats obtenus de cet utilisation ont été la base pour une meilleure définition des nécessités d'édition. Ainsi, a été détecté le besoin de modules spécifiques pour le traitement du signal, (Transformation de Fourier, quotients spectraux, etc....), qui complètent les outils du système général d'édition.

Dans la seconde version de l'EDS /4/ a été incorporé la possibilité de "modules d'utilisateur". Pour travailler avec ces modules, on a incorporé une nouvelle instruction dans l'EDS; dont la syntaxe est:

```
UTILISATEUR SOUS-COMMANDE [(PARAMETRES-D'UTILISATEUR)]  
SEGMENT [(POSITION)]
```

Où, dans le champ SOUS-COMMANDE (opérateurs "arithmétiques") on indique le traitement particulier à réaliser sur le SEGMENT avec une liste optionnelle de paramètres PARAMETRES D'UTILISATEUR. Le résultat du traitement peut être superposé, optionnellement, à partir de POSITION.

Ce mécanisme est maintenant en train d'être utilisé pour l'implémentation d'un module qui permet l'incorporation dans l'EDS de certaines procédures pour les traitements précis du signal, avec un ensemble de primitives pour couvrir les opérations algébriques basiques sur les signaux /4/.

Les résultats obtenus jusqu'à présent avec les différentes versions du EDS, on montré la possibilité d'introduire d'importantes améliorations parmi lesquelles sont déjà proposées pour la prochaine révision, les suivantes:

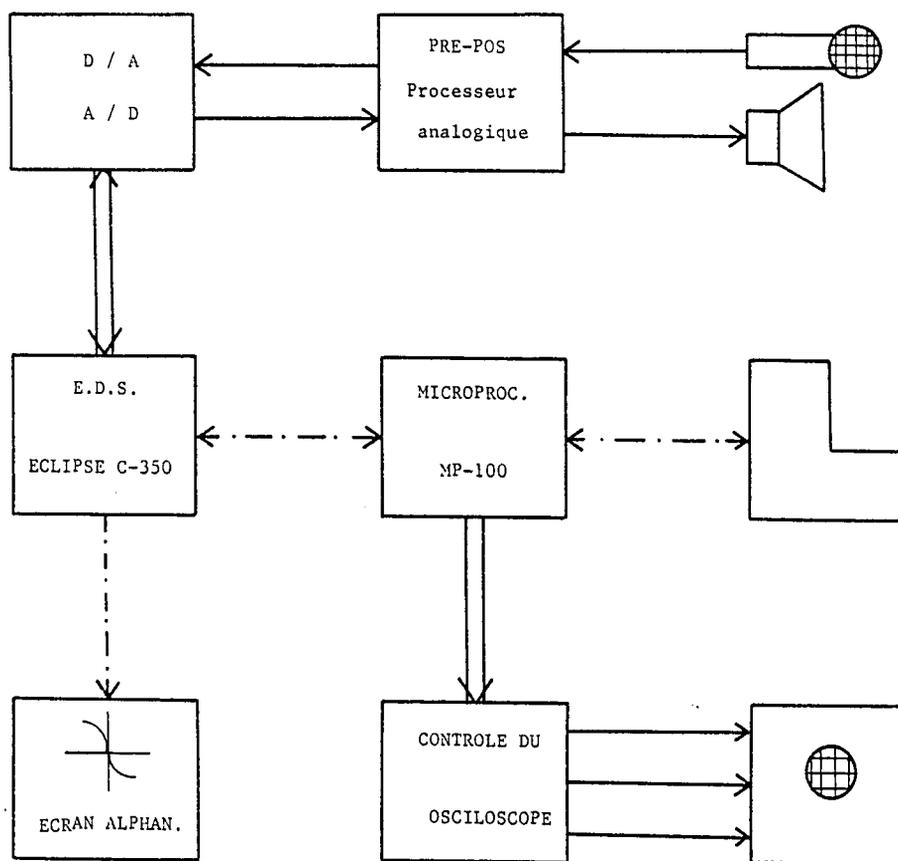


fig 6.- Matériel et logiciel associé avec l'EDS.

- Possibilitat de variables numèriques, dans tous les champs qualificateurs des sentences EDS. Ce qui permettra de varier automatiquement les positions et les segments utilisés en édition et/ou en traitement répétitif des signaux.
- Possibilitat de programmation, pour améliorer l'actuation de l'EDS en traitement réitératif. Ce-ci, avec l'existence des variables numèriques, rendra possible l'utilisation de l'EDS comme un langage de programmation pour traitement/manipulation - des signaux.

- /8/ N. WIRTH: "Algorithms + Data Structures = Programs"; Prentice Hall Inc. New York.
- /9/ F. ZURCHER: "Techniques de codage en vue de la compréhension et du stockage de la parole"; C.N.E.T./L.A.A. Nancy 1981, 10 19-20.
- /10/ L. SCHOLL: "Propositions pour une initiation à l'algorithmique"; Univ. Scientifique et Médicale, Institut National Polytechnique de Grenoble 1975.

6. REMERCIEMENTS

Notre gratitude au Prof. I. Ramos pour toute l'aide qu'il a apportée dans la conception et développement de ce travail.

7. BIBLIOGRAPHIE.

- /1/ A.V. AHO, J.D. ULLMAN: "The theory of Parsing, translation and compiling",-- Prentice Hall, 1973.
- /2/ J.M. BENEDI, E. VIDAL: "Un Editor de Señal"; V Congreso de Informática y Automática, Madrid 1982.
- /3/ J.M. BENEDI: "Un Editor de Señal"; Tesis de Licenciatura, Facultad de Físicas, - Universidad de Valencia, 1983.
- /4/ J.M. BENEDI, E. VIDAL: "A signal Editor: Present status and new implementations for its use in signal processing"; Workshop on signal processing and applications, Sitges 1983.
- /5/ J.E. HOPCROFT, J.D. ULLMAN: "Introduction to automata theory, languages and computation"; Adison Wesley, 1979.
- /6/ L.R. RABINER, R.W. SCHAFER: "Digital processing of speech signals"; Prentice Hall, 1978.
- /7/ E. VIDAL, B. TORRES, F. CASACUBERTA: "An interactive graphical system for unidimensional signals representation:SCOPE"; Workshop on signal processing and applications, Sitges 1983.