

## 5S-2

マルチメディアを考慮したオーサリングシステム L I L A C S<sup>(\*)</sup> の設計と開発<sup>(\*)</sup>楨本 淳子<sup>1)</sup>、岡田 孝<sup>1)</sup>小倉 紳吾<sup>2)</sup>、本田 洋二<sup>2)</sup><sup>1)</sup>関西学院大学・情報処理研究センター<sup>2)</sup>(株)ABC

## 1. はじめに

従来型の文字情報のみによるCAIコースウェアの限界が論じられている中で、マルチメディア対応のコースウェアを作成するオーサリングシステムの開発が、要望されている。このような状況のなかで本学では、コンピュータと、音声を制御するLL装置と、映像情報を制御する光ディスク装置との3つを結合させたマルチメディア対応のTSS教室(図1:次ページ)が完成した。これにともないCAI利用環境が飛躍的に整備された。この環境をさらにグレードアップさせるためにエンドユーザ向けのマルチメディア対応のオーサリングシステムを設計・開発した。以下このシステムをL I L A C Sと呼ぶ。

## 2. システムの構成とフロー

L I L A C Sにおけるコースウェア作成過程とその実行環境は、図2の通りである。このシステムでは、フレーム型CAIコースウェアを作成することを基本としている。コースウェアは、標準マクロを用いて記述する。しかし、標準マクロだけでは不十分な場合は、オーサが自由にユーザマクロを作成することができる。必要に応じてC言語によるプログラムを作成し、追加することも可能である。オーサのコースウェアの作成作業は、以下の通りである。

- (1) コースウェアの設計
- (2) 表示テキストの作成→エディタ/ワープロを使用
- (3) 主教材の作成→ " " "
- (4) 外部教材(音声教材・映像教材)の作成
- (5) マクロ展開、Cの翻訳とリンク

(1)以外の作業は、すべて分担作業が可能である。また、表示テキストの作成、および主教材の記述にはエディタまたは、日本語ワープロを使うことを原則としており、オーサの負担を軽減することに配慮している。

## 3. オーサからみたシステムの特徴

L I L A C Sは、フレーム型CAIのもつ基本機能に加えて、オーサからみて特記すべき4つの機能をもっている。

- (1) POP-UP画面によるヒント表示機能

オーサがヒント画面を作成することにより、学習者は、簡単なキー操作でPOP-UP画面上にヒントを

表示できる。複数のヒント提示も可能である。

- (2) キーワードヘルプ機能

オーサが各画面にキーワード指定をすれば、インデックス画面を自動生成できる。学習者はインデックス画面を使って用語検索を行なうことができる。

- (3) 目次の自動生成機能

各画面のタイトルから、目次を自動生成することができるので、段階的な教材の構造化が可能である。

- (4) 音声・映像教材利用による教材の多様化

音声教材の併用により、従来の文字情報のみによる単調なCAI教材から脱皮でき、音声を主とする外国語学習などにも活用できる。また、映像教材の利用は、文字情報はもとより、グラフィック画面では実現できないリアルな映像を提示することにより、学習教材の多様化が計れる。

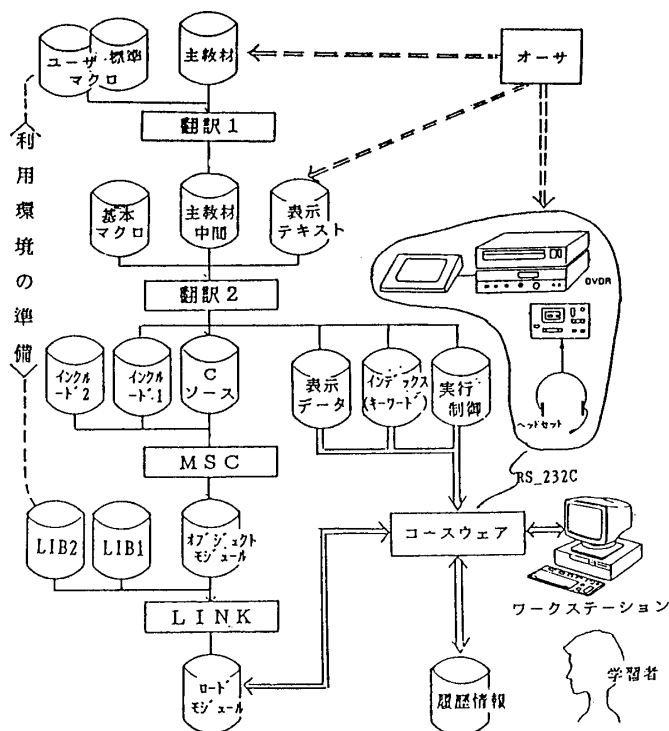


図2. コースウェアの作成と実行環境

## 4. マクロの定義とその利用

L I L A C S主教材は、通常標準およびユーザマクロで記述される。これらのマクロは、翻訳段階でC言語に展開され、L I L A C Sの実行制御モジュールや各種ライブラリー関数とリンクされる。マクロの作成と利用は、基本マクロも含めてすべて同一の文法に従う。

(\*) Authoring System "LILACS" for the Audio Visual Media Interface.

(+) LILACS: Learning and Instruction Laboratory Assisted by Computer Systems

図3にマクロ定義と利用の標準的な形式を示す。

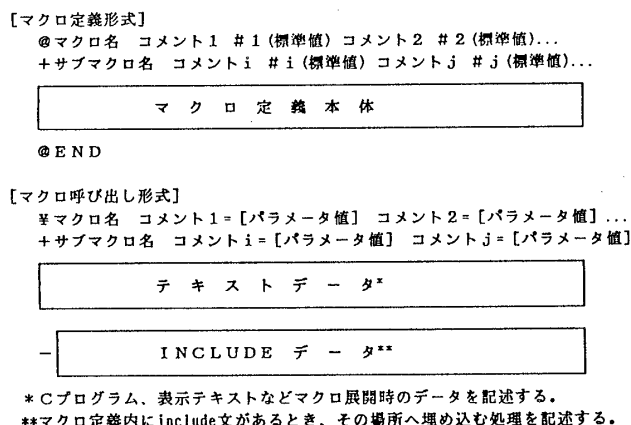


図3 マクロ定義と呼び出しの標準的形式

このような形式に設計した理由は以下の通りである。

- (1) システム作成の容易さを考慮して、マクロ展開を制御するための情報は位置パラメータ形式であたえる。
- (2) マクロ言語を利用するオーサにプログラミングの負担を少なくするため、標準/ユーザマクロの定義と利用時には、日本語の使用を可能とする。
- (3) 各パラメータの意味を、日本語の注釈としてパラメータ記述領域に自由に記述できるようにし、それらの内容を容易に理解できるように配慮する。このため実際の各パラメータ値は [ ] で囲んで指定し、定義文においては“#”記号の後にパラメータ番号を記述することとした。
- (4) まとまった機能毎に対応するパラメータ群の指定をサブマクロとして独立できるようにした。通常的位置パラメータ形式では、パラメータの数が多いときそれらの指定が煩雑となるが、本形式ではサブマクロ中のパラメータに標準値を使用する場合、対応するサブマクロの指定が省略できる。

標準マクロを利用したフレーム定義の例を図4に示す。

```

※注釈      [D002 (目次)]
※フレーム定義 NO=[2] タイトル=[目次]
+標準実行時間 時間 [300]
+標準分岐先  [next]
※テキストF  テキスト名=[D103_1]
※実行キー入力待ち 位置[2] 色=[4]
※

```

図4 フレーム定義の例

多くのオーサは、あたえられた標準マクロのみを利用すると考えられるが、LILACSシステムでは利用者が任意の新しいマクロを定義して利用することができる。利用形態としては、マクロ名の単なる変更から、出現頻度の高いパターンの登録、C言語による特殊な処理の実現にいたるまで多様な場合が存在する。このような利用形態を考慮すれば、マクロ定義中に他のマクロを記述するネスティングが多重にできることは当然必要とされる。

主教材中のあるマクロを展開する際、そのマクロに付随して主教材中に記述した一連の処理を展開形の一部に取り込む必要のあることが判明した。これを include 処理と呼び、その例を図5に示す。ここで-を付して示した部分が includeされる処理であり、マクロ展開時に include文の存在する場所に埋め込まれる。include処理は“-”記号を複数付加することにより、多重に行なうことが可能である。この処理により、種々の制御構造をLILACSマクロ言語へ簡単に取り込むことができた。

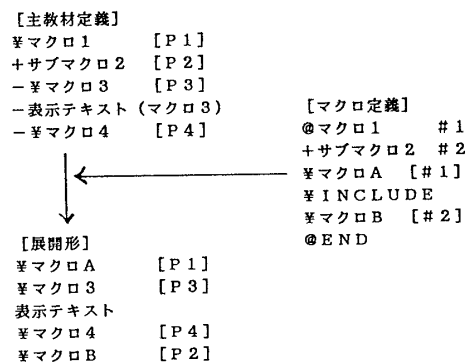


図5 include処理の例

## 5. 今後の課題

- (1) コースウェアを開発し、ノウハウを蓄積する。
- (2) さらに使い易いシステムをめざし、標準マクロ体系を改良する。
- (3) コースウェアの多様化に対応するためにC関数ライブラリーの充実をはかる必要がある。

謝辞：このシステムは、(株)日立製作所の協力のもとにLILACS/KGとして開発され、本学で稼動している。また、設計・開発にあたってのプロジェクト研究員の方々の有益な助言に対してもここに記して感謝する次第である。

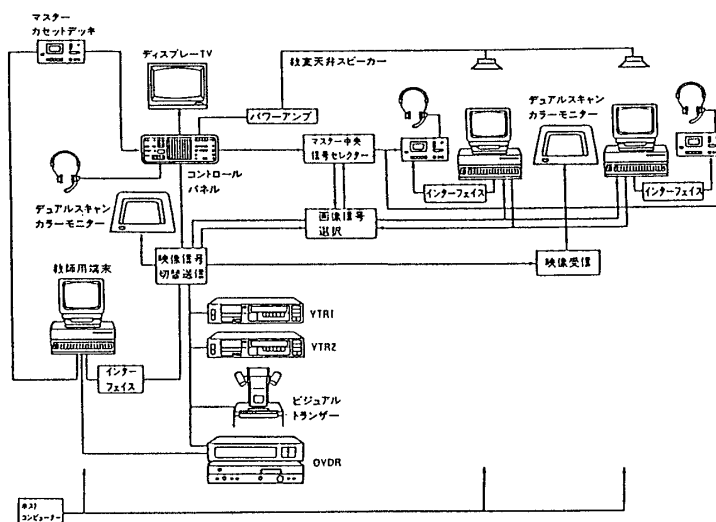


図1 マルチメディア対応のTSS教室のハードウェア構成