# オブジェクト指向ＵＩＭＳ CONCORD（2）
# 4M-7 －仕様定義言語CAISL－

松山洋一，内藤広志，間宮　悟，浅野俊昭
キヤノン(株)　情報システム研究所

## 1　はじめに

我々は，オブジェクト指向ＵＩＭＳ(User Interface Management System) **CONCORD** [1] を用いてＡＩツール **CHORUS**[2] のユーザインタフェース(**SCORE**)を開発している。通常，ユーザインタフェース（以降ＵＩと略記）の開発は，プロトタイプの作成，使用実験，ユーザ評価のフィードバックによる改良，というプロセスを繰り返すことによって行われる。従って，ＵＩＭＳではＵＩの変更が容易に行える必要がある。**CONCORD** は，ＵＩを作成するのに必要な部品を用意し，共有ＡＤＴメカニズムによりＵＩとアプリケーションの分離を容易にしている。しかし，**SCORE** の開発を進めるにつれて，**CONCORD** を使用したＵＩの作成，変更は容易でないことがわかってきた。その原因として以下のことがあげられる。

- 基本となる部品（Ｎｏｔｅ，ＡＤＴ，モデル等）を組み合わせてＵＩを作成する作業が非常に煩雑な上，これらを手続き的に記述する必要があるためにＵＩの構造がわかりにくい。
- ウインドウの配置等のように，オブジェクト間の依存関係を定義しているものがソースコード内に分散している。

このような問題を解決するために，ＵＩの仕様の定義を宣言的に行い，この定義から**CONCORD** に必要なコードを生成するための言語**CAISL**(Concord Application Interface Specification Language) を開発した。

本報告では，**CAISL** の概要を述べ，**CAISL** を用いて作成した**SCORE** のインスペクタの例を示す。

## 2　概要

ここでは，前述の**CONCORD** の問題点に対する**CAISL** での解決方法について述べる。

### 2.1　ＵＩ仕様の宣言的定義

**CONCORD** を用いて作成されるＵＩは幾つかのツール(ブラウザと呼ぶ)から構成される。ブラウザは，幾つかのサブウインドウから構成されるウインドウとして画面上に表示される。これらのサブウインドウは，オブジェクトの表示，選択等の機能を持ちビューと呼ぶ。

**CAISL** では，３つの定義マクロ**def-application, def-browser, def-view** で，以下のような項目を宣言的に定義する事によってＵＩを記述する。ウインドウの生成，配置等に必要な手続きの記述は，これらの定義をＬｉｓｐコードに変換することによって生成される。

- アプリケーションを構成するするために必要なファイル
- ＵＩモデルで使用する局所変数と関数の名前
- ＵＩとアプリケーションとのインタフェースのための関数名
- 共有ＡＤＴメカニズム[1]で使用するモデルとＡＤＴのタイプ
- オブジェクトの表示形態(リスト形式またはネットワーク形式)
- ウインドウの位置，大きさ，ボーダーの幅等の属性
- ブラウザ内のビューの構成および配置と大きさ
- ポップアップ・メニュー

### 2.2　ファイル構成

**CAISL** を用いたアプリケーションは以下の３種類のファイルから構成される。

1. アプリケーション・プログラム・ファイル
2. ブラウザ定義ファイル
3. アプリケーション定義ファイル

アプリケーションの内部処理の記述は全て１に含まれる。２には，ブラウザおよびビューの仕様が，それぞれ**def-browser, def-view** を用いて定義される。また，１で定義された関数のうちＵＩから呼ぶ必要があるものはその名前を２に記述する。これらのファイルは，アプリケーションの開発において，複数のバージョンが作成される。３では，これらのうちどれを使用するかを**def-application** を用いて定義する。

## 3　構文

### 3.1　アプリケーションの定義

アプリケーションの定義は**def-application** を用い，アプリケーションに必要なファイルの他にアプリケーションの起動および終了時に必要な処理を定義する。

```
(def-application アプリケーション名
  ((変数名 ブラウザ名 属性名 値. . . ) . . . )
  :default-pathname ファイルのデフォルト・パス
  :modules (アプリケーションファイル名 . . . )
  :browser-spec (ブラウザ定義ファイル名 . . . )
  :init-function アプリケーション起動処理関数名
  :exit-function アプリケーション終了処理関数名)
```

---

Object-Oriented UIMS CONCORD (2) —Specification Language CAISL— Youichi MATSUYAMA, Hiroshi NAITO, Satoru MAMIYA, Toshiaki ASANO
(Information Systems Research Center, Canon Inc.)

ここで，変数名はアプリケーション起動時に生成されるブラウザのインスタンスを保存するためのものであり，ブラウザ名には後述のdef-browserで定義するブラウザの名前を指定する。

### 3.2 ブラウザの定義

ブラウザの定義はdef-browserを用い，ブラウザの大きさ等のリソース値やブラウザを構成するビューの配置および大きさ等を定義する。

```
(def-browser ブラウザ名
  ((局所変数 初期値) . . . )
  :class モデルのタイプ
  :properties (属性名 値 . . . )
  :modules ファイル名
  :arrangement ビューの配置の定義
  :view-sizes ビューの大きさの定義
  :hooks ((フック名 関数名 引き数) . . . ))
```

ここで，局所変数には，ビューで選択されたオブジェクトを保存するための変数等を指定する。また，ビューの配置および大きさは以下のように定義する。

- :arrangement　ブラウザを構成するビューの名前を記号 | または / で区切って記述する事によって，ビューの配置を定義する。| で区切られたビューは横方向に接するビューを，/ で区切られたビューは縦方向に接するビューを表す。(図1参照)
- :view-sizes　ブラウザの大きさに占める各ビューの大きさの割合を幅および高さについて指定する。この時，大きさの割合と共にブラウザをリサイズした時に大きさを変更する(:rubber)か否(:chain)かも指定する。(図1参照)

### 3.3 ビューの定義

ビューの定義はdef-viewを用い，フォント等のリソースやメニューを定義する。

```
(def-view (ビュー名 ブラウザ名 ビューのタイプ)
    :properties (属性名 値 . . . )
    :data-type ADTデータ型
    :hooks ((フック名 関数名 引き数) . . )
    :menu ((項目名 実行関数名 テスト関数名) . . . ))
```

ここで，ブラウザ名にはビューが属するブラウザ名を，ビューのタイプにはCONCORDで用意されている部品(Noteと呼ぶ)名を指定する。また，CONCORDで用いる共有ＡＤＴメカニズムで使用するモデルおよびＡＤＴのタイプを指定する。

## 4 定義例

CAISLの使用例としてＡＩツールCHORUSの知識ベースの内容を調べるツールであるエージェント・インスペクタの定義の一部を図2に，その外観を図3に示す。

## 5 おわりに

CONCORDを用いたＵＩの仕様定義言語CAISLの概要を述べた。今後，SCOREの開発を通してCONCORDおよびCAISLの改良を行っていく予定である。

## 参考文献


[1] 間宮ほか，“オブジェクト指向ＵＩＭＳCONCORD (1) ― ＡＤＴメカニズム ―”，本論文集.

[2] 飛鳥井ほか，“エキスパートシステム構築ツールCHORUS(1)(2)(3)”，情報処理学会昭和６３年後期全国大会講演論文集(II).

[3] DOMAIN/Dialogue User's Guide, Apollo Computer Inc.

[4] Programming the User Interface. 7B, Symbolics Inc.


```
:arrangement (view1 / (view2 | view3))
:view-sizes
 ((view1 :width (1 :rubber) :height (1/2 :chain))
  (view2 :width (1/2 :chain) :height (1/2 :rubber))
  (view3 :width (1/2 :rubber) :height (1/2 :rubber)))
```

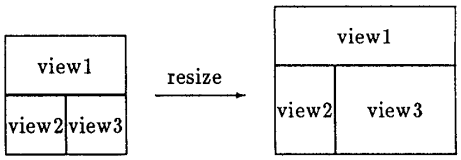


図 1: ビューの配置と大きさの定義例

```
(def-browser agent-inspector
  ((selected-agent nil) (selected-agenda nil))
  :class :flavor
  :properties (:width 400 :height 300)
  :modules ("agent-if")
  :arrangement
   ((agent | agenda-rule) / (prop-wme | rule-def))
  :view-sizes
   ((agent :width (2/5 :chain) :height (1/2 :chain))
    (agenda-rule
     :width (:rest :rubber) :height (1/2 :chain))
    (prop-wme
     :width (2/5 :chain) :height (:rest :rubber))
    (rule-def
     :width (:rest :rubber) :height (:rest :rubber)))
  :hooks
   ((popup-callback set-cursor agent-inspector)))
(def-view (agent agent-inspector
                 selectable-label-list-note)
  :properties ((:font-name "6x8" :scroll-bars :left))
  :data-type list
  :hooks ((get-data-hook get-agents nil)
          (change-hook change-agent selection)
          (init-selection-hook init-agent nil))
  :menu (("trace" :action traceon :test tracep)
         ("untrace" :action traceoff :test untracep)))
```

図 2: エージェントインスペクタの定義

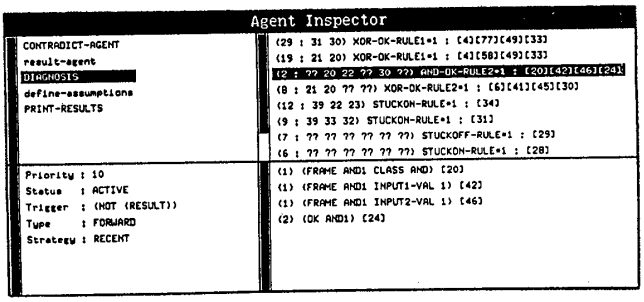


図 3: エージェントインスペクタの外観