4Q−6

# Tachyon Common Lisp における ウィンドウ・インタフェース

小島 透* 山田 雅彦* 五味 弘* 高橋 順一* 長坂 篤**

*(株)沖テクノシステムズ ラボラトリ
**沖電気工業(株)

## 1 はじめに

我々は、Tachyon Common Lisp[1]上に、Motif[2]対応のウィンドウ・インタフェースを開発した。これにより、LISP上で、Motifの関数を用いたウィンドウアプリケーションの開発が行なえるようになった。

また、ウィンドウ・インタフェースを用いて、デバッガやトレーサ等のツールをウィンドウ対応にした。これによってユーザはより快適に、デバッグなどが行えるようになった。

本論文では、まずこのウィンドウ・インタフェースの特徴を示す。次にツールのウィンドウ対応を行なう方法を述べる。例として Tachyon Common Lisp のデバッガを挙げる。

## 2 ウィンドウ・インタフェースの構造

ウィンドウ・インタフェースは、Tachyon Common Lisp 本体と Motif や X ツールキット[3] を使用するためのサーバプロセスから構成されている。

LISP プログラムからウィンドウ・インタフェースの関数を呼び出すと、Tachyon Common Lisp からプロセス間通信によって、関数番号や引数などの情報が、サーバプロセスに送られる。

そして受け取った情報をもとに、サーバプロセスは Motif などの関数を呼び出して実行する。

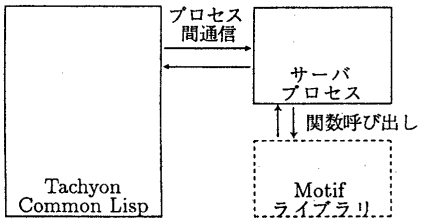


図 1: ウィンドウ・インタフェース実現方法の概要

Window Interface System in Tachyon Common Lisp
Tohru KOJIMA, Masahiko YAMADA, Hiroshi GOMI
Junichi TAKAHASHI*, Atsushi NAGASAKA**
*Oki Technosystems Laboratory, Inc.
**Oki Electric Industry Co., Ltd.

## 3 ウィンドウ・インタフェースの特徴

このウィンドウ・インタフェースの特徴として、次の四つが挙げられる。

1. 対話的にプログラミングが行なえるため、プロトタイピングに適している。コールバック関数の設定等が動的に行なえ、アプリケーション作成にかかる時間も短くなる。

2. ウィンドウ部分以外は高機能な LISP 関数が使える。そのため、ウィンドウアプリケーションのコーディング量が小さくできる。

3. 別プロセスで動作するため、Tachyon Common Lisp 本体のメモリサイズが大きくならない。またサーバが異常終了しても、Tachyon Common Lisp 本体が異常終了することはない。

4. Motif の関数を約１５０個、X ツールキットの関数約３０個をサポートしている。また、Motif のほとんどのリソースが使用できる。

## 4 ツールのウィンドウ対応

Tachyon Common Lisp は、テキスト端末でも使用するため、デバッガやトレーサ等のツールは、テキスト端末でも X 端末でも動作する必要がある。

しかし、テキスト端末用と X 端末用とを個別に作成することは、効率及び操作性の面から好ましくない。

Tachyon Common Lisp では、入出力部のみを拡張することで、ツールを両対応化している。

以下に例としてデバッガのウィンドウ対応の方法を示す。

### 4.1 入力部の拡張

入力されたコマンドは、常に文字列に格納するようにした。ウィンドウデバッガを使用している場合は、押したボタンに対応したコマンドを、文字列に入れておく。それ以外の場合は、LISP 関数 read-line を用いて、同じように文字列にしておく。

そしてその文字列を、LISP 関数 read-from-string の引数に与えてコマンドを取ることで、どちらの場合でも同じように機能するようになった。

### 4.2 出力部の拡張

デバッガ中で出力を行なうすべての関数を、その引数と共に出力を管理する関数に渡すようにした。この関数では、ウィンドウデバッガを使用している場合は、次の処理を行なう。

1. まずストリーム*debug-io*を、別の文字列ストリームに束縛する。
2. 渡された関数を実行する。
3. 最後に文字列ストリームから文字列を取り出し、ウィンドウに表示する。

デバッガの出力には、特に目立たせたい部分を強調するために、反転表示の開始と終了のエスケープシーケンスが含まれている。これをそのままウィンドウに表示しても反転表示されない。

これを解決するため3に関して、文字列中に反転表示の開始と終了のエスケープシーケンスが含まれている時は、その部分の文字列を Motif 関数 text-set-highlight を用いて反転表示する。

ウィンドウデバッガを使用していない場合は、2だけを行なう。

このようにすることで出力に関しても、どちらの場合でも同じように機能するようになった。

## 5 ウィンドウ対応デバッガの利点

ウィンドウ対応デバッガは、次のような利点を持っている。

- デバッグ作業中、エラーが起こるまでの過程は、フレームをバックトレースすることで調べる。

  このとき Motif のスクロールリストを用いることで、フレームの移動、選択はマウスで行なうことができる。

- 主なコマンドをボタン化することで、ほとんどの操作がマウスで行なえる。
- 出力ウィンドウを複数開けるため、確認しておきたい部分を常に表示させておくことができる。
- ウィンドウをリサイズすることで、表示幅、表示行、フレームリスト中の表示数が変えられる。

## 6 おわりに

Tachyon Common Lisp 上に Motif 対応のウィンドウ・インタフェースを作成した。また、それを使用することによって、デバッガなどのユーザインタフェースが向上した。今後の課題は、ウィンドウを使用したツール群を連係させることによって、さらに使いやすい開発環境を実現することである。

## 参考文献

[1] A. Nagasaka, Y. Shintani, T. Ito, H. Gomi and J. Takahashi. Tachyon Common Lisp: An Efficient and Portable Implementation of CLtL2, *Proceedings* of the 1992 ACM Conference on Lisp and Functional Programming, ACM, 1992.

[2] Open Software Foundation, "OSF/Motif Programmer's Reference Release 1.1", Prentice-Hall,Inc 1991.

[3] Tim O'Reilly & Mark Langley, "X Toolkit Intrinsics Reference Manual for X Version 11", O'Reilly & Associates,Inc 1990.

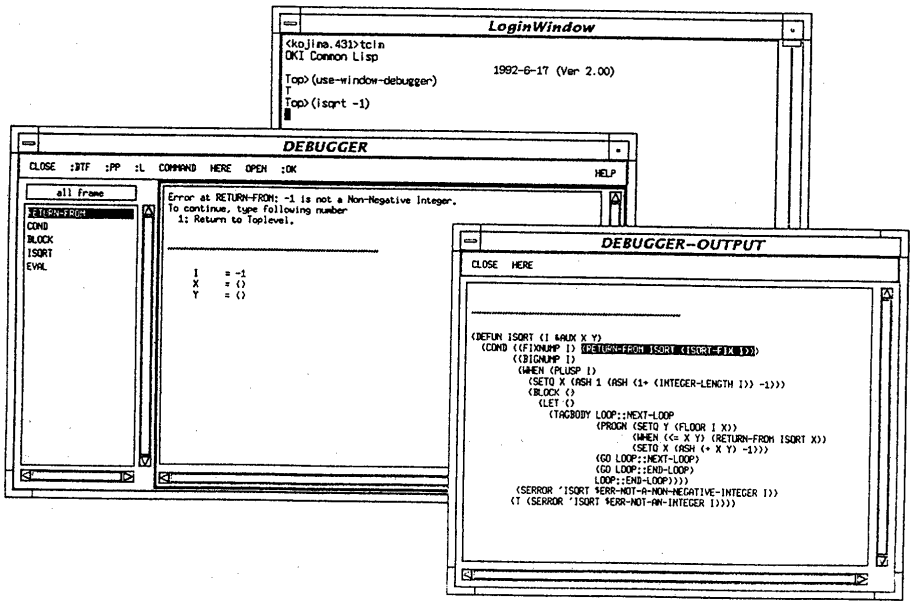


図2: ウィンドウ対応デバッガ