

段階的詳細化を基礎としたLispプログラム開発支援ツール

3S-5

FASET(5)自然言語記述による開発支援ツール

石谷 靖 松尾 正浩 仲 隆*
三菱総合研究所

1. 概要

本稿では、ソフトウェア開発において、上流工程から下流工程まで支援するツールについてその機能・構成を述べる。本ツールは、上流工程の支援のために、段階的詳細化の手法を設計の基本手法として採用し、仕様レベルでの検証などを行ない、下流工程の支援として、設計仕様からプログラムを最終的に生成する。開発は、Lispマシン(Symbolics 3640)上で行なわれ、Lispプログラムの開発を支援する。ツールの特徴は以下のとおりである。

- ・ トップダウン／ボトムアップ設計支援
機能の分割・統合という形で、仕様の段階的詳細化の過程を支援。
 - ・ 仕様の検証
分割した機能単位間の入出力データ構造の整合性の検証および推定を行なう。
 - ・ 実行可能プログラムへの変換
機能単位の記述に従って、実行可能プログラム(Lisp)に変換する。
 - ・ 仕様レベルでの再利用の支援
既存の仕様の再利用を支援する。検索は日本語文字列で可能である。
 - ・ 柔軟なユーザインタフェース
マルチウインドウ・マウス等を活用した柔軟なユーザインタフェースを持つ。
- 以下に、それぞれの説明を行なう。

2. 仕様記述言語

機能単位の記述は、ライブラリに登録された関数およびデータ型と、任意の構文要素(非終端記号に対応するもの)の代わりに日本語の記述の許されたLISPに似た言語を用いる。設計者は、仕様記述言語で要求仕様から、ミニスペックまでを、段階的に詳細化しつつ、記述することができる。

この仕様記述言語は、データ型も持つ。データ型はライブラリに登録された型とユーザが登録した型があり、記述した仕様の入出力データの検証および推定を行なうこ

とができる。

3. 機能ウインドウ

仕様の記述のためのウインドウが多数用意されている。

・ データ構造ウインドウ(図1a)

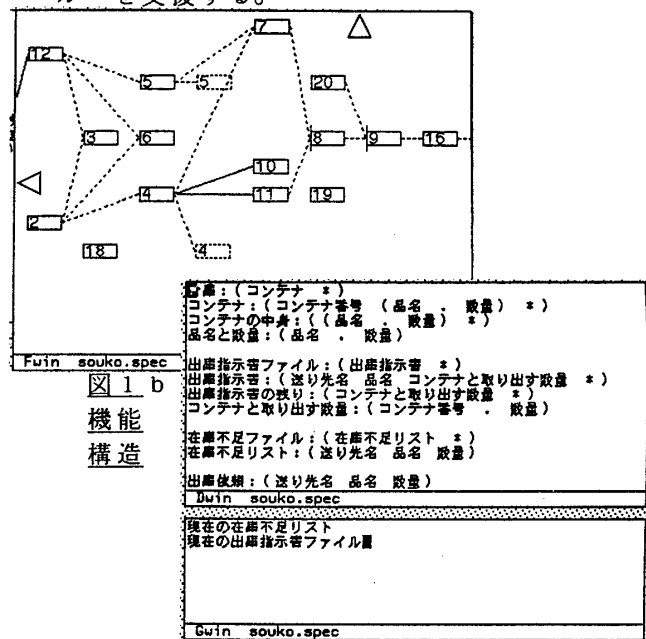
データ構造を記述・登録するためのウインドウ。データディクショナリの作成を行なうエリアである。データ構造はデータ構造名をアトムとし、任意個の繰り返し記号(*)を含むS式で記述する。

・ 大域変数ウインドウ(図1a)

ユーザが大域変数として使用する変数名を登録する。

・ 機能構造ウインドウ(図1b)

機能単位をノード、それらの呼び出し関係をアークとする木構造で、機能の構造を視覚化し、開発プログラムのウォークスルーを支援する。



・LISPコードウインドウ

変換された実行可能プログラムを表示する。ユーザは、ウインドウ内で必要に応じてコードを変更することもできる。

下の図2に示す機能単位記述ウインドウの内部は、上から、機能単位名、入力仕様、出力仕様および機能単位の仕様を記述するエリアである。

受付係	
コンテナに格納されている指定された品名の数量	
IN	品名 コンテナの中身
OUT	数量
(	OUT
(let ((品名と数量 (161 指定された品名と数量を取り出す)))	
(cond ((null 品名と数量) 0)	
(t (cdr 品名と数量))))	
N	
Nwin-9 souko.spec	

図2. 機能単位記述ウインドウ

4. ポップアップメニュー

ツールに対する指示は、ポップアップメニュー中の項目をマウスで選択することで行なう。

Open Close Move Copy Delete Comment Check Lock Lisp	Exit Save All Creat Fwin Edit Lib Data Edit Lib Functions Collect Garbage
システムメニュー	
ノードメニュー	Close Move Home Extend Reshape Show Syntax Show Example
ウインドウメニュー	

図3. ポップアップメニュー

・システムメニュー

ウインドウの外部でマウスをクリックすると表示される。

・ウインドウメニュー

ウインドウの内部でマウスをクリックすると表示される。

・ノードメニュー

機能構造ウインドウ内のノード上でマウスをクリックすると表示される。

図3でそれぞれの表示イメージを示す。

5. ノードメニュー

4節でふれたノードメニュー内の選択肢に、解析(check)、保存(save)、読み込み(load)などがある。これらのうち、「保存」は編集した仕様記述をファイルとして保存する機能であり、「読み込み」は、あらかじめ編集してある仕様記述を読み込むための機能である。

「解析」は、クリックされたノードの仕様記述に関して、その記述および入出力データの整合性の検証および入出力データの推定を行なう。マウスで示された機能単位だけでなく、それよりも下位の機能単位も検証される。入出力のデータ構造の検証には、ライブラリで定義された関数・データおよびデータ構造ウインドウで作成したデータディクショナリを使用する。

ライブラリ(関数・データ)は、仕様記述と実際のLispコードとを結ぶ情報である。ライブラリは、ライブラリ編集の専用のウインドウでエディットすることができる。

6. おわりに

本ツールは、日本語での記述および仕様の階層化を利用して、プログラム設計段階で開発を支援するのみならず、仕様の保守をも容易にできると考えられる。

現在、本ツールは、仕様の検索・仕様へのプログラムの変換を除く、上流工程支援部分について完成している。

参考文献

[1] Kant, E. and Barstow, D. R.,

"The refinement paradigm: the interaction of coding and efficiency knowledge in program synthesis",

IEEE Trans. on S.E., vol. SE-7, No. 5, pp 458-71, Sept. 1981.