ソフトウェア開発における協調支援環境Vela
—— (6) TMSの利用実験と評価 ——

2G−6

太田 剛　山口 高平　落水 浩一郎

静岡大学

1. はじめに

本稿では,Velaの仮説管理フェーズにおけるTMS[1]の有効性を実験を通して示す．なお，実験には文献[2]の電文解析問題をJSPで設計するプロセスを題材としてとりあげた．

2. 実験の概要

今回の実験において使用した知識をPrologで記述した例を以下に示す．(A)は前向き実行制御知識,(B)(C)はオブジェクトレベル知識である．なお，今回の実験では知的バックトラック機構は実装未了であるため，後向き実行制御知識は取り扱わない．

```
jsp法(Problem,Program) :-                    —(A)
        入力木作成(Problem,In),
        出力木作成(Problem,Out),
        プログラム木作成(In,Out,Prog),
        演算子数え上げ(Problem,Ops),
        演算子割り当て(Ops,Prog,OpTree),
        プログラム作成(OpTree,Program).

プログラム木作成(In,Out,Prog) :-             —(B)
        プログラム木作成本体(In,Out,Prog).
プログラム木作成(In,Out,[Prog1,Prog2]) :-    —(C)
        中間ファイル作成(In,Out,Tmp),
        プログラム木作成本体(In,Tmp,Prog1),
        プログラム木作成本体(Tmp,Out,Prog2).
```

3. TMSの利用実験例

2節に示した知識を用いて，電文解析問題をJSP法により設計する実験経過を示す．なお，この実験終了時にTMSが管理している構造を図1に示す．

(1) ユーザは2節に示した知識を含むJSP用のプロセス記述を読み込んだメタインタープリタを起動する．

(2) メタインタープリタは前向き実行制御知識(A)を実行し，ユーザに対して入力木作成の指示を出す．この問題においては，入力は電報の列ともブロックの列ともみなせるが，ここではユーザが経験不足のために誤りをおかし，電報の列であるという仮説1を採用したとする．ユーザが仮説1と入力木1をシステムに与えると，TMSは入力木1のjustificationとして問題文および仮説1を記録する．

(3) メタインタープリタは(A)により，次に出力木作成の指示を出す．ユーザは問題文に従って，出力が各電報に対する報告の列であるという仮説2をおく．ユーザがシステムに仮説2と出力木とを与えると，TMSは出力木のjustificationとして問題文および仮説2を記録する．

(4) 次にメタインタープリタは(A)によりプログラム木作成の指示を出すが，ここでは次にオブジェクトレベル知識(B)(C)が両方とも実行可能であるため，判断をユーザにあおぐ．ここでは，ユーザは入力木および出力木を吟味した結果構造不一致はないと判断し，これを仮説3として採用する．ユーザが仮説3とプログラム木1とをシステムに与えると,TMSはプログラム木1のjustificationとして入力木1，出力木および仮説3を記録する．

(5) (A)に従ってメタインタープリタは次に演算リスト作成の指示をする．ここでは特に必要とする仮説はないので，演算リストのjustificationとして問題文のみがTMSに記録される．

(6) 次にメタインタープリタは(A)により演算割り当ての指示を出す．このときユーザは，問題文中の「紙テープはread block命令によってアクセスされる」という記述に対応する演算が，プログラム木1へ割り当て不可能であることを発見し，このいきづまりをTMSに矛盾として通知する．

(7) TMSは，演算付プログラム木をcontradictionとし矛盾の原因を取り除こうとする．今回の実験では，矛盾の原因として仮説1をユーザが指定した．そして，新たに入力がブロックの列であるという仮説4を導入して，これをTMSに投入する．

(8) TMSは矛盾の原因となっている仮説1のラベルをoutにし，justificationをたどることによって，入力木1およびプログラム木1のラベルをもoutにする．そしてメタインタープリタの実行を(2)に戻す．

(9) (2)の実行に戻ったメタインタープリタは,入力木1のラベルがoutであるためユーザに修正を指示する．ユーザはメタインタープリタの指示に従い，入力木1を修正して入力木2を作成する．TMSは入力木2のjustificationとして問題文および仮説4を記録する．

Vela: A Software Development Environment for Cooperation Support

Tsuyoshi OHTA, Takahira YAMAGUCHI and Koichiro OCHIMIZU

Shizuoka University

(10) メタインタープリタは(A)により，次に出力木作成を実行する．このとき，出力木のラベルがinのままであるため，修正の必要はないと判断して実行を次に進める．

(11) メタインタープリタは(A)により，プログラム木作成を実行する．このとき，プログラム木1のラベルがoutとなっているためこの修正を指示する．

(12) ユーザはプログラム木1を修正してプログラム木2を作成する．このとき，構造不一致を発見するのでTMSに矛盾として通知する．TMSはこの通知により，プログラム木2を矛盾状態に変える．

(13) ユーザが矛盾の原因である仮説3を指定すると，TMSはこの仮説のラベルをoutにする．そして，ユーザは構造不一致があるという新たな仮説5をTMSに通知する．その後,TMSはメタインタープリタの実行を(4)に戻す．なお，この場合にはjustificationをたどっても，これ以上のラベルの変更はない．

(14) (4)の実行に戻ったメタインタープリタは(B)のオブジェクトレベル知識では設計に失敗したことを知り，次に(C)の知識の利用を試みる．すなわち，中間ファイルを用いる方法に方針を変更して，最終的に電文解析問題をJSP法により設計することに成功する．

4. 評価

本稿では，Velaシステムの「仮説管理フェーズ」におけるTMSの利用例を述べた．今回の実験では，ユーザが与える入出力木等の生成物と，それを構築するために必要となる仮説との依存関係および真偽値をTMSを用いて管理した．そして，ユーザの与えた仮説に設計のいきづまりの原因がある場合に，TMSを用いることにより対処できることが確認された．しかしながら，次に述べる不十分な点があり，今後改良していく予定である．

第一の点は，さまざまな仮説の中から設計のいきづまりの原因となっている仮説を特定するための機構の充実にある．TMSの機能そのままでは，全く関係のない仮説が原因として選ばれてしまうことがある．しかし，ソフトウェアの設計におけるやり直しにおいては，原因の究明が重要なポイントであり，これを支援できる機能が必要である．

第二の点は，「失敗情報の解析」機能の充実である．TMSが保持するnogood nodeは「前向き実行制御知識」におけるルールおよびユーザが導入した仮説の集合を管理している．前者を解析することにより組合わせ禁止のルール集合が得られるため，これを「前向き実行制御知識」にフィードバックさせることにより，過去の類似問題設計時におかしたものと同じ誤りにおちいることを防ぐことが可能となる．また，後者を解析することにより，ユーザが新たな仮説を導入しようとした際に，過去の類似問題において同様の仮説が導入されていきづまりを生じたという事実を通知できる可能性がある．

参考文献

[1] 成松, 山口, 落水: ソフトウェアプロセスの実行機構, 第41回情報処理学会全国大会予稿集 (1990).

[2] M.A.Jackson: 構造的プログラム設計の原理, 日本コンピュータ協会 (1980).

謝辞 本研究はSDAコンソーシアムの補助金と，科研費重点領域研究 (1)(課題番号 02249109) の一部の援助のもとに行なわれた。記して謝意を表する。

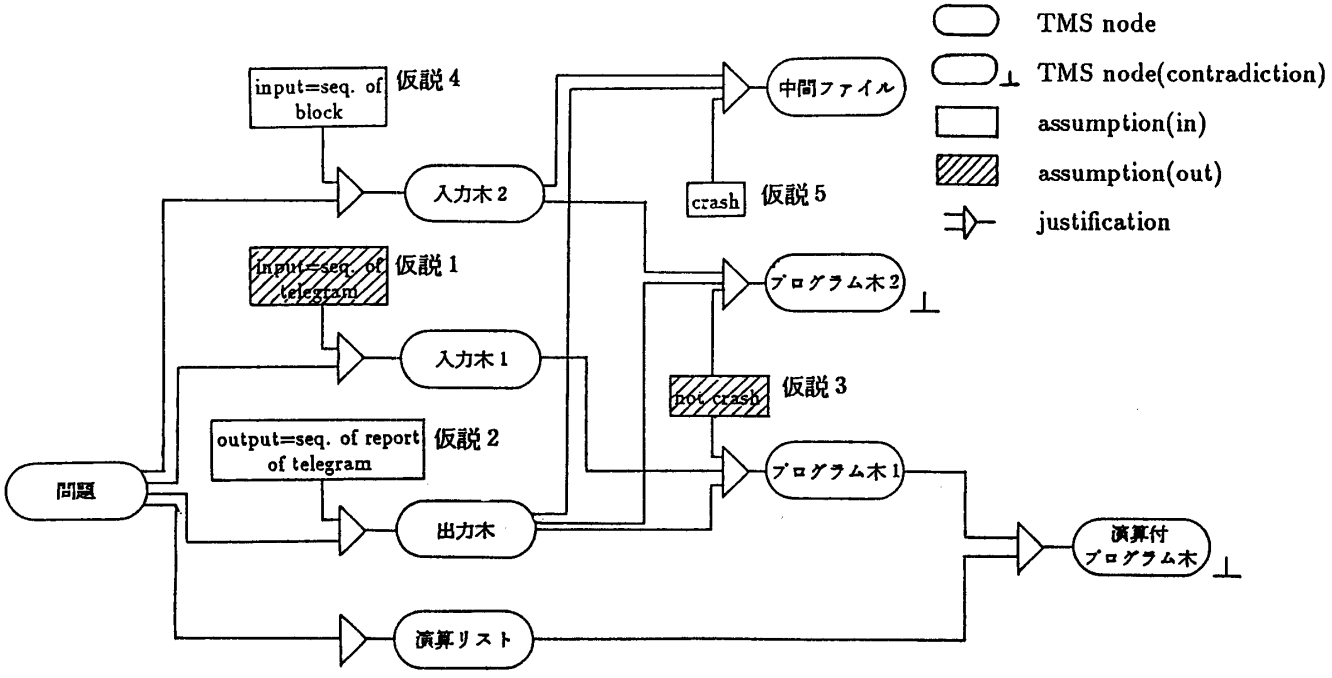

図1．実験終了時にTMSが管理している構造