7R-3

Ｋａｐｐａデータベース機能の検査システム

吉武淳*，　金枝上敦史*，　佐藤靖彦**，　平岡公一**，　河村元夫***

*：三菱電機（株）コンピュータ製作所，　**：（株）インテック，***：（財）新世代コンピュータ技術開発機構

1．　はじめに

Ｋａｐｐａ（Knowledge Application Oriented Advanced DBMS/KBMS）は、分散知識ベース管理システム[1]であり、ＥＳＰ(Extended Self-contained Prolog)で１２０Ｋ行からなる大規模なシステムである。Ｋａｐｐａは、高度な知識情報処理システムの基盤となる大量かつ複雑な知識（例えば電子化辞書、数学の知識など）を蓄積して自然言語処理、定理証明処理など広範な応用への利用が見込まれている。

Ｋａｐｐａは、データベース層、知識ベース層、ユーザインタフェース層、ユーティリティ層から成るが、今回そのうちのデータベース層のコマンド機能を対象に検査を行った。

本報告ではその検査で行った、自動化のための検査のシステム化について述べる。

2．検査の方針

今回の検査の目的は、一般の検査[2]と同様に、ユーザの立場に立ってＫａｐｐａが正しく動作するかを確認することである。検査対象は、Ｋａｐｐａのデータベース管理システムのプログラムインタフェースであるコマンドである。そして、大規模なシステムの検査を効率良く行うこと、また今後のＫａｐｐａのバージョンアップの際にも検査を効率良く行うことを考えて、検査のシステム化、自動化をはかった。また、開発者が行う試験で個々のコマンドと簡単なコマンド列の正常動作は確認されているため、大量で複雑なコマンド列を１つの検査項目として設定している。なお、検査項目は、全部で８９項目であった。

3．　検査システムの概要

3．1　特徴

Ｋａｐｐａの試作版の検査では、検査項目を考え、検査仕様を作成し、その通り動作するようなプログラムを組み、それを実行する手間や、実際の検査の実施の手間が非常に大きく、またコマンドの仕様が変更になるとプログラムまで修正する必要があった。

Ｋａｐｐａは今回ユーザリリースされるため、本格的な検査を、効率的に実施することが重要になってきた。そのため、手間のかかる検査仕様から検査プログラムへの変換および検査の実施を効率的に行うような検査システムを作成した。このシステムでは、決められた形式で検査仕様を作成し、システムに登録するだけで、検査の実施が可能になる。ここで検査仕様とは、どのようなコマンド列をどのように検査するかということであり、一連のコマンド名、引数値、正しい実行結果の値で定められるものとする。そして検査の実施も、システムに登録された検査項目を選択するだけで、検査を実施し、その結果を見ることもできる。また、コマンドの仕様が変更になっても、それまで作った検査仕様に手を加えることなく対応することも可能になった。

3．2　構成

検査システムの構成を図１に示す。検査システムは大別して、検査仕様をプログラム（検査項目サブモジュール）に変換するジェネレータと、それを管理、実行する管理モジュールからなっている。

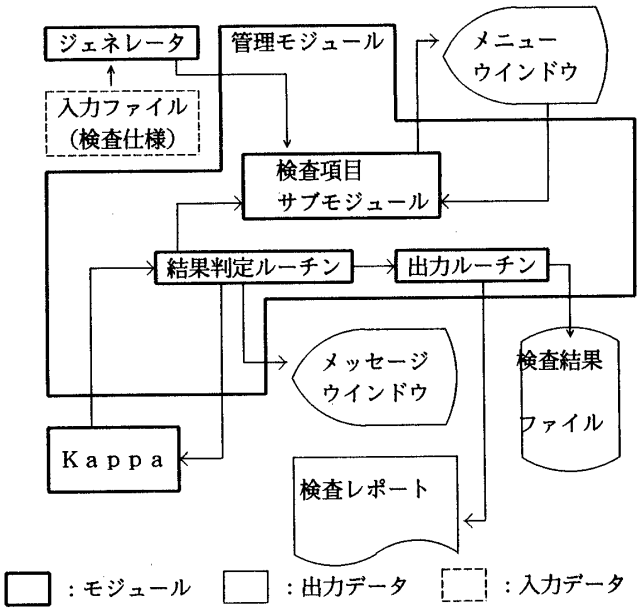

図１　検査システムの構成

3．3　検査の流れ

検査を行う際には、まず項目別に検査の手順を考え、検査システムが決めた形式の検査仕様を作成する。ここでは、それを検査項目ファイルと呼ぶ。

それは、基本的には検査時に発行するコマンドを順に並べたものであり、そのコマンドは、コマンド名、引数、結果から成り立っている。また、百件、千件単位のレコード処理のために、ループの制御構造を記述できる。さらに、エラー検査のために、コマンドの実行が失敗したほうが正しい場合の記述もできる

The inspection system of the database module of Kappa

Jun YOSHITAKE[1], Atsushi KANAEGAMI[1], Yasuhiko SATO[2] Koichi HIRAOKA[2], Motoo KAWAMURA[3]

1:Computer Works, Mitsubishi Electric Corporation. 2:INTEC Inc.

3:Institute for New Generation Computer Technology

```
COMMAND > read__record
ARGUMENT> (tap1, __, Result1)
RESULT  > (tap1, __, (apple, 'red'))

COMMAND > read__record
ARGUMENT> (tap2, __, Result2)
RESULT  > (tap2, __, (lemon, 'yellow'))
  :              :
  :              :
```

図2　検査項目ファイル

このファイルをジェネレータに通すと、検査項目ファイルの記述はＫａｐｐａのコマンドに変換され、それに結果の出力、エラー時の処理などが付加されたＥＳＰのプログラムになる。

検査仕様の修正は、検査項目ファイルを修正し、再びジェネレータに通すことで行う。

ジェネレータにより変換されＥＳＰのプログラムとなった検査項目は、管理モジュールにそのサブモジュールとして登録され管理される。管理モジュールは、いくつかの検査項目の実行要求をまとめて受け、逐次実行し、検査の状況や結果をウインドウに表示したり検査結果ファイルに書き出したりする。また、必要に応じて実行結果をレポートとしてプリンタ出力もできるようになっている。あるコマンドの実行結果が正しくなかった場合には、その検査項目の実行は中断され、次の検査項目が実行される。

4.　ＥＳＰの特徴と検査システム

ＥＳＰとは、論理型言語Ｐｒｏｌｏｇをベースにオブジェクト指向に基づくプログラムのモジュール化機能を導入した言語である。そして全ての検査項目サブモジュールは、それぞれ１つのインスタンス・オブジェクトで同じメソッドを持つ。しかし、それぞれがどの様に動作するかは、各インスタンス・オブジェクトによって異なる。このような所にＥＳＰのオブジェクト指向の機能を活用している。

さらに、論理型言語であるＥＳＰのバックトラック機能を利用している。コマンド実行中にエラーが発生した場合に残りのコマンド列を実行しないでｆａｉｌさせ、別のオルタナティブ（枝）でＫａｐｐａからのエラー情報を自動的に取り出すメカニズムが容易に記述できた。

5.　考察

検査システムの作成によって、Ｋａｐｐａの検査は、試作版Ｋａｐｐａの検査に比べて、大幅な効率化ができ、しかも本格的な検査が可能になった。

試作版Ｋａｐｐａの検査は、次のような流れであった。

検査仕様決定 → 検査プログラムの作成 → 検査プログラムのデバッグ → 検査の実施

試作版Ｋａｐｐａの検査においては、検査プログラムの作成とそのデバッグに費やす時間が多く、検査の流れを熟知した人でなければ困難であった。

これに対し検査システムを作成したことにより次のような流れになった。

検査仕様決定 → 検査システムへの登録 → 検査の実施

これにより、検査プログラムの作成とデバッグのステップの省略を可能にした。この効果は非常に大きく、検査項目や検査仕様に対する検討に時間をかけることができ、試作版Ｋａｐｐａの検査に比べ、より本格的な検査を短時間で行うことができるようになった。

また、システム化したことにより、検査の実施に人手を多くかける必要もなくなり、その面でも効率化がはかられた。検査期間中の´８８年１１月から´８９年１月までの間に、数回Ｋａｐｐａのバージョンアップが行われたために、開発チームに早く検査結果をフィードバックする必要があり、３節で述べたような検査の自動化がおおいに効果を果たした。

6.　まとめ

自然言語処理、定理証明処理等への応用を考えたＫａｐｐａは、ＥＳＰで記述された最大規模のソフトウエアである。このようなシステムの機能、性能の確認を迅速かつ的確に行うことは、非常に重要である。本報告では、そのための工夫の第一歩を述べてきたが、今後は、５節で述べた考察を踏まえて、さらなる工夫を重ねていく。

具体的な今後の課題としては、以下のようなことが挙げられる。

- ループ以外の制御構造も記述できるようにする。できれば、検査用言語を作成する。
- 検査用データベースとしてＫａｐｐａを利用し、コマンド名、検査結果等を格納する。
- 検査項目の削除等のユーザインタフェースを充実させる。

謝辞

本検査システムの設計及び製作にあたり、多大なご助力を頂いたＫａｐｐａ開発メンバーの方々に感謝します。

参考文献

1）溝口徹夫、横田一正、内田俊一：知識ベース管理システムＫａｐｐａ－データベースから知識ベースへ－，情報処理学会第３５回全国大会，３Ｍ－３．１９８７

2）菅野文友：ソフトウェア・エンジニアリング，日科技連，１９７９