2K－1

オブジェクト指向設計によるCASEツールの開発

藤波　武起[+]　谷田　耕救[+]　西木　健哉[++]　児玉　孝次[+++]
（株）日立製作所ソフトウエア開発本部[+]　（株）日立製作所システム開発研究所[++]
日立東北ソフトウエア（株）[+++]

1.背景

ソフトウエアの開発規模が増大し、生産性向上が求められている中、オブジェクト指向設計が注目を浴びている。また、CASEツールもバッチ型のものよりインタラクティブ型が求められるようになってきた。インタラクティブなものは、オブジェクト指向で表しやすいといわれる。そこで、CASEツールの開発にオブジェクト指向設計を用いることにより、オブジェクト指向設計が現状使用可能かどうかを検証することにした。

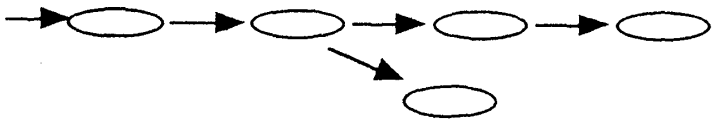
2.開発環境

開発言語としては、C＋＋を使用。C＋＋を使用することにより多くのC言語プログラマがプログラミング言語の学習時間を削減することができる。

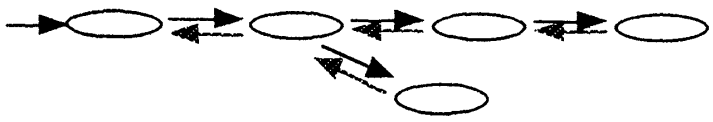
また、ユーザインタフェースとしては、X－Window上でMotifを使用した。

3.実現上の問題

X－Window上のアプリケーションは、イベントドリブンで処理を行う（非同期処理の）ためオブジェクト指向設計で扱いやすい。しかし、C＋＋ではメッセージパッシングを関数コールで実現しているため、非同期処理ではなく同期処理が行われる形になる（図１）。このため、わかりやすい設計にするためには、メッセージの流れを把握できるようにする必要がある。つまり、通常オブジェクト指向設計では気にならないメッセージの流れが重要になる。



非同期処理



Ｃ＋＋の処理系（同期処理）

図1.非同期処理とC++の処理系

4.メッセージの制御方式

メッセージの流れをつかむ為に、制御オブジェクトという特殊なオブジェクトを設けてメッセージの管理を行うことにする。この制御オブジェクトの役割としてオブジェクトの生成、メッセージの制御等を行う。また、この制御オブジェクトにより非同期処理の部分と同期処理の部分を明確に分けることによりメッセージの流れをわかりやすくすることができる。

しかし、この方法では、制御オブジェクトの扱うオブジェクトの数が多くなると制御オブジェクト自体の振る舞いがわかりにくくなり、また、メッセージの数が多くなるために実行速度に影響が出ることが十分に考えられる。

Applying Object Oriented Design to CASE tool development

T.Fujinami[1] T.Tanida[1] K.Nishiki[1] K.Kodama[2]
HITACHI,Ltd.[1]　HITACHI TOHOKU SOFTWARE,Ltd.[2]

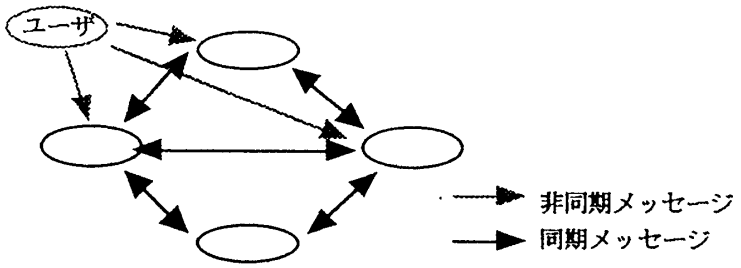


図2.通常のメッセージのやりとり

そこで、本開発では密接に関連するオブジェクト同士をオブジェクトグループとしてまとめ、オブジェクトグループ内では、直接メッセージを送る方式を採用した（図3）。ただし、同じオブジェクトグループ内にあるオブジェクトに対するメッセージパッシングでも、メッセージの流れを把握するためには重要なものにたいしては、制御オブジェクトを通すことにした。

また、この制御オブジェクトを状態遷移表で記述することにより、設計段階での検討もれを少なくした。

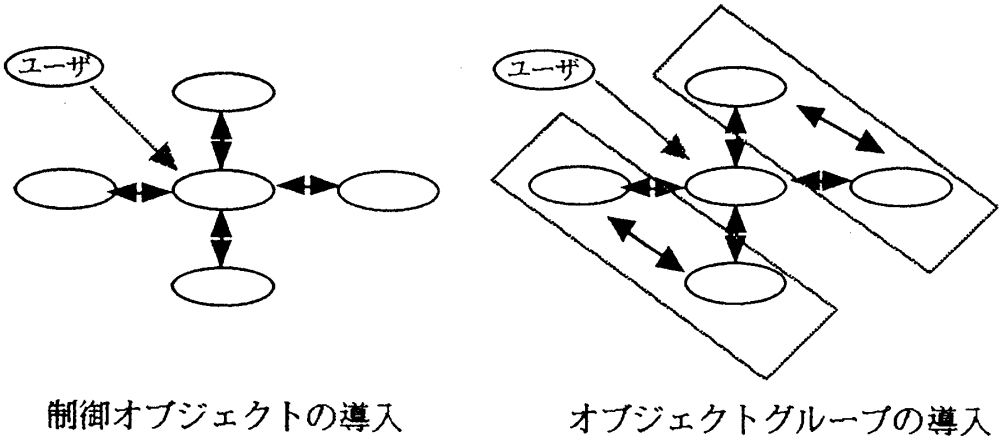


図3.制御オブジェクトの導入とオブジェクトグループ

5.評価

ＯＭＴ[1]では、制御方式の実装として、”手続駆動システム”、”事象駆動システム”、”並行タスク”の3通りをあげている。本例では、事象駆動システムを中心に手続駆動システムを混在させている。このため、事象駆動が簡略化されてよりわかりやすくなった。

今回の開発で作成したCASEツールは、速度の面で十分に現実の使用に耐えられるものになった。ただし、今回作成したものは、対話型のアプリケーションであるために、実行速度に関してシビアなOSや通信系のソフトウエアに適用する場合の検討が別途に必要である。

また、オブジェクトグループの概念を導入することにより、導入以前に比べてメッセージの流れを把握するのが容易になった。これは、メッセージの数が少なくなることや関連の深いオブジェクトが明確になるためである。

参考文献　[1]「オブジェクト指向方法論 OMT」Ｊ.ランボー他、トッパン