ＯＺ：オブジェクト指向開放型分散システムアーキテクチャ

6Q−4

－－－　ロボットの協調作業実験への応用　－－－

塚本享治（電総研）　澤田則和（ＶＩＣ東海）　笠井裕之（ＶＩＣ東海）
水谷功（住友電工）篠原弘樹（松下電器）近藤貴士（シャープ）梶浦広行（シャープ）

１．はじめに

異機種計算機間での通信プロトコルの標準化が進む中、その指標の１つとしてオブジェクト指向開放型分散システムＯＺを構築してきた。

しかし、これまでＯＺによる分散環境を評価するための適当なアプリケーションプログラムが存在しなかった。そこで、２台のロボットによる簡易協調作業への応用を試み、実装した（写真１）。

本稿では、オブジェクト指向型のＯＺ記述言語によるプログラムの構成・概要を述べ、更にその特徴の１つである拡張性を確かめるために実装後作業条件を変更したので、そのことについても述べる。

２．作業内容とシステム構成

2.1 作業内容

題材として取り上げた課題は、２台のマニピュレータが協力してハノイタワーを積み替える作業である。

この実験では、小さな円盤の上に大きな円盤を載せないという条件のもとで、任意の初期状態及び終了状態を与えることを許し、更に２本のマニピュレータが協調作業できるようにした。

2.2 作業環境の準備

この実験に使用するマニピュレータとしては、三菱電機製のムーブマスタ２台を使用し、相対させて設置した。

作業の対象となる円盤については、材質はアルミとし、重量を軽くするため、外部から見えない部分をくり抜いた。また、指先の稼働範囲以上の径の円盤が扱えるように、円盤の上部に小さい把手をつけた。

ムーブマスタ２台を相対させたときの共通稼働範囲は狭いので、特殊な作業台を作り、実験によってムーブマスタの設置場所を決めた。

2.3 ムーブマスタとのインタフェース

ムーブマスタ・コントローラは文字単位で指令と応答を交換するのに対し、ＯＺ実行系はパケット単位で指令と応答を交換する。本実験では、ＯＺ基本ソフトに、ムーブマスタ・ＯＺ実行系間のインタフェースの変換を行うデーモンを追加した。そのため、デーモンの起動／終了とパケットの送受信を行うメソッドのみを持つタイプを定義した。次にその下位タイプとしてタイプmovemasterを定義し、ムーブマスタの各種コマンドに対応するメソッドを備えた。

３．ＯＺ言語によるアプリケーションプログラムの作成

3.1 プログラム構成

3.1.1 主なタイプ

プログラムで用いる主なタイプについて示す。

①hanoi_monitor:	プログラムの全体管理
②planner:	状態入力・作業シーケンス作成
③scheduler:	単位作業要求に対する応答
④manipulator:	作業単位取得とmovemasterへの動作指令
⑤movemaster:	ムーブマスタの駆動
⑥tower:	円盤が置かれる支柱の状態管理

3.1.2 プログラムの流れ

インスタンスを単位とした流れを示す。以下、番号は［図１］中の番号と対応する。また、文中の各インスタンスは、前述の小文字による同名のタイプのインスタンスである。（例：HANOI_MONITORはタイプhanoi_monitorのインスタンスである。）

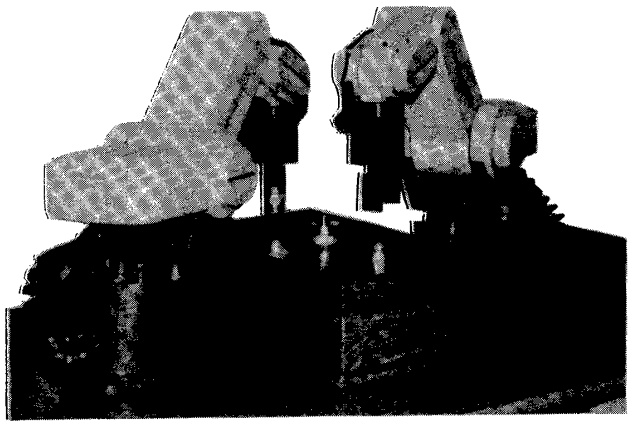

写真１．ムーブマスタ２台によるハノイタワー

OZ: Object Oriented Open Distributed System Architecture -- An Application to Cooperative Work by Robots
Michiharu TUKAMOTO(1), Norikazu SAWADA(2), Hiroyuki KASAI(3), Isao MIZUTANI(4), Hiroki SHINOHARA(5), Takashi KONDO(6), Hiroyuki KAZIURA(7)
(1)Electrotechnical Laboratory (2)VIC Tokai,Ltd. (3)VIC Tokai,Ltd. (4)SUMITOMO Electric Industries,Ltd. (5)MATSUSHITA Electric Industrial Co.,Ltd. (6)SHARP Corporation (7)SHARP Corporation

① HANOI_MONITORに作業開始メッセージを送る。
② PLANNERに作業シーケンス作成要求メッセージを出す。
③ 作業シーケンス，TOWERなどの初期状態を受け取る。
④ SCHEDULERに応答開始メッセージを出す。
⑤ 各MANIPULATORに始動メッセージを出す。
⑥ SCHEDULERに作業単位取得メッセージを出す。
⑦ 作業で使用するTOWERに占有予告メッセージを出す。
⑧ 作業単位を受け取る。
⑨ 作業で使用するTOWERに占有メッセージを出す。
⑩ MOVEMASTERに動作指令メッセージを出す。
⑪ 占有していたTOWERに解放メッセージを出す。

以下⑥〜⑪を繰り返すことによりデモが実行される。

3.2 プログラミングにおける拡張性

こうして作成したハノイタワー・デモ・アプリケーションを、ＯＺ言語の特徴がより明確になるように次の２点について改良を行った。

① 実験対象である円盤の把手は円筒状のもののみであったが、ムーブマスタが円盤を動かす際、把手の形状に合わせて動作を変えさせるため、従来の円筒状のものに加え、四角柱状のものを作ることにした。ムーブマスタは把手が四角柱の円盤を摑む場合、指先の構造上、面合わせをして摑まなくてはならない（図２）。プログラム上では、把手が四角柱の円盤のタイプは、従来の円盤のタイプを継承し、把手に関する情報を新たに加えて定義した。

② 上記の変更にともない、相対した２台のムーブマスタには、共通の動作部分と左右対称の動作部分ができるため、腕のオブジェクトに継承関係を持たせ、共通部分は上位タイプにまとめた。

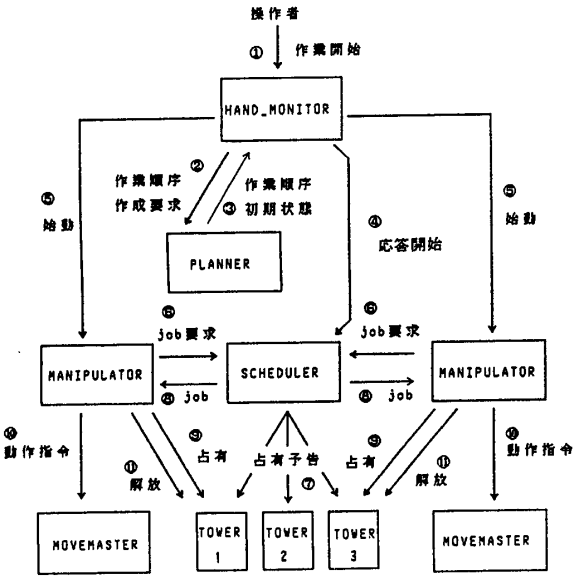

図１．インスタンスの構成

3.3 プログラムの工夫

２台のムーブマスタを使用して作業を行う場合、次のような問題が生じる。１つは、動作範囲の重複による衝突である。もう１つは、２台が並行に処理を進めるため、先に作業を受け取ったムーブマスタが実際に動作を起こす前に、通信速度の違いなどの影響で、後から作業を受け取ったムーブマスタが追い越して作業をしてしまう可能性があるということである。これらの解決策を、次に挙げる。

①衝突を避けるために、３本の支柱を中心とする３つの円筒状の空間を考え、それぞれの空間には２台のムーブマスタが同時に入ることができないように、相互排除を行った。

②SCHEDULERから取得した単位作業の実行順序の逆転を防ぐため、SCHEDULERは単位作業取得を要求されると自身をロックし、渡すべき単位作業で使用する２つの支柱（円盤の移動元、先）に対して占有予告を通知する。そののちロックを解除して単位作業をMANIPULATORに渡す。これにより、占有予告されていない支柱にムーブマスタが進入しても待たされる。

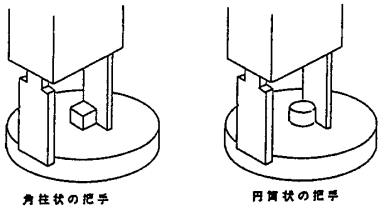

図２．ムーブマスタの指先と円盤の把手形状

４．むすび

以上、ＯＺを使って実験した、２台のロボットによる簡単な協調作業システムについて述べた。プログラムの作成に際し、3.2のようないくつかの拡張・改良を行ったが、タイプの継承機能を利用することにより、容易に変更を行うことができた。

参考文献

1)M.Tsukamoto etal.: The Architecture of Object-Oriented Open Distributed System: OZ, Interoperable Information Systems ISIIS '88, Ohmsha, PP153-166(1988,11)
2)塚本他: OZ:ｵﾌﾞｼﾞｪｸﾄ指向開放型分散ｼｽﾃﾑｱｰｷﾃｸﾁｬｰ ｵﾌﾞｼﾞｪｸﾄ指向型分散ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語とその実装、情報処理学会 ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語研究会、21-4 (1989,6)
3)塚本他: OZ:ｵﾌﾞｼﾞｪｸﾄ指向開放型分散ｼｽﾃﾑｱｰｷﾃｸﾁｬｰ LLCﾀｲﾌﾟ3を活用する通信ｱｰｷﾃｸﾁｬ、実装、およびその評価、情報処理学会 ﾏﾙﾁﾒﾃﾞｨｱ通信と分散処理研究会、43-4 (1989,9)
4)塚本他: OZ:ｵﾌﾞｼﾞｪｸﾄ指向開放型分散ｼｽﾃﾑｱｰｷﾃｸﾁｬｰ ｵﾌﾞｼﾞｪｸﾄ指向型分散ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語の複数ﾕｰｻﾞ環境への拡張、情報処理学会 ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ言語研究会、22-4 (1989,10)