2G−8　並列オブジェクト指向プログラミング言語WARASAに於ける並列分散自律オブジェクトの実現方法

江　允　　牧之内　顕文
九州大学工学部

1. はじめに

マルチプロセッサ計算機のためのソフトウェア開発環境が重要な課題として注目されている。我々は、メモリ共有型マルチプロセッサ計算機に於ける永続データ，分散データをオブジェクト指向パラダイムに従って容易かつ効率的に扱うため「出世魚」プロジェクトを推進している。その第3層のサブシステムとして、並列オブジェクト指向永続プログラミング言語WARASAを開発している[1]。

従来のWARASAは、並列機能しかサポートしておらず、オブジェクトがただ単一サイトマルチプロセッサ計算機上のみで並列実行している。その上に、「出世魚」のWAKASIサブシステムの機能を利用した分散処理機能を追加した。この機能によって、従来の単一サイトマルチプロセッサ計算機上で並列実行可能な自律オブジェクト群がそのまま多サイトのマルチプロセッサ計算機のマルチスレッドとして相互に同期をとりながら並列的に動作できるようになった。

本稿では，WARASAに於ける自律オブジェクトの並列・分散環境での実現方法について述べる。

2. 単一サイトでの並列機能

複数のオブジェクトが単一サイトのマルチプロセッサ上で相互に協調しながら並列に実行する応用プログラムを書けるようにWARASAでは、3種類のオブジェクトを導入した。自律オブジェクト(Autonomous Object),排除オブジェクト(Exclusive Object)、条件同期オブジェクト(Synchronous Object)がそれである。

自律オブジェクトは共有メモリ型マルチプロセッサ計算機での並列実行単位である。複数の自律オブジェクトがマルチスレッド上で並列的に実行できる。自律オブジェクト同士が排除オブジェクトを用いることによってデータを競合しでアクセスできる。また、自律オブジェクト間の同期をとる必要がある場合に（例えば、ある自律オブジェクトが他の自律オブジェクトの操作を持たなければならない時に）、条件同期オブジェクトにメッセージを出すことによって同期をとることができる。

排除オブジェクトは自律オブジェクト群の共有データをカプセル化したものであり、データの一貫性を保持するため、複数の自律オブジェクトから要求がなされる場合、一時にメッセージを一つしか受け付けない機能を提供している。

条件同期オブジェクトは自律オブジェクト群間のチャンネルとして、同期機能（wait, signalなど機能）を提供している。例えば、同期条件を満足できない時、実行している自律オブジェクトを一時停止させる。また、条件を満足したら、待ち状態のオブジェクトの実行を再開させる。

3. 多サイトでの分散処理

自律オブジェクト群を分散環境で実行するため、我々は、分散共有永続ヒープサーバWAKASIを用いてWARASAに分散処理機能を追加した。この機能によって、従来単一サイトマルチプロセッサ計算機で並列実行する自律オブジェクトは、複数台のマルチプロセッサ計算機がつながる分散環境でも並列・分散的に実行できるようになっている。しかも、自律オブジェクト群はこのような多サイト上で動いていながらあたかも単一サイト上で動いているように見せることが可能である。しかし、この機能を実現するには分散共有永続ヒープサーバ WAKASHIを利用することと条件同期オブジェクトに分散同期機能を追加することが必要である。以下には、WAKASHI の利用と分散機能の追加について簡単に述べる。

3.1 分散共有永続ヒープサーバの利用

分散共有永続ヒープサーバ WAKASHIは[2]，「出世魚」の一番低いレベルのサブシステムである。仮想メモリ方式と分散共有メモリ方式と結合することによって、WAKASHI は分散共有永続ヒープと分散共有揮発ヒープを提供してる。従って、WAKASHI サーバが提供する以下の機能を利用すれば、多サイトの分散環境で動作している自律オブジェクト群にデータを共有させることが可能となる。

（1）多サイトの複数のクライアントのそれぞれの仮想メモリ上にファイルと一対一に対応する分散共有永続ヒープを提供している。

（2）分散共有メモリ空間のデータが複数のクライアントにアクセスされる時に、そのページ状態の変換によってメモリコヒーレンスを保持している。

3.2 条件同期オブジェクトに分散同期機能の追加

自律オブジェクト群が多サイト上で動いていながらあたかも単一サイト上で動いているように見せることを実現するためは、条件同期オブジェクトに分散同期機能を持たせなければならない。そのため、条件同期オブジェクトに以下述べるような2つの機能を追加している。

（1） 分散同期変数をカプセル化する

分散同期変数は分散環境上でのサイト間の同期をとるために使われる変数である。まず、WAKASHI サーバを用いて多サイトに用いられる分散共有永続ヒープ上の変数として分散同期変数を定義する。そして、条件同期オブジェクトはその分散同期変数をカプセル化する。即ち、分散同期変数（例えば、例1のshared_t *shared_p）が条件同期オブ

A Realization Method on the Parallel and Distributed Autonomous Objects in the Parallel Object-Oriented Programming Language WARASA
Yun JIANG and Akifumi MAKINOUCHI
Faculty of Engineering Kyushu University

ジェクトの私的なデータとして追加され、分散同期変数に関する同期操作もうこの条件同期オブジェクトのメソッド中に追加される。このような分散同期機能を追加する条件同期オブジェクト（以下は分散条件同期オブジェクトと呼ぶ）がサイトごとに使われているので、分散同期変数を通じて異なるサイトでの同期がとられれる。例１に示すのは１つの分散条件同期クラスの宣言の例である。

例１：

```
sync class dis_sync{
   private:
       shared_t     *shared_p;
       mutex_t       sync1;
       conditiont_t  sig1;
       int           s_num;
       int           total_s;
       int           total_p;
       int           ready_a;
   public:
       dis_sync(int s_n, int t_s, int t_p);
       ~dis_sync();
       void broa_wait();
};
```

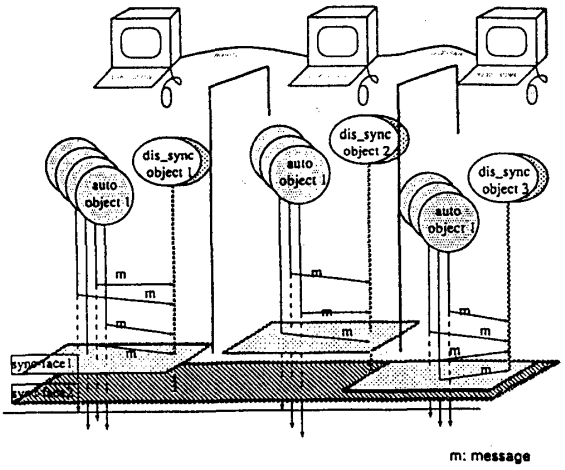

図１　条件同期オブジェクトの分散同期機能の例

（２）　２層の同期機能をサポートする

分散条件同期オブジェクトは、分散同期機能を提供するため、図１に示すような並列処理、分散処理の２層の同期機能をサポートしている。一層は同じサイトの複数のスレッド間の同期、もう一層は分散環境での複数サイトでのスレッド群の間の同期である。自律オブジェクトの単一サイトでの実行にオーバヘッドをかけないようにするため、２層同期の操作を併せて１つの同期メソッド（例１の broa_wait()メソッド）の中に定義する。さらに、並列かつ分散的に実行している自律オブジェクトの同期要求に対して、分散条件同期オブジェクトは、まず、Condition_t 型同期変数を用いて、サイト内の同期をとる。それは従来の同期とる方法と同じ[3]である。次に、分散同期変数を利用して、サイト間の並列・分散自律オブジェクト群間の同期をとる。

４．　並列・分散自律オブジェクト

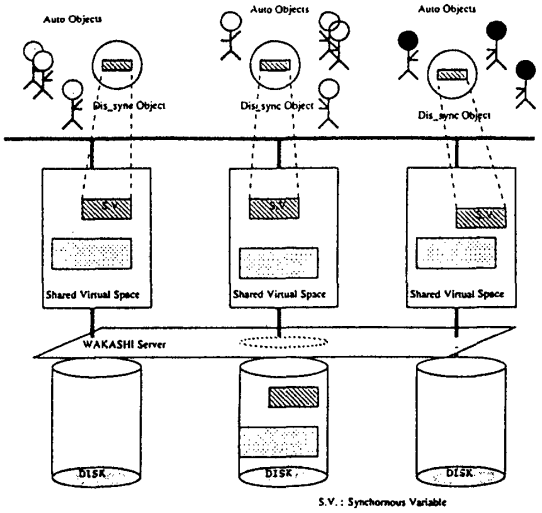

図２　並列・分散的な自律オブジェクトの実現

図２は多サイトマルチプロセッサ分散環境で実行されているWARASA自律オブジェクトの様子を示している。分散条件同期オブジェクトは分散環境での同期機能を提供しでいるので、自律オブジェクトは分散条件同期オブジェクトを用いれば、分散環境で並列かつ分散的に実行できる。また、分散条件同期オブジェクトの同期機能によって、自律オブジェクトが多サイトマルチプロセッサ計算機上で実行しても、単一サイト上の並列実行と分散環境上での並列実行の区別をする必要がない。自律オブジェクトは自分のサイトでの分散条件同期オブジェクトを通じて、他のサイトの自律オブジェクト同士とデータを共有しながら相互協調的に実行する。また。サイト間の自律オブジェクトがデータを互いに共有する時には、WAKASIの分散共有ヒープ上のデータを排除オブジェクトとしてカプセル化すればよい。

５．おわりに

WARASAの自律オブジェクトを複数のメモリ共有型マルチプロセッサ計算機を用いる分散環境で分散的に実行可能とする方法について述べた。WAKASHI の機能を用い、従来のWARASAの条件同期オブジェクトを拡張して分散同期機能を追加した。これを使えば、自律オブジェクトが複数台のマルチプロセッサ計算機上で動いていながらあたかも単一サイト上で動いているように見せることが可能である。WARASAの永続性の検討が今後の課題である。

謝辞　検討したくださった九州大学工学部の白氏に感謝します。

参考文献

[1] 江，牧之内：" 並列オブジェクト指向永続プログラミング言語WARASAの並列機能"，情報処理学会第４３回（平成３年後期）全国大会，４M−１１，平成３年１０月.

[2] G. Bai and A. Makinouchi: "Implementation and Evaluation of Persistent Data-Towards Virtual-Memory Databases", Proc. of Far-East Workshop on Future Database Systems, pp. 211-220, Apr. 1992.

[3] Y. JIANG and A. Makinouchi:"WARASA: An Enhanced C++ for Concurrent Programming on Shared Memory Multiprocessor Computers", Pro. of 16th IEEE Conf. on COMPSAC'92, pp. 257-262, Sep. 1992.