# 並列分散永続プログラミング言語 INADA

6G－6　富永浩之*　永淵美智男†　山本完‡　牧之内顕文‡
富士通九州通信システム（株）　（株）アドバングラフィックス　九州大学情報工学科　九州大学情報工学科

## 概要

近年の計算機システム環境の急速な進歩や利用形態の多様化にともない，応用プログラムに要求される機能も複雑かつ高度になってきている．出世魚プロジェクトの目的は，これらの変化に対応するためのプログラミングシステムの開発である．

『出世魚』全体は階層化されたシステムで構成される．その最下層にあるのが WAKASHI という，分散共有ヒープを提供するストレージサーバである．INADA は WAKASHI 上に実装される並列分散永続プログラミング言語である．INADA はオブジェクト指向プログラミング言語 C++ を拡張した言語で，共有，永続オブジェクトの扱いを容易にしている．

本論文では，並列分散永続プログラミング言語 INADA の実装方式について述べ，今後の検討課題を探る．

## 1　出世魚概要

出世魚は３つの階層からなっている．

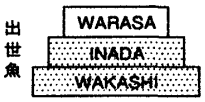


図 1: 出世魚の階層

まず，INADA のベースとなっている WAKASHI について簡単に述べる．前述したように，WAKASHI とは分散共有ヒープを提供するストレージサーバである．具体的にはヒープ領域の一部を，ネットワーク上で共有可能にする機能と，そのヒープ領域を永続化させる機能を持つ．また，ロック，トランザクション，リカバリなどのＤＢ機能も有する．INADA は，ユーザがこれらの機能をプログラミング言語レベルで扱えるようにするためのものである．

## 2　INADA

INADA は C++ 言語をベースとし，様々な機能追加を行なっている．以下にそれぞれの拡張機能について簡単に述べる．

### 2.1　GH(Global Heap) 領域

従来の C++ におけるヒープ領域に加えて，INADA では２種類の分散共有可能な GH(Global Heap) 領域を提供する．１つは永続化可能な PGH(Persistent GH) 領域で，もう１つは揮発な VGH(Volatile GH) 領域である．これらの GH 領域は WAKASHI の分散共有メモリの機構を利用し，実現している．GH 領域のイメージを図 2に示す．

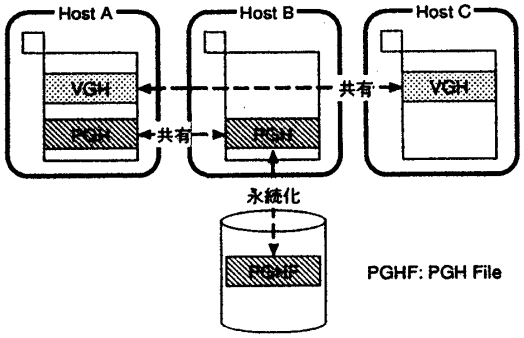


図 2: GH 領域の共有と永続化

### 2.2　GO(Global Object) と GP(Global Pointer)

上記の GH 領域に配置される GO と，GO を参照するための GP を提供する．GO は GP による参照のみで，従来の C++ の構文を踏襲している．また，GP の宣言は従来のポインタ宣言の前にキーワード"global"を記述することにより可能となっている．例えば，

```
global Person *a;
```

のように記述する．

### 2.3　並列分散処理

INADA ではオブジェクトのメソッドを他サイトで実行できる．ユーザは"parallel do"文と Agent クラスによりこの機能を利用することが可能となる．

並列分散処理機能を実装するにあたって，今回は INADA Server と Method Server（後述）を導入した．

### 2.4　マルチタイプオブジェクト

マルチタイプオブジェクトとは，１つのオブジェクトに対して複数の見方を与える機能のことである．例えば，クラス Person のインスタンスに対して，クラス Employee の属性を付与したり，削除したりすることが可能となる．ユーザは，"transforms"，"as"，"of"等のキーワードによりこの機能を利用することができる．

### 2.5　検索処理

GO を再利用したい時に，検索処理が必要となる．INADA では"for all"文を提供し，それを利用することにより，型による検索，名前による検索等ができるようになっている．例えば以下のように記述する．

```
for all Employee* x in eset such that
  x->age >= 50 && x->dept == "design"
    do {x->salary += 100;}
```

上記の例では，集合体 eset に含まれる Employee 型のインスタンス x で，such that 以下の条件を満たすインスタンスに対して，do 以降を実行する．

---

Parallel, Distributed and Persistent Programming Language INADA

*Hiroyuki Tominaga, Fujitsu Kyushu Comunication Systems Ltd.

†Michio Nagafuchi, Advanced Graphics, inc.,

‡Kan Yamamoto & Akifumi Makinouchi, Department of Computer Science and Communication Engineering, Kyushu University

## 3 実装

図3にINADAのコンパイル方式を示す．INADAは，拡張C++言語であるので，C++コンパイラのプリプロセッサとして実装している．その時，INADA独自の機能はINADA Libraryとして提供しリンクさせている．

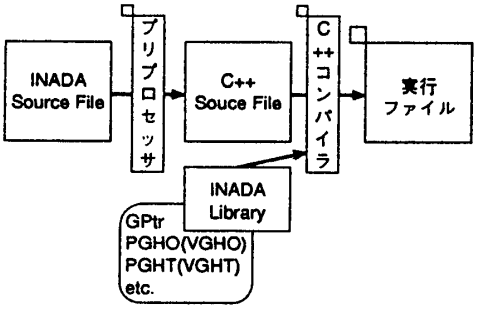


図 3: 実装概要

今回，Solaris 2.3 上にINADAを実装した．以下にそれぞれの機能の実装方式について述べる．

### 3.1 GH領域管理

GH領域は図4のような構成になっている．以下にGH領域における各部分について簡単に述べる．

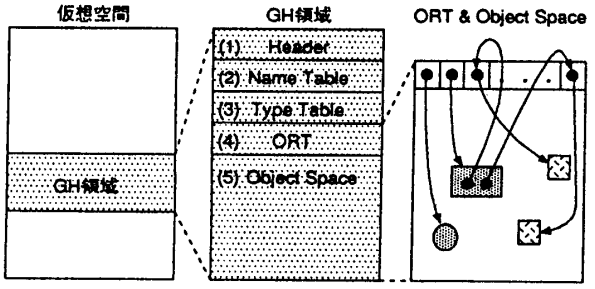


図 4: GH 領域

1. Header
   GH領域における各部分の位置情報などを保持している．
2. Name Table
   Named Object のための名前を格納する．
3. Type Table
   GH領域に格納されるオブジェクトの型名を保持する．型名を保持することにより，"for all"文での型名による検索が可能となっている．
4. ORT(Object Reference Table)
   GH領域を仮想空間の任意の位置にマッピングできるように，オブジェクトの参照は間接参照を行なっている．ORTは間接参照を実現するためのテーブルである．
5. Object Space
   GOが格納される領域．ORT領域が不足した場合，この領域内に追加ORT領域がとられる．

GH領域の管理は，PGHO(or VGHO)クラスにより実装される．そのクラスのインスタンスが，個々のGH領域を表す．また，前述したGH領域内の各部分についてもクラスとして実装される．

### 3.2 INADA Server

INADA Serverは，メソッドの並列分散実行のために各ホスト上で動作している．具体的に，並列分散処理をどのように実装したかを図5を使って述べる．

まず，ユーザはAgentクラスのインスタンスを生成することにより，メソッド実行の準備要求を行なう．Agentクラスのインスタンスが生成された時，RPCによりINADA Serverと通信を行ない，Method Serverの起動とMethod Serverと通信を行なうためのポート番号の取得を行なう．そして，parallel do文により，メソッド実行をMethod Serverに対して依頼する．

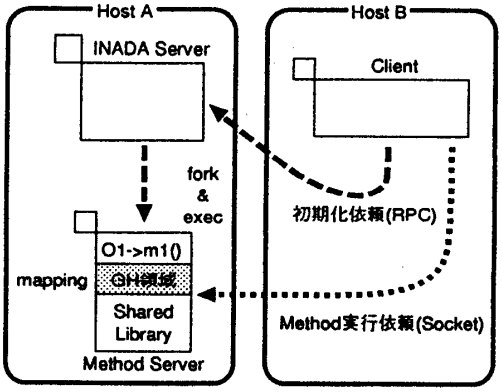


図 5: INADA Server

### 3.3 for all

前述したように，INADAではGOに対する検索処理を提供している．"for all"文が検索の対象としているのは，集合体のみである．ここで言う集合体とは，INADAが規定しているインタフェースをもつもののことを指す．例えば，INADAが標準で提供している 集合体クラスライブラリとGH領域を表すクラス(PGHO, VGHO)は"for all"文でオブジェクトを検索できる．\par すなわち，INADAが規定するインタフェースを利用すれば，\verb"for all"= 文をプリプロセッサによりC++に展開できる．

## 4 おわりに

今後の課題を以下に挙げる．

- Method Server の Immigration
  現在，Method Serverは予め実行するホスト上に用意する必要がある．今後はMethod実行コードのImmigrationを検討していく．
- 評価及び改善
  実装したINADAの評価を行ない，改善する．
- マルチメディアデータベースとしての可能性
  今後，ネットワークの高速化にともないマルチメディアデータベースの需要が高まってくると思われ，出世魚をマルチメディアデータベースとして，今後研究を進める．

また，本研究を進めるにあたって御助言，御協力頂いた九州大学工学部情報工学科牧之内研究室の皆さん，富士通研究所の龍主管研究員グループの皆さんに感謝致します．