IN86－130

論理型言語を用いたサービスベースシステム

THE SERVICE BASE SYSTEM IMPLEMENTED WITH LOGIC PROGRAMMING LANGUAGE

荻野　正　　田中　英彦
Tadashi OGINO　　Hidehiko TANAKA

東京大学　工学部
Faculty of Engineering, University of TOKYO

Abstract - In this paper, we describe the basic concept of the Service Base System (SBS) and explain the experimental system using logic programming language (prolog).

Using the Service Base System, a user can integrate and use services distributed in computer networks without knowing where the services actually exist. In the Service Base System, the specification of necessary services must be described beforehand. We also propose a new method to describe computer resources such as program and data.

1．はじめに

最近の計算機技術の発達により、複数の計算機をネットワークを介して接続し、全体として一つのシステムとして使用することは、もはや常識となりつつある。

この分野の研究は、大きく広域網に関する研究と、ＬＡＮ等の局所網に関する研究とに分類することができる。広域網に関する研究としては、異なる機種の計算機同士を接続する場合のプロトコルの変換、ファイル転送、リモートジョブエントリー等、計算機同志を接続するというレベルの研究が主であり、また、最近脚光を浴びているＬＡＮに関する研究では、その構成法や通信方法に関する研究が主流となっている。[1] しかし、様々な計算機が数十台、数百台と接続された時に、その巨大なシステムをユーザがどのように使用するかについての研究はあまり行なわれていない。

本稿では、複数台の計算機をネットワークで接続したという環境の下で、

・ユーザは、各計算機の提供する機能を、その分散性にわずらわされずに、自由に組み合わせて使用する事ができる。

・網中の各ノードでは、他の計算機とは独立に機能の拡張を行なうことができる。

等を目標として開発しているサービスベースシステム（ＳＢＳ）について発表する。

2．サービスベースシステムの概要[2]

2.1 サービスベースシステムの目標

複数台の計算機がネットワークで接続されているという状況で解決しなければならない問題は、大きく次の２つに分けることができる。

・分散性

　計算機資源が網中に分散して存在する。

・異種性

　各ノードの計算機に相違がある。

ＳＢＳは、

①計算機－計算機間及びユーザ－計算機間の論理的な通信をすべてサービスの要求と応答というシンプルなモデルに統一する。

②サービスに関する情報を三層ビューという構成で管理する。

事によって上記の問題を解決しようというものである。

2.2 サービスの定義

ＳＢＳでは、計算機がユーザに提供する機能をすべてサービスと呼ぶことにする。サービスは模式的に、「作用と、作用の対象となるデータを与えることによって提供される機能」ととらえることができる。

ＳＢＳでは、「データ」とは、構造と静的な性質を持ったものとして定義する。構造としては、Relationとstreamに限定して考える。Relationの場合は、attribute, domain,key等を決定する事で構造を定義することができる。streamの場合は、読み書きする単位となるデータの集合や、並び方に対する規則を決定することで構造を定義する。静的な性質とは、そのデータの名前とか所有者、アクセス権、データの内容等構造に反映されない性質を指す。

「作用」とは、０個以上のデータを入力として与えた時に、１個以上のデータを出力として得るものとし、入力データとして許されるデータの記述の集合と、出力データの記述の集合及び、静的な性質に関する記述で定義する。静的な性質に関する記述とは、名前、実行環境、意味の記述等を指す。通常我々が使用している計算機では、作用はコマンドに、データはファイルに対応している。

サービスの要求とは、「作用」と「（対象となる）データの集合」を指定する事である。すべての計算機の機能が、このサービスの要求のみで実現できれば、細かい使い方の差異は吸収することが可能である。各ノードは、自計算機に存在するサービス及び自計算機を介して使用することのできるサービスに関する情報を予め持っていなければならない。そして、ユーザ或いは他の計算機からのサービス要求があった時、自計算機に存在するサービスであれば自計算機で実行し、別の計算機に存在するサービスであれば、新たにサービス要求を行なう。この時、ユーザがサービスの実際の存在場所を知らなくていいのと同様に、各計算機はどの計算機にサービス要求をすればよいかを知っているだけでよく、サービスの存在場所は知らなくてもよい（図１）。

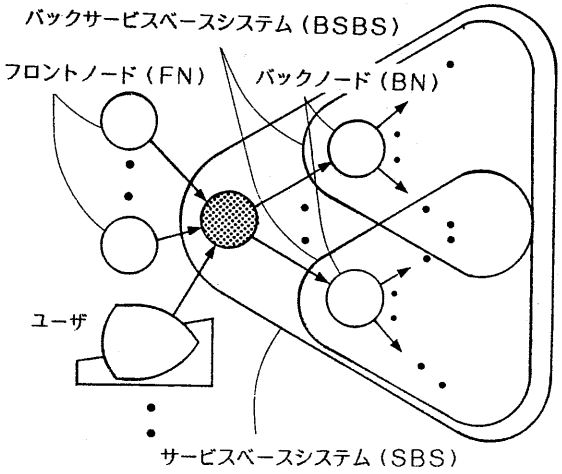

図１　ＳＢＳの構成

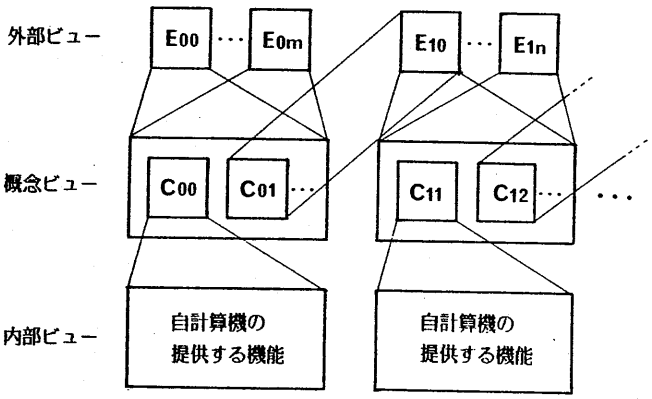

図２　ＳＢＳの三層ビュー

2.3 三層ビュー

各計算機では、サービスに関する情報を次の３つのレベルに分けて記述、管理する（図２）。

①外部ビュー
②概念ビュー
③内部ビュー

内部ビューは、各計算機が独立に提供するサービスに関するビューであり、各計算機に１つだけ存在する。

概念ビューは、自計算機の内部ビューと他の計算機の外部ビューを統合したビューであり、分散性をここで吸収する。

外部ビューは、その計算機を使用するユーザあるいは他の計算機に見せるビューであり、それぞれのユーザあるいは計算機ごとに存在する。

この様に、サービスを三層ビューという形で管理している為、各計算機では他の計算機とは独立にサービスの拡張を行なうことができる。

３．サービスベースシステムの構成

サービスベースシステムの各ノードの計算機の構成は図３（次ページ）のようになっている。各計算機には、その計算機固有のＯＳ、及びＤＢＳが予め存在する。これらは、その計算機に局所的な機能という意味でローカルＯＳ（ＬＯＳ）、ローカルＤＢＳ（ＬＤＢＳ）と呼ぶ。しかし、ＬＯＳ、ＬＤＢＳが分散環境を意識した機能を提供することを否定するわけではない。また、他の計算機との通信を行う為の通信機構をＣＭＳと呼ぶ。異なるコード体系の変換や、ファイル転送等の機能はＣＭＳが提供する。

ＳＢＳでは以上の部分は予めそれぞれの計算機に存在するという仮定をおく。これらの上にＳＢＳを構築するために、サービスの処理系とサービスの記述管理部を設

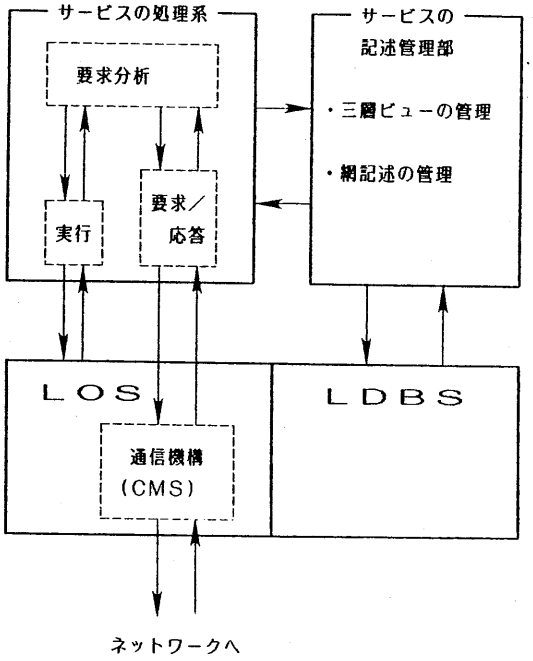

図3　SBSのノードの構成

ける。

サービスの処理系は、

・サービスの要求を受けとり、

・サービスの要求を分析し、

・自計算機に存在するサービスであれば、環境を設定し、

・サービスを実行する。

・他の計算機に存在するサービスであれば、サービスの要求を行ない、

・サービスの応答を待つ。

という処理を行なう。

サービスの記述管理部は、

・三層ビューの管理（サービスの検索、追加、変更、削除）

・網に関する情報（ノード名、ノードＩＤ、コストなど）の管理

・サービスの組み合わせに関する記述の管理

等を行なう。実際のデータは、ＬＤＢＳに存在する。

サービスの記述管理部を記述する言語に特に制限はなく、関数型言語を用いた実験も行なった。本研究では、「サービス記述＝計算機の機能に関する知識」ととらえ、知識処理に適している論理型言語を用いて記述するものとする。また、サービスの処理系も、記述管理部とのインタフェースを容易にするため同じ言語で記述する。

４．実験システム第１版の構成

ここでは、ＳＢＳの有効性を示すために構築した実験システム第１版の構成を示す（図４）[3]。実験システムは、東京大学大型計算機センターのＭ２８０Ｈ（現在Ｍ６８０Ｈ）、ＶＡＸ－１１／７８０（現在ＶＡＸ８６００）、及び、当研究室のＶＡＸ－１１／７３０を接続して構成した。ＯＳは、Ｍ２８０ＨがＶＯＳ３、２台のＶＡＸがＵＮＩＸである。Ｍ２８０Ｈ－ＶＡＸ－１１／７８０間の通信用ソフトウェアとして、ＣＶＯＳ及びＣＶＯＳ２（ＣＶＯＳ２は現在使用不可）を使用し、またＶＡＸ－１１／７８０－ＶＡＸ－１１／７３０間の通信用にｐｃｏｍというソフトウェアをＣ言語で開発した。処理系の記述言語としてはＣ－ｐｒｏｌｏｇ及びＣ言語を使用した[4]。

この実験システムでは、サービスに関する記述は、

ｓｅｒｖｉｃｅ（サービス名、属性名、属性値）

というｐｒｏｌｏｇのｆａｃｔの形で記憶されている。属性としては、次のものが実装されている。

ｎａｍｅ	サービス名
ｍａｐ	他のビューでのサービス名

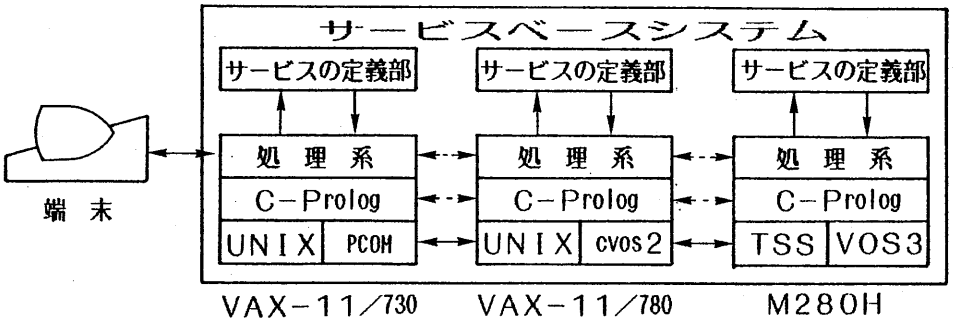

図4　実験システム第１版の構成

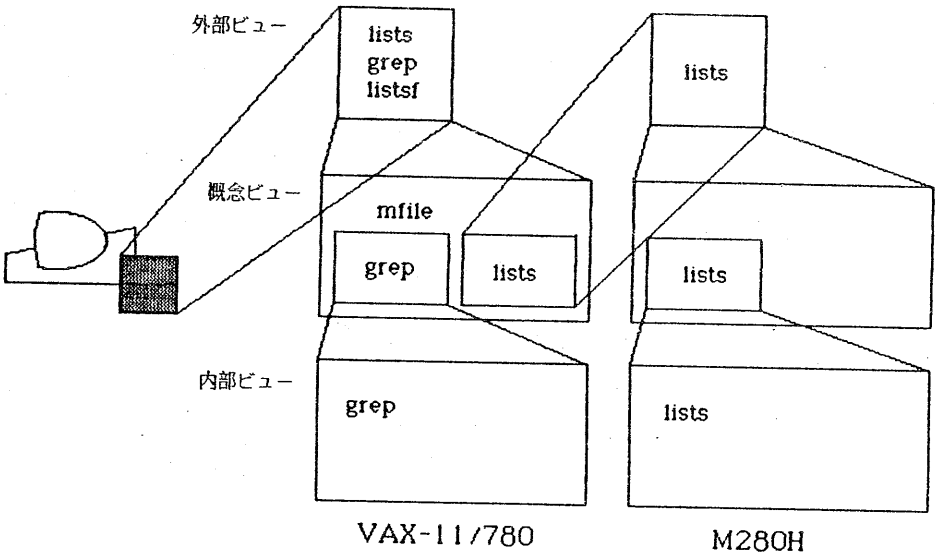

図５　三層ビューの構成例

```
| ?- listsf('%',temp),grep('VDATA',temp).

PO        114         99     ARCHIV  B0595.LISPLIB.COMP.VDATA
PO         95         38     ARCHIV  B0595.LISPLIB.VDATA
PO         95         28     LD0005  B0595.PROLOG.VDATA
PO         95         71     LD0024  B0595.PROLOG.VDATA.OLD
PO        171        113     ARCHIV  B0595.SB.VDATA
PS         19         18     ARCHIV  B0595.SBLIB.VDATA
PO         19         10     ARCHIV  B0595.TEST.VDATA
PO         76          3     LD0023  B0595.UTIL.NEW.VDATA

listsf('%',temp),grep('VDATA',temp) is true?
```

図６　実行例

ｐｌａｃｅ　　サービスの存在場所
ｏｕｔ　　画面への出力の有無

また、サービスの登録をする為の述語として次のものを用意した。

（１）自計算機のサービスの定義
ａｓｓｅｒｔ＿ｉｎｔ（サービス名、
内部ビューでのサービス名、
属性名（属性値）のリスト）

（２）自計算機のサービスの定義
ａｓｓｅｒｔ＿ｅｘｔ（サービス名、
外部ビューでのサービス名、
サービスの存在場所、
属性名（属性値）のリスト）

以下にサービスの定義の例を示す。[5]

ＶＡＸ－１１／７８０での定義

```
assert-int(grep(X,Y),grep(X,Y),[out(stdout)]).
assert-ext(lists(X),lists(X),1,[]).
listsf(X,F) :- mfile(lists(X),F).
```

Ｍ２８０Ｈでの定義

```
assert-int(lists(X),lists(X),[]).
```

これは、三層ビュー上で示すと図５のようになる。この時のサービスの実行例を図６に示す。

この実験システム第１版によってＳＢＳという構成で分散しているサービスを自由に利用できることは示せた。しかし、この実験システム第１版では、次に示すようないくつかの不十分な点がある。

・サービスの記述として、その名前と存在場所に関するものしか実装されていない。

・サービス記述が属性とその値というｆｌａｔな構造になっており、複雑な情報を記述することが難しい。

・Ｍ６８０Ｈを通常のユーザとして利用しているため、複数のプロセスの起動ができない等、制限が多い。また、コマンド体系やファイルシステム等が繁雑すぎる。

・Ｍ６８０ＨとＶＡＸ－１１／８６００の間の回線が

本しかないため要求と応答の関係が何重にもなると、新しいプロトコルを決めないとうまくいかない。

そこで、サービス記述に関する新しい方法について考察し、より柔軟性のある（インプリメントしやすい）実験システム上でその実装と評価を行なっていくことにした。

5．サービス記述[6]

5.1 サービスの記述内容

サービス記述の方法について検討する前に、どのような情報を持っていなければならないかについて考える。前述のようにサービスは作用とデータに分けて考えることができる。

まず、データについて知らなければならない情報とは、それが「どのようなデータ」であるかという情報であり、「どのような」とは、ユーザ側からみれば実験結果であるとか、英語で書いた論文であるとかいう『データの意味』を指し、計算機側からみれば、『物理的な存在場所』とか『データのフォーマット』などを指す。また作用について知らなければならない情報とは、「どのような入力データに対して、どのような処理を施して、どのような出力データを得るか」という情報であり、ユーザ側からみれば実験データの平均と分散を求めるとかソースプログムをコンパイルして実行可能なモジュールにするとかいう『作用の意味』に関する情報であり、計算機に必要なのは『入力ファイルのフォーマット』とか『実行の方法』に関する情報である。この様にユーザに提供する情報と計算機が使用する情報があり、前者をdictionary、後者をdirectory と呼ぶ。dictionaryとdirectory は厳密に区別することは困難であるが、ここではサービスに関する情報としてはdictionaryとdirectory があるものとする。

5.2 サービスの記述方法

次にdictionaryとdirectory の記述方法について考察する。以前の実験システム第１版の方法でいくつかのサービスを記述した結果からは経験的に次の様な傾向が得られている。

・属性の種類は非常に多く、かつその必要性はかたよっている。

・属性の値として複数の値をとる場合がある。

・「～の～の～の値は～である」等ネストした関係を記述したい場合がある。

そこで、サービスに関する情報を次のようなリストの形式で持つことにする。

属性名	属性値	内容
data-description	自然言語	データの意味
func-description	自然言語	作用の機能
arg-infs	リスト	入出力データの内容
e-name	サービス名	外部ビューでの名前
c-name	サービス名	概念ビューでの名前
i-name	サービス名	内部ビューでの名前
combination	ｐｒｏｌｏｇ	組み合わせ方
data-structure	リスト	データの構造
att-name	属性名	属性名
domain	ドメイン名	ドメイン名
type	ｆｉｌｅ属性	テキスト、オブジェクト等
lang	記述言語	Ｃ、ｆｏｒｔｒａｎ等
permission		パーミッション
format	リスト	ｆｉｌｅのフォーマット
owner	ｕｓｅｒ名	所有者
update-time	時間	変更時間
environment	リスト	環境
execute-way		実行方法

表１　属性の例

（属性名　属性値１　属性値２　…　属性値Ｎ）

この形式では、属性の値をリストで持っているので、複数の値を持つような属性についても記述ができる。また、属性の値として再びリストを持ってくることにより、ネストした情報の記述も容易である。不必要な属性に関しては、記述しなくてもよい。現在、属性として考えているものを表１に示す。

ここで、例としてＵＮＩＸのＣコンパイラを起動するコマンドｃｃを考える。ｃｃを実行する時に与える入力データは、Ｃまたはアセンブラで書かれたテキストファイルであるか、すでにコンパイルされているオブジェクトファイルのどれかである。この、入力データに関する情報を上の記法で記述すると、

(arg-infs (arg-inf①　(type text) (lang c asm②))
　　(arg-inf③　(type object)))

となる。②の部分は、「Ｃまたはアセンブラ」のＯＲ関係を示しており、①と③で「～テキストファイル」と「～オブジェクトファイル」のＯＲの関係を示している。また、属性同志の関係（上の例では、arg-infsとarg-inf 、arg-inf とtypeなどの関係）は、システムに固有の関係であり、これも上と同様の記法で記述してある。そのため、別のシステムで実装する場合には、この記述を変更する事で対応でき、他に大きな変更を行うことなく

外部ビュー

サービス名	分類	dictionary	directory

概念ビュー

サービス名	分類	ノードＩＤ	dictionary	directory

内部ビュー

サービス名	分類	directory

図７　三層ビューのＲｅｌａｔｉｏｎ

容易にＳＢＳを構築する事ができると思われる。

６．実験システム第２版の実装

新しい実験システム第２版の実装について述べる。実験システム第２版は、当研究室の２台のＶＡＸ－１１／７３０から構成されている。このうちの１台は、旧実験システムと同じものであり、必要があれば大型計算機センターのＶＡＸ８６００、Ｍ６８０Ｈとも接続することが可能である。ＯＳは、ＵＮＩＸ　４．３ＢＳＤである。４．３ＢＳＤは、ネットワークに関する機能が強化されており、ｓｏｃｋｅｔという通信プリミティブを使用することにより、分散したノード間のプロセス間通信を比較的容易に行なうことができる。処理系の記述言語としては、Ｃ－ｐｒｏｌｏｇにｓｏｃｋｅｔの機能の一部を追加したｄｐｒｏｌｏｇを使用している。ただ、ｄｐｒｏｌｏｇでは、ノード内のプロセス間通信しかサポートしていないので、ノード間の通信用プロセスはＣ言語で記述する。

３層ビューに関する部分は、ｄｐｒｏｌｏｇで関係データベースシステムを作製し、この上で管理をしている。この関係データベースでは、１個のタプルは、

relation名（値１、値２、…値Ｎ）

のというfunctor の形式でｐｒｏｌｏｇの内部データベースに蓄えられている。各ビューに対応するrelationの構成は図７の様になっている。dictionary、directory に関しては実際の記述はファイルに存在し、attribute の値としては、そのファイル名がはいる。また、directory の記述は５章で示した方法によっているが、dictionaryについては自然言語で記述されており、計算機がその内容を直接扱うことはしていない。

７．今後の課題

今後の課題としては、新しい実験システム第２版の上で、

・サービス記述／管理用ｔｏｏｌの作成
・サービスの実装によるサービスの記述方法の検討

を行なっていく予定である。また、属性に関する情報をメタな情報としてまとめることにより、ＳＢＳの処理系を変更せずに、このメタな情報の変更のみで別の計算機に対応する方法についても検討する予定である。

さらに、現在はユーザが利用しているサービスの意味に関する情報を計算機に扱える形にすることにより、ユーザの要求から推論をして、最適なサービスを選択して実行することも可能になると思われる。

参考文献

［１］Stalling,W.,“Local Networks”,Computing Surveys,Vol.16,No.1,March,1984

［２］深沢他、「サービスベースシステムの概念と基本構成」、ＥＣ８２－４４

［３］荻野他、「サービスベースシステムにおける論理型言語向きサービス記述」、第２９回情報処理学会全国大会、６Ｈ－７

［４］荻野他、「論理型言語を用いたサービスベースシステムの実装」、第３１回情報処理学会全国大会、７Ｑ－７

［５］荻野他、「サービスベースシステムに於けるサービスの実装」、第３２回情報処理学会全国大会、３Ｄ－８

［６］荻野他、「サービスベースシステムにおける分散情報管理」、第３３回情報処理学会全国大会、３Ｕ－３