SERVICE BASE SYSTEM BASED ON FUNCTIONAL LANGUAGE

TOWARDS THE INTEGRATION OF DISTRIBUTED SYSTEM

Tadashi OGINO, Tomoo FUKAZAWA, Hidehiko TANAKA, Tohru MOTO-OKA.

Faculty of Engineering, University of Tokyo

The purpose of Service Base System is to integrate distributed services in the network with keeping the autonomy of each system in the network. Each computer communicates with each other, when it is needed in processing the service which is requested by the user in the mixed form of distributed services. To implement the SBS, we used lisp as the interface language between computer and computer, and human and computer. In this paper, the organization of service base management system, and the implementation of an experimental system are described.

1. はじめに

将来的計算機システムでは、ユーザは、蓄積されたサービスを組合せて使用するといった利用形態が増加すると思われる。こういった計算機で、網を構成した場合、ユーザは、サービスの分散性を意識することなく、網中のサービスを組合せて利用できることが望ましい。一方、拡張性を考えると、各計算機で独立にサービスの蓄積が行なえることが望ましい。そこで、計算機網中に分散して存在する種々のサービス（プログラムやデータ）をユーザに提供する時に、各計算機の独立性を保つつつ、かつ、ユーザには分散性を意識させないユーザインタフェースを提供することを目的としたシステムをサービスベースシステム（SBS）と呼び、研究を進めている。

SBSでは、計算機-計算機間通信、及び、人間-計算機間通信をすべて、サービスの要求と応答として取り扱っている。このサービスの要求と応答を形成する言語として、関数型言語を用いたSBSの実現システムを構成したので、本稿で紹介する。

2. サービスベースシステムの概念
2.1 サービスのモデル

計算機の提供するサービスは、「データ」に「作用」をほどこすことである。したがってSBSでは、サービスを「データ」と「作用」にわけて取り扱う。また、データと作用を組合わせた「サービス」の単位でも取り扱う。

2.2 SBSにおける網のモデル

SBSでは、各計算機は、その計算機が提供するサービスと、その計算機を通じて利用可能な全ての計算機が提供するサービスを、ユーザが区別なく利用できる環境を提供する。ここで、向計

バックサービスベースシステム（BSBS）

フロントノード（FN） バックノード（BN）

ユーザ

サービスベースシステム（SBS）

図1 SBSにおける網の論理モデル
算機Nに着目した場合、Nに要求を出する計算機をFN（Front Node）と呼び、Nが要求を出す先の計算機をBN（Back Node）と呼ぶ。BSNでは、BNは、BNの提供するサービスと、BNを通じて利用可能なすべての計算機の提供するサービスを、計算機Nが区別なく利用できる環境を提供するように構成される。即ち、BNが計算機Nに対してBSNを構成しているわけであり、これを計算機NのBSN（Back Service Base System）と呼んでいる。計算機Nは、BNから見るとFNのBNに見え、計算機Nの構成するBSNは、FNからは、BSNと見える。これらの関係は、形式的には、以下の様に書ける。

\[
\begin{align*}
BSN(N) &= BSN(BN_i(N)) \\
BN_i(N) &\in BSN(N) \\
BSN(N) &= N \cup \cup BSN_i(N) \\
FNS(N) &= (K \rightarrow N) \\
FN_i(N) &\in FNS(N)
\end{align*}
\]

但し、I→Jは、計算機Iから計算機Jにサービス要求が可能なことを示す。BSNでは、この関係（上記の→関係）による論理的な関係を考えるが、これは、一般的には、物理的な構成とは独立に構成される。尚、BSNにおいて「サービス要求が可能である」との定義は後で述べる。

BSN_i(N) は、ノードNのBSNの1つを構成するノードの集合を表す。

BSN(N) は、ノードNの全てのBNからなる集合（直接サービス要求を行なう要求先の計算機の集合）を表す。

BN_i(N) は、ノードNのBNの1つを表す。

BSN(N) は、ノードNが構成するBSN内に含まれる全てのノードの集合を表す。これらの式から、BSN(N)を構成する為には、各ノードの全てのBN(N)上の情報を全てのBN(N)上の情報だけを知れば十分であることが暗黙に示唆される。

FNS(N) は、ノードNの全てのFNからなる集合を表す。

FN_i(N) は、ノードNのFNの1つを表す。

2.3 各ノードの論理構成

BSNの各ノードの主な構成要素は、「サービスの処理系」と「サービスの定義部」である。

・処理系の機能

処理系は、定義部の記述に基づいて、サービス要求を解釈し、自ノードが提供するサービスは、自ノードで実行し、BSNに存在するサービスはBNにサービス要求を行う応答を解釈する。この様にBSNでは、各計算機が自動的にサービスの存在する計算機にサービス要求/応答を繰り返すことにより、分散して存在するサービスを組合せたサービスを実行する。

・定義部の構成

BSNではサービスを次の3層にわたって記述する。

i）内部ビュー
各計算機が単独で提供するサービスの記述

ii）概要ビュー
自計算機の提供するサービスと、BSNの提供するサービスを統合したサービスの記述

iii）外部ビュー
自計算機を通してFNに提供するサービスの記述

[図3 サービスの定義部の構成]

1) は,
①各計算機のOSの提供する機能（コマンド等）
②各計算機のOSに管理されているサービス（ロード・モジュール）
③処理系が直接実行するサービス（粗詰込みサービス）

からなる。

ii）は、①基本サービスと②合成サービスとに
分類できる。①は、自動製造機の内部ピューを取扱う、BNの外部ピューに記録されたサービスを、概念ピュー上で定義したサービスであり、②は、自動製造機の概念ピュー上のサービスを組み合わせて、新たに定義するサービスである。

③（もへ）、④基本サービスと⑤合成サービスに分類でき、⑤は、自動製造機の概念ピュー上で定義されているサービスの定義であり、⑤は、自動製造機の外部ピュー上のサービスを組み合わせ定義するサービスである。（③）⑤のピューは、各計算機内のピューを一つずつ用意するが、⑤は、各計算機内、その計算機のFN、あるいはその計算機を直接使うユーザ毎に一つずつ用意する。

以上のように、各計算機の概念ピューは、自動製造機の内部ピューとBNの外部ピューから構成される。BNでは、他機に3層にわたってサービスが記述されているので、BNの外部ピュー上には、BSBSが自動製造機に提供する全てのサービス記述されていることになる。即ち、各ノードは、自動製造機に関する情報と、BN上の情報だけを知っていればよい。

前節で述べた、あるノードが「BSBSを構成している」とは、外部ピューを提供することであり、「サービスを要求することができる（→）」とは、他計算機の外部ピューを自動計算機上の概念ピュー上で定義しているということになる。この様にピューを構成することにより、前節で述べた論理的な構成が構成されることができる。

2.4 BSBSにおける言語

BSBSを実際に作成する為には、以下の言語が必要である。

(i) 処理系記述言語
(ii) サービスの実体の記述
(iii) サービス要求言語
(iv) サービス定義言語
(i) は、
・OSとのインタフェース
・下位通信機関とのインタフェース
・サービスの管理制御
・サービス要求言語の解釈

図4  計算機間の論理的な通信（要求と応答）の例

等を記述し、処理系を作成する為の言語である。

ii) は、各計算機が提供する言語プロセッサに応じて、サービスの実体を作成する為に用いる。

iii) は、BSBS特定の言語である。

- 対話性
- 記号処理向き
- 、記号処理向きであることが要求される。

iv) は、各ピューを記述する為の言語であり、各ピュー上の呼び出し名（サービス名）と、その定義体との対応を与える機能が必要となる。定義するサービスが基本サービスの場合、定義体は、別のピュー上の名前となる。合成サービスを定義する場合の定義体の記述は、サービス要求言語による記述（婆則、婆則系）の記述と似ており、プログラミング的な作業が必要となるが、一般にiii)とiv)は独立がよい。

本稿では、次章以下、iii)、iv)として、関数型言語を用いた場合について紹介する。
３．関数型言語に基づく処理系間の通信

サービス要求言語を関数型言語とし、サービスの要求と応答を、関数呼び出しと、return-valueに
対応させることにより、分散して存在するサービスの実行を自然に行うことができる。この時の
計算機間の通信の様子を図４に示す。

４．処理系の構成とサービスの処理 ［３］

処理系記述言語として、C言語とLispを用い
て、実験システムを作成した。

C言語は、OSとのインタフェース、通信機能
とのインタフェースを記述する。これにより、
①system
②send
③receive
等の関数をLispから関数として呼び出す様にした。
①は、OSとのインタフェース用の関数であり、
②③は、通信機能とのインタフェースである。
Lispでは、サービス要求言語を読み込み、ピ
ューの記述を参照して、それを解釈・実行する機
能を実現した。即ち、合成サービスの場合は、L
ispのsemanticsに従って基本サービスに分解し
ていく。基本サービスの場合は、それぞれに応じ
て、
①）Lisp関数の起動
②）他計算機との通信
③）OSの機能の呼び出し
を行なう。このうち、①、Lispの機能をそ
のまま用い、③）は、上述の①を用いる。②）
では、①②を用いて、サービス要求と応答を行
なう。この時一覧には、サービス要求の結果とし
て、新たなサービス要求が返ってくる場合がある
（サービス要求のネスティング）、これに対処す
る必要がある。これらの実現例を以下に示す。
・サービス要求の形式
（<s-name> , <arg-list>)
<s-name>…要求するサービス名
<arg-list>…引き数リスト・サービスの応答の
形式（結果の場合）
（throw <return-value>）・サービス要求後の
処理
（catch
（prog （）
loop
（print (eval (read) )）
（go loop))）…（＊）
（read）…相手からのメッセージを読む。
（eval）…メッセージの評価（解釈）
（print）…評価した結果の相手への応答
（catch ＜S－式＞）…＜S－式＞の中で
（throw …）という関数が評価されるとき
）
（＊）を本システムでは「フレーム」と呼んで
いる。BNにサービス要求を行なった後、サービ
ス要求をした計算機は、フレームを作って待つ。
相手からのメッセージが新たなサービス要求の
場合、このフレーム内で処理される。新たなサー
ビス要求は、フレーム内の（eval）で処理される。
が、処理中に更にBNにサービス要求を行なう必
要が生じた場合は、サービス要求後、別のフレー
ムを作成する。（eval）はこの様な動作が可能な
様に新たに作成した。（Lispの（eval）の拡張）
BNからのメッセージがサービスの応答の場合
は、（catch ）を抜けて、次のサービスの処理に
移る。要求がステインしている場合は、それ
ぞれの要求に応じて、別々のフレームを作成す
ので、任意のレベルのネスティングが可能である。

５．副作用を伴うサービスの取り扱い

関数型言語では、ファイル操作や、外部との入
出力は、「副作用」として扱われる。ＳＢＳで特
に考慮する必要のある副作用は、サービスの実行
中に、外部と入出を行なう場合であり、
・ファイル転送
・サービス要求／応答用の通信路を用いての入
出力
等に対して特別な扱いが必要となる。これらの為に、
本システムでは、副作用を伴なうサービスを受
け入れる側で、
①）特殊サービス開始
②）特殊サービス終了
を示す２種類の特殊なサービスを用意し、①と
②）の間は、計算機間で直接サービス要求のネス
ティングを禁止している。実験システムでは、特
殊サービスとして
①file
②transparent
の２つの属性を用意した。①は、サービス提供者
の通信路への出力を、サービス要求側で一時ファ
イルで受け取ることを示し、②は、FNとBNを透
過的に接続することを示す。これらの属性は、サ
ービスを提供する側でサービスを定義する時に指
定する。①、或いは、②が指定されたサービスを
実行する時は、処理系が自動的に、サービス提供
者からサービス要求者に対して、副作用のための
特殊処理をする様に要求する。この時のプロトコ
ルを図５に示す。

６．サービスの定義方法
分散サービスを統合するにあたって、概念ビューの働きが重要なので、本章では、実験システムの概念ビューの記述について述べる。尚、実験システムは、ユーザー人用なので、概念ビューと外部ビューを区別していない。

i）基本サービスの定義

・自計算機の提供するサービスの定義
(defs <c-sn> [<i-sn>] [<se-type>])

・外部計算機の提供するサービスの定義
(defs <c-sn> <hostid> [<e-sn>] [<arg-type>])

図5 計算機Aが、計算機Bに、副作用を伴うサービスの要求した場合のプロトコル

図6 実装システム構成

図7 サービスの定義・要求例

文字列“LOAD”がファイル名に含まれているVOS3上のファイル名の表示

→ (srep 'LOAD (lists "*"))

VAX11/730 VAX11/780 M280H

7. サービスの定義・実行例

図6の構成で、両計算機の提供するサービスを組み合わせて利用する例を図7に示す。

서비스の定義
unix側 (defe lists 1)
VOS3側 (defe lists (file))
8. 検討・評価

8.1 サービス定義・要求言語と関数型言語

サービス要求／応答と、関数呼び出し／return valueを自然に対応させることにより、処理系間の通信機能を容易に実現できた反面、

i) ファイル転送など、基本的なサービスを「副作用」という形で実現しなくてはならない。

ii) ハイレベルなユーザインタフェースの提供がまだ不十分といった問題点がある。i,ii) に関しては、関数型言語の範囲内でも改善できるが、現在、サービス定義・要求言語として「論理型言語」を提供するシステムも検討している。論理型言語で、サービス定義を行うことがより、サービスに関する情報の知識化が可能になり、よりハイレベルなユーザインタフェースの提供が可能となるであろう。

また、SBSの概念自体は、言語に依存しないので、計算機－計算機間において、適した言語を用いるといった構成も考えられる。

8.2 並列サービスの実現 [2]

SBSでは、

i) 効率の向上

ii) 計算機同士の協調動作の制御

の為に、並列サービスの起動が必要となる。

i) の実現にあたっては、

①処理系内でのみ、（できる範囲で、或いは、必要な場合にかぎり）並列処理を行なう。

②サービス定義・要求言語として、陽に並列サービスを記述する制御構造を提供する。

の2つのレベルが考えられる。現在は、ファイル転送などの、並列処理が必要な場合に限って、①を行なっている。

②の実現に関しては、

サービス定義・要求言語として、陽に並列サービスを記述する制御構造を提供する。

に関して、サービス定義・要求言語による記述方法と、その実現方法について検討している。

8.3 効率

サービスの処理にかかる時間は、

①通信にかかる時間

②処理系の処理のオーバヘッド

③サービスの実行時間

の和である。これらの割合は、サービス毎に異なるが、実装システムの例では、①が最も多く、②は極めて小さい。①は、SBSの下位通信機能の性能に依存する部分であり、SBSの処理系の効率自体は、十分よいといえる。②の負荷を減らす為には、処理系で、あらかじめ、サービスの処理を行う順序をスケジューリングする必要がある。実装システムの処理系では、サービス要求言語を解釈するとすぐ実行しているが、この様な方式を「インタプリタ方式」と呼び、効率化などの為に、スケジューリングや最適化を行う方式を「コンパイラ方式」と呼ぶ。コンパイラ方式の処理系の作成も、今後の課題のうちの1つである。

8.4 定義部（ビュー）の管理 [1]

現システムでは、ビューの管理は、ファイルに頼っているが、サービスの増大に対応したり、より系统的な管理を行うために、ビューをデータベース化することを検討している。SBSでは、データとして、関係データベースをサポートするので、定義部をrelationで表現することにより、ビューをデータと同様に管理することができる。

ビューをrelationで表現するため、原則として、サービス名をkeyとしたタブレットの集計として表す。

ビューを蓄えるrelationの具体的なスケーミに関連して、現在検討中である。

9．おわりに

本稿では、関数型言語に基づくSBSの概念と実現、その具体例について述べた。残された問題は幾つかあるが、SBSによって、一般的の機械の拡張サービスを利用しやすい形で簡単に提供することが可能である。この時、関数型言語でサービス要求／応答が容易に実現できることを示した。

今後、関数型言語によるサービス記述の蓄積と並行して、記述のハイレベル化、並行サービスの記述などに通じた、サービス定義・要求言語の検討を行なっていく予定である。更に、種々のサービス要求・定義言語を基にして同時に使用できる強力なSBSのモデルとその構成方法についても検討する予定である。

<参考文献>

[1] 矢部、深沢、田中、元岡、「サービスベースシステムにおけるデータ管理についての一考察」、情処全大、1 J－2、pp.781－782、1983.10
[3] 萩野、深沢、田中、元岡、「サービスベースシステムの実装」、情処全大、4 D－10、pp.855－856、1984.3
[4] 深沢、萩野、田中、元岡、「論理型言語としての利用」、情処全大、サービスベースシステムの概念と実現、1984.9
[5] 深沢、田中、元岡、「サービスベースシステムの概念と基本構成」、電子通信学会、電子計算機研究会、1982.10