CORBA環境におけるプリンティングファシリティ技術

Printing Facility Technology for CORBA Platform

森田　哲也*　　丹羽　雄一*　　池上　宗光*
Tetsuya MORITA　　Yuichi NIWA　　Munemitsu IKEGAMI

要　　旨

Printing Facilityは，ネットワーク環境で印刷業務システムやプリンタ装置管理ソリューションなどを構築する際に利用可能な分散オブジェクト群をCORBA（Common Object Request Broker Architecture）標準のIDL（Interface Definition Language）を用いて記述したものである．本文ではリコーがCORBA技術の標準化団体であるOMG（Object Management Group）に提案したPrinting Facility技術について紹介する．

ABSTRACT

Printing Facility, defined in the CORBA-IDL(Common Object Request Broker Architecture - Interface Definition Language), is a set of distributed objects which are designed to develop business printing systems and printer management solutions under the network environment. Ricoh proposed the Printing Facility to OMG (Object Management Group), the global consortium for adopting a CORBA standard.

* 画像システム事業本部 プリンタ事業部 第二設計室 設計1グループ
1st Designing Section, 2nd Designing Dept
Printer Business Division, Imaging System Business Group

1. 背景と目的

近年，ネットワーク環境で構築される企業情報システムではCORBA[注1)](Common Object Request Broker Architecture)やDCOM[注10)](Distributed Common Object Model)を利用した分散オブジェクトコンピューティングが盛んになっている．これは情報システムへの要求の複雑化，構成するサーバーマシンのマルチベンダ化や大規模・分散化に伴いソフトウエア開発・維持管理のコストが増大してきたことが背景にある．

分散オブジェクトシステムとは，様々なソフトウエア・オブジェクトが，その物理的な位置とは無関係にお互いのサービスを利用しあえるシステムであり，ソフトウエア部品(オブジェクト)のAPI統一化による再利用性の向上や，部品単位の開発・デバッグ・評価が可能となる．また優れたソフトウエア部品を外部から調達したり，オブジェクト継承と呼ばれる手段によって独自拡張部分を差分開発のみで対応できるため大規模なシステムの機能拡張や開発期間短縮が可能となる．

Printing Facilityは，ネットワーク環境で印刷業務システムやプリンタ装置管理ソリューションなどを構築する際に利用可能な分散オブジェクト群をCORBA標準のIDL(Interface Definition Language)を用いて記述したものである．本文ではリコーがCORBA技術の標準化団体である OMG[注1)](Object Management Group)に提案したPrinting Facility技術について紹介する．

2. オブジェクト指向プログラミングとCORBA

あらゆるコンピュータプログラムは，データとそのデータを処理するための命令という2つの要素から構成されている．コンピュータ初期の時代からごく最近まで，命令を実行順に記述する「手続き型プログラミング手法」が主流となっていたが，この手法におけるデータ表現は個々の命令処理に依存した形式で設計される傾向にあるため，プログラムのサイズと複雑さが増大するとともに，データに対する意図的でないアクセスがプログラムの完全なテストを阻害するようになった．

一方，オブジェクト指向プログラミング(OOP)はデータ中心のプログラミング手法である．まずプログラムで扱う対象物(オブジェクト)の役割を概念的にモデル化し，その役割を実現するために必要なデータ表現(以下，属性という)と，オブジェクトに対する操作(以下，メソッドという)を定義する．例えばプリントシステムの場合，"プリンタ"，"ジョブ"，"文書"などをオブジェクトとし，プリンタの能力(紙サイズ，両面機能有り無し，等)を属性で表現したり，ジョブに対する操作(キャンセル，等)をメソッドとして定義したりできる．

OOPではオブジェクト内部のデータが隠蔽(データに対する直接操作ができないようにすること)されるため，データの安全性やデバッグの局所性が高まる．またソフトウエアの部品化・再利用を実現しやすいという特徴をもつ．

CORBAはOMGという世界800社以上で構成される非営利団体において標準化された技術であり，PC/WSサーバーやメインフレームなど異なるプラットフォームで構成される分散システムにおいて，アプリケーションの連携やプログラム生産性・移植性の向上を実現するためのオブジェクト指向ソフトウエア技術である．現在，CORBA製品は SUN[注2)]，IBM[注3)]，HP[注4)]，富士通[注5)]，Unisys[注6)]，Visigenic[注7)]，IONA[注8)] などからリリースされており，ミッションクリティカルな大規模システムにおいて実績を挙げている．

2-1 CORBAの特徴

CORBAの主な特徴を以下に挙げる．

(1) 分散オブジェクト環境

ネットワーク上で分散したオブジェクトのサービスをローカル/リモートの意識をせずに利用することができる．

(2) 言語中立性

オブジェクトのインターフェースは開発言語中立なIDLという言語で記述される．開発に際しては，サーバー/クライアントそれぞれについて開発者が使用するプログラミング言語に変換(マッピング)されたコードを利用できる．例えばサーバープログラムはC++で開発し，クライアントはC言語やJava言語で開発することなどが可能である．

またIDL-to-Java，Java-to-IDLといった言語マッピングや Java-RMI[注2)](Remote Method Invocation) over IIOP[注1)](Internet Inter-ORB Protocol)などCORBA環境とJava環境の融合が進んでいる．

(3) マルチプラットフォーム

MVS[注3)]，Windows[注10)]，Solaris[注2)]などの汎用OSから，pSOS[注9)]，VxWorks[注11)]などの組み込みOSまで様々なプラットフォームがサポートされている．またMicrosoftのDCOMとCORBAの連携に関しても標準化が進められている．

(4) インターフェースの継承

既存のオブジェクトを継承して新しい機能拡張が可能である．実装も拡張部分のみを追加すればよい．

(5) 異ベンダー製品の相互接続性

CORBA1.2まではORB内部の通信プロトコルが正確に規定されていなかったため，異なるベンダーのORB製品は相互接続できなかった．この問題を解決するためCORBA2.0では，TCP/IPをベースとしたORB間のプロトコルとしてIIOP(Internet Inter-ORB Protocol)が

規定され，異なるORB(Object Request Broker)製品間の通信が可能となった．

(6) CORBAservices[注1)]

Naming Service, Event Service, Security Serviceなど様々なソフトウエア部品(CORBAservices)のインターフェースが標準化されCORBAベンダーから提供されているため，アプリ開発が省力化できる．

最近では国内のCORBAベンダー数社が開発したORB(Object Request Broker)環境で，各社のTransaction Serviceの相互運用性が検証され話題となった．

このように統一されたサービスインターフェースを持つ分散オブジェクトが異なる実装でありながら利用可能になるということは，ソフトウエア部品ビジネスが実用化時代に入りつつあるという点で大きな意義を持つ．

2-2 CORBAアプリケーションの構成

CORBAはクライアント/サーバー間通信技術だが，その核となるソフトウエアはORBである．ORBはサーバーとクライアントにおいて動作し，位置の透過性(どのサーバーにオブジェクトがあるのかを意識しないこと)を持つことから，クライアントにとっては様々なサーバー・オブジェクトが接続された論理的なソフトウエア・バスであるとも言える．

CORBAを用いた通信の仕組みをFig.1に示す．クライアントからサーバーへの呼び出し方法としては，静的起動と動的起動の２種類がある．前者はクライアントの実行開始前にあらかじめサーバーのインターフェース(呼び出し関数群)を確定しておくため静的起動と呼ばれ，後者はクライアント実行開始後に必要に応じてリクエストオブジェクトを生成し起動するため動的起動と呼ばれる．

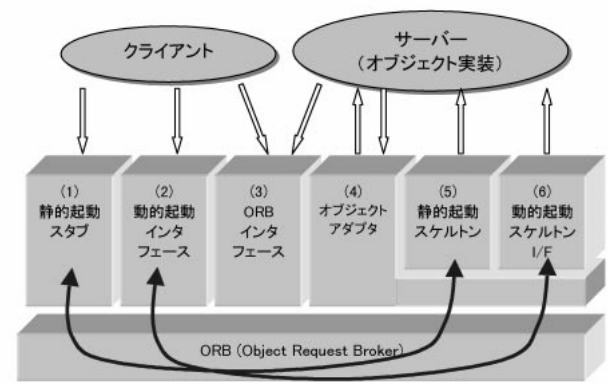

Fig.1 CORBA modules.

静的起動では，クライアントが静的起動スタブ関数(1)と呼ばれるサーバー呼び出し関数を実行すると，ORB経由でサーバー側の静的起動スケルトン関数(5)が起動される．また動的起動では，クライアントがサーバーへのリクエストオブジェクトを生成し，動的起動インターフェース(2)を実行すると，ORB経由でサーバー側の動的スケルトン起動インターフェース(6)が実行され，リクエストの内容に基づいてサーバーが起動される．

またORB自体の機能として，ORB環境の初期化などを行うORBインターフェース(3)と，セキュリティやサーバーオブジェクトのメソッド起動や実装の活性化・非活性化などを行うオブジェクトアダプタ(4)がある．前述のSecurity Serviceはこのオブジェクトアダプタに実装される．

3．Printing Facility

3-1 RFP(Request For Proposal)

OMGでは標準化プロセスの第1ステップとして，標準提案に対する要求項目(RFP)が提示される．Printing Facilityに関する主要な要求項目を以下に挙げる．

(1) プリンタを複数ユーザで共有できること
(2) 文書データを転送(Push,Pullなど)できること
(3) ジョブのスケジューリング，実行ができること
(4) ジョブの監視，イベント通知ができること
(5) セキュリティ機能(認証)を持つこと
(6) インストール，プリンタ管理ができること
(7) 実装が隠蔽されていること

以上は必須項目であり，他にWant項目として課金や大規模なジョブのハンドリングなどが要求されている．

これらの要件を満たす印刷サーバーによって，論理プリンタ(負荷分散印刷，代行印刷，分冊印刷など)や，印刷ジョブの効率的スケジューリングなど稼動状況管理や，プリンタの障害管理やサプライ管理などの機能を構築できる．

さらにリコー提案仕様の特徴として，紙切れやプリンタ障害発生時の障害回避機能およびユーザとのインタラクション機能をサポートしている．またモデルとしてスキャナなどプリンタ以外の画像機器を利用する際の拡張性も考慮した．

3．成果

3-1 CORBA Printing Facility(PF)提案モデル[4]

OOPではオブジェクト間の関係(所有関係，継承関係，等)を表すオブジェクトモデルによってソフトウエアの構造を記述する．Fig.2はPrinting Facilityのオブジェクトモデルを OMT法に基づいて示したものである．

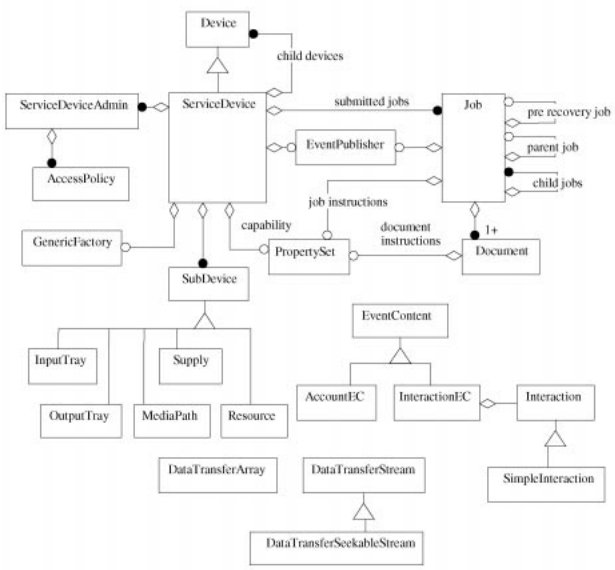

Fig.2 Printing Facility Object Model.

所有関係:

四角はオブジェクト(クラス)を表し,[A]◇—●[B]は,オブジェクトAが0個以上のオブジェクトBを持つことを示す.また[A]◇—○[B]はAが1個のみBを持つことを示す.

継承関係:

△は継承関係を表し，下位クラスが上位クラスの能力(属性，メソッド)を引き継ぐことを意味する．継承は正しく実装されると上位クラスのコードを2回実装する必要がなくなるため，下位クラスでのコード開発を省力化できる.

例えばFig.2において，ServiceDeviceはプリンタやスキャナなどの装置を表すオブジェクトであり，Deviceオブジェクトを継承し，0個以上のJob，SubDevice（トレイ，排紙先等)，ServiceDeviceAdmin（アクセス制御のための管理機構）と，1個ずつのEventPublisher(イベント通知機構)，PropertySet(装置能力)，GenericFactoryを持つことが示されている.

Printing Facility(PF)の主要なオブジェクトの動作について以下に述べる．また属性・メソッドの一覧をTable 1に示す.

Table 1　主要なPFオブジェクト

オブジェクト名	説明	属性・メソッドの例
Device	装置を特定するために必要最小限の属性を持つオブジェクト.	属性：デバイス名称，機種情報 メソッド：なし
ServiceDevice	Device を継承し，Document I/Oサービスを行うプリンタ，スキャナなどの装置を表す.	属性：装置の能力（装置タイプ，スケジューリング方式など） メソッド：Jobの投入，スプール操作
Job	ServiceDeviceによって生成され，生成～消滅までのStatusの変遷が規定されている．Jobは親子関係をサポートしており，論理プリンタと実プリンタのジョブ関係を表すことができる.	属性：Job ID，Instructions，Status，セキュリティ情報，開始/終了時刻，進捗状況，親ジョブ/子ジョブ，Document リスト，消耗品リスト メソッド：ジョブのキャンセル・停止・再開，代替デバイスによる障害回避
Document	印刷文書，条件などを表す．Property Serviceにより柔軟性・拡張性に対応.	属性：文書名，印刷条件，利用可能な文書データ転送方式. メソッド：なし
EventPublisher	Publish/Subscribeモデルによる非同期イベントの通知ができる．イベント内容はEvent Contentオブジェクトが使用される.	属性：通知可能なイベントリスト，イベントデータの転送方式 メソッド：イベントの購読/取消し（subscribe/unsubscribe）
DataTransferXXX	イベント通知，印刷データの転送に利用される通信プロトコルの定義で，個々の実装に応じて拡張できる.	構造体：転送方式名称，使用プロトコル，転送方向Push/Pull指定

(1) Device, ServiceDevice Object

DeviceはDocument I/Oサービスを提供するプリンタ，スキャナなどデバイスを現わし，PFの実装されていないプリンタでも装置を認識できるように名称と機種情報など装置を特定するために必要最小限の属性を持っている．そのためPF機能を持たないプリンタを子デバイスとしてPF環境で取り扱う場合などに使用することができる．PF ClientはDeviceを継承したService Device にアクセスする.

ServiceDeviceは，装置の能力情報(装置タイプ，利用可能な言語，スケジューリング方式など)と状態情報(状態，トータル枚数など)を持つ．Capabilityは先述のCORBAserviceの一つであるProperty Serviceを利用しており個別開発において拡張が可能である．またServiceDeviceは論理プリンタを構成するために親デバイス・子デバイスの属性を持っている．子デバイスの属性をどのように親デバイスに反映するかなどの関係は実装依存としている.

(2) Job Object

JobはServiceDeviceによって生成され，消滅に至るまでのStatusの変遷(life cycle)が規定されている.
Jobに対してキャンセル・停止・再開，代替デバイスを用いた障害回避などが可能である．Jobのlife cycle図をFig.3に示す.

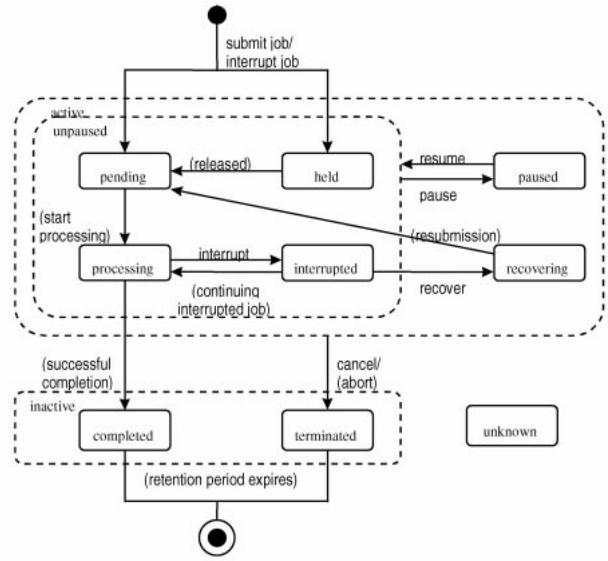

Fig.3 Job Status Diagram.

投入されたJobはpendingまたはheld(実行待ち)を経て processing(実行中)に移る．heldはJobの実行開始時間が指定されている場合などの実行待ちで，pendingはスケジューリングされ実行する装置が確定するまでの実行待ちである．processing中にエラーや割り込みJobが発生するとinterruptedに移行し，エラーが回避されrecover()が呼ばれるとpendingになる．Jobが正常に終了するとcompletedとなり，cancel()されたり異常終了するとterminatedに移行する．

ServiceDeviceと同様にJobも親子関係をサポートしており，例えば論理プリンタと実プリンタのジョブ関係を示すことができる．Fig.4に親子関係にあるJobとServiceDevice(図中では，Serverと表す)の例を示す．

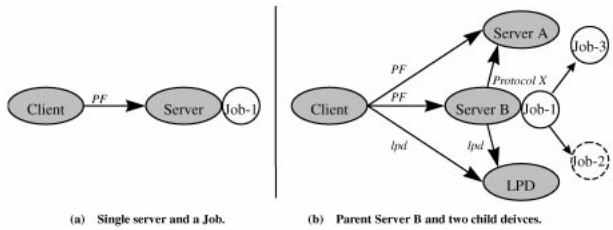

Fig.4 Server and Job Hierarchy.

(a)は親子関係のない単一のServerとJobである．(b)はServer B(例えば，平行印刷可能な論理プリンタ)に対してClientが親Job(Job-1)を投入しているが，実際の動作はServer AとLPD(Line Printer Daemon) Printerが行う例を示している．Server AはPF Objectであるため，Clientから子Job(Job-3)に関する情報を取得することができる．一方，Server BはLPD Printerを用いて仮想的に子Job(Job-2)を動作させることができる．ここでLPD PrinterはPF機能を持たないためClientからPF Objectとして扱うことはできないが，親ServerであるServer Bが２つの子Server(Server A, LPD Printer)の動作状況を合成してClientからの問い合わせやキャンセルメソッドなどを実行できるかもしれない．

job instructionsもProperty Serviceを利用しており，必要に応じて論理プリンタの印刷条件などを拡張することができる．

代替装置を用いた障害回避(Recovery)はFig.5のようにして行われる．(a)のようにClientがServer AにおいてJob-1を実行している際，Server Aで障害が発生するとJob-1はその時点までの情報を保持するためにpre_recovery_job_holderとしてJob-2を生成し，さらに代替装置Server BでJob-3を生成して実行を継続する．

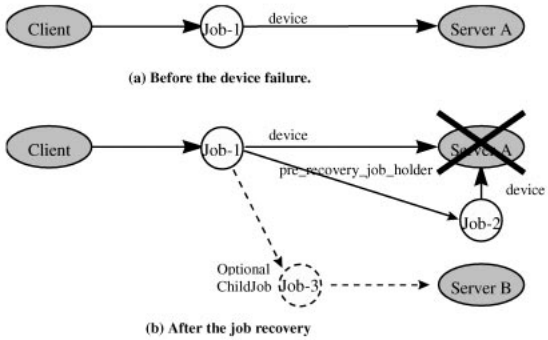

Fig.5 Job Recovery Mechanism.

(3) Document Object

PFにおけるDocument Objectは印刷データ自体を隠蔽しており，PDLデータやラスタ画像データなどは後述するDataTransferMethodによって直接転送される．document instructionsも拡張性・柔軟性に対応するためにProperty Serviceを利用している．なお，実用上必要と思われる最低限のInstruction(紙サイズ，文書フォーマット，両面，後処理，圧縮方式，ジョブ操作など)が推奨属性として定義されている．

(4) EventPublisher Object

PFではpublish/subscribe modelによる非同期イベント通知方式を採用している．Publisherは通知可能なイベントリストを持ち，Subscriberが通知して欲しいイベントをあらかじめ登録しておく．従ってServiceDeviceやJobの進捗や異常などを通知する場合，ServerがPublisherになり，ClientがSubscriberとなる．

イベント情報の表現には，EventContentもしくはこれを継承したObjectが使われる．例えば，課金情報を通知するためのAccountECやユーザとの対話を要求するためのInteractionECなどはEventContentを継承している．これらのObjectを転送する際には文書データ転送と同様，DataTransferMethodが使用される．

(5) DataTransferXXX Object

実システムではイベント通知や印刷データの転送に既

存通信プロトコルを利用されるため，PFではデータ転送方式を識別する属性 （TransferMethod構造体）のみが規定されており，個々の実装に応じて拡張できる．
PFではORB経由でのデータ通信を行う方式として，生データを転送する方式（DataTransferArray）と，streamライクな転送方式（DataTransferStream）が用意されている．

4．結論と今後の展開

Printing Facilityは汎用ソフトウエア部品として提供されるため，異なるプリンタベンダーが開発したサーバーオブジェクトを混在利用したり，クライアントソフト開発において統一したインターフェースを利用できるため開発工数を低減できる．

今後もネットワーク環境トータルな印刷能力の向上と管理コストの低減を目指したプリントサーバー技術に対するニーズが多様化すると考えられるため，部品提供だけでなくアプリケーション開発環境を拡充したい．

謝辞

本仕様開発および標準提案活動にあたり，多大なるご支援をいただきました研究開発本部および画像システム事業本部他の関係者各位に厚くお礼申し上げます．

参考文献

[1] "Document Printing Application （DPA)", ISO/IEC 10175-1 and 10175-2, （1996）
[2] "Printer Management Information Base - MIB", IETF RFC1759 （1995）
[3] "Internet Printing Protocol - Model and Semantics.", IETF Internet Draft （June.1998）
[4] "Ricoh/Novell Revised Submission to the Printing Facility RFP : orbos/97-09-03.doc ,pdf, rtf, ps" , http: // www. omg.org / library / schedule / Printing_ Facility_RFP.htm(Sep.1997)

注1） OMG, CORBA , CORBA IDL, IIOP, CORBAservicesは Object Management Group,Inc.の登録商標または商標です．
注2） SUN, Solaris, Java, Java-RMIはSun Microsystems,Inc.の登録商標または商標です．
注3） IBM, MVSはInternational Business Machine,Corp.の登録商標または商標です．
注4） HPはHewlett Packard Companyの登録商標です．
注5） 富士通は，富士通株式会社の登録商標です．
注6） UnisysはUnisys Corp.の登録商標です．
注7） VisigenicはINPRISE Corp.の登録商標です．
注8） IONAはIONA Technologies,Inc.の登録商標です．
注9） pSOSはIntegrated Systems,Inc.の商標です．
注10）DCOM,WindowsはMicrosoft Corp.の登録商標です．
注11）VxWorksはWind River Systemsの登録商標です．