システムソフトウエアと
オペレーティング・システム 60−13
マルチメディア通信と分散処理 61−13
（１９９３．７．８）

試行評価のためのメディア複合サービス開発環境

金井 敦, 三宅 延久, 中山 隆二

NTT ソフトウェア研究所

様々なメディアを組み合わせたサービス（メディア複合サービス）を開発する場合, 単一のメディアを使用したサービスに比較してユーザインタフェースや機能の組合せが複雑となるためその仕様を確定させるのは非常に難しい. このような場合は, まずサービスを簡易に作成し, 試行的に運用・評価することにより仕様を確定する事が望ましい. 本稿では, メディア複合サービスを対象とした上記の試行評価を行なうためのプラットホーム[1]について述べる. このプラットホームは様々なメディアを柔軟に組合せることが可能なブロック構造のシステム構成と汎用性があり抽象度の高いライブラリを構築可能なソフトウェア階層を持っている. さらに, メッセージシーケンスを用いて複数プロセスからなる実構造を忠実に再現するとともにエラー処理を自動生成することによりサービス開発者が試行に耐えるサービスを簡易に開発することを可能としている. また, 具体的サービスとして定型電子・音声メール発信サービスを例により容易にメディア複合サービスを試行できることを示す.

Media Complex Service Development Environment for Experimental Evaluation

Atsushi Kanai, Nobuhisa Miyake and Ryûji Nakayama

NTT Software Laboratories

This paper describes a platform for experimentally evaluating media- complex services, which combine several kinds of media. It is much more difficult to define a specification for media-complex services than for single-medium service. In this case, a service should be experimentally operated and evaluated by easily developing the service. This platform has a system and software structure designed to enable services to be easily modified. Furthermore, highly reliable services that are actually used in an experimental environment can easily be developed by describing multiple processes faithful to a real system by message sequences and by generating error sequences from ordinary service sequences. An example of a fixed-statement electric/voice mail service, which is a media-complex service, can be easily developed.

[1] 実行環境と開発環境を合わせてプラットホームと呼ぶ.

1 はじめに

多種のメディアを組み合わせたサービスは，ユーザ要求が明確でなく使用形態もさまざまであるため，突然大規模なサービスを開発するのはリスクが大きく危険である．そのため，試行的にサービスを運用し評価することがより重要となる．従来のマルチメディア技術に関する研究では，通信路，データベースあるいは端末などをマルチメディア化したり，メディア間の関係を表現した通信制御方式[1, 2]などが研究されており，メディアを統合して取り扱うことを目的としているものが多い．これに対して，本稿では，様々なメディアをうまく組み合わせて利用する最上位のアプリケーション(AP)の開発および試行を行なうプラットホームの方式を検討の対象としている．ここでは，APでのメディア操作の観点からAPからみて論理的に一つのメディアにみえる場合（例えばテレビメディアのように音声と画像が一つのメディアとして扱えるような場合）はたとえ複数のメディアが使用されていても一つのメディアとして取り扱う．

本プラットホームは試行運用と複数のメディア操作を効率よく実現する必要がある．要求条件としてはこの二つの独立した要求を満たす必要がある．それぞれ，以下の要求条件を仮定している．

（1）試行のためのサービス開発についての要求

- 信頼性の高いサービスが作れること
- 簡単に作れること
- 設計が容易であること

（2） 複数のメディア操作についての要求

- 様々なメディアが組み込み可能であること
- メディアの組合せや自由な使い方が可能であること

上記の条件に対する実現方式としては，以下に示す技術を用いて要求を実現している．

- 並行プロセスの動作を自動検証する(1).
- エラー処理を自動生成する(1).
- 各種メディア制御をプロセスとしてメディア毎に独立に追加・修正・操作を可能とする(2).
- AP構造を実世界の構造と一致させる((1),(2)).

現在，本プラットホームは電話交換処理，音声応答蓄積処理，電子メール処理，ボイスメール処理，FAX処理を対象メディアとしてる．なお，メディアという言葉は様々な意味に用いられることがあるが，以下では，通常の媒体という意味を拡張し，あるメディアや機能を処理する装置あるいはプロセスの意味で用いる場合もある．

本稿では，今回実現したプラットホームの全体像について，主にシステムアーキテクチャ，ソフトウェアアーキテクチャ，APアーキテクチャおよび開発技術について述べる．さらに，複合メディアサービスの例として定型電子・音声メールサービスの開発例について述べる．

2 システム構成

メディア処理装置はメディアごとに様々な機能のものが存在し，技術の進歩状況に応じて完全に統合されたマルチメディア装置や通信路は常に実現できないと思われる．このため，最先端のメディアを組み合わせることを目的としたプラットホームは常に複数のメディア処理装置や処理ソフトウェアの組合せで構成されると想定される．このため，本プラットホームは以下に示すシステムアーキテクチャを前提としている．

- メディア処理装置は独立した装置（ボード等を含む）とする． ここで，メディア処理装置とは，音声や画像といったメディアが複合した処理装置であっても論理的に一つのメディアとして扱われる装置である場合は一つのメディア処理装置として取り扱う．
- 一つのメディア処理装置はそれを制御する一つの独立したプロセスとしてのネットワーク型ドライバ（ソケットインタフェースなどを用いて他ノードからドライブが可能）を持つ．
- メディアの通信路は物理的に分離しているかどうかに関わらず，メディア処理装置毎に論理的な通信路を持つものとする．

このような構成とすることにより，ハードウェアの増設などにも柔軟に対応できる．図1にシステム構成の概念を示す．

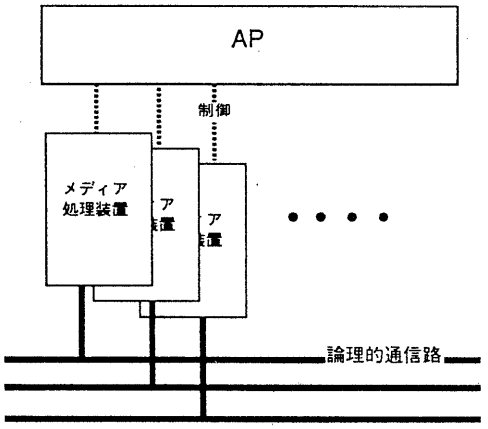

図 1: 前提とするシステムの論理構成

2.1 ソフトウェア構成

本プラットホームの基本的なソフトウェア構成を図 2に, AP と各メディアプロセス間の通信の実現構造を図 3) に示す. メディア処理装置毎に装置を制御しアプリケーションにインタフェースを提供するプロセス (以下メディアプロセスと呼ぶ) とメディアプロセスが提供する汎用インタフェースを, よりサービスドメインよりのインタフェースに変換するライブラリおよび AP から構成される. メディア処理装置の機能追加・変更時に, その影響範囲を小さく抑えることが, 多様なサービスを迅速に実現するために重要である. メディア種別毎にプロセスを配置することにより, 各機能の実現 (処理, データ構造等) の隠蔽が可能となり, 影響範囲を個々のメディアプロセスに限定することが可能となり, サービス実現に必要な機能が追加されることを容易にする. また, ドライバの変更, 通信のデータ形式の変更がサービス記述に影響を及ぼすことを防ぐ目的で, 階層構造をとった. 以下に各階層の機能を示す.

通信層 OSI モデルの AP 層より下のレイヤをサポートする.

GF 層 サービスに依存しない, メディアの汎用的な制御機能機能を提供する. 例えば, 音声応答蓄積プロセスでは, ガイダンスの再生, メッセージの録音等

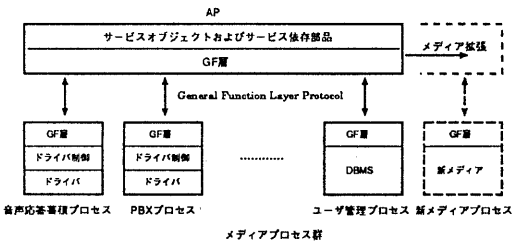

図 2: 基本的なソフトウェア構成

である. AP は, GF 層間で定められたプロトコル (GFL プロトコル) を用い, 各メディアプロセスで提供する機能を使うことができる.

SDM 層 GF 層で提供される機能は, サービス記述の観点から見ると汎用的であるが, 低レベルである. すなわち, GF 層のみでサービス記述を行なうと, 決まりきった一連の処理を何回も記述しなければならない場合や多くの不必要なパラメータを設定しなければならない場合などがあり, AP 記述上効率的でない. サービスに依存した一連の処理を部品化し, サービスを迅速に記述することを目的に機能を提供するのが SDM 層である.

サービス層 GF 層, および SDM 層で提供される機能を用い, サービスの実行を行なう.

SDE-OS 層 次節で述べる AP アーキテクチャのモデルを忠実に再現することおよび UNIX のプロセス管理機構を用いずより簡易に高速に多くのプロセスを処理することを目的とした機構 [?] である.

2.2 AP 構成

メディア複合サービスは複数のメディアを組み合わせたサービスであるため, 複数のメディアプロセスを並行して動作させることが AP の主な機能となる. このようなサービスを 1 つのプロセスで実現をすると, その記述は複雑で繁雑になる. 特に, 複数のメディアが組合わさり, サービスの修正が頻繁に発生する状況では, その記述の繁雑さは大きな欠点となる. サービスを, 各メディアプロセスごとにその制御を行なう内部プロセスと, そ

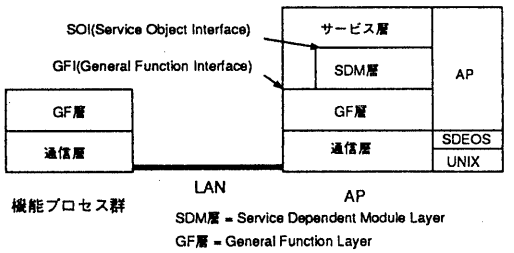

図 3: プロトコル階層とソフトウェア階層

れらの間を接続するチャネルにより並行プロセスにモデル化 (図 4) して考える. このモデル化は現実のシステム構成と AP プロセス構成を忠実に一致させることを意味している. このような構成とすることにより, 設計時に現実の構成とのギャップが小さくなり上記問題点が解決できる. 上記モデル化は, 各メディアプロセスや他の AP プロセスをオブジェクトとみなすことにより分散オブジェクト指向プログラミング [3] と捉えることができる.

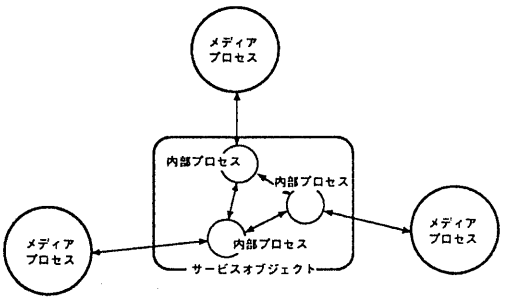

図 4: AP アーキテクチャ

3 開発環境

2 章で示した AP アーキテクチャのプログラムを開発するには並行プロセスの動作を直接記述することができる開発環境が有効である. 並行プロセスの動作をメッセージシーケンスの集合として記述してプログラムを自動生成する開発環境 SDE[2][4] が開発されている. 本プラットームでは SDE を用いて AP 作成を行なっている. また, 技術的スキルの少ないサービス仕様設計者でも短期開発が可能となることを目的として SDE 環境の上にエラー処理を自動生成する機構を実現している. 本章では, SDE とエラー処理自動生成について概要を述べる.

3.1 概要

SDE は次の特長を持つ通信ソフトウェア開発環境である.

仕様の形式的記述: 複数プロセス間のシーケンス仕様記述を言語 SAL[3] により, 形式的に記述する.

仕様の自動合成, 自動検証: SAL により記述された仕様の正しさ (実現可能, 両立可能, 要求外動作なし) を検証する.

プログラム自動生成: 与えられた仕様から, プロセス毎の動作仕様 (SDL) へ変換し, さらにプログラムへ変換する.

SDE では, プログラムは互いに通信しながら動作する有限状態機械 (プロセス) の組 (複数プロセスの表現が可能) としてモデル化される. 各プロセスは他のプロセスおよび外界との間に通信路を持っている. SDE のシステムイメージを図 5 に示す. メディアプロセスや他の AP プロセスとのメッセージシーケンスを SDE を用いて記述することにより AP を作成する.

3.2 エラー処理自動生成

エラー処理シーケンスの生成ルールを, 正常仕様に網羅的に適用して正常仕様からエラー処理シーケンスを自動生成する方法である [5]. この方式を適用することにより, 本来サービス開発者が意図した中心的な動作である正常仕様のみを記述することにより, 量が多くサービス開発者にとって煩わし処理を記述する必要がなくなる. この方式の概念を図 6 に示す.

[2] Systems Design Environment

[3] Service Addition Language

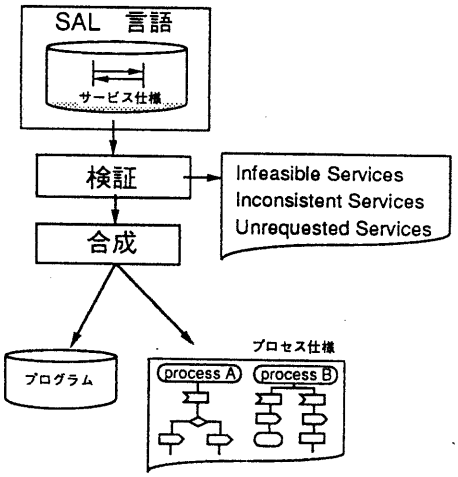

図 5: SDE の構成

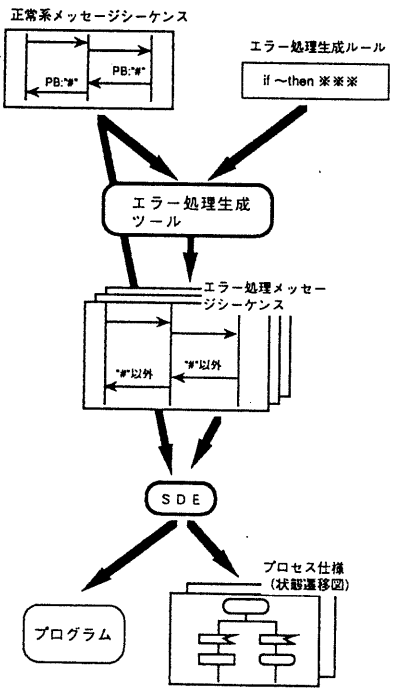

図 6: エラー処理の概要

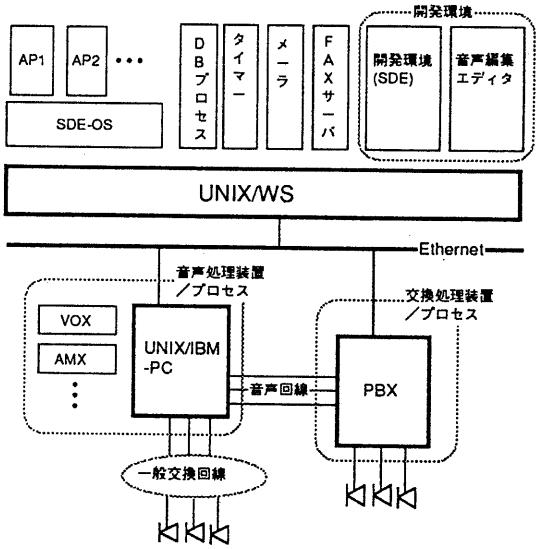

図 7: プラットホームの全体構成

4 プラットホームの概要

本章では，上記で述べたプラットホームの概念の具体的インプリメントについて述べる．本プラットホームは図7に示すように，大きく，開発環境系と走行系に分かれる．開発環境系，走行系ともに UNIX を OS とするワークステーション上で走行する．音声処理のハードウェアは IBM-PC と処理ボードから構成される．PBX はコンピュータ制御インタフェースを持ちワークステーションから制御可能なものである．メーラおよびデータベースはワークステーション上のソフトウェアであり特別な装置は必要としない．FAX プロセスは FAX ボードをワークステーション上に搭載して実現している． 各装置はイーサネットで接続され，AP からの制御はイーサネットを通じて行なわれる． 電話の音声回線はイーサネットとは独立に PBX および音声処理装置に接続されている．音声メールは MIME[4] フォーマットで電子メールとして取り扱われる．

[4] Multipurpose Internet Mail Extensions

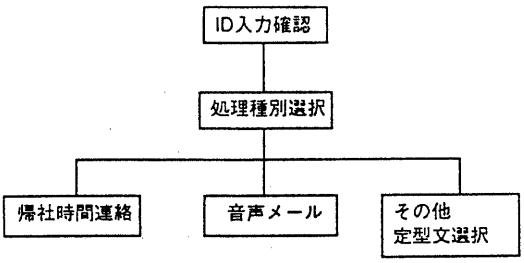

図 8: サービス仕様の概要

5 サービス例

5.1 概要

出先の路上や自宅から同僚や上司と連絡をとりたい場合に通常電話を使用するしか方法はない．この場合，連絡相手が不在であった場合などの対処が面倒である．これに対して電子メールのような非同期の蓄積型通信では連絡相手が不在の場合の煩わしさが無い反面，電話からの自由なアクセスができない．

定型電子・音声メール発信サービスは，電子メールの持つ非同期性と，電話のアクセスポイントの豊富さを兼そなえたサービスである．本サービスは，センターに電話をし，PB 操作をすることにより，予め決めた相手に予め決めた定型文の発信と必要ならば音声メール発信を行なうサービスである．サービス仕様の概要（正常仕様）を図 8に示す．

5.2 AP 構造

メディアプロセスとしては，音声処理，メーラ，データベースを使用する．AP は，これらのメディアを制御するためのプロセスとこのサービスのユーザを表す仮想的なプロセスおよびメディアプロセスに対応するプロセスから構成される．仮想ユーザプロセスは試験手順の生成やユーザから見た使用手順などを自動生成することに用いることを目的として設けている．図 9に典型的なメッセージシーケンス例を示す．

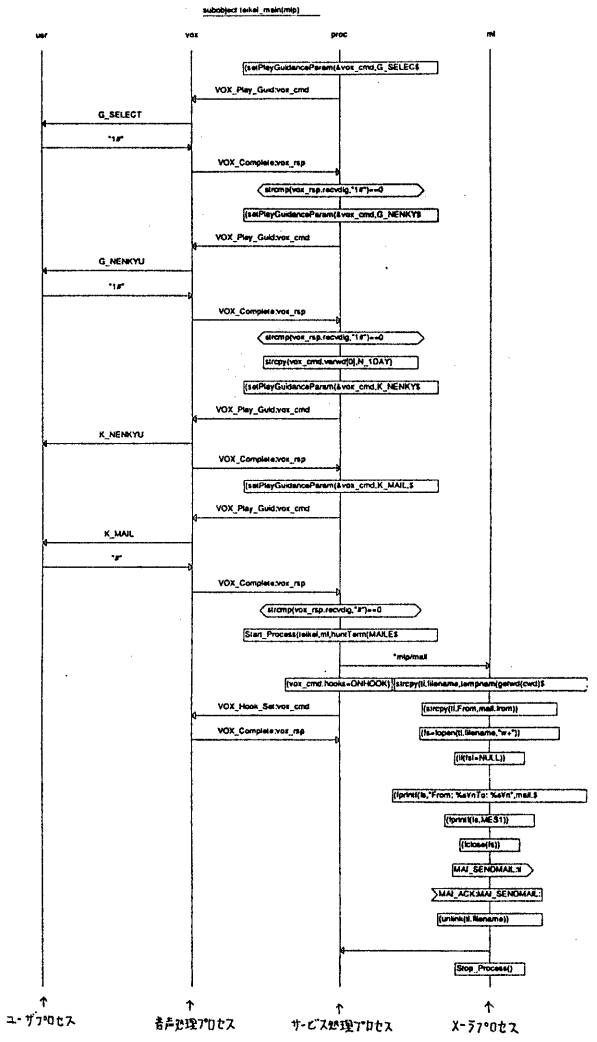

図 9: メッセージシーケンス例

5.3 エラー処理自動生成とプラットホームの効果

6種類のエラー生成ルールを使用してエラー処理を自動生成した. この時使用したルールの種類を以下に示す.

- ユーザ入力時の途中切断
- ユーザ入力時間切れ
- 入力データ確認時に誤入力
- 登録時のユーザ切断
- ユーザ入力タイムアウト
- システムエラー

ユーザが記述した正常系 (ユーザ記述), その正常系記述中のSDM層マクロ部品を展開した記述 (マクロ展開後), さらにEHGENにより生成したエラー系を加えた記述 (EHGEN適用後), それぞれの記述量を, 状態数で評価した結果を[5], 表1に示す. マクロ展開も含めると本プラットホームがない場合と比較して記述量がほぼ3分の1に削減されている事がわかる.

表 1: SDM 部品・エラー系自動生成の効果

	状態数
ユーザ記述	124
マクロ展開後	246
EHGEN 適用後	363

既存のメディアプロセスを使用し要求仕様が完全に決定されている状況で, 上記サービスはガイダンス作成 (ガイダンス作成は専用ツールを使用) を含めてほぼに1人週程度で開発, 試行開始することができた.

6 まとめ

多種のメディアが複雑に絡み合ったサービスの開発・試行を目的とするプラットホームとそのAPの構成および開発環境を述べた. また, 本プラットホームを用いた実際のサービス記述例を紹介し, 既存のメディアプロセスを使用する場合は, 簡単にサービスを開発, 試行することができることを示した.

今後は, 実験をさらに行ないメディア制御方式をより明確化するとともに多数のユーザがインタラクティブにメディアを複合して使用するCSCWなどのAPへの適用も行なう.

[5] ユーザ記述, マクロ展開後の記述はメッセージシーケンスチャートを使い表現されている. ここでは, 比較のため状態図への変換を行い, 状態数を算出した.

謝辞

本検討を進めるにあたり, 終始ご指導頂いたNTT情報通信処理研究所小池グループリーダと同グループの皆様ならびにNTTソフトウェア研究所市川グループリーダと同グループの寺内氏に深謝いたします.

参考文献

[1] 梶浦正規, 玉置政一, 菅野政孝: “マルチメディア通信プロトコルにおけるメディア間関係制御 方式の検討”, 9 1991, 情報処理学会研究会報告, マルチメディア通信と分散処理研究 会 52-27.

[2] Gerd Schürman and Uwe Holzmann-Kaiser: “Distributed Multimedia Information Handling and Processing”, 11 1990, IEEE Network Magazine.

[3] 宮本衛市, 武村功司, 渡辺慎哉: “オブジェクトのモジュール化による複合オブジェクトの生 成”, 9 1987, 情報処理学会ソフトウェア工学研究会.

[4] H. Ichikawa, M. Itoh et al.: “Incremental Specification and Development of Cmmunication Software”, IEEE Trans. on Computers, Vol.40, No.4.

[5] 寺内敦, 山中顕次郎, 加藤順: “通信サービスにおけるエラー処理の自動生成”, 10 1992, 情報処理学会第4 5回全国大会, Vol.1.