インターネットサービス連携システムにおけるサービスレベルを考慮した運用管理方法

権藤　夏男† 　楊　　巍† 　小山　貴夫† 　松田　栄之†

† 株式会社ＮＴＴデータ 開発本部 技術開発部
〒 104-0033　東京都中央区新川１－２１－２

E-mail: †{gondo,yang,kym,matu}@rd.nttdata.co.jp

あらまし　インターネット上のサービスを組合せて，1 つのサービスとして提供する連携サービスが提案・実現されつつある．連携サービスのトランザクション管理や QoS 確保のためには，動的に変化する連携サービスと個別サービスとの関連や連携サービスの処理フローの進行状況を管理し，個別サービスの障害時・性能劣化時に影響の分析およびフロー制御を行うことが重要である．本発表では，サービス連携システムに要求される運用管理機能の分類および実現方式を提案し，これらの機能を実現したプロトタイプについて述べる．

キーワード　サービス連携、サービスレベル管理、性能監視

An approach of Internet Service Collaboration System Management at Service Level

Natsuo GONDO†, Wei YANG†, Takao KOYAMA†, and Shigeyuki MATSUDA†

† Department of Information Technology, Research and Development Headquaters
NTT DATA CORPORATION
Shinkawa 1-21-2, Chuo-ku, Tokyo, 104–0033 Japan

E-mail: †{gondo,yang,kym,matu}@rd.nttdata.co.jp

Abstract　Collaborated service composed by multiple indivdual services across the Internet is widelynoticed and in the progress of being realized. Transaction management and QoS are indespensable to provide such kind of service. In order to achieve these two functions, management of relationships between collaborated and individual service and progress of service execution are both requried. In addition, impact analysis of service loss, caused by individual service's trouble,and flow control of service execution according to the analysis result are necessary for service construction. In this paper, we propose an approach of service management which consists of function classification and method for realization. A prototype based on our approach is also mentioned.

Key words　Service Collaboration System, Service Level Management, Performance Monitoring

1. はじめに

インターネットが発展し、それをベースにサービスを提供する企業が現れ発展を遂げている．近年は、XML の普及とともに Web サービスという形態も現実のものとなりつつある．Web サービスが広く普及すれば、インターネットはさらに発展することは言うまでもない．たとえば、現在のポータルサイトは固定的なサービス提供者を静的に結合しているが、Web サービスを用いることで動的にサービスを取り込むことができる．つまり、ユーザの要求に対して最も条件に合うサービスを、リアルタイムで検索しかつそのサービス提供サイトへ逐次接続してサービスの提供を受けることが可能となる．Web サービス提供者にとってはビジネスチャンスが広がり、サービスの利用者にとっては、より自分の要求にマッチしたサービスを享受できると言う点で、今まで以上にメリットがある．また、ユーザみずからサービスを探す手間を省くために、従来どおりポータルサイトの役割は大きい．

しかしながら、Web サービスを利用することを前提としたポータルにおいて、単一のサービスを提供していては、ポータルサイトの特色を出すことができない．つまり、単純にフライトの予約をするサービスでは、どのようなプロバイダでも同一のサービスになってしまう．そこで、複数サービスを連携し、新たなサービスを生み出すというような差異化要素が必要となる．具体例として、フライトの予約、ゴルフ場の予約、ホテルの予約というサービスを一連の処理で行うことで、ひとつの旅行サービスが提供できる．

このように、Web サービス等を組み合わせて新たなサービスを創り出す仕組みを、筆者らはサービス連携 [1] と呼ぶことにする．このサービス連携を実現するには、サービスのフローを制御する仕組みや、インターネット上のサービスを検索する技術が必要となる．また、従来の静的な結合により実現されるシステムと異なり、サービス提供サイトが頻繁に現れ、サービス内容が更新されることが予想される．連携サービスを提供するポータルサイトが、高いレベルのサービスを提供するには、これらの動的な構成変化を監視し運用を可能とする運用管理の仕組みが重要となる．本稿では、サービス連携システムの実現方法について検討し、特に高いサービスレベルを維持するための運用管理方法について考察する．また、この考察に基づいて構築したプロトタイプシステムの実装についても述べる．

2. サービス連携の概要

本章では、サービス連携の概念を説明し、さらにサービス連携システムの実現のための要件について述べる．

2.1 サービス連携とは

サービス連携においては、図 1 に示すように ”サービス利用者 ”, ”連携サービスプロバイダ (CSP) ”, ” 個別サービスプロバイダ (ISP) ”の 3 つの立場に分類できる．以下、それぞれの立場について説明する．

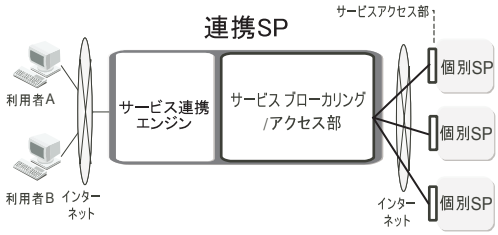

図 1　サービス連携を構成する要素の概念

○サービス利用者
連携サービスプロバイダが提供するサービスの利用者である．連携サービスプロバイダを使うメリットとして、利用者自身が個別サービスの所在やサービス内容を把握している必要が無く、連携サービスプロバイダが仲介役としてサービスの検索や予約処理などを行った結果をサービスとして受けられる点があげられる．

○連携サービスプロバイダ
サービス利用者の希望条件を収集し、実際のサービスのブローカリング（検索など）および実際のサービスのアクセス（実行依頼）を仲介する．従来のサービス提供者を募って作成されたポータルサイトと異なる点は、UDDI [2]/SOAP [3]/ebXML [4] といった Web サービスを実現する仕様に準拠した個別サービスプロバイダに対し、逐次システムを接続することを可能としている点である．この特徴により、利用者は希望条件に最も近いサービスを享受することが可能となる．また、個別サービスの提供者は、人気の高い（サービス利用者の多い）連携サービスプロバイダにサービスを提供することで、ビジネスチャンスを広げることが可能となる．（以下、連携サービスプロバイダを簡単のため、連携 SP 又は CSP と記述する．）

○個別サービスプロバイダ
主にインターネット上でサービスを提供するプロバイダを想定している．従来、Web ページによりサービスを提供して来たプロバイダは、Web サービスに対応したインタフェースを Web ページと同様に準備することで、個別サービスプロバイダとなることが可能となる．そのメリットは、前述のようにビジネスチャンスを広げることが期待できる点にある．また、Web サービスでは、莫大な広告費を投入して顧客獲得をねらう大企業と横並びでサービス検索の対象となることが可能である．（以下、個別サービスプロバイダを簡単のため、個別 SP または ISP と記述する．）

以上、それぞれの立場について整理した．ここで示した、サービス連携システムのモデルでは、複数の独立したシステムが疎結合によりサービスを提供することになる．サービス品質を保つためには、従来のシステムと同様に統合的な運用管理を行う必要がある．しかしながら、システムを運営する主体が、連携サービスプロバイダとは異なるため、従来と同様の運用管理手法の適用は非常に困難である [5].

次節以降では、サービス連携を実現する手段について検討をすすめる．その後、サービス連携システムのアーキテクチャにより提供するサービスを考慮した運用管理手法について述べる．

2.2 サービス連携を実現するための要素

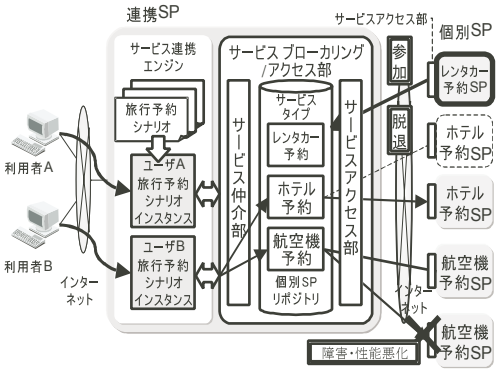

図 2　サービス連携システム実現のための構成要素

サービス連携を実現するための構成要素を図 2 に示す．図中の各用語は、以下のように定義する．

連携サービスプロバイダ (連携 SP)：サービス連携システムのポータルとして，利用者からの要求を受け付け，連携サービスを提供するサービスプロバイダ．

個別サービスプロバイダ (個別 SP)：ネット上に分散し，個々にサービスを提供するサービスプロバイダ．

シナリオ：連携サービスを構成するサービスの組合せおよび実行順序の定義．

シナリオインスタンス：利用者の要求ごとに，シナリオから作成される連携サービスの実体．

サービス連携エンジン：利用者の要求に従い，シナリオからシナリオインスタンスを作成し，連携サービスの実行を制御する．

サービスタイプ (ST)：航空機予約やホテル予約など，サービスの種類でまとめた個別 SP のグループ．

サービスブローカリング/アクセス部：サービス連携のための基本的な機能を提供する．主に以下の構成要素からなる．

- 個別 SP リポジトリ：個別 SP が提供するサービスやそのサービスタイプを格納したデータベース．
- サービス仲介部：サービス連携エンジンからの要求に従い，個別 SP リポジトリからの個別 SP の検索およびサービスの実行を行う．
- サービスアクセス部：連携 SP と個別 SP との通信を行う．ルーティングやセキュリティ機能を提供する．

2.3 ソフトウェアコンポーネント

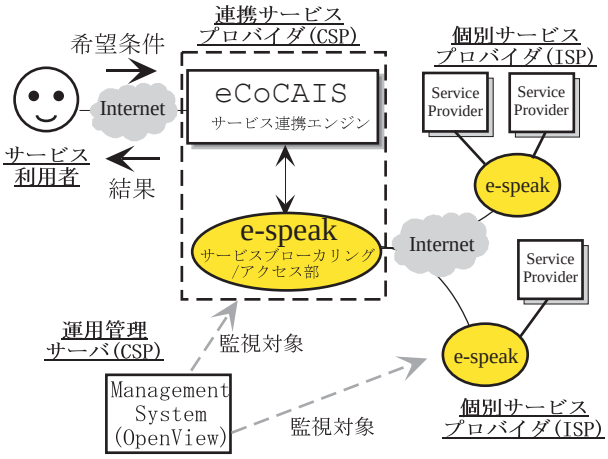

図 3　構成要素とソフトウェアコンポーネントの対応

前節で述べたサービス連携システムの機能を実現するために用いたソフトウェアコンポーネントを図 3 に示す．まず、サービス連携エンジンとして、”eCoCAIS(注1) ”を用いた．eCoCAIS は、フロー制御にトランザクションの考え方を導入しており、シナリオ実行中に障害が発生した場合でも、システム復旧後に中断したサービスタイプからシナリオを続行できるという特徴を有する．このフロー制御に関しては、次節で詳細に説明を行う．

サービスブローカリング／アクセスには、必要となる機能を全て実装している e-speak(注2) を用いた．e-speak は、インターネット上で秘匿性の高い通信を実現するためのセキュリティ機能や、特定のサービスタイプ毎に検索対象にするなどの機能も有している．また、Java により実装されており、多様なプラットフォーム上で実行することが可能である．さらに、運用管理用のイベント通知を行うための機能を有してる．本システムでは、この機能を用いてサービスの参加・脱退、および稼動・非稼動状態を通知するための通信経路として利用した．

2.4 シナリオフロー制御とサービスタイプ

図 4 に、本システムでのシナリオフロー制御とサービスタイプの関係を示す．図中の、”ホテル予約 ”と ”航空券予約 ”がサービスタイプ (ST) を表す．サービス利用者の要求毎にシナリオがインスタンス化されて個別に管理され、処理の進行状況を RDB のトランザクション処理機能を用いて、サービスタイプの開始と終了時点をチェックポイントとして不揮発記憶に永続化している．よって、例えばホテル予約が終了後に連携 SP にシステム障害が発生しても、復旧後に航空券予約の処理から再開が可能となっている．また、

(注1)：eCoCAIS は NTT 情報流通プラットフォーム研究所が開発した製品です
(注2)：e-speak は Hewlett-Packard 社の製品です．

サービスタイプごとの処理実行結果を受けて、処理の流れを分岐処理することが可能である．図の例では、ホテル予約が出来なかった場合に、航空券予約をキャンセルし、処理を終了させることなども可能となっている．

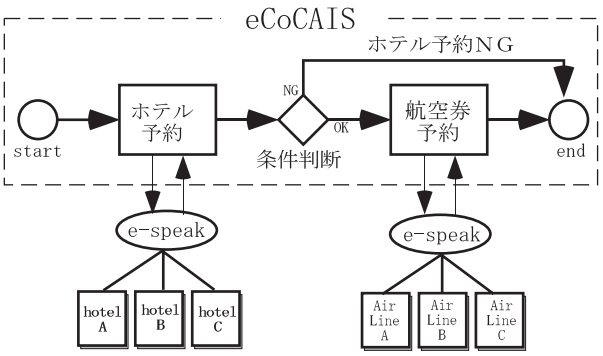

図 4　サービスフロー制御

本システムにおいて、サービスブローカリングやサービスアクセス（予約依頼）はサービスタイプ毎に、e-speak を介して実行している．e-speak は与えられた条件に対して、複数のサービス候補を検索することが可能である．今回のプロトタイプの実装では、最初に返されたサービス候補を無条件で利用することにした．

2.5　利用対象個別 SP の限定

e-speak においては、e-speak コアを１つ用意し、LAN やインターネットを介して全てのサービスプロバイダがその１つのコアに接続するシングルコア構成がある．また、各々のプロバイダで e-speak コアを運営し、e-speak の持つグルーピング機能により複数のコアがシームレスに通信することで論理的に１つのコアとして扱うことが可能なマルチコア構成方式がある．それぞれの構成においてメリットはあるが、本プロトタイプでは、各サービスタイプ毎に個別 SP を検索する対象を制限することを可能とする目的でマルチコア構成を採用した．マルチコア構成では、シナリオインスタンスからのサービス検索ごとに、検索対象とするコアのグループ構成を変更することが可能となる．

本システムでは利用対象とする個別 SP の取捨選択を実現するために、図５に示すような認証機構を設けた．具体的な流れとしては、個別 SP が１回目に連携 SP へのサービス提供を開始する際に、連携 SP へ認証を求め、認証された場合にサービスを提供していくと言う流れである．ここで、仮に連携 SP 側から参加の拒絶を受けると、サービスを開始しても、連携 SP 側でのサービス検索対象にならないため、サービスを利用されることは無い機構にしている．また、いちど登録されたサービスに関しては、個別 SP 側から明示的に脱退の処理を行わなければ継続的にサービス検索対象として利用される．つまり、個別 SP 側で運用のためシステムを停止しても、再起動の際に、再度参加のための認証を受ける必要はない．ここで、システム停止・再起動などは e-speak のイベント伝播機能を用いて、連携 SP 側へも通知される．

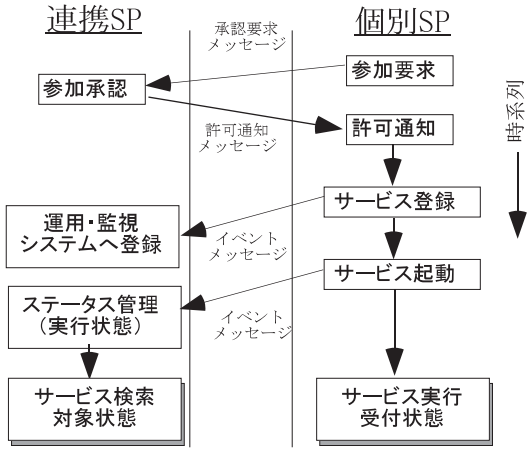

図 5　サービス参加認証の概念図

3.　運 用 管 理

サービス連携システムでは、複数の疎な結合のシステムを監視し、サービス利用者がサービスが利用できるか否かと言う観点での運用管理が重要となってくる．さらに、疎な結合の中で性能を監視し、現在利用しているユーザがどの程度の時間でサービスを受けられているかといったサービスレベルの把握も必要となる．このようなポイントを鑑み、以下に課題を整理する．

3.1　運用管理における課題

3.1.1　システム構成の動的変化

サービス連携システムの運用中に，個別 SP が新規に参加したり，参加中の個別 SP が脱退したりすることがある．したがって，個別 SP の参加・脱退に応じて監視対象を追加・削除する必要がある．さらに，参加・脱退する個別 SP と ST を関連付けて管理し，シナリオの利用可・不可を監視する必要がある．

3.1.2　シナリオ実行の管理

シナリオには利用する ST が定義されている．同一の ST に属する個別 SP は複数存在し，連携サービス実行時に個別 SP を動的に選択する．そのため，同一シナリオでも実行条件により連携する個別 SP が異なる．したがって，個別 SP の可用性・性能情報を用いて，個々の連携サービスの可用性・性能を監視し，必要に応じて個別 SP の選択および再選択や実行中止などの制御を行う必要がある．

3.1.3　個別 SP の監視

個別 SP はそれぞれの運用主体により独立して運用されている．また，インターネットではネットワーク帯域が限られる．したがって，連携 SP から大量で詳細なシステム構成および性能情報を取得したり，システムを制御したりすることは不可能である．よって，

個別 SP の監視では，サービスを単位とする可用性や性能の監視を行う仕組みが課題となる．以上の課題を解決するために，運用管理の各項目に求められる要件を次節で示す．

3.2 運用管理の要件

（1） **構成管理：**動的な個別 SP の参加・脱退を検知し、監視対象の管理へ反映させる．個別 SP・ST およびシナリオの関連を把握し、個別 SP の参加・脱退によるシナリオの利用可否を管理する．

（2） **状態監視：**個別 SP の稼動・非稼動の監視のために定期的なポーリングおよび非同期なイベントの受信を行う．また、実際の連携サービスおよび個別 SP の性能を測定する．

（3） **障害・性能分析：**シナリオおよび実行中の連携サービスごとに個別 SP 障害の影響を分析する．また、シナリオ実行と個別 SP の応答時間の関係を統計的に分析し、シナリオ実行の所要時間を推定する．

（4） **障害対処・回避：**連携 SP 側の制御で、連携サービスに対する障害対処および障害回避を行う．

3.3 実 現 方 式

前述の要件を満足する運用管理方式を項目ごとに示す．

（1） **構成管理：**連携サービスの構成として管理する構成情報および取得方法を表 1 に示す．

表 1　構成管理方式

構成情報	取得方法
個別 SP の参加・脱退	個別 SP リポジトリへの個別 SP の登録・削除に対してイベントを発生させ、監視対象の追加・削除と連動させる．
シナリオと ST との連携	シナリオから連携サービスとそれを構成する ST およびその順序を取得する．
ST と個別 SP との連携	個別 SP リポジトリから、個別 SP が提供するサービスの ST を取得する．

（2） **状態監視：**連携サービスの可用性を維持するために個別 SP の稼動・非稼動を監視する．また，性能を維持するために連携サービスおよび全ての個別 SP の応答時間を監視する (表 2)．

（3） **障害・性能分析**

（a） **個別 SP 障害の影響分析：**構成情報および監視情報を用いて，シナリオおよび実行中のシナリオインスタンスへの影響を分析する．分析項目および判断基準を表 3 に示す．

（b） **性能分析：**連携サービスおよび個別 SP の応答時間を統計手法により解析し，予め決められた時間内にサービスを終了することが可能な個別 SP を選択するなどに用いる．

（4） **障害対処・回避：**障害・性能分析結果を受け，表 4 の障害対処および障害回避を実施する．

表 2　状態監視方式

監視項目	監視手法
個別 SP の稼動・非稼動	連携 SP からの定期的なポーリングおよび個別 SP からの障害イベントの受信を併用する．
個別 SP の応答時間	サービスブローカリング/アクセス部における実サービスの監視およびダミーのトランザクションを実行する．
連携サービスの応答時間	サービス連携エンジンにおいて実サービスの開始・終了時刻を監視する．

表 3　障害影響分析方式

分析項目	判断基準
シナリオインスタンスと個別 SP との関係	サービスブローカリング/アクセス部からシナリオインスタンスと個別 SP の対応を取得する．
シナリオ実行可能性	個々の ST に属する全ての個別 SP が非稼動状態であれば、当該 ST を含むシナリオは実行不可能である．
シナリオインスタンス続行可能性	障害が発生した個別 SP を利用中のシナリオインスタンスは続行不可能である．
	障害が発生した個別 SP を利用予定のシナリオインスタンスであり、かつ他の個別 SP により代替可能な場合、続行が可能である．
	障害が発生した個別 SP を利用予定のシナリオインスタンスであり、かつ他の個別 SP により代替不可能な場合は、続行が不可能である．

表 4　障害対処・回避方式

制御項目	制御内容
シナリオ受付停止	実行不可能と判断したシナリオについて、新規のサービス受付を停止する．
シナリオインスタンス実行中止	続行不可能と判断されたシナリオインスタンスに対して、即座に既に完了したサービスのキャンセルなどの中止処理を実行する．
個別 SP の再割り当て	続行可能と判断したシナリオインスタンスに対して、利用者の要求を満たす個別 SP の再検索、再割り当てを行い、サービスを継続させる．

4. プロトタイプシステムの実装

前述の運用管理方式を実現するプロトタイプシステム [8] の開発を行った．図 6 に示すシステム構成の

とおり，サービスブローカリング/アクセス部として，前述のように Hewlett-Packard 社の e-speak を利用した．また，運用管理モジュールには，同社の運用管理ツール OpenView を利用し，他モジュールから構成情報および監視情報を収集する．

以下、運用管理部分についての実装について、運用管理要件ごとに説明する．

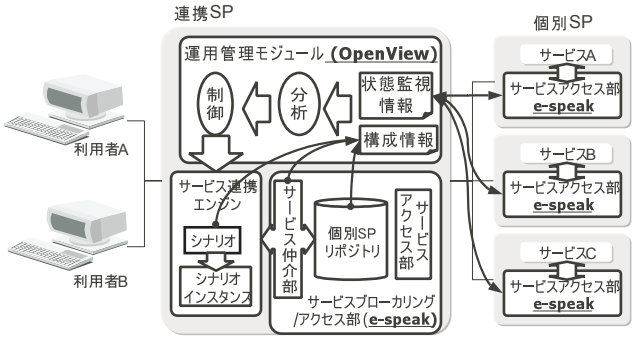

図 6　プロトタイプシステムの構成

（1）　構成管理：OpenView が提供するサービスエンジンの機能を利用して実装を行った．サービスエンジンは、図 7 に示すような階層構造での管理が可能であり、各ノードは配下のノード状態の AND または OR 論理演算結果を反映することができる．サービスタイプの階層では、配下の個別 SP の何れかが正常に動作していれば正常稼動状態として扱うこととした．シナリオ階層では、配下のサービスタイプのいずれかでも非稼動状態であれば、シナリオを非稼動状態とした．個別 SP は、サービスへ参加する際に、運用管理ツールに自動的に反映されるようした．これは、個別 SP を登録する際に、個別 SP の名称とサービスタイプなどを e-speak のイベント伝播機能で通知させることで実現した．図の階層構造の状態は、OpenView の Viewer により確認できる．また、シナリオの状態やサービスタイプの状態は、外部のプログラムから OpenView の API を用いて取得することが可能となっている．

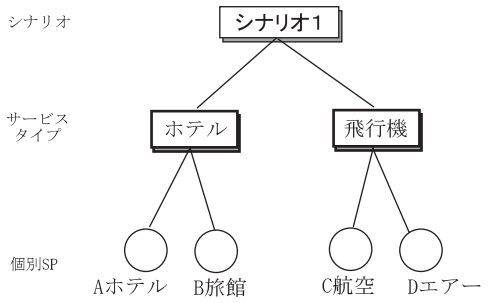

図 7　階層構造を用いた構成管理の実装

（2）　状態監視：構成管理情報に基づき、5 分間隔でポーリングを行い個別 SP の稼動状態を監視する．ポーリングに関しては、OpenView の基本的な機能により実現されるため、特に実装を行う必要は無い．ただし、インターネット上の個別 SP では、firewall が存在することによりポーリングを行えない場合もある．本プロトタイプでは、構成管理と同様に、e-speak のイベント伝播機能により、個別 SP の停止や再起動を通知し運用管理ツールに反映することを可能としている．よって、ポーリングが出来ない個別 SP に関しても、イベント通知により、大企業のように運用によるサービスの停止や再開を検知することが可能である．

（3）　障害・性能分析：障害の影響分析に関しては、構成管理の項目で既に述べたように、サービスエンジンの論理判断の機能を用いて実現している．また、性能の分析に関しては、サービスが実行させる毎に個別 SP とレスポンスタイムおよびサービス提供時刻を対比させて取得し、時系列で統計分析できるようにしている．レスポンスタイムは、e-speak にサービス実行依頼を発行し、応答が返るまでの時間とした．

（4）　障害対処・回避：シナリオの状態を OpenView から取得し、使用不可能な場合には、その旨をユーザに返却する．シナリオインスタンス実行中止と個別 SP の再割り当て機能に関しては、現在のところ運用管理ツールとの連動した実装は行っていない．

5. ま と め

本稿では，サービス連携システムを提案し、特に連携サービスプロバイダでの運用管理方式について詳細に考察した．本方式により，インターネット上のオープンな環境において，連携サービスの可用性および性能の維持が可能となる．今後は，インターネットデータセンタなどで連携 SP と個別 SP が同居するような環境についての統合運用管理システムについて検討する予定である．

謝辞　今回の研究を進めるにあたり，e-speak および OpenView に関してご協力および貴重な情報を提供頂いた日本ヒューレット・パッカード株式会社に深謝いたします．

文　献

[1] 吉田英嗣，横山和俊，元田敏浩，畑島隆，箱守聰，”シナリオを用いたサービス連携システムの実現と評価，”第 61 回情報処理全国大会論文集 分散アプリケーション (2)-5
[2] uddi “URL:http://www.uddi.org/”
[3] soap “URL:http://www.w3.org/”
[4] ebXML “URL:http://www.oasis-open.org/”
[5] M.Hamada, T.Inuzuka, T.Fujisaki, K.Kageyama, “Developing Platform for Cooperative Internet Management”, NOMS 2000 - 2000 IEEE/IFIP Network Operations and Management Symposium, pp.505-518, 2000
[6] e-speak “URL:http://www.e-speak.net/”
[7] 権藤 夏男，楊 巍，小山 貴夫，箱守 聰，松田 栄之,”サービス連携システムにおける運用管理方式,”情報処理学会第 62 回 (平成 13 年前期) 全国大会 第 3 分冊 ,page 335-336, 2001
[8] “URL:http://www.jpn.hp.com/partners/si_plaza/focus_4.html”