

特別研究報告書

ユーザの操作例を用いた
Webサービスの自動連携

指導教員 石田 亨 教授

京都大学工学部情報学科

宮浦 宏暢

平成18年2月10日

ユーザの操作例を用いた Web サービスの自動連携

宮浦 宏暢

内容梗概

現在，多くの Web サービスが利用可能である．Web サービスは，機械可読なインタフェースを持っているため，プログラムが利用可能なサービスである．機械可読なインタフェースを持っているため，Web サービスをコンポーネントとみなすことができる．複数の Web サービスを連携させることにより新たなサービスを作成することが可能である．Web サービスの連携が実用化され始めている．

既存の Web サービス連携の研究は Web サービス連携プロセスの実行や実現可能性に主眼を置いている．Web サービス連携プロセスとは Web サービスをどのように連携させるかを記述したものである．BPEL4WS は Web サービス連携プロセスを実行することに主眼を置いている．プランニングは全自動で Web サービス連携プロセスを作成する実現可能性に主眼を置いたものである．ユーザが使いやすい Web サービス連携プロセス作成の支援をすることには重点を置いていない．このため，ユーザが Web サービス連携プロセスを作成することは手間がかかり困難な作業となっている．

本研究では，ユーザが Web サービス連携プロセスを作成するために障害となっている以下の 2 つの問題に取り組む．

1. 新たなインタフェースの習得が必要

既存のシステムは Web サービス連携プロセスを作成するために，ワークフロー図等のユーザインタフェースを使用している．一般のユーザはこのユーザインタフェースを使用していないので，ユーザはこのインタフェースの使い方を習得しなければならない．

2. ワークフローの習得が必要

Web サービス連携プロセスを記述するために，BPEL4WS 等のワークフロー記述言語が使用されている．Web サービスの連携を記述するためにワークフローが使用されているため，ユーザはワークフローによる処理手順を規定するための方法を習得しなければならない．

上記の問題を解決するために，本研究では Web ユーザインタフェースを使用し，Programming by Example という手法を Web サービス連携プロセスの作成

に適用する。Programming by Example とはシステム自身のユーザインタフェース上においてプログラムを作成する方法である。プログラムが行うべき作業の例を記録し、その例を用いてプログラムを作成し、再び実行できるようにする。

具体的には、上述した問題を解決するために、以下の2点の方法を使用する。

1. 既知のインタフェースの使用

ユーザが普段 Web ブラウザ上で行っているリンクのクリック、コピー&ペースト、ボタンのクリック等の操作を使う Web ユーザインタフェースを使用する。これはユーザが普段行っている既知の操作になる。ユーザにとって既知のユーザインタフェースを使用することにより、新たなインタフェースを学ぶ負担をなくすことが可能となる。

2. 操作例を用いた Web サービスの連携

Web ユーザインタフェース上で連続して行われた操作を記録する。その操作を例として使用し、Web サービス連携プロセスを作成する。これにより、ワークフローの知識がなくても、Web サービス連携プロセスを作成することが可能となる。

本研究では、メタデータが付与された Web ページを作成し、上記の手法を実現するシステムの開発を行った。Web ページ上でのユーザの操作例として、Web アプリケーションの実行、リンクのクリック、データのコピーという操作を記録する。次に、これらの連続して行われた操作例を基にして、Web サービス連携プロセスを作成する。ユーザが操作例において使用したデータのメタデータをなぜユーザがそのデータを使用したかというユーザの意図を示すものとする。異なるデータに対しての操作であっても、データに付与されているメタデータが同じならば、ユーザが意図したとおりの操作であるとみなす。これはデータがメタデータによってパラメータ化又は一般化されたことになる。これにより、作成された Web サービス連携プロセスを実行する際に、使用するデータを変更できるようになる。

実装したシステムを用いて、Web ユーザインタフェース上での操作例に基づき、Web サービス連携プロセスを作成及び実行した。このシステムを使用して、特別な知識を持たないユーザでも Web サービス連携プロセスの作成及び実行することが可能となる。

Automatic Composition of Web Services using User Operation Sequence

Hironobu MIYAURA

Abstract

Many Web services are available now. A Web service, which has a machine-readable interface, is a service that can be used by software programs. Because of having a machine-readable interface, a Web service can be regarded as a component. Composition of two or more Web services enables us to make a new service. The composition of Web services are starting to be put to practical use. Previous researches focus on execution or feasibility of Web service composition process. Web service composition process describes a process of composing Web services. BPEL4WS aims at execution of Web service composition process. Planning focuses on full automatic composition of Web services. Both approaches do not give priority to easy-of-use of Web service composition. Therefore it is time-consuming and difficult for users to compose Web services. In this research, I deal with the following two problems that are obstacles for users to create Web service composition process.

1. The necessity of learning a new interface

Existing systems use user interfaces such as workflow chart etc. in order to create Web service composition process. Because ordinary users do not use these new interfaces, he/she must learn how to use them.

2. The necessity of learning workflow

Workflow description languages like BPEL4WS etc. are used so as to describe Web service composition process. Because workflow is used to describe it, he/she must learn how to describe procedure by workflow.

In this research, I use web user interface and apply “ Programming by Example ” method to the composition of Web services to solve the above issues. Programming by Example is a way to program on system ’s own user interface. The system first records example of what program should do, then creates a program using the example, and finally makes it possible to execute the program again. Concretely speaking, the following two methods are used to solve the above-mentioned problems.

1. Use of already-known interface

Web user interface, which allows operations like click of a link, click of a button, copy & paste, and etc. that a user is usually performing on a web browser, is used. These operations are known operations that users usually perform. By this, extra cost of learning a new interface is spared by using a known interface.

2. Composition of Web services using user operation sequence

The system records user operation sequence on web user interface. The system regards these operations as an example and creates Web service composition process using the example. Consequently, user can create Web service composition process without the knowledge of workflow.

In this research, I made sample web pages containing metadata and developed a system that puts the above-mentioned methods into practice. The system records actions which include click of a link, click of a button and copy on web pages as user operation sequence. Then, it creates Web service composition process based on this user operation sequence. Metadata of the data, which the user used in the user operation sequence, is assumed to indicate intention of the user or why the user used the data. It is considered that intention of operation is same even if the data used by operations is different as long as metadata of the data is same. This means that data is parameterized or generalized by metadata. As a result, it becomes possible to change the input data when the created Web service composition process is executed. I created Web service composition process and executed it with the implemented system. It becomes possible for even users who did not have special knowledge to create and execute Web service composition process by using the system.

ユーザの操作例を用いた Web サービスの自動連携

目次

第 1 章	はじめに	1
第 2 章	関連研究	3
第 3 章	Web ユーザインタフェースの使用	4
3.1	従来のユーザインタフェース	5
3.2	Web ユーザインタフェース	5
第 4 章	ユーザの操作例を用いた Web サービス連携プロセスの作成	7
4.1	Programming by Example	7
4.2	ユーザの操作例の記録	8
4.3	Web サービス連携プロセスの作成	11
第 5 章	実装	13
5.1	具体的な利用シナリオ	13
5.2	Web ユーザインタフェースとなる Web ページの作成	15
5.3	Web サービス連携プロセス作成システム	15
5.3.1	システムの概要	16
5.3.2	ユーザが行った操作の記録	17
5.3.3	Web サービス連携プロセスの作成及び実行	20
第 6 章	おわりに	22
	謝辞	23
	参考文献	24

第1章 はじめに

これまで，Web 上には人間が利用するために，数多くの Web アプリケーションが公開されているが，これらは機械可読なものではない．これらの Web アプリケーションは人間が直接利用することを前提として，Web ブラウザを介して利用されている．これらのサービスや提供される情報は人間が目で見えて解釈することは可能であるが，機械が解釈することは不可能である．このため機械が Web アプリケーションを利用することは難しい．

この問題に対処するために Web サービスがある．Web サービスは，機械可読なインタフェースを持つことにより，プログラムがサービスを利用可能なものをいう．

現在，様々な Web サービスが利用できるようになりつつある．XMethods や Web Services List では様々な Web サービスが紹介されている．他にも個々の企業による Web サービスとして，Google Web APIs，Amazon Web サービス，Yahoo! Search Web Services 等がある．

これらの Web サービスは機械による解釈が可能である．Web サービスをコンポーネントとみなして，複数の Web サービスを自動で連携させて新たなサービスを作成することが可能となる．

従来の Web サービス連携プロセスの作成は，Web サービス連携プロセスの実行や実現可能性に重点を置いている．BPEL4WS [1] は XML ¹⁾ に基づいた Web サービスの連携を記述するワークフロー記述言語である．この言語は作成された Web サービス連携プロセスを実際に行うことに主眼を置いている．一方で，プランニングは Web サービスの意味的な記述に基づいて，Web サービス連携プロセスを推論するという，Web サービス連携の実現可能性に主眼を置いている．

ユーザが使いやすい Web サービス連携プロセス作成の支援をすることには重点を置いていない．このため，ユーザが Web サービス連携プロセスを作成することは手間がかかり困難な作業となっている．

本研究では，ユーザが簡単に Web サービス連携プロセスを作成することを可能にするために，以下の 2 つの問題に取り組む．

1. 新たなインタフェースを習得することが必要

¹⁾ <http://www.w3.org/TR/REC-xml/>

従来のシステムは、Web サービス連携プロセスを作成するために、ワークフロー記述言語である BPEL4WS を使用している。BPEL4WS は XML を基にした言語で、人間が直接エディタにより編集するのは困難である。そのため、GUI を使用して、ワークフロー図によって、Web サービス連携プロセスを記述するという方法をとっている。XML を基にした BPEL4WS を直接エディタによって作成するより、これらのツールを使うほうが大幅に手間が軽減される。それでも、ユーザは新たなインタフェースの使い方を習得しなければならない。

2. ワークフローの知識を習得することが必要

Web サービス連携プロセスを記述するために、BPEL4WS 等のワークフロー記述言語が使用されている。Web サービスの連携を表現するためにワークフローが使用されているため、ユーザはワークフローによって Web サービスを実行する手順を規定するための知識を習得しなければならない。

上記の問題を解決するために、本研究では Web ユーザインタフェースを使用し、Programming by Example という手法を Web サービス連携プロセスの作成に適用する。Programming by Example とはシステム自身のユーザインタフェース上においてプログラムを作成する方法である。ユーザはプログラムがすべきことの例を与える。システムはユーザの操作を記録し、それをプログラムに変換し、再び実行できるようにする。Programming by Example とは”自分が行ったことを行う”プログラムを作成することになる。これにより、ユーザは大量に新たなことを学ばなくてもよくなり、ユーザにとって既知の操作のみを使用してプログラミングが行えるようになる。

具体的には以下の 2 点によって、上述した問題の解決を図る。

1. ユーザにとって既知のインタフェースの使用

ユーザが普段 Web ブラウザ上で行っているリンクのクリック、コピー&ペースト、ボタンのクリック等の操作を使う Web ユーザインタフェースを使用する。これはユーザが普段行っている既知の操作になる。既知のユーザインタフェースである Web ユーザインタフェースを使用することにより、新たなインタフェースを学ぶ負担をなくすことが可能となる。

2. ユーザの操作例を用いた Web サービス連携プロセスの作成

Web ユーザインタフェース上で連続して行われた操作例を Web サービスの連携とみなす。その操作例を基に Web サービス連携プロセスを作成する。

これにより，ワークフローの知識がなくても，Web サービス連携プロセスを作成することが可能となる．

本稿の構成を以下に示す．第 2 章は関連研究について述べる．上記の問題を解決するために，第 3 章では Web ユーザインタフェースの使用について，第 4 章ではユーザの操作例を用いた Web サービス連携プロセスの作成について述べる．第 5 章では上記の方法を用いた実装について述べる．最後に，第 6 章で本研究のまとめを行う．

第 2 章 関連研究

Web サービス連携プロセスを作成するための既存の研究としては，セマンティック Web によって Web サービスを拡張したセマンティック Web サービスの実現を目指すものがある．

METEOR-S プロジェクト¹⁾は Web サービスへのメタデータの注釈，Web サービスの発見，Web サービス連携プロセスの作成，Web サービス連携プロセスの実行の面において取り組んでいる．

オントロジーによって Web サービスの能力を形式的に記述し，Web サービスの意味的な記述に基づいて，プランニングによって Web サービスの連携を推論する人工知能の方法を使用するという Web サービス連携に注目した研究 [2] がある．

この研究による Web サービス連携プロセスの作成方法は以下のステップによって構成されている．

1. *Service Representation*: 利用可能な Web サービスとその能力の表示
2. *Requirements Specification*: 要求される機能の詳細化
3. *Composition*: 要求される機能を実現する Web サービス連携プロセスを作成
4. *Composite Service Representation*: 配備，発見，呼び出しを可能にするための新しい複合サービスとその能力を表示

この研究による基本的な Web サービス連携プロセスの作成方法を図 1 に示す．Service Registry は利用可能な Web サービスを格納する．Domain Ontology で定義されている用語を使用して各々の利用可能な Web サービスの能力が形式的に記述されている．

¹⁾ <http://lsdis.cs.uga.edu/projects/meteor-s/>

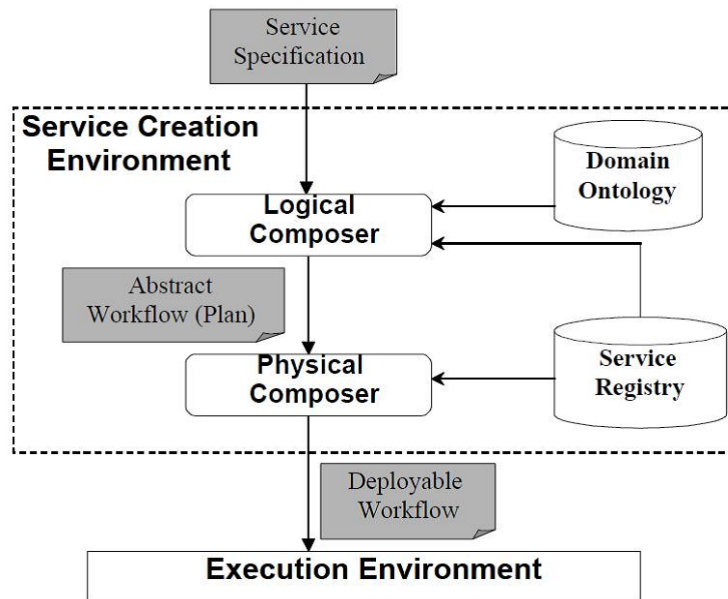


図 1: プランニングを使用した Web サービス連携プロセスの作成 [2]

Logical Composer モジュールに Service Specification が入力されると，プランニング技術による推論によって，利用可能なサービスのオントロジーに基づいた，抽象的な Web サービス連携プロセス (Abstract Workflow) ができる．この段階では，現在利用不可能な新しい機能を作成するために，オントロジーによる機能の連携を行う．これは実現可能性に主眼を置いた作業になる．

Physical Composer は配備し実行することが可能な Web サービス連携プロセス (Deployable Workflow) を生成するために，最良の Web サービスを具体的に選択する．この段階では機能外要求 (e.g. quality of service) に基づいて，Web サービスの選択を行う．これは実行に主眼を置いた作業になる．

Execution Environment はネットワークにおいて近接している Web サービスについて，Web サービス連携プロセスの実行を分割する．これによって，よりよいスケーラビリティと性能が得られる．

第 3 章 Web ユーザインタフェースの使用

本章では，Web サービス連携プロセスを作成するために使用するユーザインタフェースについて述べる．

3.1 従来のユーザインタフェース

Web サービスの連携を記述するために、ワークフロー記述言語である BPEL4WS が使用されている。BPEL4WS は XML を基にした言語で人間が直接エディタにより編集するのは困難である。そのため、GUI を使用して BPEL4WS を作成するためのツールがある。これらのツールでは、図 2 のように、ワークフロー図によって Web サービス連携プロセスを記述するという方法をとっている。XML を基にした BPEL4WS を直接エディタによって作成する必要がなくなることにより、これらのツールを使うほうが大幅に手間が軽減される。それでも、ユーザは新しいインタフェースの使い方を習得しなければならない。

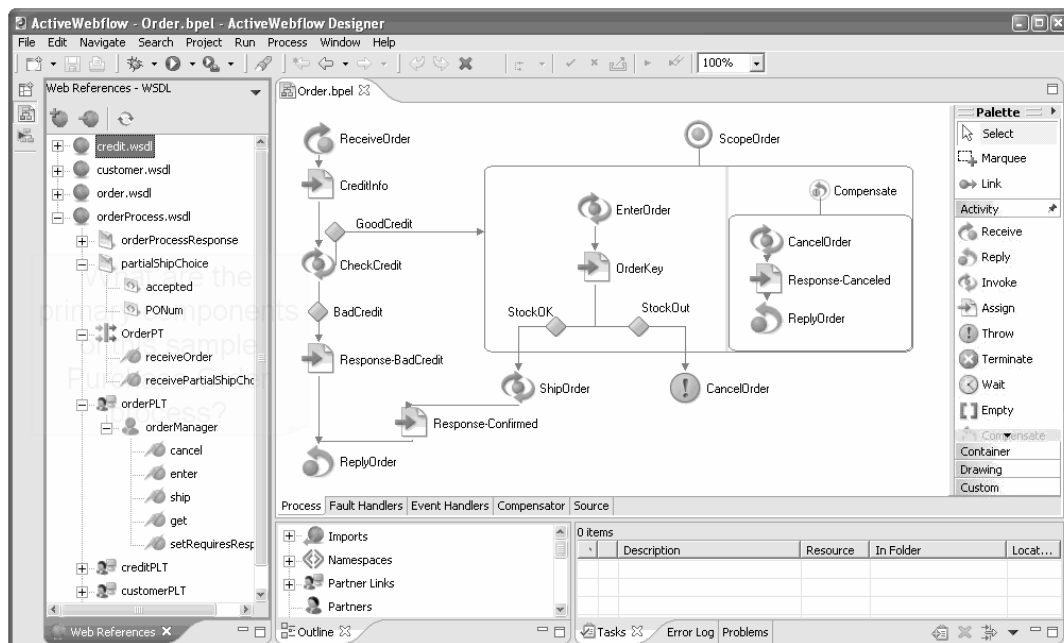


図 2: BPEL4WS によるワークフローの作成支援ツール

(<http://www.active-endpoints.com/products/index.html>)

3.2 Web ユーザインタフェース

図 3 は検索を行う Web アプリケーションである。この Web アプリケーションを実行するための操作はテキスト型のフォームにキーワードを入力し、ボタンをクリックすることである。図 4 はアフィリエイト (成果報酬型) 広告の作成を行う Web アプリケーションである。アフィリエイト広告とは Web ページに商

品の広告を載せ、その商品を販売している Web ページへのリンクを張り、そのリンクを経由して商品販売をしている Web ページを訪れ商品を購入する人がいると、Web ページの広告を掲載していた人に報酬が支払われるという広告になる。この Web アプリケーションはアフィリエイト広告を作成したい本の ISBN とアソシエイト ID を入力し実行すると Amazon のアフィリエイト広告を作成する。この Web アプリケーションを実行するための操作はテキスト型のフォームにアフィリエイト広告を作成したい本の ISBN とアソシエイト ID を入力しを入力し、ボタンをクリックすることである。図 3 と図 4 は異なる Web アプリケーションだが、使い方は同じとなる。



図 3: 検索を行う Web アプリケーション
(<http://www.google.co.jp/>)

Amazon.co.jp: Webサービスエッセンシャルズ: 本
Web サービスの中核をなすXML-RPC, SOAP, WSDL, UDDIの基礎概念から具体的な導入の方法、独自のサービスを作成するための手順を、... 最初に、XML-RPC実装を説明し、その後、インフラ技術としてのWebサービスと、それを実現する、SOAP, UDDI, WSDLの3つ ...
www.amazon.co.jp/ezc/obido/s/ASIN/4873110890-48k-キャッシュ-関連ページ

Amazon.co.jp: Building Web Services With Java: Making Sense of Xml ...
Building Web Services With Java: Making Sense of Xml, Soap, WsdL, and Uddi (Java (Sams)); 洋書の販売はオンライン通販 Amazon.co.jp。1500円以上のご注文で国内無料配送。Arts & Photography, Biographies & Memoirs, Business & Investing ...
www.amazon.co.jp/ezc/obido/s/ASIN/0672321815-45k-キャッシュ-関連ページ

Amazon.co.jp: msrx178mk2さんのプロフィール:レビュー
SOAP, WSDL, UDDIについての説明から、実際にJAVAとNETでアプリケーションを作成し、それに対する相互運用まで網羅しているので、...また一番読みたかったAXISIによるWebサービスについてはボリュームが少ないので、SOAP, WSDL, UDDIについてのみ勉強に ...

図 5: Google での検索結果

(<http://www.google.co.jp/>)

Amazonアフィリエイト広告の作成

ISBN:
 アソシエイトID:

図 4: アフィリエイト広告の作成を行う Web アプリケーション

Webサービスエッセンシャルズ



イーサン セラミ Ethan Cerami 長瀬 嘉秀

オンライン・ジャパン 2002-07
 売り上げランキング: 164,211
 ￥ 3,360
 通常2日間以内に発送

おすすめ平均 ★★★★★
 ★★★★★ SOAP, UDDI, WSDLを大変詳しく解説

[Amazonで詳しく見る](#)

図 6: アフィリエイト広告

(<http://www.goodpic.com/mt/aws/>)

図 5 は Google での検索結果の Web ページとなる。リンク先の Web ページに移動するためには、リンクの上にマウスポインタを移動させ、クリックするという操作をする。図 6 はアフィリエイト広告の例となる。このアフィリエイト広告に含まれているリンクを選択して、リンク先の Web ページに移動するためには、先ほどと同様にリンクの上にマウスポインタを移動させ、クリックをす

るという操作をすることになる。

このように、一度 Web ブラウザ上での操作を覚えると、異なる Web ページについてでも同様の操作で作業が行えるようになる。

本研究では、Web サービス連携プロセスを作成するために使用するインタフェースとしてユーザにとって既知の Web ユーザインタフェースを用いる。Web ユーザインタフェースとは、ユーザが普段 Web ブラウザ上で行っているリンクのクリック、コピー & ペースト、ボタンのクリック等の操作を使用するインタフェースのことである。

これによって、ユーザは新たなインタフェースを使う必要がなくなり、普段から使用している既知の Web ユーザインタフェースを使うことにより、新たなユーザインタフェースを習得するという負担がなくなる。

第4章 ユーザの操作例を用いた Web サービス連携プロセスの作成

本章では、Web サービス連携プロセスを作成するための方法について述べる。まず、Programming by Example とはどのようなものを述べる。次に、その手法をどのように Web サービス連携プロセスの作成に適用するかについて述べる。

本研究では、セマンティック Web の普及により、Web ページにメタデータが付与され機械可読であるとする。システムは Web ページに付与されたメタデータを用いて、ユーザの操作を記録し再現する。

4.1 Programming by Example

Programming by Example とはシステム自身のユーザインタフェースによってプログラミングを行う方法である。ユーザがすでに知っている操作のみを使用してプログラミングを行う。簡単にいうと、Programming by Example は”Do What I Did”になる。ユーザはプログラムがすべき作業の例を与える。システムが一連の行動(入力と出力の組)を記録し、その行動を用いてプログラムを作成し、再び実行できるようにする。これにより、ユーザは大量に新たなことを学ばなくても、既知の操作のみを使用してプログラミングが行えるようになる。

大量に新たなことを学ぶのを避けるために、ユーザがすでに習得していて、普段から行っていることを使用できるようにする。プログラムをより簡単に作成

するために、少しずつプログラムを作成できるようにする。そして、プログラムを作成しながらテストできるようにする。この2つが Programming by Example に欠くことのできない特徴となる。

ユーザが行った例と全く同一のことを繰り返すプログラムを作成することが役立つこともある。しかし、ほとんどの場合、ユーザは与えた例で使用されているデータ (data objects) を変更して、プログラムを実行したい。ユーザの操作例で使用されたデータをパラメータ化 (parameterize) 又は一般化 (generalize) して変更することを可能にするために、ユーザがなぜそのデータを使用したかという意図を示すためのデータの解説 (data description) が必要になる。Web サービス連携プロセスを実行する際に、このデータの解説に基づいてデータを探し、選択することによって異なるデータについてもプログラムを実行できる。 [3, 4]

本研究では、ユーザが与えるプログラムがすべき作業の例を Web ユーザインタフェース上での操作とし、作成するプログラムは Web サービス連携プロセスとすることによって、Programming by Example という方法を Web サービス連携プロセスの作成に適用する。

4.2 ユーザの操作例の記録

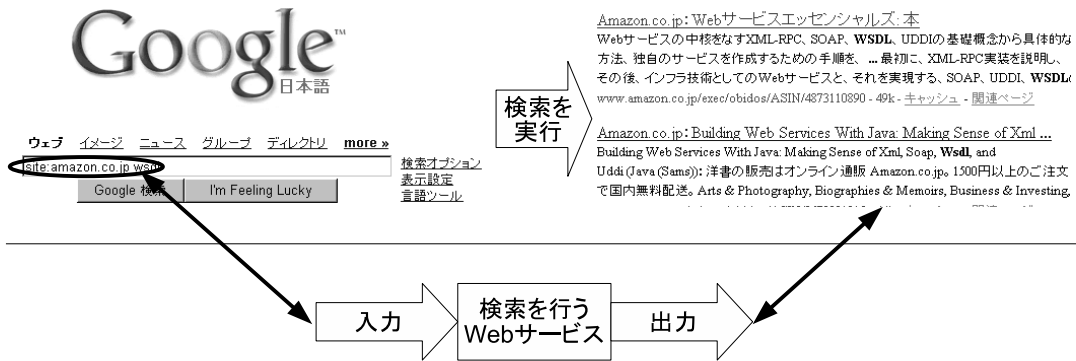
SOAP¹⁾ や WSDL による Web サービスは人間用のインタフェースを持っていない。しかし、XMethods²⁾ の Try It のように WSDL から Web ユーザインタフェースを作成するものは既に存在する。Web サービスが人間用のインタフェースを持つことにより、機械の側では Web サービスの実行であるものが、ユーザの側から見ると Web アプリケーションの実行とみなすことが可能となる。

図7は検索を行う Web サービスとその Web ユーザインタフェースの例である。検索を行う Web サービスが Web ユーザインタフェース上では検索を行う Web アプリケーションとなり、入力がテキスト型のフォームになり、出力が検索結果の Web ページとなる。Web サービスはユーザの側から見ると検索を行う Web アプリケーションがある Web ページとなる。ユーザがこの Web アプリケーションを使用するためには、フォームにテキストを入力し、検索を実行するためのボタンを押すことになる。このように Web ユーザインタフェース上で

¹⁾ <http://www.w3.org/TR/soap/>

²⁾ <http://xmethods.net/>

人間側の視点 (Webアプリケーション):



機械側の視点 (Webサービス):

図 7: Web サービスと Web アプリケーションの関係

表 1: Web サービスと Web アプリケーションの対応

	入力データ	実行されるサービス	出力データ
ユーザ側の視点	フォームに入力されるデータ	Web アプリケーションの実行	検索結果の Web ページ
機械側の視点	Web サービスに入力されるデータ	Web サービスの実行	Web サービスから出力されるデータ

行われた Web アプリケーションの操作について、表 1 のような関係を用いて、Web サービスの実行と対応付けることが可能となる。

Web サービスと Web ユーザインタフェース上の Web アプリケーションを結び付けているものがメタデータである。Web サービスに付与されているメタデータを Web ユーザインタフェースとなっている Web ページにも付与する。これによって、図 8 の例のように、メタデータによって Web ユーザインタフェース上の Web アプリケーションの実行と Web サービスの実行が結びつけられる。これにより、ユーザが Web 上で行った操作に対して、入力したデータのメタデータとデータの値、実行した操作、出力されたデータに付与されているメタデータとデータの値によって、機械側の視点から見ると、どのように Web サービスが実行されたかという情報を得ることができる。

また、ユーザは Web アプリケーションの実行結果ページからリンクのクリッ

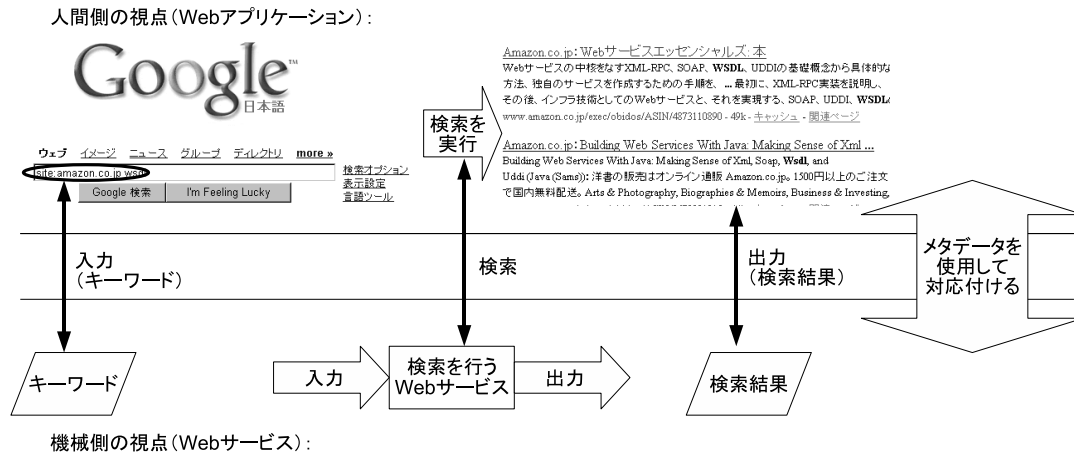


表 2: Web アプリケーションの実行以外のユーザによる操作

ユーザによる操作	入力データ	実行されるサービス	出力データ
リンクのクリック	現在の Web ページの URL	現在のページからリンクを選択	クリックされたリンクの URL, クリックされたリンクの URL のメタデータ
コピー & ペースト	現在の Web ページの URL	現在の Web ページからデータをコピー, 指定された場所にペースト	コピーされたデータ, ペーストした場所を示すデータ

クをしたり, コピー & ペーストを行い, 次の Web アプリケーションを実行する。そこで, Web サービスから作成された Web ユーザインタフェース上の Web アプリケーションの実行だけでなく, リンクのクリックやコピー & ペーストといった表 2 の操作の記録も行う。これによって, リンクのクリックやコピー & ペーストによって, ある Web アプリケーションの出力と別の Web アプリケーションの入力の間で行われるデータを選択し, 受け渡すことも記録し, 連携させることが可能となる。

表2にある2つの操作において、メタデータはユーザがなぜそのデータを取得したかという意図を表すデータの解説 (data description) を示すために用いる。

リンクのクリックは入力として、現在の Web ページの URL があり、その中からあるメタデータを持つリンクを選択するという操作となる。また、データのコピーでは現在の Web ページの URL からあるメタデータを持つデータをコピーし、ペーストはコピーしたデータがあるメタデータを持つ場所へペーストするという操作となる。

4.3 Web サービス連携プロセスの作成

ユーザによる操作例を全て記録したら、次にその操作例を用いて Web サービス連携プロセスを作成する。単純な方法としては、ユーザの各操作例を Web サービスの実行又はデータの選択とみなして、操作例で使用されたデータをそのまま使って連携プロセスを作成する方法である。

図9はメタデータを通じてユーザの操作例を使用されたデータをそのまま使用した Web サービス連携プロセスに対応付けたものである。

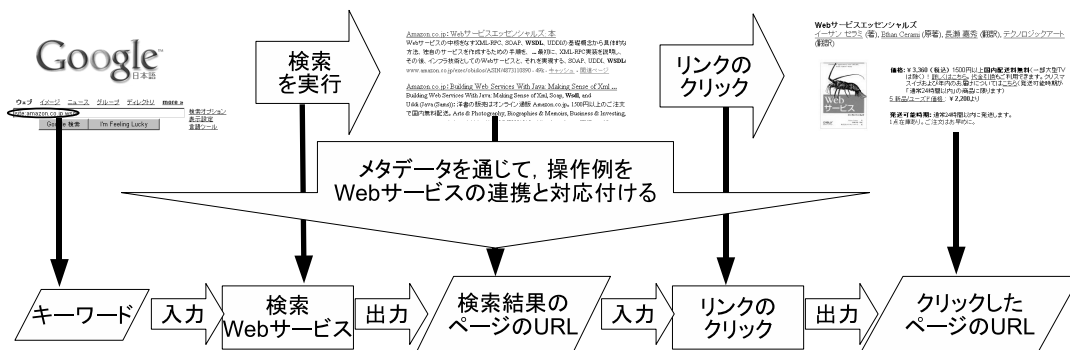


図9: 操作例のデータをそのまま使用した Web サービス連携プロセス

まず、ユーザはフォームに検索するキーワードを入力する。次に Web ページ上の検索ボタンを押して、検索を実行する。それから、検索結果が表示された Web ページから目的の Web ページのリンクを見つけ出し、クリックする。そして、目的の商品紹介の Web ページへと移動する。

システムはメタデータを使用してこの操作例を Web サービスの実行及びデータの選択に対応付ける。検索を行う Web アプリケーションの実行という操作を検索という機能を持つ Web サービスの実行とする。フォームに入力された ”

site:amazon.co.jp wsdl ”というキーワードが Web サービスの入力となり，検索結果の Web ページの URL が Web サービスの出力となる．

リンクのクリックという操作をあるリンクを選択するという機能を持つ操作の実行とする．検索結果の Web ページの URL が入力となり，商品紹介の Web ページの URL が出力となる．

この2つのサービスが連続して実行され，検索という機能を持つ Web サービスの出力である検索結果の Web ページの URL がリンクのクリックという操作の入力となっている．これにより，システムは検索機能を持つ Web サービスの出力をリンクのクリックという機能を持つ操作の入力として，データを選択して，受け渡しを行うことによって，Web サービス連携プロセスを作成する．

以上のように各操作をそれぞれ Web サービスの実行又はデータの選択とみなす．そして，あるサービスの出力データと別のサービスの入力データが同じならば，その2つの操作が連携して実行されたものとして Web サービス連携プロセスを作成する．これによって，ユーザが行った操作と全く同じことを Web サービス連携プロセスの実行として行えるようになる．

しかし，通常はユーザが行ったことと全く同一のデータを使用して同じことを繰り返す Web サービス連携プロセスではなく，ユーザによる操作例で使用されているデータ（data objects）をパラメータ化（parameterize）するか又は一般化（generalize）することによって，同じ操作を異なるデータに対して行うことができるようにしたい．

このことを実現するためには，ユーザが操作例において実際に使用したデータだけでなく，なぜそのデータを使用したかというユーザの意図を示すために，データの解説（data description）が必要となる．

本研究では，ユーザの意図を示すために使用するデータの解説に，操作例で使用されたデータに付与されているメタデータを使用する．つまり，システムはユーザがそのデータを使用した理由をそのデータに付与されているメタデータであるとみなす．これによって，ユーザが操作例において使用したデータを一般化して，同じメタデータを付与されている異なるデータについて同様の操作を行うことが可能となり，ユーザの操作例において使用されているデータを置き換えることが可能となる．

図10はユーザが行った操作と全く同じことをする Web サービス連携プロセスから，各サービスをメタデータで一般化し，変更できるようにしたものである．

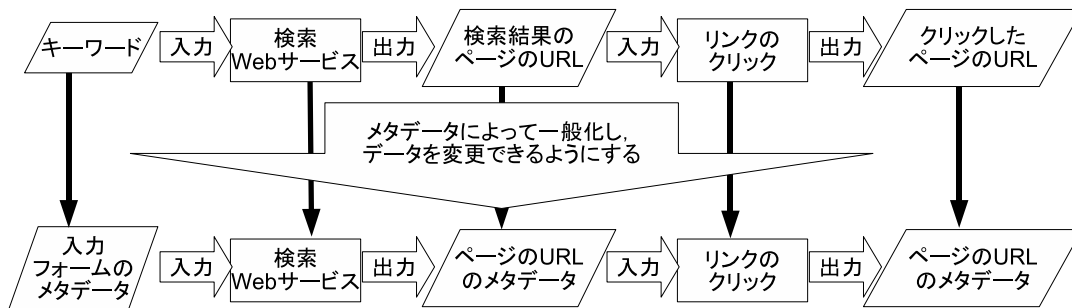


図 10: 一般化した Web サービス連携プロセス

検索という機能を持つ Web サービスの入力として、キーワードという具体的なデータではなくて、入力フォームに付与されているメタデータを使用する。出力としては具体的な検索結果の Web ページの URL ではなく、例で使用した Web ページの URL のメタデータを持つ Web ページの URL となっている。

リンクのクリックという操作の入力としては具体的な検索結果の Web ページの URL ではなく、検索結果の Web ページの URL のメタデータを持つ Web ページの URL となっている。出力は具体的な商品紹介ページの URL ではなくて、商品紹介ページの URL であるというメタデータを持つ Web ページの URL となっている。

このように、具体的なデータからそのデータが持つべきメタデータによって、ある Web サービスからの出力を別の Web サービスの入力に渡すという連携を記述する。これによって、使用されるデータを変更して、Web サービスを連携させることが可能となる。

第5章 実装

本章では、第3章、第4章の方法による Web サービス連携プロセスの作成を実現するために実装したシステムについて述べる。実装したシステムでは Web ユーザインタフェースやユーザの操作例を用いて Web サービス連携プロセスを作成及び実行する。

5.1 具体的な利用シナリオ

A さんはあるキーワードに関連した商品のアフィリエイト広告を自分の HP に掲載したい。

Amazon¹⁾のキーワードによる検索を行い、その検索結果に基づいた本の Amazon アフィリエイト広告を作成するサービスは既に存在する²⁾。しかし、Amazon のキーワードによる検索では本の題名や著者名等に基づいた検索結果しか返さない。本の紹介ページにあるレビュー、目次、カスタマーレビュー等の情報を用いた検索は行わない。

Aさんはレビュー、目次、カスタマーレビュー等の情報も含めてキーワードに関して検索を行い、そしてその検索結果に基づいた本のアフィリエイト広告を作成してくれる機能を持つサービスが欲しい。

検索を行ったり、ISBNからその本のアフィリエイト広告を作成する Web サービスは既に存在する。そこで、Aさんはこれらの Web サービスを連携させて欲しい機能を実現したい。Aさんはインターネットをよく使用するので、Web ブラウザ上での操作はできるが、Web サービスを使用したことや連携させたことはない。そこで、Web ブラウザ上での操作例を基に、同様の操作を行う Web サービス連携プロセスを作成してくれるシステムを使用して、Web サービス連携プロセスを作成する。

Aさんは欲しいサービスを実現するために、Web サービス連携プロセスが行うべき作業を Web ブラウザ上で行いシステムに操作例として与える。具体的な操作は以下ようになる。キーワード (i) に関して検索を行う (1)。検索結果の Web ページにある URL 一覧 (ii) から Amazon の商品紹介ページ (iii) をクリックする (2)。そして商品紹介ページ中の ISBN (iv) をコピーする (3)。その ISBN (iv) と Amazon のアソシエイト ID をアフィリエイト広告作成してくれる Web ページに入力し (4)、アフィリエイト広告 (v) を得る。

この操作例を基にシステムは同様の操作を行う Web サービス連携プロセス (図 11) を作成する。(i) ~ (v) は入出力データとして、(1) ~ (4) は Web サービスとデータの受け渡しとしてこの Web サービス連携プロセスと操作例が対応している。

¹⁾ <http://www.amazon.co.jp/>

²⁾ <http://www.goodpic.com/mt/aws/>

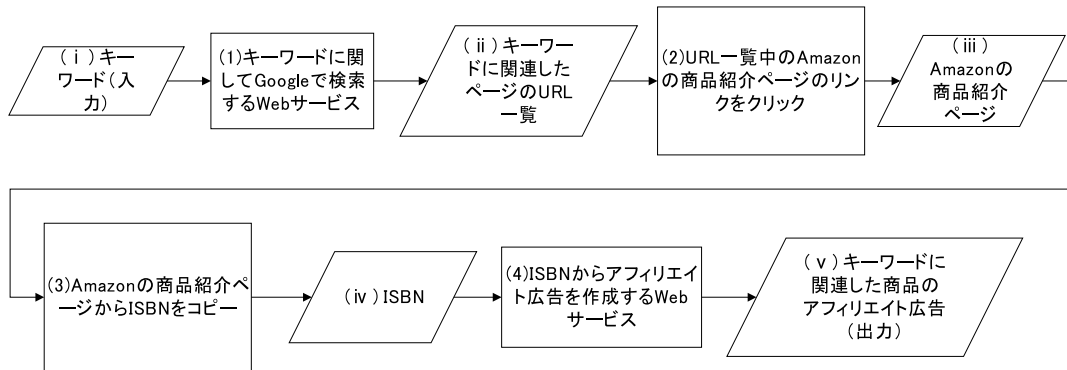


図 11: Amazon アフィリエイト広告作成のための Web サービス連携プロセス例

5.2 Web ユーザインタフェースとなる Web ページの作成

XMethods¹⁾の Try It のように WSDL から Web ユーザインタフェースを作成するサービスは既に存在する。現状では、この方法で提供される Web ユーザインタフェースは必ずしもユーザにとって使いやすいものではない。今回は、Web サービスと同じ機能を提供する使いやすい Web ページが既に存在するので、メタデータをその Web ページに付与する。そして、その Web ページを Web ユーザインタフェースとみなして使用する。

メタデータを Web ページに付与する方法として、今回は Web ページにメタデータを直接埋め込む方法をとった。具体的には、図 12 のような方法でメタデータを付与した。本文にないデータについては meta 要素によって記述し、本文中にあるデータについては ex:prop 属性で記述した。

RDF [5] では、メタデータをリソース、プロパティ、値、の 3 つ組で表現する。今回の方法では、リソースが Web コンテンツ自身の URI となり、値自体はコンテンツとして記載されていて、プロパティは XHTML タグの ex:prop 属性で関連付けられる。²⁾

5.3 Web サービス連携プロセス作成システム

本節では、第 3 章、第 4 章で示した方法によって、Web サービス連携プロセスの作成するために実装したシステムについて述べる。

¹⁾ <http://xmethods.net/>

²⁾ <http://www.net.intap.or.jp/INTAP/s-web/data/15-semanticweb-report.pdf>

```

<html
xmlns:ex="http://www.lab7.kuis.kyotou.ac.jp/
example/">
<head >
<title>Webサービス入門</title>
<meta schema="RDF" name="ex: keywords "
content="Webサービス入門" />
</head>
<body>>
<span ex:prop="ex:isbn"> 4894715902</span>
.....

```

本文中に無いデータについては
meta要素で記述

本文中にあるデータについては
ex:prop属性で記述

図 12: メタデータで注釈された Web ページ

5.3.1 システムの概要

システム構成図を図 13 に示す。本システムは、Web ブラウザ上でユーザのシステムへの操作を受け付け、そして Web ページ上におけるユーザの操作に関する情報を取得するクライアントの部分がある。そして、ユーザの操作例を記録し Web サービス連携プロセスの作成を行うサーバの部分がある。

処理の流れとしては、Web ブラウザ上でのユーザの操作例を記録し、それを用いて Web サービス連携プロセスを作成及び実行することになる。

ユーザの操作については、Bookmarklet を使用して Web 上でのユーザの操作を取得し、取得した情報をサーバ上にある Perl で書かれた CGI に送信して記録するという方法をとる。Web ブラウザ上でのユーザの操作を取得するために使用する Bookmarklet とは、Bookmark の URL 部分に JavaScript で書かれたプログラムを記述する。そして、登録された Bookmark をクリックすることにより、JavaScript で書かれたプログラムを実行することができるものである。この Bookmarklet という方法を使用することで、ユーザが使用しているブラウザにユーザの操作を取得するという機能を追加することができる。

次に記録されたユーザの操作例を基に、Web サービス連携プロセスを作成

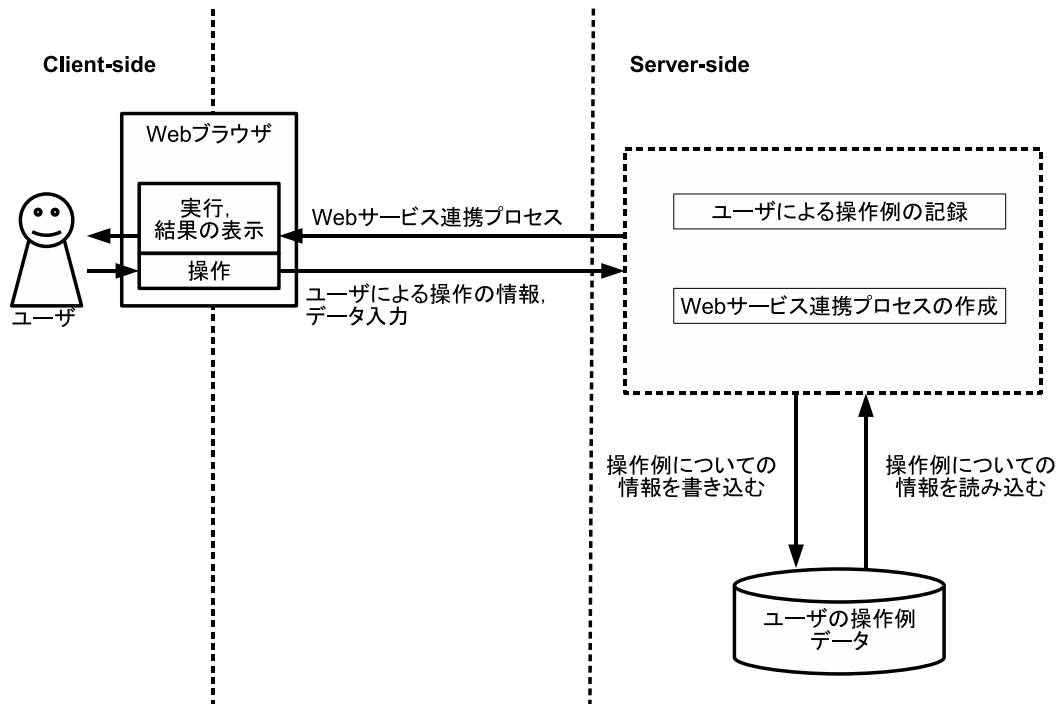


図 13: システム構成図

し、Web サービス連携プロセスを実行する際に使用するデータをユーザに入力してもらう。そして Web サービス連携プロセスを実行するためのプログラムを JavaScript で生成し、クライアント側に送信し、ユーザのブラウザ上で動作させ、動作経過や動作結果をユーザに表示する。

今回実装したシステムによりユーザの入力したデータを変更して、Web ユーザインタフェースを持つ Web サービスが連続して実行される Web サービス連携プロセスを作成できる。分岐 (branching) や反復 (iteration) のような制御構造を持つ Web サービス連携プロセスの作成は今回行わなかった。

5.3.2 ユーザが行った操作の記録

システムは Web ページ上におけるユーザの操作を記録し、操作例とする。ユーザからの Web サービス連携プロセス作成システムに対する操作は Bookmarklet (Bookmark 機能を使用して JavaScript で書かれたプログラムを実行する方法) によって行われる。

図 14 はユーザによる Web ユーザインタフェース上で行われた連続した操作例を全て取得するまでのシーケンス図となる。まず、最初に "連携プロセスの作成を開始" を実行し、サーバ側にあるユーザの操作例を記録するためのデー

タを作成又は初期化する．これにより，ユーザの操作例を記録するための準備が整ったことになる．

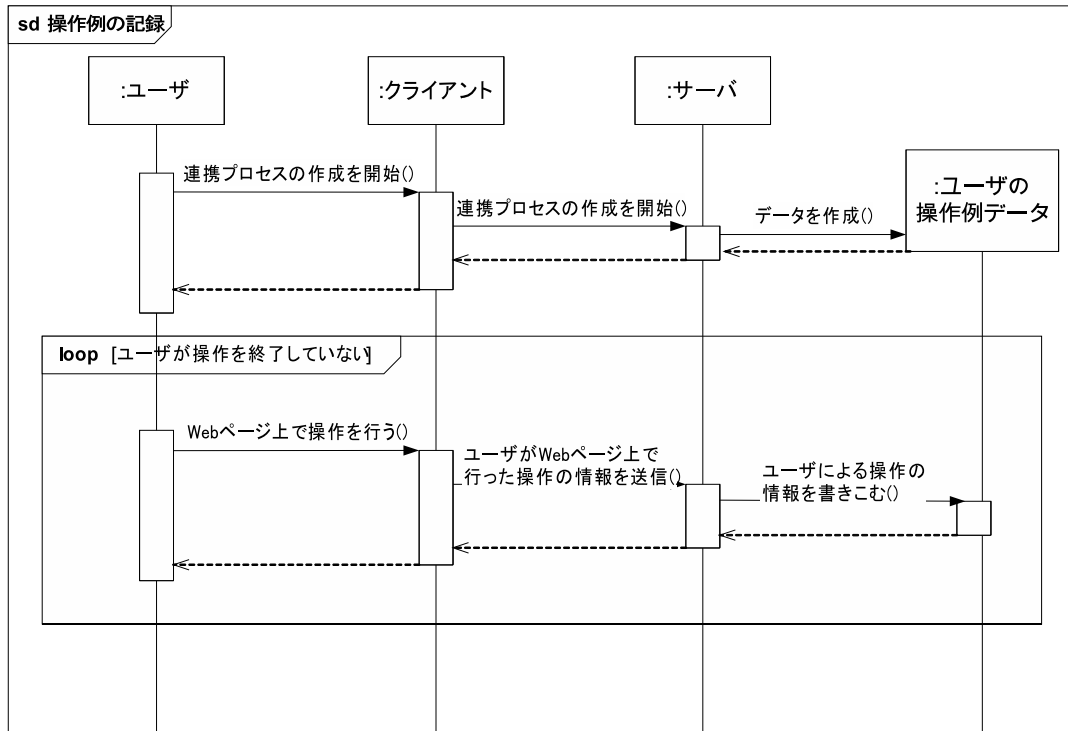


図 14: ユーザの全操作例を取得するまでのシーケンス図

次に，操作を行う Web ページへ移動し，“このページ上でのユーザの操作を記録”を実行する．これによって，各操作により生じるイベント毎にイベントハンドラを付与し，ユーザによる操作が行われる度にイベントハンドラを呼び出してユーザの操作についての情報を記録する．これによって Web ページ上でのユーザの操作を取得し，記録することが可能となる．

ユーザは全操作が終了するまで，各々の Web ページ上で同様の作業を繰り返し行う．これによって，システムはユーザの操作例を全て記録できる．

今回実装したシステムでは，Web アプリケーションの実行，データのコピー，リンクのクリックという 3 つの操作を操作例として記録する．上記の 3 つのイベントが起こる毎にイベントハンドラを呼び出して，Web ページに付与されているメタデータを基にしてユーザの操作を取得する．

各操作毎に，実行した操作，その操作に与えられた入力，その操作からの出力という情報を得る．実装したシステムで記録している情報は表 3 になる．

表 3: ユーザの操作から取得する情報

実行した操作の情報	操作に与えられた入力	操作からの出力
実行した操作の種類 (Web アプリケーションの実行), 実行した Web アプリケーションがある Web ページの URL, 実行した Web アプリケーションに付与されているメタデータ	入力データ, 入力データをを入力したフォームに付与されているメタデータ	Web アプリケーションの実行結果ページの URL
実行した操作の種類 (データのコピー)	コピーを実行した Web ページの URL	コピーされたデータ, コピーされたデータに付与されているメタデータ
実行した操作の種類 (リンクのクリック)	リンクのクリックを実行した Web ページの URL	クリックされたリンクの URL, リンクの URL に付与されているメタデータ

Web アプリケーションの実行という操作が行われたときに記録している情報は次のようになる。操作の種類として Web アプリケーションの実行, そして実行した Web アプリケーションがある Web ページの URL とその Web アプリケーションに付与されているメタデータを取得する。これによって, どの Web サービスを実行したかを特定することが可能になる。Web サービスへの入力としては, その Web アプリケーションに入力されたデータとその入力データをを入力したフォームのメタデータを取得する。Web アプリケーションからの出力データは Web アプリケーションの実行結果ページの URL となる。

データのコピーという操作が行われたときに記録している情報は次のようになる。操作の種類としてデータのコピーという情報を取得する。これによって, 実行した操作の特定が可能になる。この操作の入力はデータのコピーを実行した Web ページの URL となる。また, この操作の出力としては, コピーしたデータとコピーしたデータに付与されているメタデータとなる。

リンクのクリックという操作が行われたときに記録している情報は次のようになる。操作の種類としてリンクのクリックという情報を取得する。これによって、実行した操作の特定が可能になる。この操作への入力データはリンクのクリックを実行した Web ページの URL となる。また、この操作の出力としては、クリックされたリンクの URL とそのリンクの URL に付与されているメタデータとなる。

5.3.3 Web サービス連携プロセスの作成及び実行

システムはユーザの操作例を用いて Web サービス連携プロセスを作成し、再び実行できるようにする。

ユーザの操作例を用いて作成される Web サービス連携プロセスはユーザの行ったことと全く同じことを繰り返すのではなく、操作例において行った操作を一般化することによって、Web サービス連携プロセスの実行時に入力するデータの変更を可能とする Web サービス連携プロセスになる。

この与えられた操作例を一般化するために、ユーザがなぜそのデータを使用したかという意図を示すデータのデータの解説 (data description) が必要となる。このシステムではこのデータの解説に Web ページに付与されているメタデータを使用する。システムはこのユーザの意図とみなしたメタデータに従って、データを選択しながら、Web サービス連携プロセスを実行する。これによって、ユーザは与えた操作例で使用されているデータを変更することができるようになる。

図 15 はユーザの操作例を用いて、Web サービス連携プロセスを作成及び実行するシーケンス図となる。

ユーザが全ての操作を終えてシステムに操作例を与え終わったら、“連携プロセスの作成及び実行”を実行する。これによって、Web サービス連携プロセスが作成及び実行される。まず、システムはユーザの操作例を記録したデータを読み込み、ユーザが変更する可能性のあるデータを選び出す。Web サービスを実行する際に、入力となるデータはユーザが直接入力したデータと前に実行した Web サービスから出力されたデータの 2 種類が存在する。Web サービス連携プロセスの実行時に与えた操作例で使用されているデータを変更するために、読み込んだユーザの操作例データを基に、変更する可能性のあるデータを選び出す必要がある。変更する可能性のあるデータは、ユーザがコピー＆ペーストしていない直接入力したデータになる。

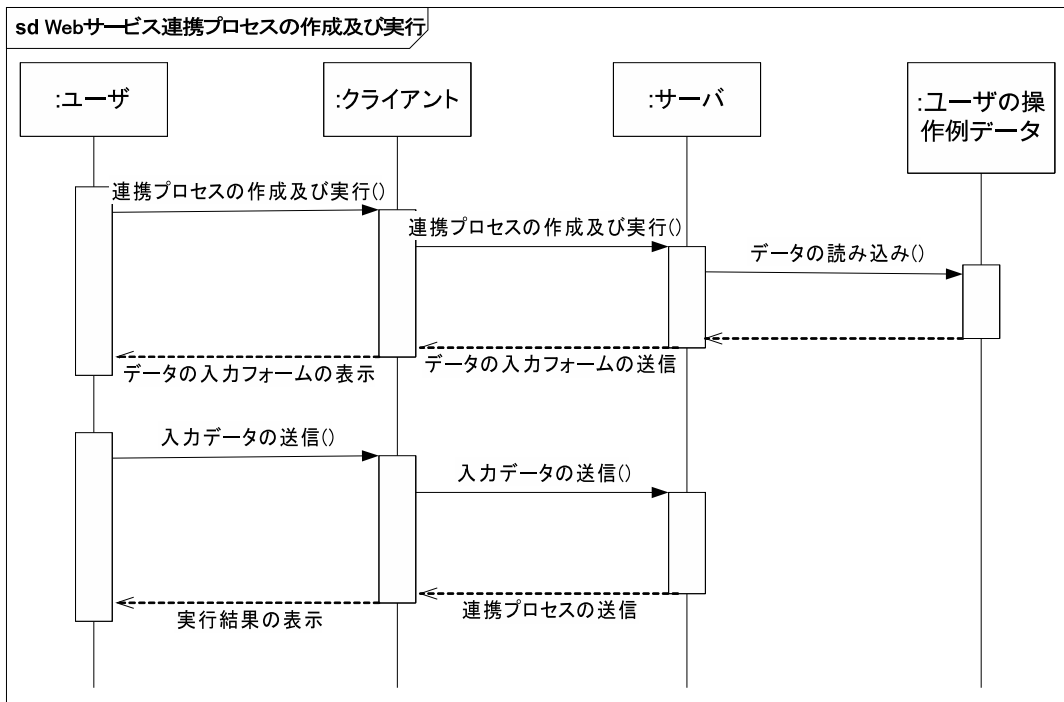


図 15: Web サービス連携プロセスの作成及び実行のシーケンス図

次に、ユーザが操作例において直接入力したデータを提示し、変更するかどうか確認を求める。具体的には、図 16 のようにフォームを表示する。フォームには操作例のときに使用した値が予め入力してあり、ユーザが変更して実行したい部分のみを変更して Web サービス連携プロセスの実行ボタンをクリックする。

input_googleSearch =

input_associateID =

図 16: Web サービス連携プロセスを実行するとき使用するデータの入力

そうすると、システムは入力されたデータを使用して Web サービス連携プロセスを作成する。ユーザが直接入力したデータはそのまま使用し、Web サービス間で受け渡されているデータについては操作例で使用されたデータと同じメタデータを持つデータを選択し、次の Web サービスに受け渡す。

このようにして、ユーザの操作例に基づく Web サービス連携プロセスを実行

するための JavaScript によるコードを作成し，クライアント部に送信し，ユーザが使用しているブラウザ上で実行させる．Web サービス連携プロセスの実行をする際に，Web サービス連携プロセスの実行状況をユーザに提示しながら実行する．これにより，ユーザは自分が行った操作例に基づいてどのような Web サービス連携プロセスが作成されたのを確認できる．

第6章 おわりに

既存の Web サービス連携プロセスを作成する研究は Web サービス連携プロセスの実行や実現可能性に重点を置いている．ユーザはワークフロー図によるインタフェースを習得したり，ワークフローによって Web サービス連携プロセスを記述する方法を習得しなければならない．ユーザにとって手間がかかり困難である．本研究では，ユーザが簡単に Web サービス連携プロセスの作成を可能にするために，以下の2つの問題に取り組んだ．

1. 新たなインタフェースを習得することが必要

従来のシステムは Web サービス連携プロセスを作成するために，一般のユーザにとって既知のインタフェースではなく，ワークフロー図等の一般のユーザにとって使用したことのないインタフェースを使用している．このため，ユーザは新たにそのインタフェースの使い方を習得しなければならない．

2. ワークフローの知識を習得することが必要

Web サービス連携プロセスを記述するために，BPEL4WS 等のワークフロー記述言語が使用されている．Web サービスの連携を表現するためにワークフローが使用されているため，ユーザはワークフローによる処理手順を規定するための方法を習得しなければならない．

本研究では，以下の2つの方法を使用することにより，上記の問題の解決を目指した．

1. Web ユーザインタフェースの使用

ユーザにとって既知のユーザインタフェースとして，Web ユーザインタフェースを使用する．Web ユーザインタフェースとは，ユーザが普段 Web ブラウザ上で行っているリンクのクリック，コピー&ペースト，ボタンのクリック等の操作を使用するユーザインタフェースである．普段から使用

している Web ユーザインタフェースを使うことにより，新たなユーザインタフェースを習得するという負担がなくなる．

2. Programming by Example の適用

Web サービス連携プロセスを作成するための方法として，Programming by Example を使用する．ユーザが与える例をユーザが普段行っている Web ブラウザ上での操作とする．これにより，ワークフローに関する知識を習得しなくても，普段ユーザが行っている操作によって Web サービス連携プロセスの作成が可能になる．

ユーザが操作例において使用したデータのメタデータをなぜユーザがそのデータを使用したかというユーザの意図を示すものとしてみなす．操作例で使用されたデータと異なっている場合，データに付与されているメタデータが同じならば，ユーザが意図したとおりの操作であるとみなす．これによって，データがメタデータによって一般化され，作成された Web サービス連携プロセスを実行する際に，使用するデータを変更することが可能になる．

また，上記の方法による Web サービス連携プロセスの作成を実現するためのシステムを実装した．実装したシステムを用いて，Web ユーザインタフェース上での操作例に基づき，Web サービス連携プロセスを作成及び実行した．実装したシステムを使用して，特別な知識を必要とせずに，ユーザが Web サービス連携プロセスを作成及び実行できることが確認できた．

謝辞

本研究をすすめるにあたり，多大なるご指導とご助言を頂いた石田亨教授に厚く御礼を申し上げます．

また，研究全般にわたり多くの有益なご助言を頂きました CHO Heeryon 氏，岩部正明氏，谷塚俊輔氏に深く感謝致します．

最後になりますが，日頃から様々なご助言やご協力を頂きました石田研究室の皆様方に心より感謝致します．

参考文献

- [1] Thatte, S.: Business Process Execution Language for Web Services version 1.1, Technical report, IBM, BEA Systems, Microsoft, SAP AG, Siebel Systems (2003).
- [2] Agarwal, V., Dasgupta, K., Karnik, N., Kumar, A., Mittal, A. K. S. and Srivastava, B.: A Service Creation Environment Based on End to End Composition of Web Services, *Proceedings of the 14th international conference on World Wide Web*, pp. 128–137 (2005).
- [3] Halbert, D. C.: *Programming by Example*, PhD Thesis, University of California Berkeley (1984).
- [4] Myers, B. A.: Visual Programming, Programming by Example, and Program Visualization: A Taxonomy, *Proceedings of the CHI'86 Conference on Human Factors in Computing Systems and Graphic Interfaces*, pp. 59–66 (1986).
- [5] Klyne, G., Carroll, J. J. and McBride, B.: Resource Description Framework (RDF): Concepts and Abstract Syntax, Technical report, W3C (2004).