社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE

BPEL ワークフローに着目した連携 Web サービスの応答速度・稼働率の見積もり手法

前島 弘敬† 大西 洋司‡ 中村 匡秀‡ 松本 健一‡

†奈良先端科学技術大学院大学 情報科学研究科 〒630-0192 奈良県生駒市高山町 8916-5

E-mail: †{hirotaka-m, yoji-o, masa-n, matumoto}@is.naist.jp

あらまし Web サービス技術の普及に伴い Web サービスが増加してきたことによって，サービス利用者は数あるサービスの中から品質の高いサービスを選択する機会が与えられるようになった．サービスとしての品質を比較するためには Web サービスの品質を定量的に測定する技術が必要とされる．しかし，既存の研究は単体 Web サービスを対象としたものが多く，連携 Web サービスにおける有効な品質測定手法は確立していない現状にある．そこで本稿では，連携 Web サービスの品質測定の一手法として，品質を静的に見積もる手法を提案する．提案手法では，BPEL の記述から連携のワークフローを制御する論理構造要素を抽出することで，ワークフローを分解して解釈する．抽出された論理構造要素ごとに演算式を対応付け，ワークフローを数式化することで連携 Web サービスの品質見積もりを実現している．

キーワード BPEL，ワークフロー，論理構造要素，連携 Web サービス，応答速度，稼働率，見積もり

Estimating Response Time and Availability of Composite Web Services From BPEL Workflow

Hirotaka MAESHIMA† Yoji ONISHI‡ Masahide NAKAMURA‡ and Ken-ichi MATSUMOTO‡

†Graduate School of Information Science, Nara Institute of Science and Technology 8916-5, Takayama, Ikoma, Nara 630-0192 Japan

E-mail: †{hirotaka-m, yoji-o, masa-n, matumoto}@is.naist.jp

Abstract The users of Web service have been given chances to select the high quality services from many services, because of the increase of Web services with the advances of Web service technology. Some technologies which measure the quality of Web service quantitatively are needed to compare the quality as services. However, many existing studies focus on the individual Web service, therefore there is no effective methods of quality measurement is established for integrated Web service. In this paper, we propose a method which estimates the quality statistically as one of the methods of quality measurement for integrated Web service. Associating an operation with each of structural constructs derived, we translate the given workflow into a formula, from which we derive the estimation of the response time and the availability of the composite Web service.

Keyword BPEL，Workflow，Structural Constructs，Composite Web Services，Response Time，Availability，Estimation

1. はじめに

Web サービス技術が普及してきたことで，公開・提供される Web サービスの数が増加し，サービス内容に差異のない類似 Web サービスが出現するようになってきた．また，Web サービスの特徴である疎結合による連携の容易さから，複数の Web サービスを連携・統合した連携 Web サービスも提供されるようになってきた．このように，Web サービスの増加によって，サービス利用者は数あるサービスの中で最も品質の高いサービスを選択する機会が与えられるようになった．一方，サービス提供者は，利用者獲得のために品質の高いサービスを提供する必要がある．

サービスの品質を評価・比較するためには，サービス品質(QoS)を定量的に測定する技術が必要となる．近年，Web サービスの品質測定手法や，利用者・提供者間のサービスレベル合意（Service Level Agreement, SLA）などの研究が盛んになってきた[3]．しかし，これら既存研究のほとんどは，**単体 Web サービス**を対象としたものが多く，複数の Web サービスを連携・統合した**連携 Web サービス**に対して，体系的な品質の測定手法が確立していない現状にある．

本稿では，連携 Web サービスの品質測定の一手法として，連携 Web サービスの応答時間と稼働率を静的に見積もる手法を提案する．具体的には，連携 Web サービスの BPEL ワークフロー記述と，連携サービスで用いる個々の単体サービスの応答時間（または稼働率）が与えられたとき，BPEL の制御構造に基づいて連携サービス全体の

応答時間（または稼働率）を算出する．算出においてまず，BPELの記述構造からワークフローの基本的な論理構造要素を抽出・分解することで，ワークフローを細分化して考える．抽出された論理構造要素はそれぞれ演算式が対応付けられ，演算式の組み合わせからワークフローを数式化するというアプローチにより連携Webサービスの品質見積もりを実現する．品質の見積もりは導出された数式に既存の単体 Web サービスの品質測定手法によって算出された個々のWebサービスの応答速度や稼働率を代入することで表される．

提案手法では，連携Webサービスの品質をオフラインで見積もる手段を提供する．したがって，連携サービスのプロトタイピングやシミュレーション，連携構造の設計等の支援に活用することが可能である．

2. 準備

2.1 BPEL

BPEL (Business Process Execution Language) [5]は，Web サービスの連携を実現するためのプロセス記述言語であり，連携するWebサービス間の関係や**連携ワークフロー**を記述することで煩雑なプロセスを自動化することができる．現在，連携定義の仕様として標準化が進められており，標準規格として定着しつつある．

BPEL は XML をベースとした記述言語であり，タグを用いて表記する．タグで表記されたBPELの記述構造は以下の4つの要素から構成される．

サービス間の関係定義

連携するWebサービスとの関係を<partnerLinks>タグで定義する．

プロセスデータの定義

連携するWeb サービス間でのメッセージの送受信や，メッセージの受け渡しに利用される変数といった，プロセスデータを<variables>タグで定義する．

ハンドラ記述

イベントやエラーが発生した場合の処理や，それに伴う補償処理を記述する．それぞれ<eventHandlers>，<faultHandlers>，<compensationHandlers>タグを用いて表現する．

アクティビティ記述

連携されるWebサービス間のワークフローを記述する．例えば，逐次実行では<sequence>，並列実行では<flow>タグが用いられ，このようなタグの組み合わせにより連携のワークフローを表現する．

表1．BPELの論理構造要素

基本制御	逐次実行 並列実行(AND) タイマーイベント待ち プロセス即時終了
経路選択	排他選択(XOR) 包含選択(OR)
ループ制御	標準ループ マルチインスタンスループ
例外処理	メッセージ例外 タイマー例外 エラー処理 補償処理

論理構造要素

BPEL によって記述された連携 Web サービスのワークフローには，ワークフローを制御する基本となる**論理構造要素**が存在する[4]．表1に論理構造要素の分類を示す．

2.2 Web サービスにおける品質

本稿では，見積もりの対象とするWebサービスの品質として，**応答速度**と**稼働率**を取りあげる．これらは，SLA（Service Level Agreement）[2]と呼ばれるWebサービスの品質保証制度において重要視されている品質項目である．

2.2.1 応答速度

Web サービスにおける応答速度は，以下の2つのプロセスに要する時間の総和として算出される．

サーバ（Web サービス提供先）までのアクセス時間

クライアントであるユーザからサーバまで，メッセージの送受信に要する時間．

Web サービスの実行時間

クライアントであるユーザからの要求に従って Web サービスが実行され，終了するまでに要する時間．

2.2.2 稼働率

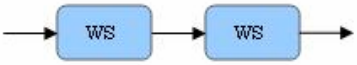
稼働率は，以下の定式から算出される．全運転時間はサービスが提供されるべき時間を表し，トラブルの発生によりサービスが提供されなかった時間は故障時間として表す．

稼働率 = 稼働時間(全運転時間 − 故障時間) / 全運転時間

3. BPEL の論理構造要素

BPEL で記述された連携Webサービスのワークフローは，**順次・分岐・反復**といった論理構造要素の組み合わせによって表現することができる．以下では，表 1 に示した各論理構造要素について，BPMN(Business Process Modeling Notation)[1]と呼ばれるプロセス表記法によって可視化されたモデルを示す．また，モデルに対応するBPEL の記述構造から，基本となるワークフローを説明する．以降では，個々の単体Webサービスを WS として表す．なお，紙面の制限のため，表1における基本制御のうちのタイマーイベント待ちとプロセス即時終了，および全ての例外処理については本稿では取り上げない．

3.1 逐次実行

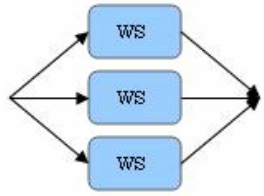



```
<sequence>
  <empty name="WS" />
  ・・・
</sequence>
```

図．1 逐次実行を表すBPMNモデルとBPEL構造

逐次実行は<sequence>タグで表現され，<empty>タグに実行されるWS が記述される．個々の WS の実行が終了するごとに後続の WS が実行される．

3.2 並列実行（AND）

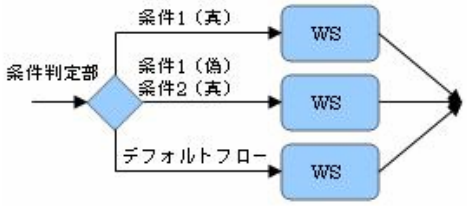



```
<flow>
  <empty name="WS" />
  ・・・
</flow>
```

図. 2 並列実行を表す BPMN モデルと BPEL 構造

並列実行は<flow>タグで表現される．<flow>タグ内に記述された WS は同時並列で実行され，全ての WS が終了することで後続の WS が実行される．

3.3 排他選択（XOR）

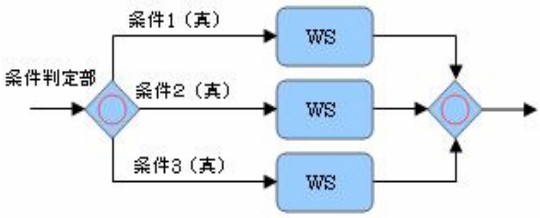



```
<switch name="XOR 分岐">
  <case condition="条件 1">
    <empty name="WS" />
  </case>
  ・・・
  <otherwise>
    <empty name="WS" />
  </otherwise>
</switch>
```

図. 3 排他選択を表す BPMN モデルと BPEL 構造

条件判定により，後続のフローが決定する．<switch>タグには分岐の説明が記述され，<case>タグにフローを決定する条件と，条件が一致した場合に実行される WS を記述する．いずれの条件とも一致しない場合，デフォルトフローとして<otherwise>タグに記述された WS が実行される．排他選択ではいずれか 1 つの条件としか一致しないため，実行される WS は 1 つだけである．

3.4 包含選択（OR）




```
<flow name="OR 分岐">
  <switch>
    <case condition="条件 1">
      <sequence>
        <empty name="WS">
      </sequence>
    </case>
    ・・・
  </switch>
</flow>
```

図. 4 包含選択を表す BPMN モデルと BPEL 構造

排他選択と同様に，条件判定により後続のフローが決定する．WS の実行を決定する条件は互いに独立しているため<switch>，<case>タグによって記述された WS の中で条件と一致したものが全て実行される．また，それらは並列実行を表す<flow>タグにより同時並列で実行される．

3.5 標準ループ


```
<while condition="ループ条件">
  <sequence>
    <empty name="WS" />
    ・・・
  </sequece>
</while>
```

図. 5 標準ループを表す BPMN モデルと BPEL 構造

<while>タグに記述されたループ条件を満たすまで，WS を逐次実行する．ループ条件を満たすことでループが終了し，後続の WS が実行される．

3.6 マルチインスタンスループ


```
<sequence>
  <while condition="ループ条件">
    <invoke>
    ・・・
    </invoke>
  </while>
  <while>
    <receive>
    ・・・
      <empty name="WS" />
    ・・・
    </receive>
  </while>
</sequence>
```

図.6 マルチインスタンスループを表す BPMN モデルと BPEL 構造

標準ループと同様に, <while>タグに記述されたループ条件を満たすまで WS が実行される. マルチインスタンスループでは, <invoke>タグによって，実行される WS のインスタンスが生成されるため，WS が同時並列で実行される．

4. 提案見積もり手法

本稿の目的は, 連携のワークフローを記述している BPEL と, 個々の Web サービスの応答速度や稼働率から，連携 Web サービスの応答速度や稼働率を見積もるための手法を提案することにある．提案手法では, 連携を構成する個々の Web サービスの応答速度や稼働率は既存の単体 Web サービスの品質測定手法によって算出できるため既知であると仮定している．

また，本稿における品質の見積もりは，サービス利用者がサービスを享受できることを前提条件として考えているため，サービスの強制終了や実行エラーの通知といった期待されない結果につながるフローは考慮していない．算出される品質はプロセスが最後まで実行された上で保証可能な水準として定め，応答速度では，プロセス実行において最も時間を要するサービスを，稼働率では最も稼働率の低いサービスを採択し，品質の見積もりを行うものと定義する．

4.1 見積もりのためのアプローチ

連携 Web サービスの品質を見積もるために，まず，与えられた BPEL に対して連携ワークフローを制御する論理構造要素に分解することでワークフローを細分化する．抽出されたそれぞれの論理構

造要素に対して演算式を対応付け，ワークフローを演算式の組み合わせからなる数式として解釈する．このように，本稿で提案する見積もり手法ではワークフローの数式化というアプローチにより連携 Web サービスの品質の見積もりを実現する．以下では見積もりの対象となる応答速度と稼働率の場合に分けて，論理構造要素ごとに対応付けられる演算式の導出方法について説明する．

4.2 応答速度の見積もり

連携 Web サービスの応答速度を見積もるために，まず，論理構造要素ごとに演算式を導出する．論理構造要素から算出される応答速度を **BPEL 応答速度**と呼び **BRT** と表し，個々の単体 Web サービスの応答速度を **RT** と表す．なお，演算式の導出には以下の論理式を用いる．

Reg（）： （）内が真であれば 1，偽であれば 0 を返す．
!： 論理式の前に添えて否定を表す．
MAX（）： （）内に指定された値の中から最大値を返す
MIN（）： （）内に指定された値の中から最小値を返す．

4.2.1 逐次実行

図.1 より，WS の実行が終了することで後続の WS が実行される場合，BRT はそれぞれの WS の応答速度である RT の和として表される．

BRT = RT1 + RT2 + ・・・ + RTn

4.2.2 並列処理（AND）

図.２より，WS が同時並列で実行される場合，本稿の応答速度の定義より，実行される複数の WS の中で最も時間を要する RT が採択される．

BRT = MAX（RT1, RT2, ・・・ , RTn）

4.2.3 排他選択（XOR）

図.3 より，条件分岐では実行される WS が未知であるため，条件の一致により実行される WS を特定する必要がある．実行される WS の特定には Reg（）が用いられ，以下のような基本式として表される．これにより実行された WS の応答速度だけを考慮することができる．

Reg(条件 1) * RT1

この基本式から演算式を導出すると，以下のように表される．

BRT = Reg(条件 1) * RT1 + Reg(条件 2) * RT2
+ ・・・ + Reg(条件 n) * RTn

4.2.4 包含選択（OR）

図.４より，条件分岐では，実行される WS を特定する必要があるため，排他選択と同様に基本式を用いて考える．包含選択では，それぞれの WS の実行は同時並列で行われるため，並列実行と同じ式構造になる．

BRT = MAX｛Reg(条件 1) * RT1, Reg(条件 2) * RT2
・・・ , Reg(条件 n) * RTn｝

4.2.5 標準ループ

図.5 より，指定したループ条件を満たすまで繰り返し WS が実行されるため，ループ条件を満たす回数だけ RT が加算される．

BRT = RT * (ループ回数)

4.2.6 マルチインスタンスループ

図.6 より，指定したループ条件を満たすまで繰り返し WS が実行される．マルチインスタンスループでは，それぞれの WS の実行が同時並列で行われるため，WS の応答速度は 1 度実行した場合の値と等価である．

BRT = MAX（RT, RT ・・・, RT）= RT

4.3 稼働率の見積もり

連携 Web サービスの稼働率を見積もるために，まず，論理構造要素ごとに演算式を導出する．論理構造要素から算出される稼働率を **BPEL 稼働率**と呼び **BAR** と表し，個々の単体 Web サービスの稼働率を **AR** と表す．

4.3.1 逐次実行

図.1 より，WS の実行が終了することで後続の WS が実行される場合，後続の WS への遷移確率は実行された WS の稼働率と等価であるため，BAR はそれぞれの WS の稼働率である AR の積として表される．

BAR = AR1 * AR2 * ・・・ * ARn

4.3.2 並列処理（AND）

図.２より，WS が同時並列で実行される場合，本稿の稼働率の定義より，実行される複数の WS の中で最も稼働率の低い AR が採択される．

BAR = MIN（AR1, AR2, ・・・, ARn）

4.3.3 排他選択（XOR）

図.3 より，条件分岐では完了される WS が未知であるため，条件の一致により実行される WS を特定する必要がある．実行される WS の特定は，応答速度と同様に Reg（）を用いて表される．しかし，稼働率は後続の WS への遷移確率を表しているため，条件に一致しなかった場合の稼働率を考慮する必要がある．本稿の稼働率の定義を満たし，かつ，BAR を実行された WS の稼働率だけで表現するために，実行されない WS の稼働率を 1 とすることで，実行の有無に関わらず後続の WS が実行されるように配慮する．これにより，条件分岐における稼働率を表す基本式は以下のように定められる．

｛Reg(条件 1) * AR1 + !Reg(条件 1)｝

この基本式から演算式を導出すると，以下のように表される．

BAR = ｛Reg(条件 1) * AR1 + !Reg(条件 1)｝
*** ｛Reg(条件 2) * AR2 + !Reg(条件 2)｝**
・・・
*** ｛Reg(条件 n) * ARn + !Reg(条件 n)｝**

4.3.4 包含選択（OR）

図.４より，条件分岐では，実行される WS を特定する必要があるため，排他選択と同様に基本式を用いて考える．包含選択では，それぞれの WS の実行は同時並列で行われるため，並列実行と同じ式構造になる．

BAR = MIN［｛Reg(条件 1) * AR1 + !Reg(条件 1)｝
, ｛Reg(条件 2) * AR2 + !Reg(条件 2)｝
・・・
, ｛Reg(条件 n) * ARn + !Reg(条件 n)｝］

4.3.5 標準ループ

図.5 より，指定したループ条件を満たすまで繰り返し WS が実行されるため，ループ条件を満たす回数だけ AR が掛け合わされる．

BAR = AR^(ループ回数)

4.3.6 マルチインスタンスループ

図.６より，指定したループ条件を満たすまで繰り返し WS が実行される．マルチインスタンスループでは，それぞれの WS の実行が同時並列で行われるが，全ての WS が正常に実行される確率（稼働率）は実行される WS の稼働率を実行回数だけ掛け合わせたものと等価であるため，標準ループと同じ式構造になる．

BAR = AR^(ループ回数)

4.4 連携 Web サービスの品質見積もり

連携 Web サービスの品質を見積もるために，論理構造要素ごとに導出した演算式を組み合わせることで連携のワークフローを表現する必要がある．論理構造要素の組み合わせる上で考慮すべき点として，以下の 2 つのケースを取りあげる．以下では，連携 Web サービスの応答速度，稼働率を **BPEL 連携応答速度**，**BPEL 連携稼働率**と呼び，CRT，CAR と表す．

4.4.1 包含

複雑なワークフローでは，論理構造要素が自身を包含するような場合が考えられる．この場合，入れ子による表現からワークフローを数式化することができる．図 8 は，連携 Web サービス WS α のワークフローを表す BPMN モデルであり，並列実行内に逐次実行がみられる包含の一例である．このワークフローに対し，演算式の導出手順を以下で述べる．

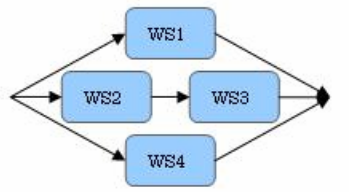


図.7 WS α のワークフローを表す BPMN モデル

まず，逐次実行に該当する部分を演算式に置き換える．演算式を入れ子にすることで，1 つの Web サービスとして解釈する．これにより，図. 7 のワークフローは論理構造要素である並列実行の形に表現され，以下のように立式される．

WSi の応答速度 = RTi (1≦i≦4)
WSi の稼働率 = ARi (1≦i≦4)
WS α の応答速度，稼働率 = CRT α , CAR α
CRT α = MAX｛RT1,（RT2 + RT3）, RT4｝
CAR α = MIN｛AR1,（AR2 * RT3）, AR4｝

4.4.2 再帰

連携サービスを構成する要素の中に連携 Web サービスが含まれている場合がある．この場合，ワークフローを記述した BPEL と連携を構成する個々の Web サービスの品質から再帰的に処理することで数式化することができる．図.8 は，図. 7 の WS α を含む連携 Web サービス WS β のワークフローを表す BPMN モデルである．このワークフローに対し，演算式の導出手順を以下で述べる．

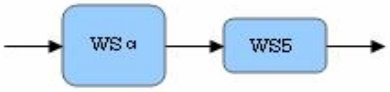


図.8 連携 Web サービスを包含した例

図.8 より，WS β のワークフローは，逐次実行の形をしているため，逐次実行に対応付けられた演算式より立式する．しかし，WS β の中には連携 Web サービスである WS α が存在するため，上と同様の手順から再帰的に処理することで以下のように立式される．

WSi の応答速度 = RTi (1≦i≦5)
WSi の稼働率 = ARi (1≦i≦5)
WS α の応答速度，稼働率 = CRT α，CAR α
WS β の応答速度，稼働率 = CRT β，CAR β

CRT β = CRT α + BRT5
= MAX｛RT1,（RT2 + RT3）, RT4｝+ BRT5
CAR β = CAR α * BAR5
= MIN｛AR1,（AR2 * RT3）, AR4｝* BAR5

5. ケーススタディ

5.1 旅行計画作成 Web サービス

ケーススタディとして，複数の Web サービスから構成される旅行計画作成 Web サービスに，提案手法を適用することで連携 Web サービスの応答速度（または稼働率）の見積もりを行った．図.9 にある (a)，(b) は，それぞれ旅行計画 Web サービスのワークフローを表す BPMN モデルと BPLE 記述例である，それぞれ同じ機能を実現しているが，記述構造が異なるため，ワークフローに違いが見られる．このサービスは，旅行計画作成に必要な日程・目的・交通手段といった情報を受け付ける旅行計画受け付けサービス，交通手段に応じた路線・空席照会の情報取得サービス，目的地近郊の旅館検索サービス，指定された旅館の空室確認サービス，作成した旅行計画の通知サービスから構成されている．

表.2 各 Web サービスにおける品質

	応答速度 (ms)	稼働率
旅行計画受付サービス（WS1）	0.3	0.998
旅館検索サービス（WS2）	0.5	0.997
空室確認サービス（WS3）	0.4	0.999
路線探索サービス（WS4）	0.6	0.995
空席照会サービス（WS5）	0.4	0.994
旅行計画通知サービス（WS6）	0.3	0.992

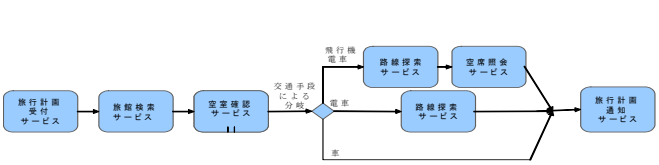
表.2 は，連携を構成する個々の Web サービスの応答速度（または稼働率）である．以下では，それぞれの Web サービスの略式である WSi（1≦i≦6）の応答速度（または稼働率）から提案手法の手順に従って品質を見積もる．なお，マルチインスタンスループによって旅館の空室確認サービスが実行されるループ回数は 3 として，排他選択における条件 1 を電車と飛行機による移動，条件 2 を電車のみによる移動と表した上で，ユーザが条件 1 を指定した場合における品質を見積もる．

図 9（a）では，BPEL 記述から論理構造要素ごとに演算式を対応付けて立式すると以下の形として表される．

CRT = W1 + W2*3 + W3 + Reg(条件 1)* (W4 + W5)
+ Reg(条件 2)*W4+ W6
CAR = W1 * W2^3* W3
*** {Reg(条件 1)*(W4 * W5) + !Reg(条件 1)}**
*** {Reg(条件 2)*W4 + !Reg(条件 2)} * W6**

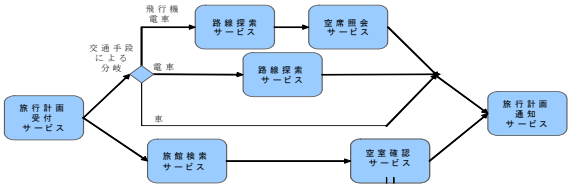
この式に対し，個々の Web サービスの品質を代入することで品質が見積もられる．

CRT = 0.3 + 0.5 * 3 + 0.4 + 1* (0.6 + 0.4) + 0 * 0.4 + 0.3= 3.5(ms)
CAR = 0.998 * 0.997^3 * 0.999 * {1 * (0.995*0.994) + 0}
*** {0 * 0.995 + 1} * 0.992 = 0.969399166**




```
<sequence>
<empty name="旅行計画受け付けサービス">
<empty name="旅館検索サービス">
  <while condition="ループ回数">
    <invoke>
    ・・・
    </invoke>
  </while>
  <while>
    <receive>
    ・・・
      <empty [name=" 空室確認サービス">
    ・・・
    </receive>
  </while>
<empty name="空室確認サービス">
  <switch name="交通手段による分岐">
    <case condition="条件1=電車と飛行機">
      <empty name="路線探索サービス">
    </case>
    <case condition="条件2=電車">
      <empty name="空席照会サービス">
    </case>
    <otherwise>
    </otherwise>
  </switch>
<empty name="旅行計画通知サービス">
```

(a) 旅館／交通逐次型フロー




```
<sequence>
<empty name="旅行計画受け付けサービス">
  <flow>
    <switch name="交通手段による分岐">
      <case condition="条件1=電車と飛行機">
        <empty name="路線探索サービス">
      </case>
      <case condition="条件2=電車">
        <empty name="空席照会サービス">
      </case>
      <otherwise>
      </otherwise>
    </switch>
    <sequence>
      <empty name="旅館検索サービス">
      <while condition="ループ回数">
        <invoke>
        ・・・
        </invoke>
      </while>
      <while>
        <receive>
        ・・・
          <empty [name=" 空室確認サービス">
        ・・・
        </receive>
      </while>
    </sequence>
  </flow>
<empty name="旅行計画通知サービス">
</sequence>
```

(b) 旅館／交通並列型フロー

図.9 旅行計画作成連携サービスの BPMN モデルと BPEL 記述

図.9（b）では，BPEL 記述から論理構造要素ごとに演算式を対応付けて立式すると以下の形として表される．

CRT = W1 + MAX[{Reg(条件 1)*(W4+W5)+Reg(条件 2)* W4}
, (W2+W3)] + W6
CAR = W1 * MIN[[{Reg(条件 1)*(W4+W5)+!Reg(条件 1)}
***{Reg(条件 2)*W4+!Reg(条件 2)}], (W2*W3)] * W6**

この式に対し，個々の Web サービスの品質を代入することで品質が見積もられる．

CRT = 1.6(ms)
CAR = 0.97915552

図 10 (a)，(b)は同じ機能を有するサービスでありながら，連携のワークフローが異なるだけで，品質に差が表れることが定量的に示される．以上のとおり，提案手法を用いることで，BPEL および単体 Web サービスの応答時間と稼動率を与えるだけで，手軽に連携サービスの品質を定量的に見積もることができる．

6. 考察とまとめ

本稿では，連携 Web サービスの応答速度と稼働率を見積もるための手法を提案した．提案手法では，まず，BPEL の記述構造に着目し，連携のワークフローを制御する基本となる論理構造要素ごとに分解して考える．分解・抽出された論理構造要素にはそれぞれ演算子が対応付けられ，演算式の組み合わせから連携のワークフローを数式化する．これにより，導出された数式に連携を構成する個々の Web サービスの応答速度や稼働率を代入することで，連携 Web サービスの応答速度や稼動率を見積もることができる ．

提案手法では，論理構造要素ごとに演算式が対応付けられているため，再帰的な処理により，任意のワークフローに対して一意に数式を対応付けることができる．しかし，応答速度の見積もりに関しては，2.2.1 で示された Web サービスの応答速度の定義より，WS へのアクセスに要される時間や，処理を行うためにプログラムの実行に要される時間がオーバーヘッドになるといった問題が考えられる．そのため，論理構造に対して，論理構造要素に応じたプログラム実行時間や WS へのアクセス時間を考慮に入れた演算式の検討が必要になると考える．

今後の課題として，上記の問題点を考慮した応答速度の演算式をあらためて検討した上で，実際に連携させた Web サービスの応答速度や稼働率を測定し，提案手法から算出された見積もりと比較することで提案手法の適切さを評価していきたい．

謝辞

この研究は，科学技術研究費(若手研究 B 18700062)，21 世紀 COE プログラム「NAIST-IS:ユビキタス統合メディアコンピューティング」の助成を受けて行われている．

文　献

[1] Business Process Modeling Notation Specification, http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf

[2] Jos J.M. Trienekens, Jacques Bouman, Jos Trienekens, Mark Van der Zwan, “Specification of Service Level Agreements,Clarifying Concepts on the Basis of Practical Research”, International Workshop on Software Technology and Engineering Practice(STEP1999), pp.169, August 1999

[3] R.D.van der Mei, H.B. Meeuwissen, “Modeling End-to-end Quality-of-Service for Transaction-Based Service in Multi-Domain Environments”, IEEE International Conference on Web Services(ICWS’06) , pp. 453-462, September 2006

[4] 特定非営利活動法人 UML モデリング推進協議会 BPMN 研究会，“ローレベル BPMN パターン”，6 月 2006 年.

[5] Web Service Business Process Execution Language version 2.0, http://www.oasis-open.org/committees/download.php/22036/wsbpel-specification-draft%20candidate%20CD%20Jan%2025%2007.pdf