OWL-S を用いた Web アプリケーションの生成支援

A Generation Support System for Web Applications Using OWL-S

海津 智宏 † 徳田 雄洋 †

Tomohiro Kaizu Takehiro Tokuda

† 東京工業大学大学院情報理工学研究科計算工学専攻

Dept. of Computer Science, Tokyo Institute of Technology

{kaizu, tokuda}@tt.cs.titech.ac.jp

Web サービスを利用する Web アプリケーションを開発する場合，WSDL ではサービスがどのような条件のもとで呼び出し可能かという情報が記述されないため，開発時には問題に気がつかず，実行時に初めて問題が発見される場合がある．この問題を解決するため，Web 遷移図を用いた Web アプリケーション生成に OWL-S 記述を取り入れる手法を提案する．OWL-S を利用して Web サービスの前提条件や効果を定義することで，Web 遷移図の作成段階で，条件を満足しない遷移を発見し，Web アプリケーションの開発を支援することができる．

1 はじめに

近年，Web アプリケーションからの Web サービスの利用が重要性を増している．Web アプリケーションが Web サービスを利用することの利点として，既存の Web サービスを利用することで Web アプリケーションを効率的に開発できることや，ビジネスロジックの実装とユーザーインターフェースの実装を明確に分離できることが挙げられる．

しかし，このような Web アプリケーションの開発を行う場合，Web サービスを呼び出す順序や渡すべきパラメータを正確に把握している必要があり，開発が容易であるとはいえない．また，Web アプリケーションの開発時には型チェック程度の検査しか行えないため，コンパイルが成功しても実行時にエラーとなってしまう場合がある．

そこで本研究では，Web サービスを利用する Web アプリケーションの開発を支援するため，Web サービスにメタデータを付加し，メタデータに基づいて設計の検査と Web サービスの合成を行う手法を提案する．これにより，意味的不整合を設計段階で発見し，修正することが可能となる．

2 Web 遷移図と OWL-S

まず，本研究で用いる Web 遷移図と OWL-S について述べる．

2.1 Web 遷移図

Web 遷移図 [2] は Web アプリケーションにおけるページとプログラムの遷移を記述した有向グラフである．グラフのノードとして固定 Web ページ，出力 Web ページ，サーバープログラム，データベースの 4 種類を，リンクとしてページ遷移リンクとデータフローリンクの 2 種類を用いる．

Web 遷移図は Web アプリケーションの全体的振る舞いを記述するためのモデルであり，プログラミングの経験が少ない開発者でも容易に理解可能である．Web 遷移図を用いて Web アプリケーションを自動生成するシステムとして，T-Web システムが提案されている [2, 3, 4, 5, 6, 7, 8, 9]．T-Web システムを利用することで，低コストで Web アプリケーションを開発できる．

2.2 OWL-S メタデータ

Web サービスに対するメタデータとして標準規格である WSDL[12] が広く使われている．しかし，WSDL では接続情報やパラメータの型のみを定義し，Web サービスの実行時の振る舞いに関する情報は与えられない．そこで，この問題を解決し，Web サービスを機械的に発見・合成するために，OWL-S[11] という規格が提案されている．

OWL-S は，セマンティック Web 関連規格のひとつである OWL 言語 [13] によって定義された Web サービス用のオントロジであり，セマンティック Web 上で構築された意味体系と Web サービスとを結びつける．

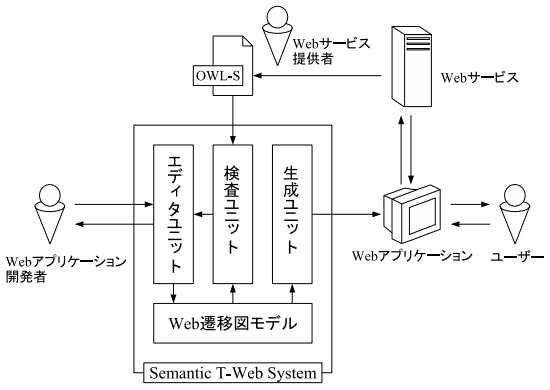

図 1: システム全体図

これは WSDL を置き換えるものではなく，WSDL と合わせて利用することでサービスに意味を付加する．

OWL-S は大きく以下の 3 つの部分から構成される．1 つ目はサービスの概要を表すサービスプロファイル，2 つ目はサービスの動作の詳細を表すプロセスモデル，3 つ目は WSDL とのマッピング情報を表すサービスグラウンディングである．本研究では主に 2 つ目のプロセスモデルの記述を利用する．プロセスモデルでは，単一の操作に対応するアトミックプロセスや複数の操作のフローを定めた複合プロセスを記述でき，それぞれに入力・出力・事前条件・効果を指定できる．

Web アプリケーションではサービス全体の実行途中で適宜ユーザーとの対話が行われる場合が多いため，OWL-S が想定している複合プロセスの振る舞いとは異なる．そこで，本研究では，アトミックプロセスの入力・出力・事前条件・効果のみを用いる．

3 セマンティック T-Web

3.1 ST-Web の概要

既存の Web サービスを利用した Web アプリケーションの開発を支援するためのシステムとして，セマンティック T-Web システム (以下 ST-Web) を提案する．ST-Web は Web 遷移図を用いて Web アプリケーションの振る舞いを記述し，Web アプリケーションを生成するシステムである．従来の T-Web とは異なり，OWL-S を用いて設計段階で Web 遷移図の意味的不整合を検出し，予想される可能な修正方法を提示する．

図 1 に ST-Web システムの全体構成を示す．Web サービス提供者は，サービス本体とともに OWL-S メタデータも提供するものとし，ST-Web システムは利用する OWL-S メタデータをあらかじめ読み込んでおく．Web アプリケーション開発者は，ST-Web システムのエディタ画面上で Web 遷移図を作成する．Web 遷移図が更新されると，システムは自動的に OWL-S を利用した検査を行い，エディタ画面に結果を表示する．Web 遷移図が完成し開発者が生成を指示すると，システムは Web 遷移図にテンプレートを適用して Web アプリケーションを出力する．

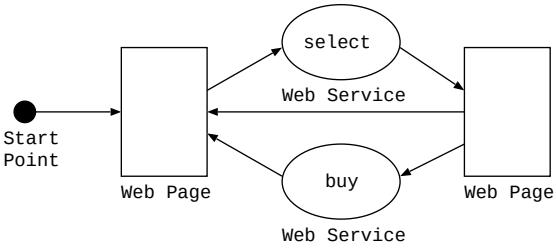

図 2: ST-Web システムにおける Web 遷移図の例

3.2 ST-Web における Web 遷移図

ST-Web システムでは Web サービスを利用した Web アプリケーションの生成に特化するため，利用する Web 遷移図に制限を加える．具体的にはサーバー側のプログラムを Web サービス呼び出しとページ出力処理に限定し，Web 遷移図のプログラムノードを Web サービス呼び出しと対応させる．

ST-Web システムでは，Web 遷移図の要素として以下のノードとリンクを用いる．

- Web サービスノード: Web サービスの呼び出しを表す．パラメータとして呼び出す Web サービスと Web サービスに与える入力を指定する．
- Web ページノード: 静的ページ及び出力ページを同一視し，Web ページノードとして表現する．パラメータとして出力するデータやフォームの情報を指定する．
- 開始点: 他のサイトからのリンクやブックマークなどによる Web 遷移図外からの遷移を，開始点からのリンクとして表現する．パラメータとして，セッション外で成り立つ状態を指定する．
- リンク: HTML のハイパーリンクとフォーム送信時のデータフローリンクを同一視し，1 種類のリンクで表現する．

3.3 検査アルゴリズム

本論文では ST-Web で行う検査項目のうち，サービスの事前条件が成り立っているかどうか，サービスの実行に必要なパラメータが揃っているか，の 2 点について説明する．これらの項目を検査するため，Web アプリケーションの実行中に Web サービスのセッションがどのような状態になるかを Web 遷移図と OWL-S の記述に従って計算し，事前条件やパラメータの情報と比較する．セッションの状態はサービスの実行やフォームへの入力によって変化するので，それぞれのノード間にあるリンクごとに状態を計算することにする．それぞれのリンクが持つ状態に関して設計時に完全な知識を得ることは困難なので，実際にはそのリンクを通過するときに必ず成り立つことがわかる状態だけを考えることとする．

Web 遷移図から計算可能な各リンクの状態は以下の通りである．

- Web サービスノードの出力リンクは，入力リンクの状態にサービスの効果を加えた状態を持つ．
- Web ページノードの出力リンクは，入力リンクの状態にフォームの入力欄の値が既知であるという情報を加えた状態を持つ．これは Web サービスノードにおける効果と同一視できる．
- 開始点の出力リンクはセッション外で成り立つ状態を持つ．

この規則に従って，開始点から順にリンクをたどって状態を更新していくことで Web 遷移図上の各リンクの持つ状態を計算できる．Web 遷移図内に閉路が存在する場合，同一ノードを複数回たどることになるが，その場合には計算済みである入力リンクの共通部分を全体の入力状態として出力リンクの状態を計算し，出力リンクの状態が変化しなくなるまで繰り返す．具体的には次のアルゴリズムで各リンクの状態を計算することができる．

1. 開始点からのリンクの状態を「セッション外の状態」とし，それ以外の全てのリンクの状態を unknown という状態に初期化する
2. 更新すべきノードの集合 S を考え，開始点からのリンク先を S に加える
3. S が空集合となるまで以下を繰り返す
   3.1. S からノード v を 1 つ選ぶ
   3.2. v の unknown でない入力の共通部分を取り出し，効果を追加し矛盾を削除した状態を出力リンクに代入する

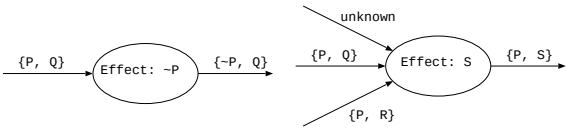

図 3: 出力リンクの状態の計算例

   3.3. 出力リンクの状態が計算前と変わっていれば，リンク先のノードを S に追加する
   3.4. S から v を削除する

ここで，上記 3.2 の矛盾の削除の方法には注意が必要である．推論の結果矛盾が存在することを発見できたとしても，どの条件を削除すればよいのかは一意には決定できない．そのため，以下の場合のみ処理を行う．

- functional なプロパティの値を指定した場合にはそのプロパティの他の値を削除する
- あるインスタンスがクラス C に含まれると指定した場合にはそのインスタンスがクラス C と disjoint なクラスに含まれるという記述を削除する
- インスタンス I_1 と I_2 が同一，もしくは異なると指定した場合にはそれ以前の I_1 と I_2 の同一性の記述を削除する

以上のアルゴリズムによって求めた状態集合を用いて，以下の Web 遷移図中の意味的不整合を検出する．

- 状態が unknown であるリンクがあった場合，そのリンクは開始点から到達不可能である．状態が不明なため他の検査はできないが，開始点から到達可能にするような修正を促す．
- Web サービスノードに関して，入力リンクの状態を調べ，事前条件を満たさないリンクがある場合や必要なパラメータの揃っていないリンクがある場合は，実行時にエラーが発生することが予想されるので，警告を発する．
- Web ページノードに関して，入力リンクの状態を調べ，出力するデータが揃っていないリンクがある場合は警告を発する．

3.4 検査結果からの修正方法の提示

求められた各リンクの状態と警告内容から，可能な修正方法を提示する．

事前条件が満たされていない Web サービスノードが発見された場合，その事前条件を効果として持つ Web サービスを列挙する．例えば，ログインが必要なサービスについて入力リンクが条件を満たさない場合，ログイン中であるという状態を効果に持つサービスを列挙する．ここで列挙されるサービスは通常ログインサービス 1 つのみであり，開発者は提示されたログインサービスを選択・追加するだけで問題点を修復できる．

必要なパラメータの揃っていない Web サービスが発見された場合，そこへ遷移する Web ページに入力欄を追加するように促す．

また，意味的不整合が検出されない場合でも，リンクの状態を開発支援に利用することができる．特定の位置に Web サービスノードを追加する場合，その入力リンクの状態が前提条件を満たす Web サービスを提示すれば，利用可能なサービスを効率的に選択できるようになる．

3.5 OWL-S メタデータの作成方法

Web サービスにメタデータを付加する手法については盛んに研究されているが [1, 10]，現在はまだ記述コストが高い．そこで、ST-Web での利用を念頭に置いた OWL-S メタデータの簡易的な生成法を提案する．

ST-Web での OWL-S の利用はサービスの実行順序の制御が主な目的なので，実行順序の記述から必要なメタデータを生成することを考える．

1. まず，サービスの実行順序を正規表現で記述する．正規表現で用いるアルファベットを各サービスに対応させる．
2. 正規表現を受理するオートマトンを生成する．このオートマトンはサービスを入力として状態遷移を行うものである．
3. 各サービスについて，対応するリンクの遷移元の状態を前提条件，遷移先の状態を効果とすることで OWL-S に対応させる．

例えば，最初にサービス a を 1 回呼び出し，サービス b もしくは c を任意の回数呼び出し，最後にサービス d を 1 回呼び出す，という場合を考えると，この条件は正規表現で `a(b|c)*d` と書くことができる．OWL-S メタデータの作成者はこの正規表現だけを書けばよい．これを受理するオートマトンは a を実行する前の state1，a を実行後で d を実行前の state2，d を実行後の state3 の 3 状態を持つ．生成される OWL-S では，サービス a は事前条件 state1・効果 state2，サービス b およびサービス c は事前条件 state2・効果 state2，サービス d は事前条件 state2・効果 state3 となる．

この OWL-S メタデータを読み込み，セッション外での状態に state1 を指定すれば，サービス a,b,c,d がこの正規表現に従った順序で呼び出されることを検査できる．なお，正規表現に現れないサービスに関しては任意のタイミングで呼び出すことが可能である．

この手法はサービスの意味を明確にするという本来の OWL-S の目的を果たしていないが，ST-Web での利用を目的とする限り有用な手法である．

4 評価

提案手法を用いて ST-Web システムの一部を実装した (図 4)．このシステムを利用することにより，サービスの事前条件をリアルタイムに検査しながら Web 遷移図を作成することができる．

例として，Web サービスとしてユーザー認証のある掲示板アプリケーションを考える．このアプリケーションにはログイン・ログアウトの処理があり，ログイン中のユーザーのみが掲示板に書き込まれた情報を閲覧したり新たに情報を追加したりすることができる．

このアプリケーションは以下の 5 種類の Web サービスによって実装される．

- login: ユーザーのログイン処理を行う.「ログイン中」プロパティが false であることを前提条件とし,「ログイン中」プロパティが true であることを効果とする．
- logout: ユーザーのログアウト処理を行う.「ログイン中」プロパティが true であることを前提条件とし,「ログイン中」プロパティが false であることを効果とする．
- getMessages: 書き込まれた情報を取得する.「ログイン中」プロパティが true であることを前提条件とする．
- addMessage: 新たに情報を追加する.「ログイン中」プロパティが true であることを前提条件とする．
- addUser: 新たにユーザーを追加する．

ST-Web で Web 遷移図を作成し，ログインせずに到達可能な点に logout や getMessages，addMessage

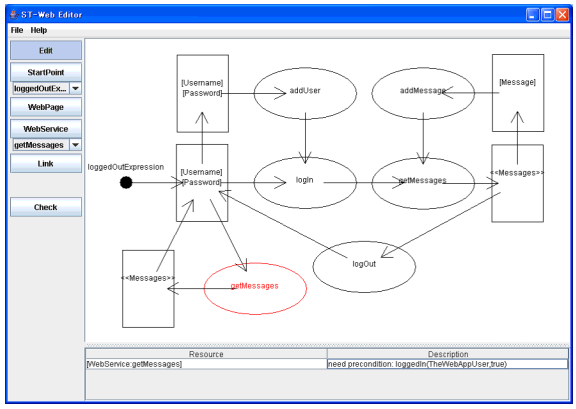

図 4: ST-Web システムの動作画面

を置くと事前条件が成り立たない旨の警告が表示されることを確認した．従来の T-Web システムでは Web サービスでのログイン状態の管理はできなかったが，提案手法によって設計段階で意味的不整合を発見することができる．

5 おわりに

Web サービスを利用する Web アプリケーションの開発を支援するため，OWL-S メタデータを用いて設計の検査と Web サービス合成の支援を行う手法を提案した．提案手法を用いることで，従来は実行時まで発見できなかった意味的不整合を開発時に発見・修正することができるようになり，より効率的に Web アプリケーションを開発することができるようになった．メタデータで記述する情報を増やし，ST-Web システムでの検査・支援項目を増やすことで，さらに Web アプリケーション開発を容易にできると考えられる．

今後の課題として，セッション管理などの Web アプリケーション特有の問題に対処することが考えられる．例えば，一定時間アクセスがなかった場合に自動的にセッション終了処理を行う場合に，任意のページからの終了処理を推論したり，戻りボタンの使用によって Web 遷移図では意図していない遷移が発生したときに，エラーを表示すべきかそのまま作業を継続できるかを自動的に判断することが考えられる．これらの問題も，状態の変化という観点から解決できると考えられる．

参考文献

[1] A.Heß, E.Johnston, N.Kushmerick: ASSAM: A tool for semi-automatically annotating Web Services with semantic metadata. *Proc. of the 3rd International Semantic Web Conference*, 2004.

[2] K.Jamroendararasame, T.Suzuki, T.Tokuda: A Generator of Web-based Transaction Systems Using Web Transition Diagrams. *Proc. 17th Japan Society for Software Science and Technology*, 2000.

[3] K.Jamroendararasame, T.Matsuzaki, T.Suzuki, T.Tokuda: Generation of Secure Web Applications from Web Transition Diagrams. *Proc. of the IASTED International Symposia Applied Informatics*, pp.496-501, 2001.

[4] K.Jamroendararasame, T.Matsuzaki, T.Suzuki, T.Tokuda: Two Generators of Secure Web-Based Transacion Systems. *Proc. of the 11th European-Japanese Conference on Information Modelling and Knowledge Bases*, pp.348-362, 2001.

[5] K.Jamroendararasame, T.Suzuki, T.Tokuda: JSP/Servlet-Based Web Application Generator. *18th Conference Proceedings Japan Society for Software Science and Technology*, 2C.1, 2001.

[6] K.Jamroendararasame, T.Suzuki, T.Tokuda: A Visual Approach to Development of Web Services Providers/Requestors. *Proc. of the 2003 IEEE Symposium on Visual and Multimedia Software Engineering*, pp.251-253, 2003.

[7] M.Taguchi, T.Susuki, T.Tokuda: Generation of Server Page Type Web Applications from Diagrams. *Proc. of the 12th Conference on Information Modelling and Knowledge Bases*, pp.117-130m 2002.

[8] M.Taguchi, T.Suzuki, T.Tokuda: A Visual Approach for Generating Server Page Type Web Applications Based on Template Method. *Proc. of the 2003 IEEE Symposium on Visual and Multimedia Software Engineering*, pp.248-250, 2003.

[9] M.Taguchi, K.Jamroendararasame, K.Asami, T.Tokuda: Comparison of Two Approaches for Automatic Construction of Web Applications: Annotation Approach and Diagram Approach. *Proc. of the 4th International Conference on Web Engineering*, pp.230-243, 2004.

[10] A.Patil, S.Oundhakar, A.Sheth, K.Verma: METEOR-S Web Service Annotation Framework. *Proc. of the 13th International WWW Conference*, 2004.

[11] DAML Services: OWL-S 1.1. `http://www.daml.org/services/owl-s/1.1/overview/`, 2004.

[12] World Wide Web Consortium: Web Services Description Language (WSDL) 1.1. `http://www.w3.org/TR/wsdl`, 2001.

[13] World Wide Web Consortium: OWL Web Ontology Language Semantics and Abstract Syntax `http://www.w3.org/TR/owl-semantics/`, 2004.