0-004

サービス指向集合知のための複合サービス制御フレームワーク

A Composite Service Control Framework for Service-oriented Collective Intelligence

田仲 正弘† 村上 陽平† 林 冬惠† 石田 亨‡†
Masahiro Tanaka Yohei Murakami Donghui Lin Toru Ishida

1. はじめに

サービスコンピューティング技術の広がりに伴い，Web 上で様々なプログラムやデータがサービスとして提供され始めている．これらを集積し，活用するサービス指向集合知を実現するには，サービスの利用に関するサービス提供者のポリシーを充足できる枠組みを整えることによって，より多くのサービス提供者が参加しやすくする必要がある．サービス提供者のポリシーは，データ転送量の制約，ビジネス上の戦略によるユーザの制約や，連携されるサービスの制約など様々である．

そこで，サービス指向集合知の実現するプラットフォームでは，関与する全てのサービス提供者のポリシーを満たしながら実行できるようなサービスを選択して連携させる必要がある．しかし，サービス選択に基づくサービス連携を行う従来の研究では，以下の問題がある．

・ユーザの要件と提供者のポリシーを同時に満たすようなサービスの組み合わせがないことがある．
・複合サービスで利用される各サービスの実行状態に関する制約は，サービスの選択のみでは解決できない．

これらの問題を解決するため，本研究では，複合サービスにおけるサービスの選択・適応・協調を包括的に行うアーキテクチャを提案する．必要な機能要件とサービス提供者のポリシーに基づく制約の両方を満たすサービスが存在しない場合，サービス動作の実行時適応によって機能要件や制約を変更する．これにより，より多くの状況でサービス提供者のポリシーを満たすように複合サービスが実行できる．また，複合サービスのメタレベル制御により，実行状態を常に監視しながら，サービスの実行順序の制御を行う．

以降で，上記に述べた問題の例と，本研究で提案するアーキテクチャ及びそれを実現するためのアルゴリズムについて述べる．

2. サービス連携における制約

本研究が扱う問題について，サービス指向集合知のためのプラットフォームである言語グリッド[4]で利用される辞書連携翻訳複合サービスを例に説明する．この複合サービスでは，形態素解析サービス，機械翻訳サービス，専門用語辞書サービスが連携される．このような複合サービスは通常，インターフェースのみを定めた抽象サービスに基づいて構成され，実際に呼びだされる具象サービスのエンドポイントは実行時に割り当てられる．このような複合サービスの実行に当たっては，以下に示す 3 種類の制約を考慮する必要がある．

1 つ目は，サービスの組み合わせについての制約である．ある辞書サービスの提供者がポリシーとして，ある翻訳サービスと連携させることを禁止していることがある．従って，明示的にユーザから具象サービスを指定されなかった抽象サービスについては，システム側で全てのサービス提供者のポリシーが満たされるように具象サービスを選択する必要がある．

2 つ目は，原子サービスの仕様についての制約である．例えば，翻訳サービスに定められた文字列長の上限を超える入力が与えられるなど，実行時に前段のサービスの出力が後段のサービスの仕様に適合しないことがある． このような場合，入力を分割してから翻訳を実行するなどの適応処理が必要になる．

3 つめの点は，複数のサービスの実行状態に関する制約である．複合サービスの中で，2 つの辞書サービスを並列実行するように定義されているとする．しかし，選択したサービスによっては，サーバへの負荷を避けるため，同じクライアントからの複数の同時アクセスを禁止しているとする．このような場合，複合サービスで並列実行として定義された処理を，直列的に実行されるように制御する必要がある．

3. アーキテクチャ

前章で挙げた 3 種類の制約は相互に関連しており，ある制約を解決させるための処理が，別の制約違反を起こすことがある．そこで，3 種類の制約をそれぞれ解決する 3 つのレイヤが，相互に連携するアーキテクチャを提案する．図 1 に，提案するアーキテクチャの概要を示す．

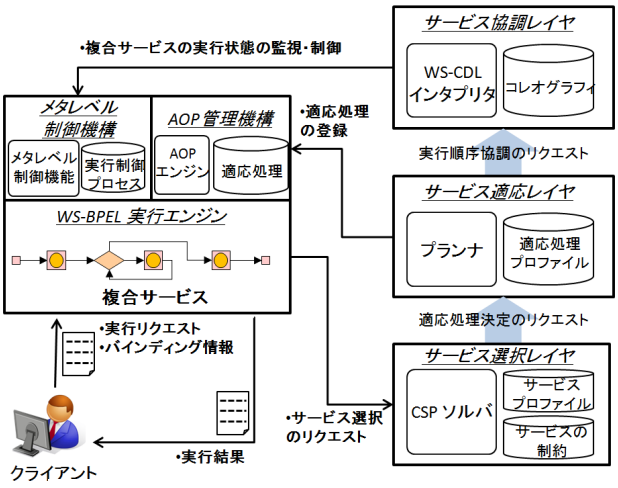

図 1 実行制御アーキテクチャ

このアーキテクチャは以下のように動作する．まず，クライアントが複合サービスの実行リクエストを，サービスのバインディング情報と共に WS-BPEL エンジンに送信する．次に，具象サービスの指定されていなかった抽象サー

† （独）情報通信研究機構 言語グリッドプロジェクト
‡ 京都大学情報学研究科 社会情報学専攻

ビスについての具象サービスの選択，サービス提供者のポリシーに適合しない具象サービスの適応，及び実行順序の制御が図 1 右側の 3 つのレイヤによって行われる．各レイヤの主な機能は以下の通りである．

サービス選択レイヤ　サービスの組み合わせについての制約を満たすように，それぞれの抽象サービスに対応するを具象サービスを選択する．選択は後述の制約充足問題としての定式化に基づいて行われる．制約を満たすサービスの組み合わせが存在しない場合には，上位レイヤでサービスの適応を実行する．

サービス適応レイヤ　複合サービスに含まれる原子サービスの適応を行うことによって，連携される他のサービスの提供者のポリシーを満たすように，サービスの属性を変更する．プランニングによって登録された適応処理を組み合わせた上で，適応の対象となるサービス呼び出しの前後にアスペクト指向プログラミングに基づいて処理を挿入することによって実現する．このレイヤで制約を満たすことができない場合，上位レイヤでサービスの協調を実行する．

サービス協調レイヤ [3]で提案された複合サービスのメタレベル制御機能を用いて，各サービスの実行が WS-CDL で記述されたサービス実行プロトコルに従うよう実行順序を変更する．このレイヤでの制御によって制約が満たすことができない場合，複合サービスの実行は失敗する．

3. 適応対象サービスの選択

サービス選択レイヤでは，制約を満たすサービスの組み合わせを発見するだけでなく，制約を満たす組み合わせがない際に適応を行う，その原因となるサービスを特定する必要がある．そこで，サービス選択を制約充足問題（CSP）として以下のように定式化する．

変数: 複合サービスに含まれる抽象サービスに相当する．

ドメイン: 抽象サービスに対応する具象サービスの集合に相当する．

制約: サービス提供者によってポリシーとして課された制約に相当し．サービスのプロファイルや実行時のサービスの入出力に基づいて定義される述語として表現する．

前章に示したように，全ての制約を満たすサービスが存在しないとき，適応処理が適用される．しかし，プランナを用いた適応処理の発見は，実行のパフォーマンスを低下させる可能性がある．従って，適応処理の対象となるサービスを効率的に特定する必要がある．

そこで本研究では，Open CSP[1]の解法を適用する．[1]では，解が存在しない CSP をドメインの拡張によって解決する場合，ドメインの拡張が必須となる変数は，バックトラックによって解の探索を行ったとき，探索木で最も深い位置にあったものであることが示されている．そこで，その変数に相当するサービスの適応が必要となる．また，いずれの具象サービスに適応処理を行うかを決定するため，Partial CSP[2]のアイデアを適用する．すなわち，選択時にもっとも制約違反度が小さくなるような具象サービスに対して適応を試みる．

これを行うアルゴリズムを図 2 に示す．このアルゴリズムは，深さ優先探索でサービスの組み合わせを探索する．もっとも深いノードのインデックスは k として記録される（6 行目）．また探索の過程で，最小の制約違反度とそれを与えるサービスの組み合わせはそれぞれ N 及び

```
function: searchCombination(X, D, C)
入力: X: 変数 ({X1, ..., Xn}), D: ドメイン ({D1, ..., Dn}), C: 制約

 1: i ← 1, k ← 1, N ← ∞
 2: while(i > 0)
 3:   if Di の全値をチェック済み then reset xi, i ← i-1
 4:   else
 5:     xi ← Di の次の値
 6:     if ({x1, ..., xi}) がC を満たす then k ← max(k, i+1) end if
 7:     i ← i+1
 8:     if i > n
 9:       if ViolationCount({x1, ..., xn}) = 0
10:         return {x1, ..., xn}
11:       end if
12:       if N > ViolationCount({x1, ..., xn})
13:         N ← ViolationCount({x1, ..., xn})
14:         currentSelection ← {x1, ..., xn}
15:       end if
16:       i ← i+1
17:     end if
18:   end if
19: end while
20: xk'← xk に適応を適用して得られたサービス
21: Dk ← Dk ∪ {xk'}
22: 変数の順序を変更し Xk を X1 とする
23: return searchCombination(X, D, C);
```

図 2 適応対象サービスの選択アルゴリズム

currentSelection として記録される（13, 14 行目）．サービス x_k がサービス適応レイヤに渡される．適応処理によって性質が変化した x_k' は，新たな変数としてドメインに追加される（20, 21 行目）．

4. おわりに

本研究では，複合サービスにおけるサービスの選択・適応・協調を包括的に行うアーキテクチャを提案した．サービス提供者のポリシー全てを満たすサービスの組み合わせが存在しない場合には，適応処理によってポリシーを満たすことを試みる．また，適応処理の対象となるサービスを効率的に特定するアルゴリズムを示した．

謝辞

本研究は，総務省戦略的情報通信研究開発推進制度の補助を受けた．

参考文献

[1] B. Faltings and S. Macho-Gonzalez, “Open constraint programming,” Artificial Intelligence, vol. 161, no. 1-2, pp. 181–208, 2005.

[2] E. C. Freuder and R. J. Wallace, “Partial constraint satisfaction,” Artificial Intelligence, vol. 58, no. 1-3, pp. 21–70, 1992.

[3] M. Tanaka, T. Ishida, Y. Murakami, and S. Morimoto, “Service supervision: Coordinating web services in open environment,” *IEEE International Conference on Web Services (ICWS-09)*, 2009, pp. 238–245.

[4] T. Ishida, “Language Grid: An infrastructure for intercultural collaboration,” *IEEE/IPSJ Symposium on Applications and the Internet (SAINT-06)*, 2006, pp. 96–100.