

仕様記述言語を用いたAPI自動変換方式の一検討

6J-4

阿川雄資 清原正幸

NTT情報通信網研究所

1. はじめに

異なるAPI(Application Programming Interface)をもつアプリケーション間の流通性を向上させるために、ネットワーク管理(CMIS)を対象として、同一の機能をもつ複数のAPIを相互に変換する方式を提案する。また、具体的な変換方式について検討し、実際にシンタックスレベルのAPI変換ルーチンを作成したので報告する。

2. APIの相互変換

ソフトウェアの大規模化に伴い、効率的にAPを開発するために、通信やデータベースなどの機能を提供するミドルソフトが広く用いられている。上位のAPはミドルソフトの定義するAPIを通じてその機能を利用する。APIの標準化に関しては、様々な分野において検討が進められているが、標準が確立するにはまだかなりの時間が必要である。実際に、OSIネットワーク管理のための共通サービス規定(CMIS)を提供するミドルソフト(CMISE)においても、同一の機能を提供するために異なった複数のAPIが定義され利用されている。しかし、APIを相互に変換する技術を確立することにより、現状の様々なAPIを統一的に扱うことができるようになり、AP-ミドルソフト間の組合せが増え、ソフトウェアの流通性が大巾に向上する。

我々は、同一の機能を提供するミドルソフトでは、基本的にAP-ミドルソフト間で受渡しされる情報のセマンティクスが同一であることに着目し(図1)、

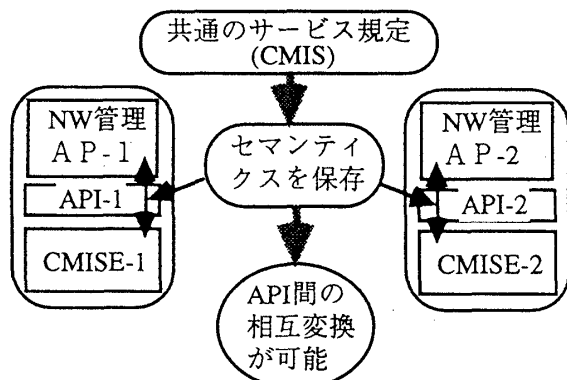


図1. 複数APIの相互変換

A Study on a Automatic Translation Method for API's
using a Specification Language
Yuji AGAWA and Masayuki KIYOHARA
NTT Network Information Systems Laboratories

まずCMISEのAPIであるCTRON-APIとXMP-APIをターゲットとして、各API間の相互変換を効率的に行う方式とそのルーチン化を検討する。

3. API変換方式

複数のAPIを変換する方式としては、

(1)相互に変換可能であること

(2)効率的な変換ができること

が望ましい。そこで図2のようなAPI変換モデルを考える。このモデルは以下のような考え方に基づいている。

○API変換ルーチンをAP、ミドルソフトから独立させる

→AP、ミドルソフトの内部構成に関与することなくAPIの変換が可能

○変換ルーチンの内部構成として、各APIを一旦その基となっているCMISサービス定義の項目(以後、セマンティック定義と呼ぶ)に変換する

→複数のAPI間での効率的な相互変換が可能

○変換ルーチンを生成する「生成系」を用いる

→複数の変換ルーチンを効率的に作成可能

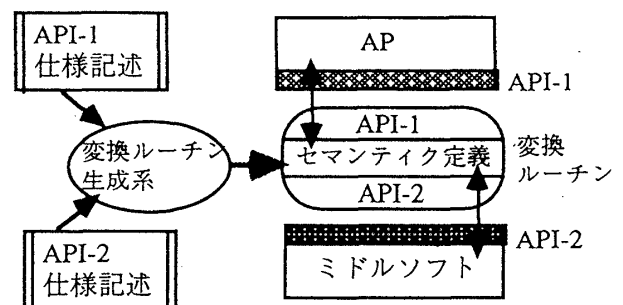


図2. API変換モデル

4. 変換ルーチンの実装

4.1 実装モデル

APIを変換する際、単純に変換できるものとそうでないものがある。前者の例としては、

○関数名の対応付け

○パラメタ形式の変換

などの、シンタックスレベルの変換があり、後者の例としては、

○プログラミングインタフェースの違い

(関数コール、メッセージ、イベントハンドラ等)

○メモリ等の資源管理

○関数内部における制御移行
などが挙げられる。

本稿では、まずAPI変換の全体像をつかむという観点から、上述のシンタックスレベルの変換を中心に具体的に延べる。図2の変換モデルを実際にライブラリとして実装するための実装モデルを図3に示す。

4.2 ユーザ定義

変換を行うために必要な項目を以下のように分類した。

(1)API定義項目

a.共通定義

- 関数名、サービス特定パラメタ、データ型など

b.CMIPパラメタ関連定義

- ASN.1定義からくるもの
→基本的に対応がとれる

c.CMIPパラメタ以外定義

- アソシエーションやコンテキストなど
→対応がとれるもの、とれないものがある

(2)変換規則定義

- セマンティック定義への変換規則
- セマンティック定義からの変換規則

これらのうち(1)に関しては、図3中の「ユーザのAPI定義」で、また(2)に関しては、図3中の「ユーザの変換規則定義」で表現する。CMISでは、そのサービス仕様を仕様記述言語 (ASN.1) で定義している。このASN.1定義を拡張することで、上記の項目を定義する方式も考えられるが、今回はユーザの利便性を考え、(1)の「API定義項目」のみを定義するフォーマットを新たに作成した。

4.3 ルーチン生成系

本研究の最終的な目標は、API変換の自動化であり、そのために図3中の「ルーチン生成系」を自動作成することを考えている。しかし節4.1でも述べたように、シンタックスレベルの変換であれば、ルー

チン生成系を直接プログラム言語で記述することは比較的容易である。そこで、実際にCMISのサービス・パラメタの1つである、「Class」を変換するルーチン生成系をC言語で作成した。また、図3「ユーザのAPI定義」部の解析にはyaccを用いた。

5. APIの静的・動的側面

4節では、2つのCMISE-APIを例に、シンタックスレベルの変換を行う方式について具体的に述べた。しかし、完全なAPI変換にはこの他にAPIの実行中の動作を考える必要である。このために、APIを「静的」「動的」、という2つの面から考えてみる。

「静的」とはAPIの定義から導出される情報で、本稿の中ではシンタックスレベルと呼ぶものである。

「静的」なAPIとして、今回はパラメタ (Class) の例を示したが、一般にCMISEのパラメタでは (識別名の指定など) 比較的簡単なデータ構造が複雑にネストして表現される場合が多い。しかし、これらの機能を実現する「ルーチン生成系」を一般の言語で記述するのは困難である。これに関しては、各データ構造を「状態」、ネストしていく過程を「遷移」として捉え、変換規則を定式化することによって、仕様記述言語等により効率的に変換を行うことが可能と考えられる。

「動的」とは、APやミドルソフトを実行することによって動的に変化する情報や状態、例えばパラメタの値やポインタ、データ構造、各関数間の相互関係、関数内部における制御移行などである。これらに関しては、APIの振舞と情報のやり取りをモデル化することにより、「静的」と同様に仕様記述言語等を用いることにより、「ルーチン生成系」を効率的に作成可能と考えられる。

6. おわりに

2つのCMISE-APIを例に、シンタックスレベルの変換を行う方式について具体的に述べた。今後は、今回実装したモデルの評価、および効率的な変換方法について検討を進めていく。

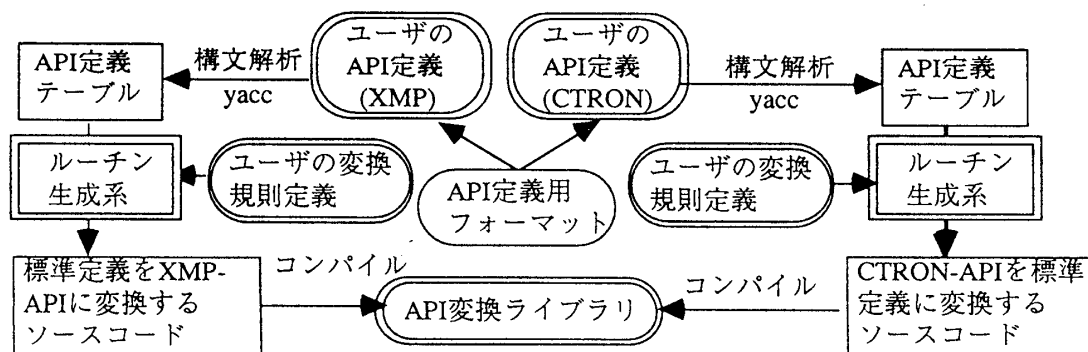


図3. API変換実装モデル