B-011

クラウド上でのデバイス・コンフィギュレーション（ADHoCC）に適用するＡＰＩの検討

Consideration on API for ADHoCC (Architecture for Devices on Holistic Cloud Configuration)

大橋　正†
Tadashi Ohashi†

1．まえがき

システムの如何なるアルゴリズムでもハードウェア部品オブジェクト及びソフトウェア部品オブジェクトを組み合わせることで実現できると考える．例えば既存デバイスの機能に関して，利用者の要求に適合した機能変更に伴うアルゴリズムの変更を部品オブジェクトの入れ替えで実現できると考える．所与デバイスのシステム構成をクラウド上で展開しオブジェクト部品の相互は LD (Linked Data) [1]により結合し，従来の部品オブジェクトを入れ替えることの可能な API (Application Program Interface)を提供することで，この要求を満たすことができる．その結果クラウド上に機能変更のデバイスのコンフィギュレーションを一時的に展開し，動作確認後にクラウドからデバイスへダウンロードして実稼働可能とならしめる UdAD[2]のシステム創生又は再創生に関しての API 案を検討したので報告する．

2．現状の課題とその解決

デバイスのハードウェア及びソフトウェア部品の結合関係にセマンティクスを用いて徴表できるかの技術的，経済的課題が散在しており，ましては膨大な部品を LD で結合をする効果についての疑問も認めえざるを得ない．この課題は技術の発展とともに解決すると考え，既存問題としてクラウドで処理する際に API が具備されていないという卑近の問題がある．その解決のために以下の API 案を提唱し，実現する必要があると考える．

(1) 上位機能(Strong)及び下位機能(Weak)版数を持つデバイスの検索を可能とする API の提供
(2) 稼働要件を満たしているかを判別する API の提供
(3) FPGA データなどのオブジェクトを入手する API の提供等

3．システムの全体構成とＡＰＩ

本論を展開する事前準備として以下の前提を設定しておく．所与のデバイスはハードウェアとソフトウェア品で構成されており，ハードウェア部品はソフトウェアを保持して内部で実行できるコンポーネント例えば FPGA(Field Programmable Gate Array)と入出力インターフェイスを組み込める IO ポートから構成される．システムのアルゴリズムの実現はこれらのハードウェア部品とその上に実装されたソフトウェア部品により実現する．

†アイリクト　http://www.ilict.jp

また FPGA によって実現される各種演算命令は FPGA に提供されるゲート制御プログラムで処理される．以下にクラウド側とデバイス側の双方を大別して，各々の構成を述べる．

3．1　クラウド側の構成

図 1 に示したクラウド側の構成は本論文の主要な部位を示している．デバイスの構成をクラウドで実現したクラスの集合で実現している．つまりハードウェア部品やソフトウェア部品はクラスのインスタンスである．クラウド上では所与のデバイス側の実質的なメモリー空間でのインスタンス即ちオブジェクトの写像の原型 ADHoCC[2]として種々の API を介して実現することになる．

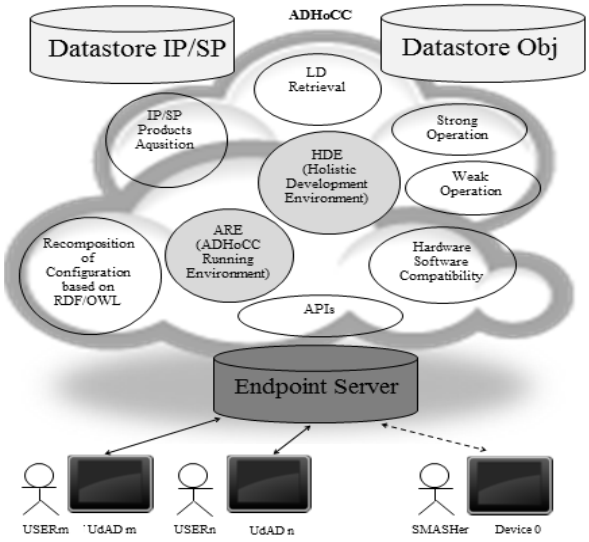

図 1 クラウド側構成図

3．2　デバイス側の構成

ADHoCC で形成されたデバイスのブジェクトはハードウェア部品即ちオブジェクトを格納できる機能とその実行を確認する機能を具備する．デバイス側のシステム構成情報は ADHoCC の原型となっており，この原型をクラウド上で再創生させ，その結果へ新たなオブジェクトとしてダウンロードさせることを意図している．しかしデバイス側での API は現時点で検討段階にある．

3．3　構成部品のセマンテック

デバイスのハードウェア部品及び ソフトウェア部品から構成される部品間の結合関係を RDF[3]又は OWL[4]等を用いてセマンテックスで具体的に表現する．詳細は引用文献を参照．各部品は雛形となるクラスと実際の部品と

なるインスタンスで実現させられる．ハードウェア部品は FPGA で実現される無形ソフトウェア部品と実際に有形ハードウェア部品からなる．これらが RDF/OWL でインスタンスとしてアルゴリズの実現に向けて API で容易に操作できる仕組みを具備する必要がある．

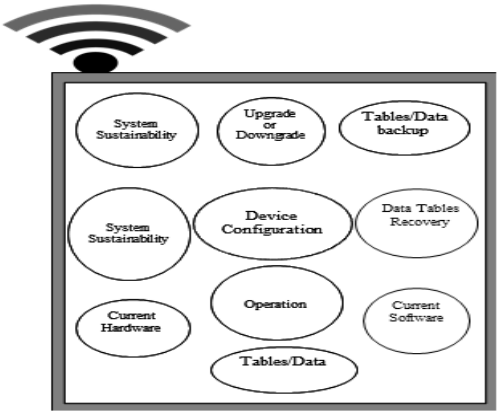

図 2　UdAD デバイス構成図

3. 4　クラウド側ＡＰＩ

ADHoCC で実現するため必要な API を以下に列挙する．しかし API を実現するソフトウェア言語に限定されない．以下に幾つかのクラスと各々のクラスで具備されているメソッドについて案を述べる．

(1) Component クラス
① makeComp メソッド
ハードウェア及びソフトウェア構成創生機能
②remakeComp メソッド
ハードウェア及びソフトウェア構成再創生機能

(2) SearchProducts クラス[5]
①esarchStrongOperation メソッド
所与部品の上位世代部品検索(Strong)機能
②searchWeakOperation メソッド
所与部品の下位世代部品検索(Weak)機能

(3)Compatibility クラス
①hardwareCompatiblity メソッド
Strong/Weak 操作後のハードウェア適合検査機能
②softwareCompatibility メソッド
Strong/Weak 操作後のソフトウェア適合検査機能

(4) Billing クラス
①setUser メソッド
ユーザ及び所属などを設定する機能
②setSecurityLank メソッド
所与ユーザの課金及びセキュリティランク設定機能
③setBilling メソッド
所与部品の課金及びセキュリティー設定機能．

4　動作概略

4. 1　クラウド側の動作概略

クラウド側ではデバイスからの Strong 又は Weak 要求を受け付けるとデバイスへハードウェア及びソフトウェアの構成情報を RDF や OWL 等の形式で徴表する．デバイスの構成情報を入手すると，対象となるプロダクトに対する Strong を創生できる．又は更に Weak 要望の構成情報を再創生する．SPARQL[5]は検索対象となるプロダクトの上位または下位の機能を持ったプロダクトの構成情報のメタ情報を検索する．上位または下位のプロダクトを検索すると，プロダクトの動作条件のコンパティビリティを確認する．もし確証結果が真であれば Strong 又は Weak 操作を継続する．コンパティビリティが偽であれば Strong 又は Weak 操作を中断する．動作条件のコンパティビリティは論理的判断を優先して確証を得る．動作条件にはハードウェアの環境も入る．新たなハードウェアの追加が必要な場合は FPGA のファームで代替させる．

4. 2　デバイス側の動作概略

デバイス側のユーザは，現デバイスはハードウェア及びソフトウェア構成情報を RDF 又は OWL 等生成する．ユーザからの対象となるプロダクトの Strong/Weak の要望を上位系へ要望する．この要望を受理すると現状のデバイスの構成情報を要求し，この情報を獲得する．Strong 及び Weak の機能を実現できることを判断し，その処理可能通知をデバイスへ伝達する．もし不可能であると判断すると，その不可能通知を他デバイスへ発信して一連の処理を中断して終了する．

5. まとめと課題

（１）現段階での API はクラウド上だけに限定した HDE(Holistic Development Environment)と ARE(ADHoCC 1 Running Environment)であるが，いずれかの環境でも有効な仕様案をまとめることができた．今後はデバイス側での API を検討する予定である．

（２）他 Endpoint サーバとの連携

実際にはクラウド上の多種の Endpoint Server を横断的に SPARQL で検索する必要がある．現時点での API はあくまでも同一内のサーバでのみで検索の対象に限定している．マルチ Endpoint サーバへのブロードキャスティングの導入が余儀なくされる．

（３）検索速度

システムのコンフィギュレーションのトリガーは UdAD のユーザが発することで検索が開始され ADHoCC 上で実構築するとなると検索時間をかなり要する．検索時間速度の問題は当該システムに限らずあらゆるシステムの共通テーマとして解決が必要となる．

（４）課金とセキュリティー

全てのウェアがフリーになるわけでなく特定品に対しては IP として課金も必要な場合がある．更にはセキュリティーの課題もある．今回は論じることができなかったが今後の課題としたい．

参考文献

[1]LD https://www.w3.org/wiki/LinkedData
[2]大橋 正，UdAD(User-driven Architecture for Devices)のクラウド・アーキテクチャの検討，第 79 回情報処理学会全国大会，2017
[3]RDF https://www.w3.org/RDF/
[4]OWL https://www.w3.org/2001/sw/wiki/OWL
[5]SPARQL https://www.w3.org/TR/rdf-sparql-query/