クラウド管理のアーキテクチャー

Open Cloud Standards Incubator のホワイト・ペーパー

バージョン:1.0.0

位置付け:DMTF 参考情報

公開日: 2010-06-18

文書番号:DSP-IS0102

概要

このホワイト・ペーパーは、DMTF のクラウド・インキュベーターから受領した 2 点の Phase 2 成果物の 1 つであり、クラウド・サービス・プロバイダーとクラウド・サービス利用者とのインターフェースに関連する参照アーキテクチャーについて説明したものである。DMTF のクラウド・インキュベーターの目的は、企業とクラウド・コンピューティングの相互運用可能な管理を統合するアーキテクチャー上の意味体系（セマンティック）を定義することにある。

謝辞

Open Cloud Standards Incubator に貢献いただいた次の方々に感謝するものである。

- Kane, Valerie – Advanced Micro Devices
- Vancil, Paul – Advanced Micro Devices
- Kowalski, Vincent – BMC Software
- Lipton, Paul – CA, Inc.
- Moscovich, Efraim – CA, Inc.
- Waschke, Marvin – CA, Inc.
- Armstrong, Joseph – Cisco
- Sankar, Krishna – Cisco
- Serr, Frederick – Cisco
- Siefer, Mike – Cisco
- Tomsu, Peter – Cisco
- Wheeler, Jeff – Cisco
- Pardikar, Shishir – Citrix Systems Inc.
- Sirjani, Abolfazl – Citrix Systems Inc.
- Landau, Richard – Dell
- Ericson, George – EMC
- Linn, John – EMC
- Durand, Jacques – Fujitsu
- Song, Zhexuan – Fujitsu
- Aguiar, Glaucimar – Hewlett-Packard Company
- Cook, Nigel – Hewlett-Packard Company
- Maciel, Fred – Hitachi, Ltd.
- Baskey, Michael – IBM
- Johnson, Mark – IBM
- Pendarakis, Dimitrios – IBM
- Cox, Billy – Intel Corporation
- Li, Hong – Intel Corporation
- Raghu, Rekha – Intel Corporation

- Parchem, John – Microsoft Corporation
- Carter, Steve – Novell
- Jose, Jaimon – Novell
- Carlson, Mark – Oracle
- Vambenepe, William – Oracle
- Yu, Jack – Oracle
- Owens, Ken – Savvis
- Qualls, Richard – Sun Microsystems, Inc.
- Norbeck, Don – SunGard Availability Services
- Ronco, Enrico – Telecom Italia
- Galán Márquez, Fermín – Telefónica
- Bumpus, Winston – VMware Inc.
- Lamers, Lawrence – VMware Inc.
- Matthews, Jeanna – VMware Inc.

目次

図

表

1 適用範囲

本書は、標準化されたインターフェースおよびデータ・フォーマットに基づくクラウドの管理方法について記述した、関連する 2 つの文書の 1 つである。本書では全体的なアーキテクチャーを重点的に取り上げ、もう 1 つの文書では相互作用およびデータ・フォーマットを扱っている。

アーキテクチャーに関する本書の適用範囲を以下に示す。

- **クラウド管理の参照アーキテクチャー**。この参照アーキテクチャーは、DMTF のホワイト・ペーパー「相互運用可能なクラウド」（DSP-IS0101-J）で紹介されている。この文書では、この概念をさらに調査し、説明している。
- **要件**。これらの要件は、設計するインターフェース全般を対象としたものであり、プロトコル、リソース・モデル、セキュリティー・メカニズムなどに対する要件がある。
- **ポリシーと制約の役割**。クラウド実装の内部詳細ではなく、クラウド管理のインターフェースに重点を置いた場合、ポリシーと制約を使用してクラウドに必要な機能を定義することが、有用な抽象化となる。これらは、クラウド・サービス・プロバイダーは「何」を提供すべきか（それを「どのように」提供すべきかではなく）を示すものとして解釈できる。
- **相互作用のパターン**。クラウド利用者とクラウド・プロバイダーとの相互作用で見られるこれらのパターンは、さまざまな操作とデータ・ペイロードを伴う多数のユースケースにわたり、そのユースケースに応じて繰り返し発生する。
- **クラウドのエコシステムの例**。理解を深める意味で、設計上の考慮事項を交え、クラウド・サービス・プロバイダーによる管理インターフェースの設計例を取り上げる。このセクションは規範を示すものではなく、あくまでも参考であることに注意を要する。

本書と対になるホワイト・ペーパーである「クラウド管理のユースケースおよび相互作用」（DSP-IS0103）では、クラウドの管理につき、本書では取り上げていない以下のような側面について説明している。

- クラウド・サービスのライフサイクル全体にわたる管理のユースケース
- ユースケースの実装に向けたクラウド利用者、開発者、プロバイダー間の相互作用シーケンス
- 相互作用シーケンスで送受信されるデータ・アーティファクト

クラウドの次の面については、本書の適用範囲外である。

- ここのアーキテクチャーでは管理機能のみを扱い、クラウドが提供する業務サービスやクラウドでデプロイされているアプリケーションを一般的な業務アプリケーションで使用する方法については取り上げていない。
- ここのアーキテクチャーでは、クラウドに管理機能を構築する方法は取り上げていない。重点を置いている対象は、クラウドとの管理インターフェースである。これらのインターフェースの背後でのサービスの実装は、どのような形態であれ、プロバイダーに一任されている。

2 参考資料

Cloud Security Alliance, Security Guidance for Critical Areas of Focus in Cloud Computing 2.1, http://www.cloudsecurityalliance.org/csaguide.pdf

DMTF DSP0243, Open Virtualization Format Specification 1.1, http://www.dmtf.org/standards/published_documents/DSP0243_1.1.pdf

DMTF DSP1001, Management Profile Specification Usage Guide 1.0,
http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf

DMTF DSP-IS0101, Interoperable Clouds, A White Paper from the Open Cloud Standards Incubator 1.0,
http://www.dmtf.org/standards/published_documents/DSP-IS0101_1.0.pdf

DMTF DSP-IS0101-J, 相互運用クラウド, Open Cloud Standards Incubator のホワイトペーパー 1,0,

http://dmtf.org/sites/default/files/DSP-IS0101_100_J1-2-1.pdf

DMTF DSP-IS0103, Use Cases and Interactions for Managing Clouds, A White Paper from the Open Cloud Standards Incubator 1.0, http://www.dmtf.org/standards/published_documents/DSP-IS0103_1.0.pdf

DMTF DSP-IS0103-J, クラウド管理のユースケースおよび相互作用, Open Cloud Standards Incubator のホワイトペーパー 1,0, http://www.dmtf.org/sites/default/files/DSP-IS0103_1.0.0_rev1_ja_clean.pdf

ITIL® V3 Glossary, Glossary of Terms, Definitions, and Acronyms, v3.1.24, 11 May 2007,
http://www.best-management-practice.com/gempdf/ITIL_Glossary_V3_1_24.pdf

National Institute of Standards and Technology, The NIST Definition of Cloud Computing,
http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc

OASIS, ebXML Messaging Services Version 3.0,
http://www.oasis-open.org/committees/download.php/24618/ebms_core-3.0-spec-cs-02.pdf

OASIS, Web Services Make Connection (WS-MakeConnection) Version 1.1, http://docs.oasis-open.org/ws-rx/wsmc/200702/wsmc-1.1-spec-os.html

Storage Networking Industry Association, Cloud Data Management Interface Version 1.0g,
http://www.snia.org/tech_activities/publicreview/Cloud_Data_Management_Interface_Draft1.0.pdf

3 用語と定義

本書では、その目的に沿って以下の用語と定義が適用される。

3.1
クライアント
DMTF 標準を基本としたプロトコル・ソフトウェア・スタックとともに独自のクラウド・ソフトウェア・スタックを実行する。

3.2
クラウド・サービス
一般に公開されたサービス、または企業内で使用されるプライベート・サービス。

3.3
クラウド・サービス利用者
- 利用サービスに対する業務/財務支出を承認する。
- 利用するサービス・インスタンスのアカウントを維持する。
- サービス・インスタンスを要求する。また、サービス・インスタンスへの変更を要求する（通常、利用者側経営者の代理として実施）。
- サービス・ユーザーがサービスにアクセスできるようにする。

図 2 を参照のこと。

3.4
クラウド・サービス開発者
サービス・テンプレートの設計、実装、および保守を行う（図 2 を参照のこと）。

3.5
クラウド・サービス・プロバイダー
1 人以上の内部利用者または外部利用者にクラウド・サービスを提供する組織（図 2 を参照のこと）。

3.6
構成管理データベース
CMDB
情報システムを構成するすべてのコンポーネントの承認済み構成である情報を収めたリポジトリーであって、IT Infrastructure Library®のコンテキストで使用するものをいう。

3.7
データ・アーティファクト
この文書では、インフラストラクチャーを使用してプロバイダー・インターフェース全体で受け渡される制御要素および状態要素を表す。

3.8
デプロイ
予約または準備された環境にサービス・インスタンスを作成するプロセス。

注意：稼動している既存のサービスに対するソフトウェアや構成のデプロイを指す場合に、より適切な用語である。

3.9
インフラストラクチャー
本書では、クラウド・サービス・プロバイダーとクラウド・サービス利用者との間で制御要素と状態要素を取り交わすための手段を指す。

この手段で取り交わす要素をデータ・アーティファクトという。

3.10
インフラストラクチャー・アズ・ア・サービス
IaaS
クラウド利用者に提供される機能は、CPU、ストレージ、ネットワーク、および他の基本的なコンピューティング・リソースのプロビジョニング機能である。クラウド利用者はこれらの機能を使用し、オペレーティング・システムやアプリケーションなどの任意のソフトウェアをデプロイして実行できる。

利用者は基礎を成すクラウド・インフラストラクチャーの管理と制御は行わないが、オペレーティング・システム、ストレージ、およびデプロイされたアプリケーションを制御する。場合によっては、特定のネットワーク構成要素（ホスト・ファイアウォールなど）の限定的な制御も行う。（出典：NIST）

3.11
通知
注意が必要な状態が存在することをクラウド・サービス・プロバイダーがクラウド・サービス利用者に示すためのシグナル。

通知はポーリングまたは非同期送信できる。

3.12
プロファイル
管理ドメインの CIM モデルおよびそれに関連する動作を定義する仕様（サーバーの仮想化など）。（出典: DSP1001）

CIM モデルは、CIM クラス、CIM アソシエーション、CIM インジケーション、CIM メソッド、および CIM プロパティで構成される。管理ドメインは、一連の関連する管理タスクである。

3.13
プロバイダー・インターフェース
クラウド・サービス利用者が、契約したサービスへのアクセスおよび契約したサービスの監視に使用するインターフェース。

このインターフェースでは、SLO の交渉と測定、サービスへのアクセス、サービスの監視、および請求処理を扱う。また、クラウド・サービス開発者がクラウド・サービス・プロバイダーとの相互作用を通じて、サービス・カタログに追加できるサービス・テンプレートを作成するときにも、このインターフェースが使用される。サービス・テンプレートを収めたサービス商品はプロバイダーが構成する。これにより、クラウド・サービス利用者は、その用途に合わせてサービス商品を選択できる。図 2 を参照のこと。

3.14
プロビジョニング
サービス商品のインスタンスを選択、予約、または作成するプロセス。

サービス商品はサービス・プロバイダーのサービス・カタログから選択され、サービス・インスタンスにプロビジョニングされる。

またプロビジョニングは、契約を履行するために、契約に定めた特定の要求に基づいて、利用可能なプールからサービス・リソース群を選択または予約し、それらを組み合わせて構成するプロセスでもある。

たとえば、テンプレートからサーバー・インスタンスを作成し、CPU、メモリー、ストレージ、およびネットワーク・リソースを割り当て、利用者の契約要件を充足できるようにそのサービス・インスタンスを構成（パッチの適用、セキュリティーの調整、ファイアウォールの構成などを行う）できる。

3.15
セキュリティー・マネージャー
プロバイダー・インターフェース全体の操作に関連する認証情報と認証プロセスの管理責任者。

クラウドのセキュリティー要件には、ユーザー認証、ID およびアクセスの管理、データ保護、マルチテナント型サービスにおけるリソース隔離、コンプライアンスのための監視と監査、インシデントへの対応、ユーザーと顧客のプライバシー、セキュリティー構成要素の基本的な可搬性と相互運用性などがある。

3.16
サービス・カタログ
サービス・プロバイダーが提供するクラウド・サービスに関する情報のデータベース。

サービス・カタログには、サービスの説明、サービスのタイプ、コスト、サポートされている SLA、サービスの閲覧や使用が可能な人物など、サービスに関するさまざまな情報が記録される。より一般的には、サービスのライフサイクル全体を通した情報がサービス・カタログに収録される。サービス・カタログでは、サービス・テンプレート（開発者が作成）、サービス商品（プロバイダーが作成）、およびデプロイされるサービス・インスタンスを扱う。

3.17
サービス契約
クラウド・サービス利用者によるサービスの使用条件について規定した、クラウド・サービス・プロバイダーとクラウド・サービス利用者間の取り決め。

3.18

サービス・エンティティー

操作のリストを提供し、それらの操作を満足する識別可能な論理要素を指す。

たとえば、OS スタックを持つサーバー（物理サーバーおよび仮想サーバーのいずれでも可能）は、その特定の OS で定義された規定の操作をサポートするサービス・エンティティーである。また、複数のサービス・エンティティーおよびサービス要素がグループとしてこれらの規定の操作を満足するのであれば、それらのエンティティーと要素の複合体を 1 つのサービス・エンティティーとすることができる。

3.19

サービス・インスタンス

インスタンス化されたサービス要求。

3.20

サービス商品

サービスレベル・アグリーメント、制約、コスト、請求情報、およびサービス・テンプレートに記述されたサービスを利用者に提供するために必要なその他のデータをサービス・テンプレート上で組み合わせたもの。

3.21

サービス要求

利用者からの、サービス商品のインスタンス化要求。サービス要求はサービス契約を必要とする。サービス契約は、サービス要求に先だって作成されることもあれば、サービス要求と同時に作成されることもある。

3.22

サービス・テンプレート

サービス・カタログに収録されていて、クラウド・サービス・プロバイダーでのプロビジョニングが可能な項目（マシン・イメージ、接続定義、ストレージなど）の集まり（図 3 を参照のこと）。

3.23

サービスレベル・アグリーメント

SLA

クラウド・サービス・プロバイダーのコンテキストでは、クラウド・サービス・プロバイダーとクラウド・サービス利用者間で合意された、法的拘束力のある契約。

SLA では、サービスに関する合意事項およびサービスの可用性、パフォーマンス、および請求処理の責務を扱う。SLA は、関与するすべてのクラウド・サービス・プロバイダーへの波及が可能なように構成する必要がある。クラウド・サービス利用者はクラウド・サービス・プロバイダーとサービス契約を結ぶが、そのクラウド・サービス・プロバイダーはブローカーまたはフェデレーターである場合もあれば、そのどちらでもない場合もある。この SLA は、ブローカーまたはフェデレーターが別のクラウド・サービス・プロバイダーとプログラム的にサービス契約を結ぶ場合の基盤として使用される。

3.24

サービスレベル目標

SLO

SLA の独自で独立した測定可能な側面。

SLA のすべての当事者は、一連の SLO およびその測定結果、相関性、レポートが、SLA にある独自の規定に対するコンプライアンスまたは非コンプライアンスを表現するものであることに同意しなければならない。相互に関連する 1 つ以上の SLO の結果は一義的に決まるものでなければならず、SLO の結果の検証または複数の SLO の結果に見られる相関性の検証を利害関係者が実施できるようにする監査ログによってサポートされている必要がある。

3.25
サービス・マネージャー
サービス・インスタンスおよびサービス・トポロジーの管理責任者。

サービス・マネージャーは、仮想マシンのインスタンスおよびサービスを作成するための設備を監理者に提供する。またサービス・マネージャーは、サービスの作成、監視、制御、および報告のメカニズムを提供する。

3.26
強力な認証
一般的に多く使用されているユーザーID とパスワードによる認証に加えて情報を要求する認証。ユーザーID とパスワードのほかに、生体的手段（指紋）、証明書（X.509）、スマートカード、追加の質問（たとえば母親の旧姓）などを使用することで、認証が高品質であることを示す。

3.27
仮想イメージ
仮想イメージとは、目的のワークロードの実行環境を構成するうえで必要なメタデータとともに、そのワークロードの実行に必要なすべてのコードを網羅して、ワークロードをカプセル化した要素をいう（多くの場合は、オープン仮想化フォーマットを使用したパッケージの一部である）。

3.28
ワークロード可搬性
クラウド・サービス利用者に提供する機能の 1 つであって、サービス・パッケージを作成した後、それを大幅に変更することなく、さまざまなクラウド・サービス・プロバイダーでプロビジョニングできるようにするもの。

4 記号と略語

4.1
API
Application Programming Interface（アプリケーション・プログラミング・インターフェース）

4.2
CMDB
Configuration Management Database（構成管理データベース）

4.3
CMIWG
Cloud Management Interface Working Group（クラウド管理インターフェース作業部会）

4.4
CMMWG
Cloud Management Model Working Group（クラウド管理モデル作業部会）

4.5
CSP
Cloud Service Provider（クラウド・サービス・プロバイダー）

4.6
CSPI
Cloud Service Provider Interface（クラウド・サービス・プロバイダー・インターフェース）

4.7
DoS
Denial of Service（サービス妨害）

4.8
DPI
Deep Packet Inspection（ディープ・パケット・インスペクション）

4.9
HTTP
Hypertext Transfer Protocol（ハイパーテキスト転送プロトコル）

4.10
IaaS
Infrastructure as a Service（インフラストラクチャー･アズ・ア・サービス）

4.11
IDS
Intrusion-Dtection System（侵入検知システム）

4.12
MEP
Message Exchange Pattern（メッセージ交換パターン）

4.13
MOF
Managed Object Format（管理オブジェクト・フォーマット）

4.14
NIC
Network Interface Card（ネットワーク・インターフェース・カード）

4.15
NIST
National Institute of Standards and Technology（米国標準技術局）

4.16
OVF
Open Virtualization Format（オープン仮想化フォーマット）

4.17
PaaS
Platform as a Service（プラットフォーム･アズ・ア・サービス）

4.18
QoS
Quaty of Service（サービス品質）

4.19
RBAC
Role-based Access Control（ロールベース・アクセス制御）

4.20
RDF
Resource Description Framework（リソース・ディスクリプション・フレームワーク）

4.21
REST
Representational State Transfer

4.22
RPC
Remote Procedure Call（リモート・プロシージャー・コール）

4.23
SIP
Semantic Interaction Pattern（セマンティック相互作用パターン）

4.24
SLA
Service-Level Agreement（サービスレベル・アグリーメント）

4.25
SLO
Service-Level Objective（サービスレベル目標）

4.26
SNIA
Storage Networking Industry Association（ストレージ・ネットワーキング・インダストリー・アソシエーション）

4.27
SOAP
Simple Object Access Protocol（簡易オブジェクト・アクセス・プロトコル）

4.28
SSL
Secure Sockets Layer（セキュア・ソケット・レイヤー）

4.29
TCP
Transmission Control Protocol（伝送制御プロトコル）

4.30
TLS
Transport Layer Security（トランスポート層セキュリティー）

4.31
UDP
User Datagram Protocol（ユーザー・データグラム・プロトコル）

4.32
URI
Uniform Resource Identifier（統一資源識別子）

5 はじめに

本書は、米国標準技術局(NIST)による作業(「NIST Definition of Cloud Computing」バージョン 15)で得られたクラウド・コンピューティングの定義に基づいている。NIST によるクラウド・コンピューティングの定義は、必ずしも総合的ではないものの、「最小限の管理作業またはサービス・プロバイダーの相互作用によって迅速にプロビジョニングおよび解放できる、構成可能なコンピューティング・リソース(ネットワーク、サーバー、ストレージ、アプリケーション、サービスなど)で構成した共有プールに対し、利便性の高いオンデマンドのネットワーク・アクセスを実現するモデル」というものである。

NIST は次の 4 つのクラウドデプロイメント・モデルを定義している。

- パブリック・クラウド(広く一般に利用可能であるか、大規模な業界グループが利用可能なクラウド・インフラストラクチャー)
- プライベート・クラウド(特定の組織専用に運営されるクラウド・インフラストラクチャー)
- コミュニティー・クラウド(複数の組織で共有されるクラウド・インフラストラクチャー)
- ハイブリッド・クラウド(複数のクラウドを結合したクラウド・インフラストラクチャー)

Open Cloud Standards Incubator によって現在検討されている環境には、これらすべてのデプロイメント・モデルが含まれる。このインキュベーターはインフラストラクチャー・アズ・ア・サービス(IaaS)の管理面に重点を置いているが、その特徴の中には PaaS などの他のサービス・スタックに共通で適用できるものもある。このようなものとして、サービスレベル・アグリーメント(SLA)、サービスレベル目標(SLO)、サービス品質(QoS)、作業負荷可搬性、自動プロビジョニング、課金、請求処理などがある。

クラウド利用者が利用できる基本的な IaaS 機能はクラウド・サービスである。サービスの例として、セキュリティーおよびパフォーマンス上の指定された制約を充足するコンピューティング・システム、ストレージ容量、ネットワークなどがある。クラウド・サービス利用者の例として、企業のデータセンター、中小企業、他のクラウドがある。

クラウド・コンピューティングには、既存および新たな多数の標準が関連する。これらの中には、セキュリティー関連の標準のように一般に分散コンピューティング環境に適用されるものもある。また、現在、クラウド実装の重要な構成要素の 1 つと見なされる仮想化技術に直接適用される標準もある(仮想化などの各種技術が実現する動的インフラストラクチャーは、クラウドの動的なオンデマンド特性と適切に調和する)。標準の例として、SLA の管理とコンプライアンス、連携 ID、認証、クラウドの相互運用性と可搬性などがある。

図 1 は、Open Cloud Standards Incubator の適用範囲、およびその管理標準の拡張による利点を示したものである。

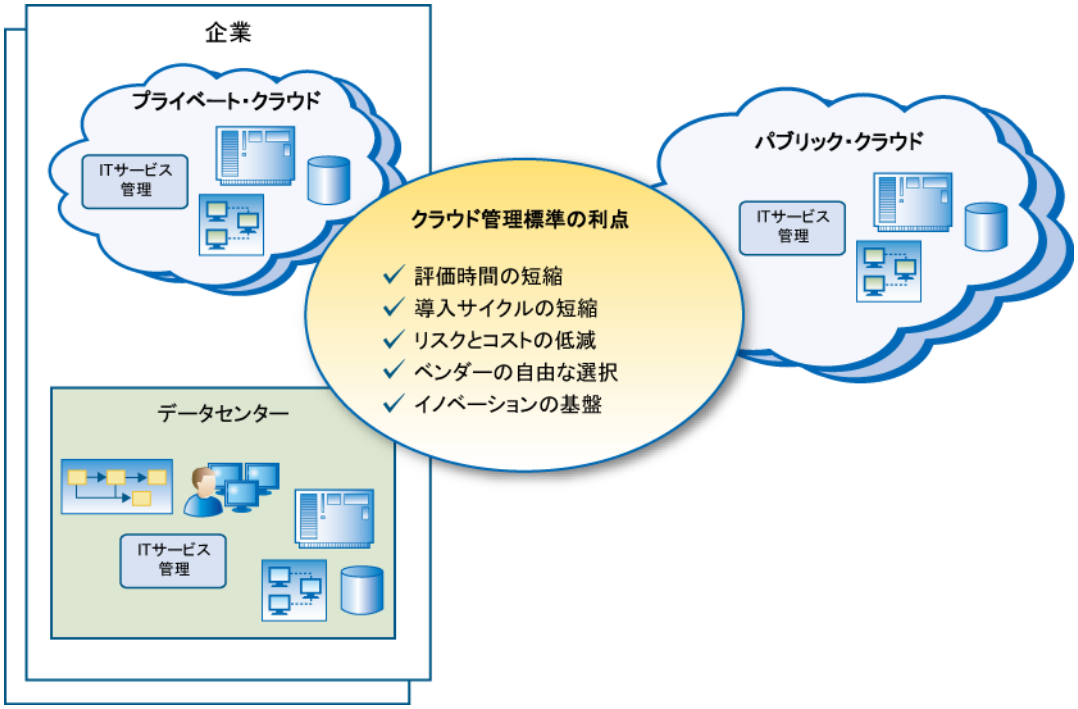


図 1 - DMTF Open Cloud Standards Incubator の適用範囲と利点

Open Cloud Standards Incubator は、クラウド・サービスのライフサイクルについて次の側面に取り組んでいる。

- テンプレートでのクラウド・サービスの記述
- クラウドへのクラウド・サービスのデプロイ
- 利用者へのサービスの提供
- 利用者による提供物利用契約の取り交わし
- プロバイダーによるサービス・インスタンスの運営と管理
- サービス提供物の廃止

既存の標準（またはその拡張）を推奨ソリューションに統合することが現実的な場合は、そのような統合が実施される。標準化領域の例として、リソース管理プロトコル、データ・アーティファクト、パッケージ化フォーマット、識別プロトコル、鍵管理プロトコル、監査フォーマット、コンプライアンス・フォーマット、セキュリティー・メカニズムなど、相互運用性を実現するための領域がある。

6 参照アーキテクチャーの概要

本書で取り上げているライフサイクルの事例では、クラウド利用者がクラウド・サービス・プロバイダーとの間で確立する必要のある機能インターフェースの例を引用している。このセクションでは、アクター、インターフェース、データ・アーティファクト、プロファイルなどの主要要素を記述するクラウド・サービス参照アーキテクチャー（図 2）について説明し、さらにこれらの要素間の相互関係を示す。このセクションでは、このアーキテクチャーの概要を説明し、セクション 9 では、このアーキテクチャーの拡張について詳しく説明する。以下に示す機能面の説明は必須なものではないが、クラウド・サービス・プロバイダーがアーキテクチャー面で用意すべき要素を判断するための検討作業で効果的である。

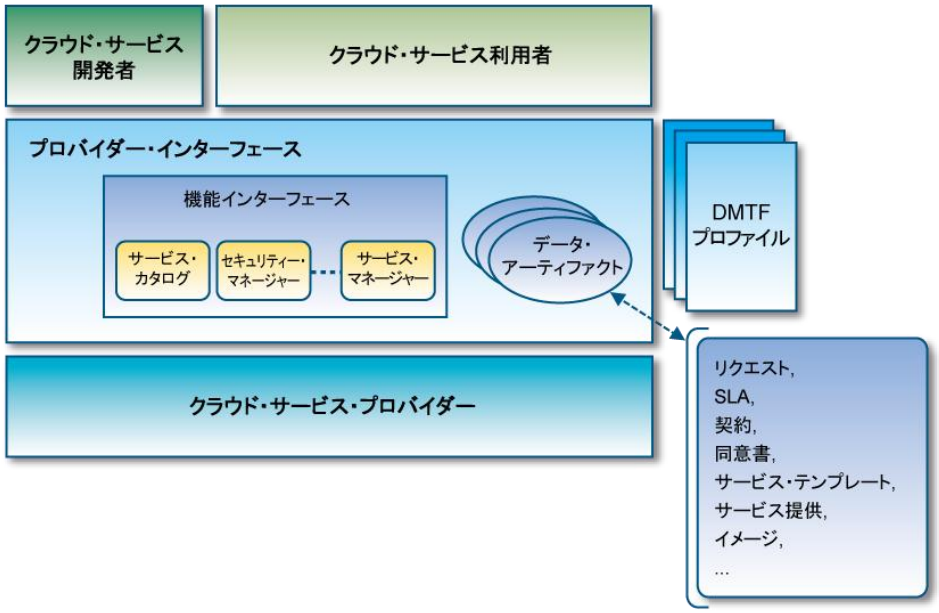


図 2 - クラウド・サービス参照アーキテクチャー

6.1 サービスのライフサイクルとユースケース

図 3 は、一般的なクラウド・サービスのライフサイクルの 6 つの状態と、それぞれの状態との関連性が高いユースケースを示したものである。

- **テンプレート**:開発者は、サービスの内容とサービスとのインターフェースを記述したテンプレートにサービスを定義する。
- **サービス商品**:プロバイダーは、制約、コスト、およびポリシーをテンプレートに適用して、利用者からの要求に応えるサービス商品を作成する。
- **契約**:利用者とプロバイダーは、コスト、SLA、SLO、具体的な構成オプションなどの合意内容を規定したサービス契約を取り交わす。
- **サービスのプロビジョニング**:プロバイダーが、利用者との契約に従って、サービス・インスタンスをデプロイ(または変更)する。
- **稼動時保守**:プロバイダーが、デプロイされたサービスとそのすべてのリソースを管理する。この管理には、リソースの監視や、重要な状況に関する利用者への通知などがある。
- **サービス終了**:プロバイダーは、サービス・インスタンスを停止し、他のサービスをサポートするためにリソースを回収して再デプロイするなどの措置を講じる。

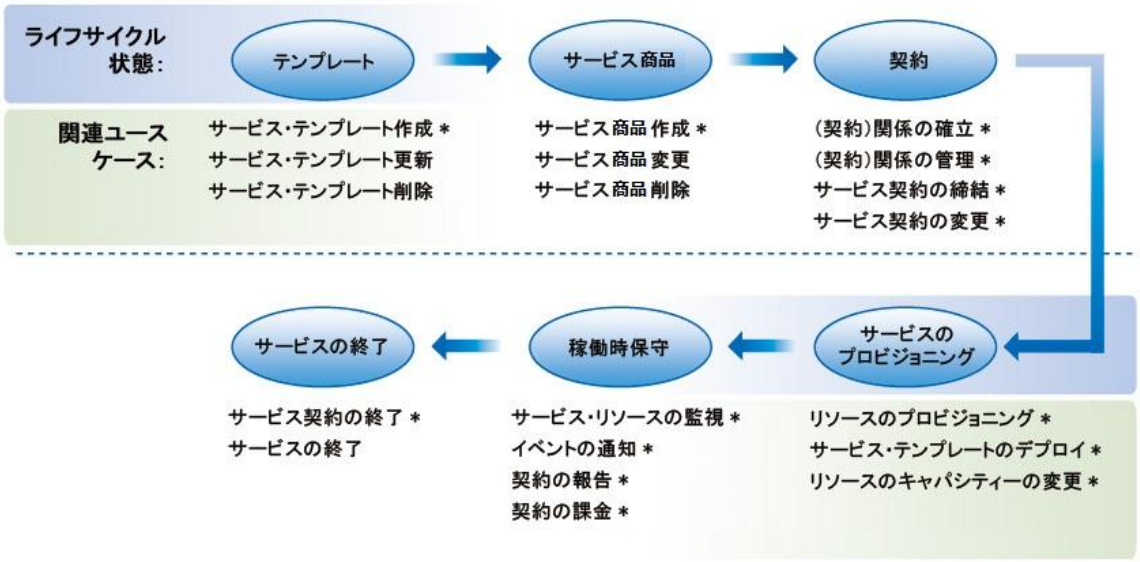


図 3 - クラウド・サービスのライフサイクルの状態とユースケース

図 3 にアスタリスクで示したユースケースについては、ホワイト・ペーパー「クラウド管理のユースケースおよび相互作用」(DSP-IS0103)に詳しい説明がある。

6.2 リソース・モデル

プログラム化されたどのような API であっても、基礎を成すリソース・モデルを明示的または暗黙的に備えている。IT 管理ドメインでは、事実上、これまでリソース・モデルを明示的なものとして、モデル要素の操作に使用するプロトコルとは明確に分離して扱ってきた。このようなプロトコルから独立したモデルの一例として DMTF の CIM モデルがあり、このモデルに対していくつかのアクセス・プロトコルが定義されて、さまざまな相互作用パターンに対応している。クラウド・コンピューティングのような大規模なドメインの場合、リソース・モデルは 1 つではなく、関連する複数のモデルが連続していることが普通である。これらのモデル間の境界は、各モデルの再利用可能性と相補性の検討に基づいて確立される。用意されているクラウド・リソースの種類に関係なく適用できるエンティティーもあれば、クラウド・リソースの種類を具体的に規定しているエンティティーもある。前者の例として、利用者、プロバイダー、サービス、ポリシーがあり、後者の例として、ホスト、ディスク・イメージがある。リソースの種類に固有のモデル・エンティティーは、提供と同時に利用される可能性に基づいてグループ化される。たとえば、ホストとボリュームは、クラウド・リソースの同じドメインに属することが普通であると考えられる(これは、一般的に IaaS、つまりインフラストラクチャー・アズ・ア・サービスと呼ばれる)。一方、アプリケーション・データ・ストレージ・リソース(SQL エンジンやその他の構造化データ・ストア)は、多くの場合、これとは別のリソース・ドメインに属するものとしてグループ化される。設計上の判断を要する重要な事項として、モデル・エンティティーのグループ化のほかに、モデルの表現形態としてのメタモデル(MOF や RDF など)の選択がある。

メタモデルとモデルを決定した後は、管理インターフェースからリソース・モデルにアクセスできるようにする方法を設計面から決定する必要がある。これらの決定は、対象とするプロトコルがどの程度モデル中心であるかに応じて異なる。モデル中心であるかどうかということは、プロトコルの粒度がリソース・モデルの粒度と正確に対応し

547 ているかどうか（リソースが個々にアドレス指定可能であり、プロトコルの論理的なエンドポイントがモデル・エンテ
548 ィティーに対応していること）、またはインターフェースでの各操作がそれぞれ異なる粒度のレベルで発生する中
549 で、モデルを使用してそれらの操作を記述および文書化しているかどうかということである。アクセス可能性や粒
550 度に関するこれらの決定は、データの取得、データの設定（直接アクセスが許可されている場合）、および操作の
551 開始に適用される。このほかの決定事項として相互作用パターンやセキュリティー要件などがあるが、これらにつ
552 いては本書の別の部分で取り上げる。

553 ### 6.2.1 サービス商品モデル

554 さまざまな種類のクラウド・リソースを管理するには、プロバイダーおよび利用者のリソース・モデルが存在する必
555 要がある。このリソース・モデルは、商品が提供される方法および利用される方法を記述する。このモデルは、用
556 意されたクラウド・リソースの具体的な種類から抽象化できる。サービス商品モデルは、サービス利用者とサービ
557 ス・プロバイダーがそれぞれ使用するエンティティーを定義する。サービス・テンプレートを使用して、プロバイダー
558 が提供できるものを一般的な形態で記述し、個々のプロバイダーがサービス・テンプレートを具体的な商品として
559 展開する。サービス・テンプレートおよびサービス商品の問合せと取得には、カタログなどのメカニズムを使用す
560 る。利用者は、サービス商品のサービス・インスタンスを要求できる。リソース・モデル要素は、要求のライフサイ
561 クルを定義する。サービスの実際の提供は、IaaS モデルにあるエンティティーのようなドメイン固有のモデル・エ
562 ンティティーによってモデル化される（6.2.3 を参照のこと）。

563 ### 6.2.2 ID モデル

564 クラウド・コンピューティングは新たなユースケースと脅威をもたらすが、それ自体では新しい ID モデルを必要と
565 しない。企業が有する既存の ID モデルは適切であるが、本書で後ほど説明するように統合が必要である。変更
566 の対象となる可能性があるものは、維持されている監査の量およびユーザー定義（個々のユーザーおよび組織）
567 の精度である。一方、クラウド・サービス・プロバイダーの業務推進要素としてのマルチテナント・モデルの導入に
568 は、より多くの検討を要する。複数のクラウド・サービス・プロバイダー利用者の ID と属性の情報を混合して扱う
569 場合は、意図しない情報の開示を防止するためにデータのモデル化に細心の注意が必要である。

570 ### 6.2.3 IaaS モデル

571 通常、IaaS ドメインの定義では、サーバー（仮想サーバーまたは物理サーバー）、ストレージ、ネットワーク接続、
572 およびネットワーク・インターフェースに公開される IP アドレス（パブリックまたはプライベート）の各種類のリソー
573 スを扱うようにする。さらに、IaaS 層のリソース・モデルでは、インフラストラクチャーの追加機能となるリソースを
574 扱うこともできる。このような機能として、ネットワーク機能（ロード・バランシングなど）、テンプレート（デプロイ向
575 け）、スナップショットなどがある。データ・ストレージ要素（ボリュームなど）は専用のサービス品質要素（重複除去、
576 暗号化、バックアップなど）を有するが、それらは、ここでの取り扱い範囲外となる別のリソース・モデル（データお
577 よびストレージ向け）で取得することが普通である。

578 ## 6.3 相互作用パターン

579 さまざまなクラウド管理のユースケースを分析することで、それぞれのユースケースを相互作用パターンの組み
580 合わせに分解できることがわかっている。相互作用パターンはメッセージのシーケンスで構成される。どのような
581 相互作用パターンでも、具体的なメッセージはユースケースごとに異なるが、アーキテクチャーのレベルでは各メ
582 ッセージの特性に類似性があり、それは特にプロトコルとセキュリティーを考慮する際に顕著である。

583 図 4 は、相互作用パターンを 4 つの大分類でグループ化したものである。

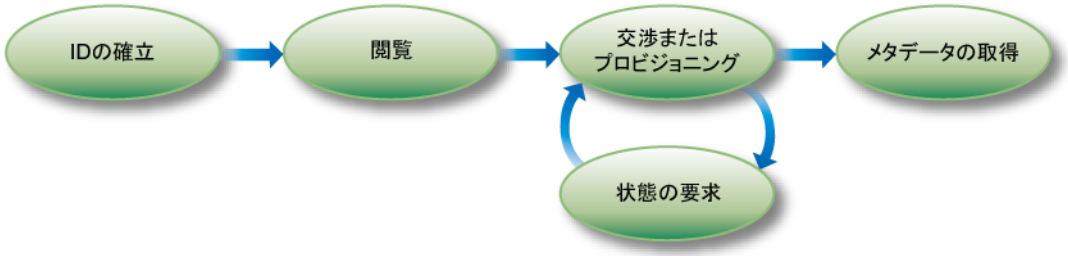
584 • **ID**：クラウド・サービス・プロバイダーと相互作用する人物やエンティティーは、それぞれの ID を確立し
585 て、セッション・トークンなどの適切な認証情報を受け取る。クラウド・サービス・プロバイダーと信頼関
586 係にある外部 ID プロバイダーから ID トークンを入手することもあり得る。これらの認証情報や ID トー
587 クンで認証された接続から操作とデータにアクセスできるようになる。

- **監理**:この相互作用パターンは、クラウド・サービス・プロバイダーとの相互作用に必要となるカタログ、ユーザー、およびその他の監理メタデータ情報を記述するデータを扱う。たとえば、監理者やオペレーターは、用意されている商品のリストを閲覧していずれかの商品を選択し、特定の目的に合わせた設定とするためにその商品のメタデータを更新して、その商品を構成するサービスのインスタンスへのアクセス方法に関する詳細を入手できる。
- **デプロイおよび更新**:この相互作用パターン(実際には、交渉およびリソースのプロビジョニングという 2 つの種類がある)は、サービスとリソースを選択してサービスに組み込む際に使用する。このパターンで扱うものとして、リソースの量と種類に関して必要なあらゆる交渉、サービスをサポートするインフラストラクチャーを含めたサービスをプロビジョニングする操作、長時間実行の操作となるものを示す状態の追跡などがある。
- **定常状態**:この相互作用パターンは、サービスとリソースのプロビジョニングが完了して使用状態になった後で使用する。このパターンで扱うものとして、レポート要求のようなクライアントが開始した要求、利用者が関心を持つ状況や修復措置を必要とする状況に関するプロバイダーからの通知などがある。

図 4 - 相互作用パターン

図 5 は、複数の相互作用パターンを組み合わせてユースケースとする例である。サービスのプロビジョニングは次の手順に従う。

1) アクセスするエンティティーの ID を確立する。
2) 用意されている商品を閲覧する。
3) サービス(リソースのサポートなど)をプロビジョニングする要求を提出し、実行する。受け入れ可能な要求として利用者とプロバイダーが合意するために、交渉を必要とすることもある。
4) 長時間を要する可能性のあるプロビジョニング・プロセスの状態を、プロセス中に要求できる。
5) 利用者は、プロビジョニングされたサービスに関するメタデータを入手する。このメタデータとして、サービスのエンドポイントとリソースにアクセスするためのネットワーク・アドレスなどがある。

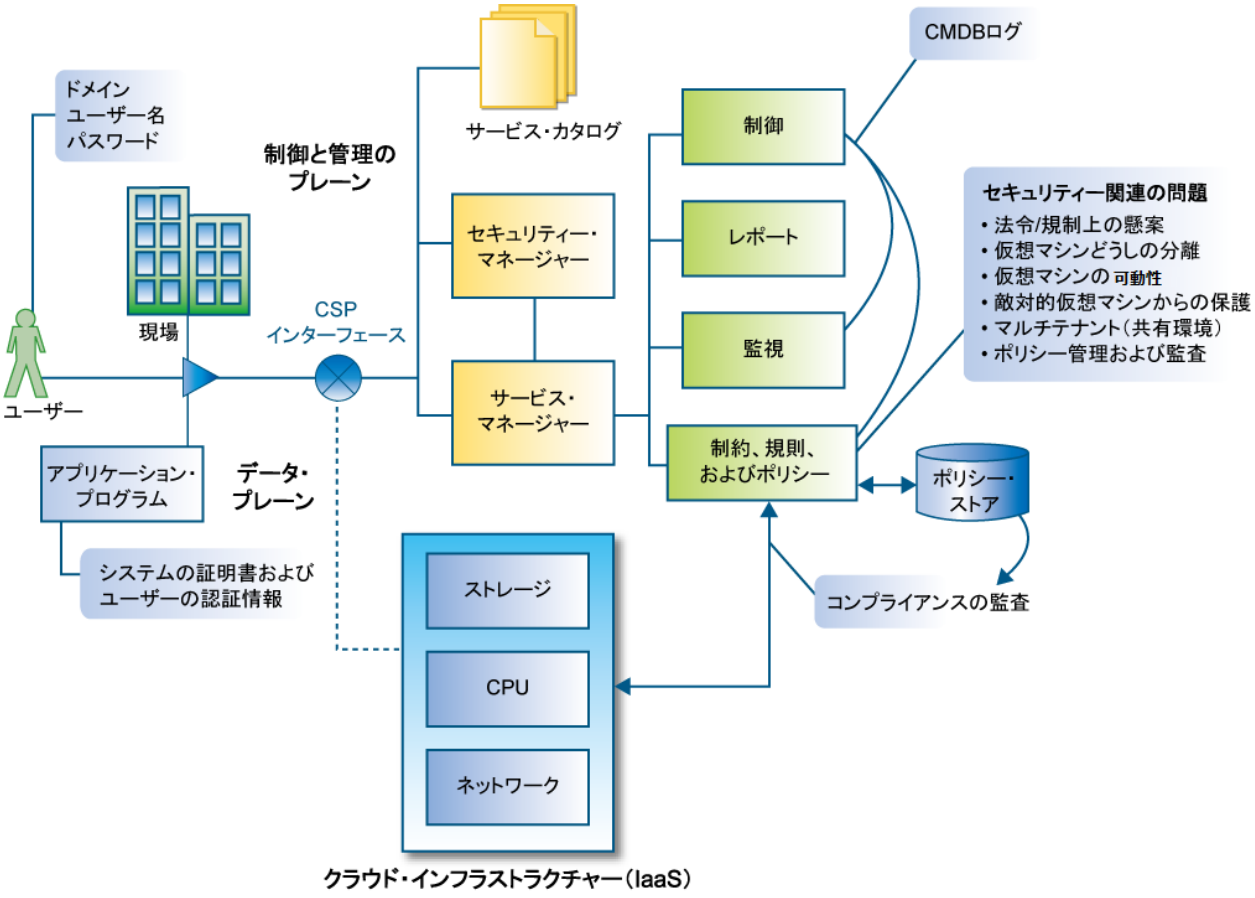


612

613 **図 5 - サービスをプロビジョニングするユースケースで使用する相互作用パターンの例**

614 6.4 セキュリティー・アーキテクチャー

615 一般的には、制御プレーンおよび管理プレーンでインターフェース、サービス、およびオブジェクトに適切なセキュ
616 リティーの基本要素を適用する。ここでは管理に重点を置いていることから、データ・プレーンのセキュリティー(た
617 とえば、ユーザーと IaaS プラットフォームで実行しているアプリケーションの間で稼動時に交わされる相互作用の
618 セキュリティー)は、プロバイダー・インターフェースの対象範囲外である。図 6 は、セキュリティーのコンテキスト、
619 クラウド・サービス・プロバイダー・インターフェースを通じた情報の流れ、およびセキュリティーで保護されている
620 オブジェクトを示したものである。



621

622 **図 6 - セキュリティーのコンテキスト**

クラウド・サービス・プロバイダー（CSP）インターフェースは、セキュリティー・マネージャー、サービス・マネージャー、サービス・カタログなどの論理エンドポイントへのアクセスを提供する。これらのエンドポイントはさまざまなサービスを提供して、サービス・エンティティー（仮想マシン、ボリューム、ネットワーク、複合アプリケーションなど）との相互作用、監査レポートの取得、およびインフラストラクチャーの履行と維持に必要なその他のアクティビティーのホスト実行を実現する。

サービス・マネージャーで管理するオブジェクトは、大きく分けて制御オブジェクト、監視オブジェクト、およびレポート・オブジェクトであり、これらへのアクセスはロールベースのアクセス制御（RBAC）で制御できる。

制約オブジェクト、規則オブジェクト、およびポリシー・オブジェクトはクラウド・インフラストラクチャーで使用され、これらのポリシー関連要素の内容をプロバイダー・インターフェースの機能で管理する。プロバイダー・インターフェースから見るとクラウド・インフラストラクチャー（IaaS）は「ブラック・ボックス」であり、ポリシーをどのように実装するかはクラウド・サービス・プロバイダーに委ねられている。ただし、さまざまな制約、規則、およびポリシーが実装されているという事実は、監査イベントで検証される。

CSP インターフェースと相互作用するアクターには、人間であるユーザーおよびアプリケーション・プログラム（管理、自動プロビジョニング、請求、監査などを実行するアプリケーション）の 2 種類がある。人間であるユーザーは、ポータル・インターフェースを通じて相互作用することもあり、その場合は Web ブラウザーを使用することが普通である。ポータル・インターフェースは、クラウド・サービス・プロバイダー・インターフェースを使用して開発する。いずれのアクターも、CSP インターフェースでセキュリティー・マネージャーによって認証されるか、セキュリティー・マネージャーに ID トークンを提示する。

人間であるユーザーは従来から認証情報としてユーザー名とパスワードを使用しているが、より強力なメカニズム（ID 連携およびアサートのプロビジョニング）を使用する必要がある。一般的に、アプリケーション・プログラムでは証明書を使用する。しかし、アプリケーション・プログラムにユーザー名とパスワードを埋め込むことが安全であるとは考えられない。この場合も、トークン化した ID を効果的に使用することで、より高度なセキュリティー標準を実現できる。

これらは一般的な例であるが、必ずしも必須なものではない。たとえば、人間が認証トークンを使用し、アプリケーションが Kerberos チケットを使用することも考えられる。環境に応じて適切なメカニズムが異なることもあり得る。信頼関係を採用して、認証メカニズムと認可メカニズムを強化できる。ここでの意図は、受け入れ可能な認証メカニズムの完璧なリストを明示することではなく、着手のための最適な手がかりを確立することである。

6.5 規則、ポリシー、および制約の役割

クラウドの管理では、ポリシーと制約が重要な役割を演じることが期待される。クラウド実装の内部詳細ではなくクラウド管理のインターフェースに重点を置いた場合、ポリシーと制約を使用してクラウドに必要な機能を定義することが、有用な抽象化となる。これらは、クラウド・サービス・プロバイダーは「何」を提供すべきか（それを「どのように」提供すべきかではなく）を示すものとして解釈できる。

ポリシーにはいくつかの種類がある。たとえば、サービスレベル・アグリーメント（SLA）、サービスレベル目標（SLO）、デプロイの制約（現状および提案中の OVF にあるものなど）、データ配置の制約、監査能力の制約、セキュリティーの制約がある。利用者とプロバイダーまたは開発者とプロバイダーが一連の制約に合意すると、それらの制約によってそれぞれ（特にプロバイダー）の行動が規制される。図 7 は、制約と規則の流れをモデル化したものである。

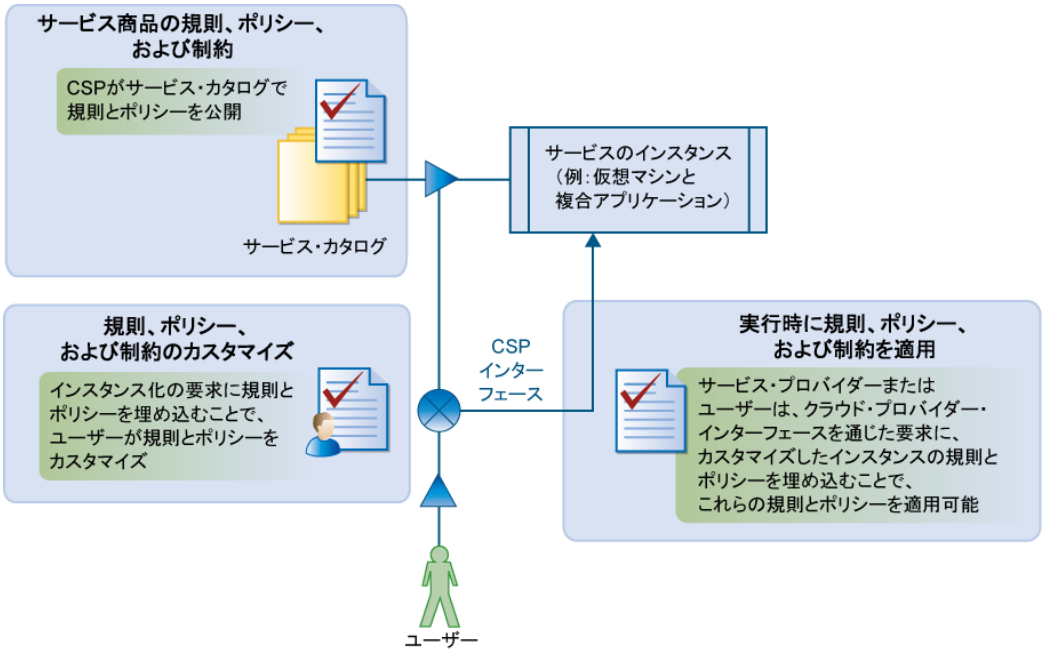


図 7 - 制約の流れ

クラウド・サービス・プロバイダーは、関連するさまざまな制約、規則、およびポリシーを公開し、それらをサービス商品の一部としてサービス・カタログに記述できる。これらは、テンプレートからサービス・エンティティー（仮想マシンや複合アプリケーションなど）をインスタンス化するときに、そのインスタンスの一部となる。

ユーザーは、サービスをインスタンス化するサービス要求の一環として、制約、ポリシー、および規則をカスタマイズできる。サービス・プロバイダーが提示しているポリシーと制約の中には、ユーザーがカスタマイズできないものがあることも考えられる。インスタンス要求には、制約のバンドルを追加するセクションおよび変更するセクションがある。これにより、インスタンスは制約、規則、およびポリシーの集合を有することになる。変更した制約は、インスタンスにそのライフサイクル全体（一時停止/再開、移行など）を通じて適用する必要がある。

サービス・インスタンス（仮想マシンや複合アプリケーションなど）を稼動に移した後、CSP またはユーザーは、制限の範囲内でポリシー・セットの変更を要求できる。稼動中のサービス・エンティティーに適用したポリシーの変更が、CSP にとって必ずしも適切ではない場合や実施可能ではない場合がある。これは、インスタンス化に先立って合意した特性がその変更によって影響を受ける場合である。

制約の例としてセキュリティー・ポリシーがある。サービス・カタログやインスタンスにある商品には、多数の多様なセキュリティー・ポリシーを適用できる。その例をいくつか次に示す。

- **アクセス制御**。サービス・インスタンスにアクセスできるユーザーを一部のクラウド・ユーザー（たとえば、ある企業に属するユーザーや適切な認可レベルを有するユーザー）のみに制限できる。さらに、インスタンスを変更できるユーザーや、インスタンスをデプロイできる場所と時期などをポリシーで指定することもできる。
- **ネットワーク・セキュリティー・ポリシー**。このようなポリシーでは、テンプレートにあるサービス・インスタンスを、クラウド外部で他のサービス・インスタンスやリソースとどのように接続できるかを指定できる。これらのポリシーは、侵入検知と侵入防止のためのトラフィック検査を構成するトラフィック・フィルタリング（ファイアウォールなど）の規則と要件の形態とすることが可能である。

684 • **セキュリティー・ポリシーの「範囲」**。商品では、インスタンスの実行場所を指定できる。たとえば、特定
685 のクラウド・サービス・プロバイダーのインフラストラクチャー、特定の企業のゾーン、または特定の地理
686 的領域に制限できる。

687 7 要件

688 このセクションでは、クラウド・サービス商品の一部としてクラウド・サービス・プロバイダーが対応する必要のある
689 要件の概要を説明する。このリストを完璧なものとしてとらえるべきではない。これは、調和のとれた規範的なクラ
690 ウド・サービス・プロバイダーの商品に存在することが望ましい機能の種類を示すものである。

691 7.1 プロトコル・スタック

692 プロトコル・スタックに関連する要件は次のとおりである。

693 • インターフェースでは、該当のドメインに関連するすべてのメッセージ交換パターン（MEP）をサポートする必
694 要がある。これには、HTTP のような要求応答転送を介した一方向 MEP（プッシュとプル）および双方向
695 MEP（同期と非同期）がある。

696 • REST や SOAP などの業界標準のメカニズムを使用して、さまざまなクラウド・プロバイダー・サービスへの
697 プログラム・アクセスを提供する必要がある。

698 • ペイロードは、数バイトからギガバイト単位までサイズが大幅に変動し、利用者とプロバイダーまたは開発者
699 とプロバイダーの間をどちらの方向にも流れる。状況に基づく適切な選択が必要であり、プロトコルはこれら
700 のどの選択でも可能にするものであることが必要である。

701 • スタックは、認証情報などの補足的なコンテキスト情報を扱うことができる必要がある。

702 • スタックは、クライアントの能力に適応し、最適なメッセージ・パターンを使用する必要がある。ここでいうクラ
703 イアントの能力として、クライアント・アドレス、ペイロード・サイズ、処理時間、およびスループットの可用性な
704 どがある。

705 • スタックは、クライアントまたはサーバーが開始するキャンセル処理をサポートし、大量の転送を再開できる
706 必要がある。

707 7.2 リソース・モデル

708 リソース・モデルに関連する要件は次のとおりである。

709 • リソース・モデルは、インターフェースおよびメカニズムから独立している必要がある。プロトコルからリソー
710 ス・モデルを独立させることで、使用者のアプリケーション（およびプロバイダーのアプリケーション）をプロト
711 コルから独立したものにすることができる。リソース・モデル（エンティティーの種類、対応するデータ、これら
712 両者間の関連付け、操作）に基づいてアプリケーションの機能を実装していれば、管理の相互作用を実行す
713 る際に使用するあらゆるプロトコルにこれらのアプリケーション機能をマッピングできる。このような独立性が
714 あることで、プロトコルの設計と実装のどの部分も変更せずに、隣接するドメインから新しいエンティティーを
715 リソース・モデルに追加することもできる。

716 • リソース・モデルは、膨大な数（100,000 個）のアーティファクトのインスタンス化とアドレス指定をサポートす
717 る必要がある。

718 • プロバイダー・インターフェースで使用可能にするあらゆるリソースは、それを直接操作することはできない
719 場合でも、少なくとも識別の目的で個別にアドレス指定できるものであることが必要な場合がある。さまざま
720 なリソースへのアクセスには、適切な認証を設定する必要がある（たとえば、検出可能なリソースでは、認証
721 を必要とせずにメタデータが利用できなければならない場合がある）。

- 数多くのアクターがリソースへのアクセスを必要とする場合がある。これらのアクターが持つ権利はさまざまなので、モデルはすべてのレベルで認可に対応している必要がある。ロールベースのアクセス制御（RBAC）などのポリシーでは、コンテキストに基づいて適用できる可視性規則を使用して、すべてのリソースへのアクセスを認可し、すべてのリソースを制御することが必要である。アクセス制御の対象とするリソースと機能の例として、各種の集合（仮想データセンター、仮想マシンなどの仮想アプリケーション・スタック要素、ネットワークとストレージ、イメージ、テンプレート、商品など）を作成および操作する機能、リソースを監視する機能、これらのリソースに関する監査情報を要求して受け取る機能がある。
- メタデータのエンドポイントがリソースのエンドポイントと同じである必要はない。たとえば、監査レポートが仮想データセンターのリソースではなくてデータベースであっても良く、パワーオン VM のインターフェースが仮想マシンに直接接続せずに仮想マシンのリソース表現（メッセージ・キューなど）に接続していても良い。
- バージョン管理に対応している場合は、すべての永続性リソースのバージョンを作成して読み取る機能をリソース・モデルが備えている必要がある。特に、複数のアクターがアクセスすることから変化が激しいリソース・モデルで必要になる。
- プロバイダーには、クライアントが必要とするリソース・メタデータを利用可能にする能力が必要である。このようなメタデータとして、データが保存されている場所の地理的な位置などがある。

7.3 採用可能性

採用可能性に関連する要件は次のとおりである。

- サービス商品は、認証やネットワーキングなどの普通に見られる IT メカニズムに統合する必要がある。
- サービス商品では、理解が容易で広く認知されているプログラミング・モデルとプログラマー・スキルを使用する必要がある。
- サービス商品は、他のクラウドと連携できることが必要である。特に、同じ標準に基づくインターフェースをサポートしているクラウドとの連携が必要である。
- 可能な限り、標準の構文、オープンな API、およびオープンな標準を使用する必要がある。

7.4 国際化

世界中のどこからでも容易にアクセスできるクラウドの特性を考慮すると、そのインターフェースでは、ユーザーの場所と言語、管理リソース、およびデータに何の制限もないことが必要である。プロバイダーは制限を実装できるが、インターフェースではプロバイダーの選択に制限を設けてはならない。

7.5 規則、制約、およびポリシー

規則、制約、およびポリシーに関連する要件は次のとおりである。

- クラウド・サービス・プロバイダーは利用者単位でポリシーを定義できるようにする必要がある。このポリシーでは、利用者に代わってクラウド・サービス・プロバイダーがサービス・エンティティー（仮想マシン、ネットワークの接続性、ストレージなど）をどのように扱うかを定義する。
- クラウド・サービス・プロバイダーのインフラストラクチャーで利用者によってサービスがデプロイされるときは、そのインフラストラクチャーでイメージを操作可能にする前に、クラウド・サービス・プロバイダーが設定したポリシーと規則が、利用者が指定する規則のポリシーで強化されることが必要である。クラウド・サービス・プロバイダーとクラウド・サービス利用者との間でポリシーの仕様に関連して矛盾がある場合は、その解決のためにこれらの矛盾をクラウド・サービス利用者に周知しなければならない。
- システムは、規則とポリシーの交換を通じてさまざまな関連する制約を明確に指定する機能、および監査レポートを通じてそのような制約の実装を検証できる能力を備えている必要がある。ポリシーを指定し、監査

情報を検査することで、セキュリティー上のさまざまな脅威が軽減される。これらの機能をどのように実装するかという点は、プロバイダー・インターフェースの適用範囲外である。

- 規則と制約では以下の内容を扱う必要がある。
  - 地理的な制約の観点でコンプライアンスを実現する弾力的な規則と制約
  - データと処理の隣接性に関する隣接性規則
  - 仮想マシン、アプリケーションなどの間でのパフォーマンスの親和性規則
  - PCI のコンプライアンスを実現するための反親和性規則
- 仮想化インフラストラクチャーの場合は、仮想マシンのセキュリティーの基本要素を反映するポリシーと規則をプッシュするメカニズムのほか、ハイパーバイザーのセキュリティー機能を取得するメカニズムが必要である。サービス・カタログとサービス商品には、これらの規則と制約が反映されていなければならない。
- インターフェースは、リソースとデータの分離や仮想化ネットワークのセキュリティーによるネットワークの分離などのマルチテナント・セキュリティー仕様に対応している必要がある。

7.6 クラウド管理インターフェースのセキュリティー

クラウド管理インターフェースのセキュリティーに関連する要件は次のとおりである。

- クラウド・サービス・プロバイダー・インターフェースのエンドポイントおよび制御リソースへのアクセスは、標準化された技術によってセキュリティーで保護されている必要がある（クラウド・サービス・プロバイダーのエンドポイントでのセキュリティーの例として、REST ベースの URL や URL ルーターなどを保護する方法がある）。
- クラウド・プロバイダーのサービスおよびコンテンツのフローに対するプログラム・アクセスおよび手動アクセスは、SSL や TLS などの業界標準の保護メカニズムを使用して保護する必要がある。
- クラウド・サービス・プロバイダーによる保証が必要となる事項として、クラウドのサービス、リソース、およびデータに対するアクセスまたは変更が成功であるか失敗であるかに関係なく、それらのすべてで監査可能な記録が生成されること、およびそれらの監査記録から、エンド・ユーザーまたはサービスが開始したアクションと相関付け可能な一貫性のある監査証跡が作成できることがある。監査記録には、ID または認可のあらゆる委任が明確に記されなければならない。
- 匿名アクセスが必要である場合は、セキュリティー・マネージャーのみを通じて匿名アクセスが許可されることが必要である。これらの状況は例外として記録し、例外とした理由とセキュリティー・マネージャーが実行したアクションの明確な説明とともに監査データベースのログに記録しなければならない。
- クラウド・サービス・プロバイダーは、標準化された技術を使用して、自身と顧客または自身と他のサービス・プロバイダーとの間に信頼関係を構築するサービスを提供する必要がある。このような標準化されたメカニズムとして、証明書、暗号化した要素（鍵など）、および root ID の交換などのほか、以降の信頼関係とポリシーの確立に使用可能なセキュリティー・ポリシーがある。
- クラウド・サービス・プロバイダーは、顧客が ID をプロビジョニングおよび管理できる ID サービスを提供する必要がある。この ID は、標準化されたメカニズムを使用してクラウド・サービスにアクセスするために、認証および認可を受けることができる ID である。このサービスでは、手動または自動手段で ID をプロビジョニングする必要性および外部 ID プロバイダーのサービスと同期または協調する必要性に配慮しなければならない。
- クラウド・サービス・プロバイダーは、その顧客がアプリケーションとデータの保護に使用している暗号化要素と暗号化方法を顧客が制御および管理できるようにする必要がある。この保護は、アプリケーションとデータを使用しているかどうかに関係なく、またサービス・プロバイダーのインフラストラクチャーを使用開始す

801 るとき、使用終了するとき、または使用中の如何に関わらず機能するものである。機能が最小限の状態で
802 は監理者が手動でこの情報を入力でき、機能を拡充するとセキュリティーで保護された同期が可能になる。

803 • クラウド・サービス・プロバイダーは、その ID プロバイダー・サービスへの業界標準によるエントリー・ポイン
804 トを提供する必要がある。このようなエントリー・ポイントにより、利用者はクラウド・サービス・プロバイダーと
805 の適切な信頼関係を確立できる。

806 • クラウド・サービス・プロバイダーは、顧客またはサービスの ID を認証して、セキュリティー・ポリシーとアク
807 セス制御を適用するためにクラウド・サービスで利用可能なセキュリティー情報を生成できるサービスを提供
808 する必要がある。

809 • クラウド・サービス・プロバイダーが提供する ID 機能では、認証トークンなどの標準化されたメカニズムを使
810 用して ID と認可の情報を委任することを認めても良い。ID と認証の情報のあらゆる表現形態は、それが直
811 接であるか委任であるかに関係なく、セキュリティーで保護して提供されなければならない。アカウント保有
812 者が委任を使用すると、該当のサービスのアカウントを持っていない他のユーザーに特定のリソースに対す
813 る時間限定のアクセス権を与えることで、そのユーザーがサービスにアクセスできるようになる。委任の過
814 程でも、このような委任されたアクセスによる使用状況の監査と追跡が実施される。委任されたクライアント
815 に対してその使用量に応じた課金請求ができるように、このようなアクセス用のアカウントを作成する機能を
816 追加することもできる。

817 7.7 データとその保存

818 データとその保存に関連する要件は次のとおりである。

819 • クラウド・サービス・プロバイダーは、データのライフサイクル全体を通じてデータの一貫性が維持されている
820 ことが顧客に明確にわかるようにする必要がある。

821 • 暗号化と鍵管理では業界標準および政府標準を使用する必要がある。クラウド・サービス・プロバイダーは、
822 データ・アクセス制御を目的として役割の管理および責務の分離を実現する必要がある。暗号化と鍵管理で
823 は、マルチテナント・ストレージでのデータの独立した取り扱いおよびサービス・プロバイダーの運営データと
824 顧客データの分離が可能でなければならない。

825 • 顧客の機密データは、保存しているときにのみ暗号化するのではなく、クラウド・サービス・プロバイダーの
826 社内ネットワークで転送する際にも暗号化する必要がある。プロバイダー・インターフェースへの転送アクセ
827 スは、サーバー証明書による TLS/SSL を使用してセキュリティーで保護できる。セキュリティーを強化する
828 ために、必要に応じ、クライアント証明書に基づいて TLS 層での相互認証を実装することもできる。

829 • 規制上の要件に従い、クラウド・サービス・プロバイダーは、内容捜索の目的でデータが第三者（サプライ・
830 チェーンや政府など）によって押収される可能性がある場合（または押収された場合）、そのデータの所有者
831 に適切に通知する必要がある。

832 – 第三者によってデータが押収される可能性があるかかどうかは、プロバイダーの商品の定義で記述す
833 ることもでき、その場合はどのような通知も意味を持たない。

834 – データが押収された場合、そのデータの所有者に通知するかどうかはプロバイダーのガバナンスに基
835 づく判断となる。

836 • データの喪失、意図しないデータの上書き、およびデータの破損を防止するために、クラウドでのデータの
837 バックアップと復元の体制を設け、効果的に維持する必要がある。

838 • データの地理的な保存場所は、必要に応じてクラウド利用者に公表する必要がある。クラウド・サービス・プ
839 ロバイダーがデータをホストし、要求されている手段を実装している場所を把握することで、境界を越えたデ
840 ータの移動を規制している地域の法律に対するコンプライアンスを確保できる。

- データを保持する機能と安全に破棄できる機能を準備しておく必要がある。サービス・プロバイダーは、削除するデータをクラウドの中で正確、効果的に同定でき、データの破棄に関して一定レベルの保証と証拠を示すことができなければならない。

7.8 ログ、イベント管理、インシデント対応、および通知

ログ、イベント管理、インシデント対応、および通知に関連する要件は次のとおりである。

- クラウド・サービス・プロバイダーは、規定のインターフェースを使用して顧客にインシデントとアラートの情報を提供する必要がある。
- クラウド・サービス・プロバイダーは、顧客のインシデント・データの機密性を維持する必要がある。
- イベント管理インターフェースは、クラウド利用者がそれぞれのシステムについて推定できるように、識別可能なデータ・ソース（アプリケーション・ログ、ファイアウォール・ログ、IDS（侵入検知システム）ログなど）による豊富な意味体系（セマンティック）を有している必要がある。これらのイベントおよび情報は、クラウド・サービス利用者が使用して、請求の検証、監査レポートの提供、規制に対するコンプライアンスの検証、機能特性の監視などができるものでなければならない。
- クラウドでサービス・エンティティー（仮想マシンやその他のサービス）を稼動している間、クラウド・サービス・プロバイダーは、そのサービス・エンティティーを監視して稼動、請求、コンプライアンスなどに関するイベントをログに記録している必要がある。クラウド・サービス・プロバイダーは、SLA に従って、この情報をリアルタイム、バッチ・モード、またはその両方でクラウド・サービス利用者が利用できるようにしなければならない。
- イベント・インターフェースは、インシデントの通知、インシデント対応、インシデントの抑制、およびインシデントからの回復を扱ううえで十分な汎用性を備えていることも必要である。
- わかりやすいエラー・メッセージを伴うエラー処理メカニズムが必要である。
- ログの保持期間を規定した SLA に従い、稼動ログを用意しておく必要がある。
- ログを保存し、その一貫性を維持する適切なメカニズムを採用する必要がある。その例として、デジタル署名やライト・ワンス型メディアを使用したストレージ・デバイスがある。

7.9 監査、法令、およびコンプライアンス監視

監査、法令、およびコンプライアンス監視に関連する要件は次のとおりである。

- クラウド・サービス・プロバイダーは、顧客がそれぞれの情報リスク管理要件の実施で利用できる数値基準と管理要素を用意する必要がある。
- クラウド・サービス・プロバイダーは、顧客のセキュリティー要件がどのように満足されているかを示す証拠を提供する必要がある。
- クラウド・サービス・プロバイダーは、必要に応じ、外部監査を実行するためのインターフェースを提供する必要がある。
- クラウド・サービス・プロバイダーは、ビジネス・モデルおよび業界規則の要求に応じて、SAS70 などの適切な監査標準に適合する必要がある。

7.10 仮想化技術に対するセキュリティー上の考慮事項

仮想化技術に対するセキュリティー上の考慮事項に関連する要件は次のとおりである。

- 仮想化オペレーティング・システムの監理的なアクセスと制御は、強力な認証、ID 管理、RBAC（責務の分離（SoD）によるもの）、およびログ記録と一貫性監視のツールに統合する必要がある。

878 • クラウド・サービス・プロバイダーは、仮想マシンを隔離してセキュリティー・ゾーンを作成する機能を利用可
879 能にする必要がある。このセキュリティー・ゾーンは、用途の種類（デスクトップとサーバーなど）、運用段階
880 （開発、本番、テストなど）、仮想マシンの存在場所、およびデータの機密性に基づいて作成する。

881 • イベント管理インターフェースによるレポート・メカニズムでは、マルチテナント環境でのデータの分離と責務
882 の分離を示すうえで必要な情報を提供しなければならない。

883 • クラウド・サービス・プロバイダーは、仮想マシンとアプリケーション・サービスの定義を可能にする必要があ
884 る。このような仮想マシンのイメージとサービスの定義は、その利用者の要求に応じて、他の利用者がその
885 イメージを参照できないように保護する必要がある。

886 7.11 セキュリティーで保護した移行の可搬性および相互運用性

887 セキュリティーで保護した移行の可搬性と相互運用性に関連する要件は次のとおりである。

888 • 仮想マシンなどのサービス・エンティティーは、組織と所有権の境界（企業とサービス・プロバイダーの IaaS
889 インフラストラクチャーとの境界など）を越えて移行できることが必要である。

890 • 仮想化インフラストラクチャーでは、セキュリティーで保護したプロビジョニング解除（別の場所または別のサ
891 ービス・プロバイダーに移行した後の仮想マシンの廃止）および部分的な移行（クラウド・バースト：新旧の場
892 所および新旧のサービス・プロバイダーの間でのセキュアーな統合）に仮想マシンが対応している必要があ
893 る。

894 • サービス・プロバイダーは、制御の効果、管理、監視、およびレポート・インターフェースの一貫性および新旧
895 の場所とプロバイダーにわたるこれらの統合の一貫性を保証する必要がある。

896 • ストレージの移行機能を提供する場合、そのサービス・プロバイダーは、データ転送機能をセキュリティーで
897 保護していることを検証しておく必要がある。このような保護措置として、暗号化、アクセス制御、鍵管理、ス
898 トレージ・デバイスの廃止、移行したデータの破棄などがある。

899 8 プロトコルの例

900 クラウド・インフラストラクチャーの検討では、特定したユースケースの解釈を、広く使用されているプロトコルに落
901 とし込む必要がある。インフラストラクチャーの観点から要件を特定するために実施した分析では、分析に向けた
902 さまざまな手法が調査されている。一連の基準を以下の概要グループに分類して使用し、それぞれのユースケー
903 スが分析されている。

904 • 相互作用の特性

905 • リソースの要件

906 • デプロイの要件

907 • セキュリティーの要件

908 ユースケースの相互作用を分析する作業の中で、相互作用パターンの単純なセットがさまざまな組み合わせで
909 繰り返されている様子がすべてのユースケースで見られた。さらに、9 種類の小規模なパターンを多彩に組み合
910 わせて、特定されたそれぞれのユースケースを構成できることも判明している。これらの構成ブロックをセマンティ
911 ック相互作用パターン（SIP）と呼んでいる。たとえば、状態要求は多くのユースケースに関与していることから、独
912 立した相互作用パターンと考えられる。他の相互作用としてリソース交渉や変更通知なども見られるが、それが
913 使用されているユースケースはさまざまである。これらのことから、上記の基準分類に基づくインフラストラクチャ
914 ー分析で SIP が使用されている。このセクションでは、メッセージ交換のパターンを重点的に取り上げ、ごく一般
915 的なプロトコルのスタイルにこれらのパターンを解釈する方法の例を紹介する。プロトコル解釈の多様なスタイル

916 の中から、ここでは転送プロトコルとして HTTP を使用し、IT 管理領域で広く見られる 2 種類のスタイルを検討す
917 る。この 2 つのスタイルとは、リモート・プロシージャー・コール（RPC）と表現状態移送（REST）である。

918 これらのスタイルを区別するいくつかの特徴を以下に挙げる。

919 • RPC スタイルでは、リソースを操作するための専用のインターフェースが、それぞれのリソース・クラス
920 で用意されていることを前提としている。この RPC として、RPC over HTTP と SOAP RPC の 2 つが
921 考えられる。どちらも、メソッドの呼び出しに XML ペイロードを使用し、HTTP の POST メソッドを使用
922 してこのペイロードを送る。

923 • REST スタイルでは、GET、POST、PUT、DELETE などの標準的なメソッドをリソースとのインターフェ
924 ースとした HTTP などのプロトコルにより、ハイパーテキスト主導のパラダイムを使用する。各リソース
925 は URI で識別する。

926 ## 8.1 メッセージ交換パターン

927 2 つのメッセージ交換パターン（MEP）を使用して、クラウド・ユーザーとクラウド・サービス間のセマンティック相互
928 作用を解釈できる。1 つは一方向 MEP であり、もう 1 つは双方向 MEP である。さらに、さまざまな接続上の制約
929 やユーザー・プロファイルを反映するために、これらの MEP に「実行モード」を定義すると効果的である。このモ
930 ードは、特定の要件や環境上の制約に関連付けることができる。このセクションでは、一方向 MEP と双方向
931 MEP に適用できる実行モードをいくつか紹介する。これらの実行モードは、相互作用を完了するために送信側と
932 受信側が実行する 2 つのアクションである「プッシュ」と「プル」に関連している。「送信側」と「受信側」という語は
933 総称的なものであり、応答を返信するエンティティーが、要求を受信したアドレス指定可能なエンティティーと同一
934 であるとは限らない点に注意を要する。

935 このプロトコルのオプションには、HTTP を使用するうえでのバリエーションがいくつか示されている。プロトコルの
936 ソリューションには、RPC スタイルとドキュメント・スタイル、RPC と REST、REST スタイルによるプレーンな
937 HTTP ソリューションと SOAP ベースのソリューションといったいくつかの側面がある。ここでは、REST と RPC と
938 の対照に重点を置く。RPC に依存しない SOAP ベースのソリューション（document/literal の SOAP スタイルな
939 ど）についてはここでは取り上げない。

940 ### 8.1.1 一方向 MEP

941 一方向 MEP の実行モードは、図 8 および以下に示すとおりである。

942 • **一方向/プッシュ MEP**（「プッシュ」モード）では、送信側が率先して受信側に業務メッセージを送信する
943 （HTTP 要求によるものなど）。このモードでは、受信側が IP アドレスを持っていることが前提となる。こ
944 の例として、通知される側（ユーザー）がアドレス指定可能な場合の変更通知の相互作用がある。

945 – RPC を使用した一方向/プッシュ MEP でクライアントおよびサービスが実行する一般的な手順:

946 1) オブジェクトのインスタンス ID、メソッド呼び出し、およびパラメーターを収録した RPC ペイロ
947 ードを持つ HTTP POST 要求をクライアント（送信側）が発行する。

948 2) 成功または失敗を示す状態コードである HTTP 応答をサービスが送信する。RPC ペイロー
949 ドは想定されていない。

950 – REST を使用した一方向/プッシュ MEP でクライアントおよびサービスが実行する一般的な手順:

951 1) 特定のリソースを表す URI を対象としてクライアント（送信側）が HTTP コマンド（PUT、
952 POST、または DELETE）を発行する。

953 2) HTTP の成功または失敗の応答によってサービスが応答する。

954 Annex A は、HTTP を使用したプロトコルの例である。

- **一方向/プル MEP**(「プル」モード)では、受信側が率先して送信側から業務メッセージをプルする(たとえば、特定のキューまたはメールボックスのプル信号を収めた HTTP 要求を開始する)。応答は、HTTP 応答を介して送信される。このモードは、業務メッセージを受信するためのアドレスを受信側が持っていなくても機能する。

  この例として、通知される側(ユーザー)がアドレス指定不可能で、プロトコルのバックチャネルで取得する通知をプルする必要がある場合の変更通知の相互作用がある(たとえば、HTTP 応答)。

  プル・モードでは、MEP を正常に完了するために、受信側が定期的にプル信号を数回送信することがある(たとえば、応答工程上の業務メッセージ)。これらのプル信号は、多くの場合、基礎となるメッセージング層で自動的にスケジュール設定される(たとえば、WS-MakeConnection(「Web Services Make Connection (WS-MakeConnection) Version 1.1」)の使用、HTTP GET の発行、「OASIS ebXML Messaging Services Version 3.0」に説明がある PullRequest 信号などの使用による)。このようなプル信号には、プルに絶対的に必要な最小限の情報(キューID、チャネル ID など)のみが収められ、それ以外の業務情報は収録されていない。

  ここでは、クライアントが呼び出しを発行する前か、発行する呼び出しの中で、クライアント側の URI を指定できることが前提となっている。また、キューID またはメールボックス ID がクライアントに対して指定され、それ以降はサービスからのすべてのプルでそれらの ID を使用するという、プルの別形態もある。

  – RPC を使用した一方向/プル MEP でクライアントおよびサービスが実行する一般的な手順:

    1) 必要に応じて、イベント ID で指定された URL に対する RPC ペイロードを持つ通知をクライアントが受け取る。

    2) 詳細を入手するためのパラメーターとしてイベント ID(またはキューID)を指定した HTTP GET または HTTP POST をクライアントが発行する。

    3) 指定のイベント ID に該当するリソース情報についてのペイロードを応答する側が返送する。

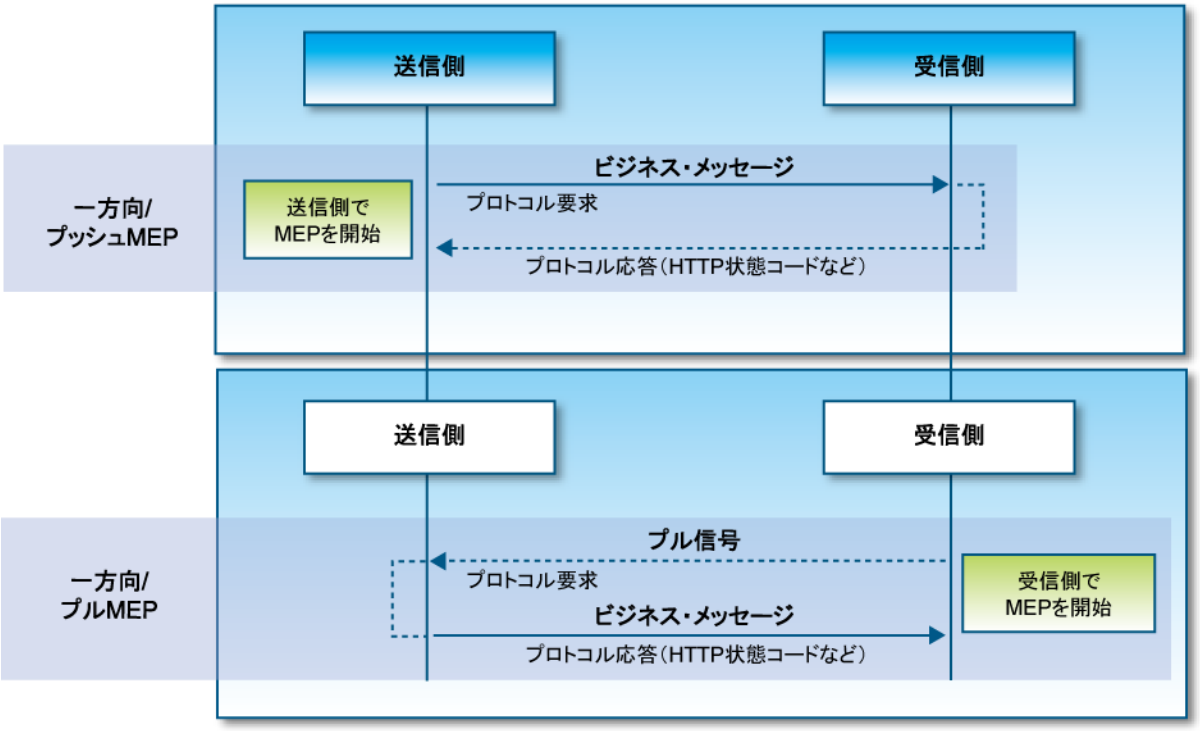
  – REST を使用した一方向/プル MEP でクライアントおよびサービスが実行する一般的な手順:

    1) 必要に応じて、イベントの URI に関する通知をクライアントが受け取る。

    2) そのイベント URI の HTTP GET をクライアントが発行する。

    3) その URI で特定されたリソースに関するペイロードを応答する側が返送する。

  Annex A は、HTTP を使用したプロトコルの例である。



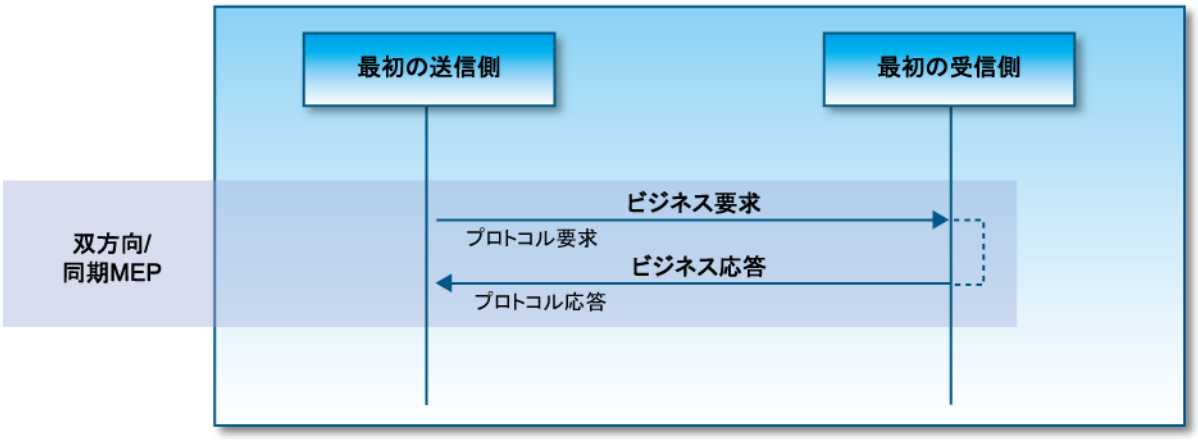
983

984 **図 8 - 一方向 MEP の実行モード**

985 ### 8.1.2 双方向 MEP

986 双方向 MEP の実行モードは以下に示すとおりである。

987 • **双方向/同期 MEP**（「同期」モード）では、送信側から率先して受信側に業務要求を送信し、その送信と
988 同じ接続上で業務応答を取得する（HTTP などの要求応答転送の場合）。このモードでは、送信側がア
989 ドレス指定可能である必要はない。この例として、状態情報を要求する相互作用において、その状態
990 情報が迅速に生成されることから単一の HTTP 要求応答の交換に対応できることを前提としている状
991 況が挙げられる。この様子を図 9 に示す。



992

993 **図 9 - 双方向/同期 MEP の実行モード**

994 – RPC を使用した双方向/同期 MEP でクライアントおよびサービスが実行する一般的な手順:

995 (これは、一方向/プッシュ MEP の例とほとんど同じである。主な相違は、この実行モードでは
996 RPC 応答のペイロードを扱っている点である。)

997 1) オブジェクトのインスタンス ID、メソッド呼び出し、パラメーターなどの要求データを収録した
998 RPC ペイロードを持つ HTTP POST 要求をクライアントが発行する。

999 2) このメソッドが成功すると、RPC ペイロードを持つ HTTP 応答をクライアントが受け取る。メ
1000 ソッドが失敗した場合は、失敗を示す HTTP エラー・コードを受け取る。

1001 – REST を使用した双方向/同期 MEP でクライアントおよびサービスが実行する一般的な手順:

1002 1) 特定のリソースを表す URI を対象としてクライアントが HTTP コマンド(GET、PUT、POST、
1003 または DELETE)を発行する。クライアントでは業務応答を受け取ることが想定される(この
1004 点は一方向/プッシュ MEP とは異なる)。

1005 2) メソッドが成功すると、リソース・データを表すペイロードを持つ HTTP 応答をクライアントが
1006 受け取る。

1007 Annex A は、HTTP を使用したプロトコルの例である。

1008 • **双方向/プッシュ・プッシュ MEP** は非同期モードであり、業務要求と業務応答がそれぞれ別々の接続
1009 で送信される(たとえば、別々の HTTP 接続上で両方をプッシュする)。送信側と受信側の両方がアド
1010 レス指定可能であることが前提となる。応答にいくらかの計算時間を必要とする場合に、このモードが
1011 効果的である。この例として、簡単なリソース交渉の相互作用において、要求されたリソースをクラウ
1012 ド・サービスが割り当てるために、1 つの HTTP 要求応答で対応できる遅延よりも長い時間を必要とす
1013 る状況が挙げられる。この場合は、コールバックで確認が送信される。この様子を図 10 に示す。

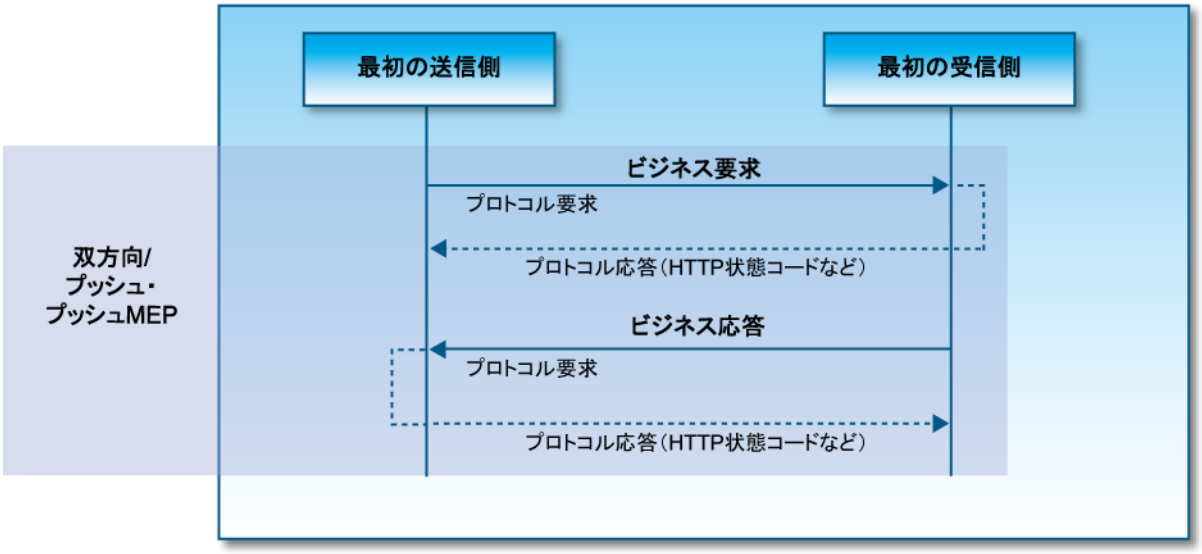


図 10 - 双方向/プッシュ・プッシュ MEP の実行モード

– RPC を使用した双方向/プッシュ・プッシュ MEP でクライアントおよびサービスが実行する一般的な手順:

1) クライアントのアドレス指定可能なエンドポイントなどの要求データを収めた RPC ペイロードを持つ HTTP POST 要求をクライアントが発行する。

2) 必要に応じ、この要求に応答するために継続中の作業を示す ID でサービスが応答する。

3) いくらかの時間が経過した後、当初のメソッド呼び出しへの応答を収めたペイロードを持つコールバックを、クライアントの URL に対してサービスが(HTTP POST などを使用して)発行する。

– REST を使用した双方向/プッシュ・プッシュ MEP でクライアントおよびサービスが実行する一般的な手順:

1) リソースを特定する URI に対してクライアントが HTTP コマンドを発行する。

2) 必要に応じ、クライアントが状態情報の取得元とすることができる URI でサービスが応答する。

3) いくらかの時間が経過した後、クライアントが登録した URI に対して、サービスが(普通は HTTP POST を使用して)コールバックを発行する。

Annex A は、HTTP を使用したプロトコルの例である。

- **双方向/プッシュ・プル MEP** は非同期モードであり、送信側が率先して業務要求のプッシュと業務応答のプルの両方の交換を実行する。このモードが「プッシュ・プッシュ」モードに比べて有利な点は、アドレス指定可能なエンドポイントを送信側に必要とせず、要求の転送準備が整う時期を送信側で決定できることである。この例として、簡単なリソース交渉の相互作用において、クラウド・ユーザーがアドレス指定可能ではなく、確認メッセージを後でプルする必要がある状況が挙げられる。この様子を図 11 に示す。

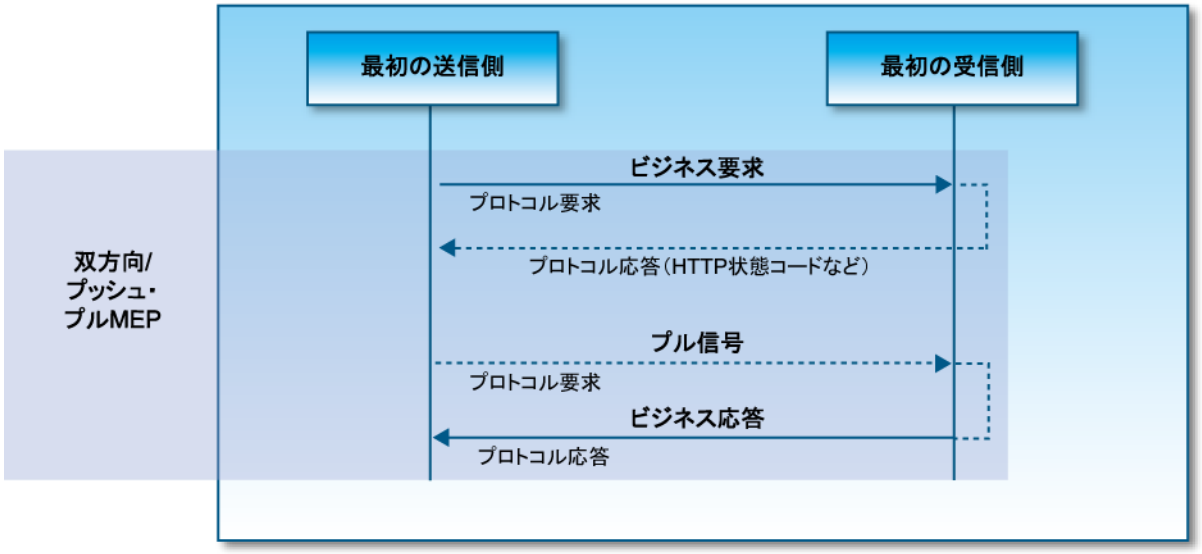


図 11 - 双方向/プッシュ・プル MEP の実行モード

– RPC を使用した双方向/プッシュ・プル MEP でクライアントおよびサービスが実行する一般的な手順:

1) オブジェクトのインスタンス ID、メソッド呼び出し、およびパラメーターを収録したペイロードを持つ HTTP POST 要求をクライアント（「最初の送信側」）が発行する。

2) 必要に応じ、ID に関する合意がまだ成立していない場合は、以降の問合せでクライアントが使用する ID を収めたペイロードでサービス（「最初の受信側」）が応答する。

3) 最初の要求に対する応答で受け取った ID を収めたペイロードを持つ HTTP POST 要求をクライアントが発行する。

4) 業務 RPC ペイロードを HTTP 応答でサービスが返送する。プルするメッセージが存在しない場合は、空のペイロードを返送する。

– REST を使用した双方向/プッシュ・プル MEP でクライアントおよびサービスが実行する一般的な手順:

1) リソースを特定する URI に対してクライアント（「最初の送信側」）が HTTP 要求を発行する。

2) 必要に応じ、以降の要求でクライアントが使用する URI を収めたペイロードでサービス（「最初の受信側」）が応答する。

3) その URI に対して HTTP GET 要求をクライアントが発行する。

4) 業務・ペイロード（通知など）を HTTP 応答でサービスが返送する。ペイロードのない HTTP 応答コードをサービスが返すだけになることもある。

HTTP を使用したプロトコルの例については Annex A を参照のこと。

8.2 MEP およびその実行モードの選択に影響するインフラストラクチャー的側面

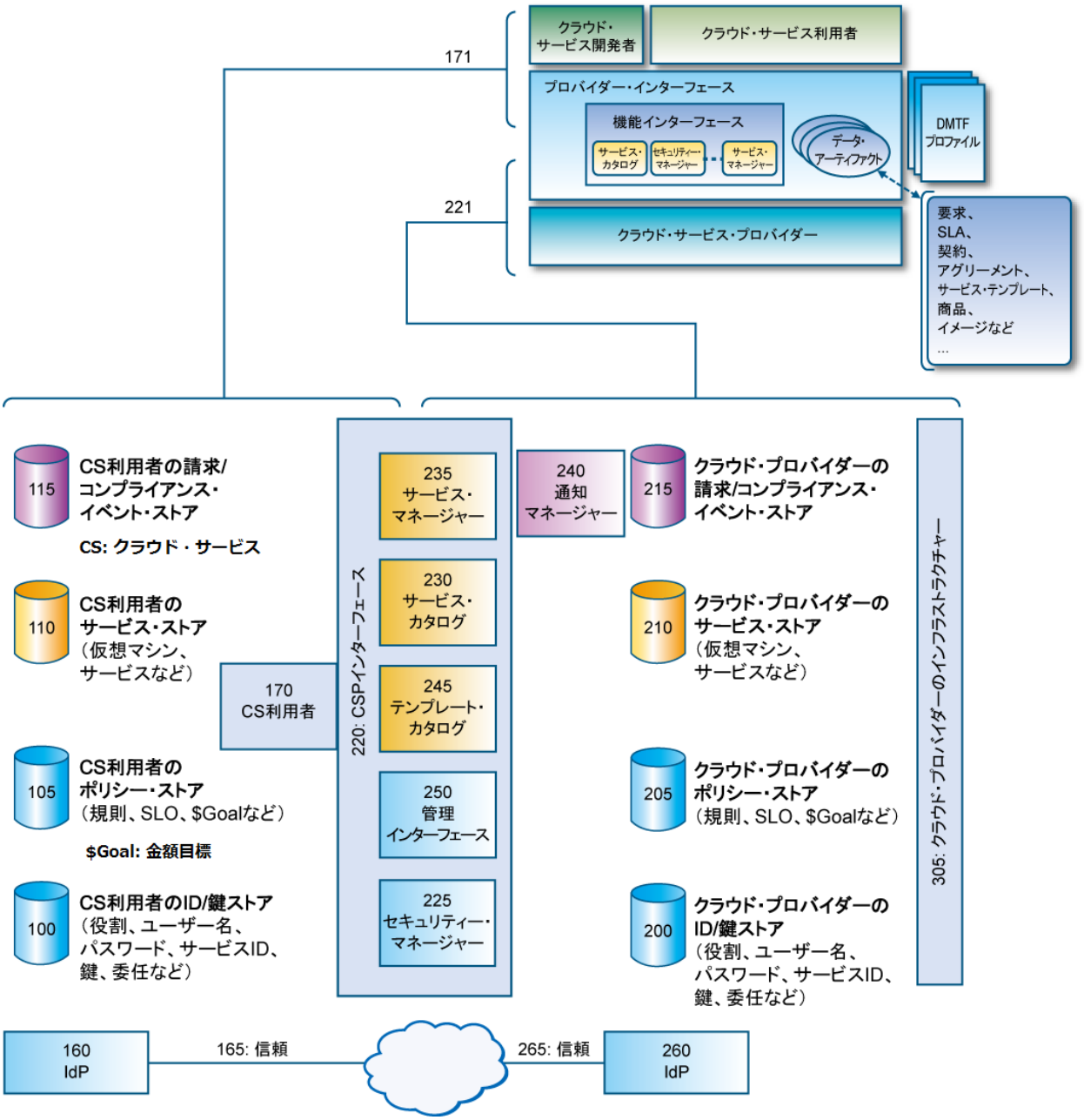
インフラストラクチャー的な側面の中には、他の側面よりも MEP の選択に深い関係を持つものがある。

- ペイロードのサイズを知ることで、相互作用に要すると考えられる時間の程度がわかり、それに応じて相互作用を同期パターンまたは非同期パターンのどちらにすることが望ましいかを判断できる。
- バックエンドでの処理に要すると考えられる時間の長さは、実行モードの選択に影響する。操作の実行に要する時間の長さは、その実行のために取得するペイロードとはまったく関係がないことがある。長い遅延が発生する場合は、双方向/同期 MEP を使用できないことがある。これは、そのような遅延が、1 つの HTTP 要求応答で対応できるタイミングよりも長くなる可能性があるためである。一方、(SLA などによって)応答時間を短く制限している場合は、双方向/同期 MEP が使用可能である。
- クライアントがアドレス指定可能であるかどうかは、実行モードの選択に影響する。クラウドによってクライアントをアドレス指定できる場合にのみ、そのクライアントに対して双方向/プッシュ・プッシュ MEP (「コールバック」によるものなど)または一方向/プッシュ MEP を使用できる。アドレス指定できないクライアントは、そこから発行した要求に対する応答をすべて非同期でプルする必要がある。
- 最後に、サイズの大きなコレクションを閲覧または列挙する機能も、閲覧操作にどの MEP を使用するかという選択に関連性を持つ。これはここで挙げた最初の項目のバリエーションであるが、名前空間での列挙操作が発生することから、処理時間に関する別の特性が対象になることがある。

9 クラウドのエコシステム

このセクションでは、本書の前半で説明した参照アーキテクチャーを詳しく取り上げ、ホワイト・ペーパー「クラウド管理のユースケースおよび相互作用」(DSP-IS0103)に説明のあるデータ・アーティファクトを使用して、参照アーキテクチャーを理論的に説明する。このセクションの内容を十分に理解するには、DSP-IS0103 の内容に精通している必要がある。このセクションを理解することで、クラウド・サービス・プロバイダーが提供する商品で必要とされるアーキテクチャー上の相互作用および機能面の能力を把握できるようになることが期待できる。ここで取り上げる情報はあくまでも参考情報であり、必須ではない。

図 12 は、「相互運用可能なクラウド」(DSP-IS0101)で紹介しているアーキテクチャーが、図 13 に示すアーキテクチャーのより詳しい図および説明とどのような関係にあるかを理解しやすいように用意されている。図 12 では、番号 221 の領域で、クラウド・サービス・プロバイダーが提供するプロバイダー・インターフェースが番号 225～250 の要素によって示され、クラウド・サービス・プロバイダーが番号 200～215、240、および 305 の各領域によって示されている。番号 171 の領域では、クラウド・サービス利用者が番号 100～170 の各領域によって示されている。番号 160、165、265、および 260 の各領域では、ID プロバイダー間の関係が示されている。番号 160 および 260 によって示されている ID プロバイダー・サービス(IdP)の機能は、番号 165 および 265 によって示されている信頼関係とともに、より詳しい図である図 13 を検討する際の一助となるように示されている。



1090

1091 **図 12 - アーキテクチャーの概要**

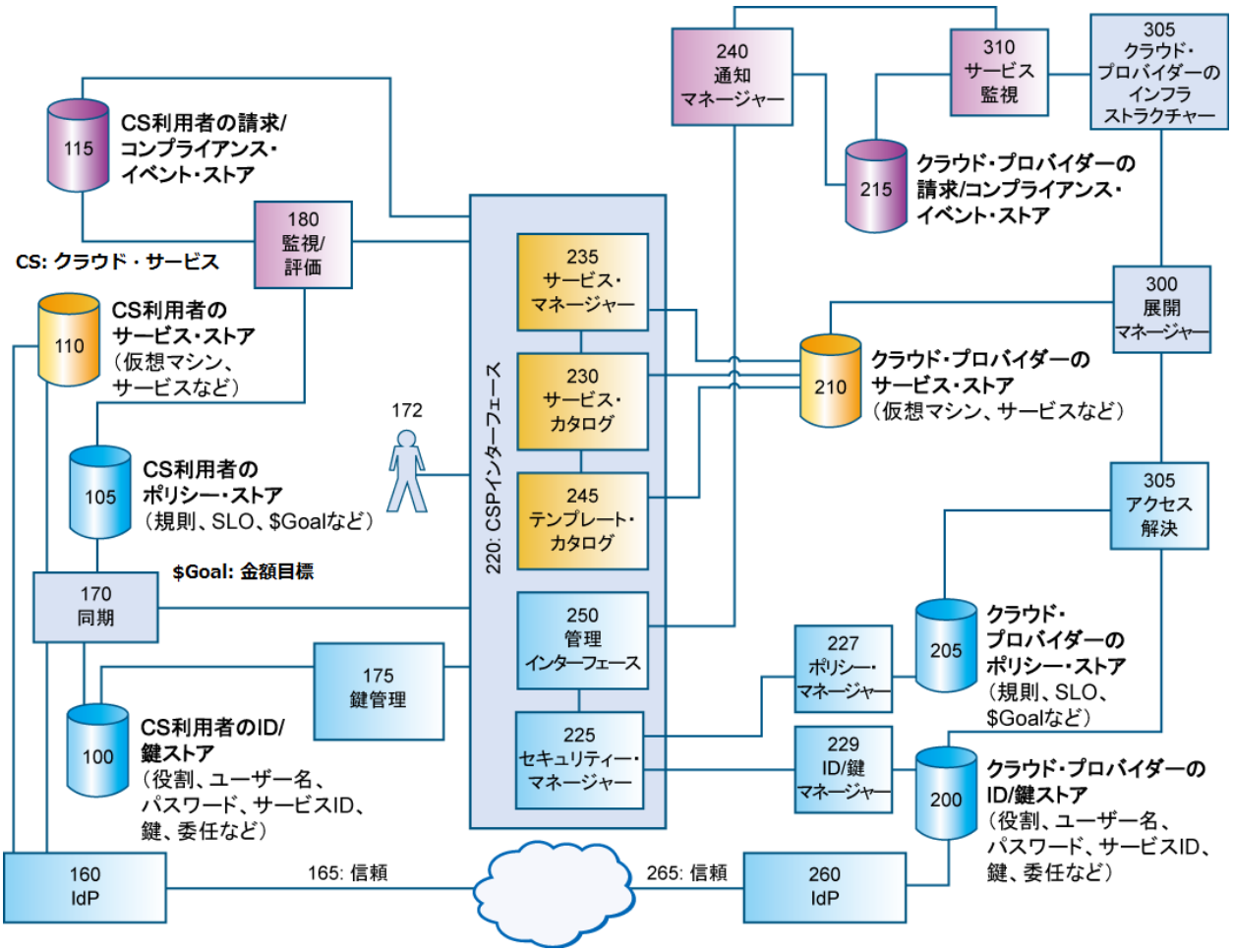


図 13 - アーキテクチャーの詳細

図 13 の説明は、インターフェース 225、230、235、245、および 250 で管理しているデータ・アーティファクトおよびこれらのインターフェースがサポートしているデータ・ストア 200、205、210、215、100、105、110、および 115 を取り上げることから始める。これらのインターフェースは、ホワイト・ペーパー「クラウド管理のユースケースおよび相互作用」(DSP-IS0103)に説明のあるデータ・アーティファクトのライフサイクル管理を担当している。

9.1 データ・アーティファクトに対するアーキテクチャーの影響

サービス・マネージャー(235)は、DSP-IS0103 に説明のある次のデータ・アーティファクトを管理している。

- サービス・インスタンス
- サービス・トポロジー
- サービス・トポロジー項目
- サービス・トポロジー関係

サービス・カタログ(230)は、DSP-IS0103 に説明がある次のデータ・アーティファクトを管理している。

- サービス商品
- サービス契約

- サービス要求

テンプレート・カタログ（245）は、DSP-IS0103 に説明がある次のデータ・アーティファクトを管理している。

- サービス・テンプレート

セキュリティー・マネージャー（225）は、DSP-IS0103 に説明がある次のデータ・アーティファクトを管理している。

- ユーザーID 情報（UserIdentityInformation）
- 認証トークン（CredentialsToken）

監理インターフェース（250）は、DSP-IS0103 に説明がある次のデータ・アーティファクトを管理している。

- サービス・レポート情報
- サービス・イベント・ログ
- サービス通知
- 上記以外のデータ・アーティファクトについては、どれもここでは取り上げない。

9.2 クラウド・サービス・プロバイダーの ID/鍵ストア（200）

クラウド・サービス・プロバイダーの ID/鍵ストア（200）は、ローカルの ID、役割、暗号鍵、委任指定などを保存および提供するためのアーキテクチャー上の抽象化をクラウド・サービス・プロバイダーに実現する。クラウド・サービス・プロバイダーの ID/鍵ストア（200）にあるデータ・アーティファクトは、そのクラウド・サービス・プロバイダーが提供することもできるが、多くの場合はクラウド利用者の監理者（172）が提供する。この管理者は、利用者のビジネスに携わってクラウド・リソースを使用する各業務担当者のユーザー名、パスワード、役割などを管理する。この監理者がクラウド・サービス・プロバイダーの ID/鍵ストア（200）をプロビジョニングすることは労力を要するので、現在の市場ではクラウド利用者が望んでいないことがわかっている。

クラウド・サービス・プロバイダーの ID/鍵ストア（200）にあるデータ・アーティファクトの別のソースとして、クラウド利用者が番号 170 の領域で用意している同期機能がある。この機能は、利用者の設備にある ID/鍵ストア（100）を基にしてユーザー名、パスワード、役割、委任指定などを同期する。このメカニズムを使用することで、承認済みクラウド・ユーザーおよび役割と委任の指定を、管理者による方法よりもはるかに短時間でプロビジョニングおよびプロビジョニング解除できる。ただし、利用者の業務とシングル・サインオンのメカニズムに認証される人物が使用するパスワードではないパスワードを番号 170 の同期機能が提供できない場合は、利用者の内部データ・システムにアクセスできるユーザー名とパスワードが露見することになる。代わりの方法として、クラウド・サービス・プロバイダーの ID/鍵ストア（200）で別のユーザー名とパスワードを提供することもできるが、ユーザー名とパスワードを覚えている人物やモニターに「貼り付けたメモ」で露見している認証情報には対処できないという問題がある。

利用者の業務のうち、ID プロバイダー（160）にアクセスできるものでは、クラウド・サービス・プロバイダーの ID プロバイダー（260）との信頼関係（165、265）を確立でき、ユーザー名やパスワードなどのプロビジョニングとプロビジョニング解除を、ID プロバイダー（160）が生成する ID 表明の属性から実行できる。この方法にも、機密性のある認証情報がクラウド・サービス・プロバイダーの ID/鍵ストア（200）で露見する可能性やユーザーに対して複数の認証情報が要求される可能性（ID プロバイダー（160）で複数のユーザー名とパスワードの使用を仲介して、ユーザーの負担にならないようにしたとしても同様である）があるという問題が存在する。

クラウド・サービス・プロバイダーに利用者の ID を提供するうえでもっとも安全なメカニズムは、クラウド・サービス・プロバイダーの ID/鍵ストア（200）にはユーザー名とパスワードを保存せず、ID プロバイダー（260）を使用する方法である。クラウド・サービス・プロバイダー・インフラストラクチャー（305）でのアクセス解決では、ID プロバイダー（160）にある ID トークンを直接使用するようにして、クラウド・サービス・プロバイダーの ID/鍵ストア（200）にはユーザーの認証情報を一切保存しないようにする。代わりのメカニズムとして、クラウド・サービス・プロバイダ

一の ID/鍵ストア（200）に ID 表明を保存しておき、（有効期間中であるなどで）ID 表明が有効な限りは、それをユーザー名とパスワードの代わりに使用するという方法もある。

同期機能（170）とクラウド利用者の監理者（172）を使用する場合、クラウド・サービス・プロバイダーは監理インターフェース（250）の機能を提供することが必要になる。この機能は、セキュリティー・マネージャー（225）を通じて ID/鍵マネージャー（229）に情報を適切に提供するインターフェースまたは API として機能する。ID/鍵マネージャー（229）は、クラウド・サービス・プロバイダーの ID/鍵ストア（200）を最新の状態に維持する。これらの機能（他のすべての機能とともに図 13 に示す）は、あくまでもアーキテクチャー上のものであり、実装のメカニズムではなく、実行すべき機能である。

クラウド・サービス・プロバイダーの ID/鍵ストア（200）に影響する別の機能として、利用者の代わりに実施する暗号鍵の保守がある。これらの鍵は、監理者（172）または鍵管理メカニズム（175）が提供できる。どちらの場合でも、セキュリティー・マネージャー（225）を通じてクラウド・サービス・プロバイダーの ID/鍵ストア（200）に鍵情報を適切に提供するために必要な機能は監理インターフェース（250）が提供する。クラウド・サービス・プロバイダーの ID/鍵ストア（200）に暗号要素を保存すると、それが外部に露見する可能性がリスクとして発生する。ID 同様に、鍵要素も ID トークンを使用して渡すことができるので、クラウド・サービス・プロバイダーの設備（160、165、260、265 など）に暗号要素をローカルに保存せずに、それらの暗号要素を提供することが可能である。

クラウド・インターフェースにおけるセキュリティー・ポリシーを示すために、図 B-1 の Annex B では組織の観点からセキュリティー・ポリシーを説明している。表 1 は、図 B-1 の Annex B にある項目と図 13 にある項目との対応関係を示したものである。

表 1 - 図 B-1 にある項目と図 13 にある項目との対応関係

Annex B のポリシー・モデル（図 B-1）	図 13 での参照対象
項目 120、121、122、および 123	項目 110、210、および 245
項目 124	項目 230 および 235
項目 126	項目 105
項目 127	項目 205
項目 125	項目 105 および 205
項目 255	項目 305
項目 310	項目 310
項目 315	項目 215
項目 320 および 325	項目 180

9.3 クラウド・サービス・プロバイダーのポリシー・ストア（205）

クラウド・サービス・プロバイダーのポリシー・ストア（205）は各種ポリシーのリポジトリーを提供する。これらのポリシーは、クラウド・サービス・プロバイダーのインフラストラクチャー（305）の使用、サービスレベル目標（SLO）、クラウド利用者と交渉した料金が関与する金額目標（$Goal）、クラウド利用者のデプロイでのファイアウォール設定などを管理する。クラウド・サービス・プロバイダーのポリシー・ストア（205）は、機能的にセキュリティー・マネージャー（225）に関連付けられているポリシー・マネージャー（227）によって保守されている。多くの場合、クラウド・サービス・プロバイダーが自身の各種ポリシーを規定しているが、それにより、クラウド利用者の監理者（172）は利用者が使用するポリシーを選択し、そのポリシーに限定的な変更を実施できる。クラウド・サービス・プロバイダーが自身のポリシー・ストア（205）を使用して利用者のポリシーを手動で保守する作業は労力を要するうえ、規模の変更にも対応が困難である。この状況は、クラウド利用者のポリシー・ストア（105）の使用を促すものである。このポリシー・ストアでは、ポリシーや規則などを利用者が管理し、クラウド・サービス・プロバイダーのポリシー・スト

ア（205）との同期を図る（これらのポリシーは、クラウド利用者の CMDB 運用の一環として利用者が保守することが普通である）。クラウド利用者側からクラウド・サービス・プロバイダー側にポリシーを同期する方法では、クラウド利用者がプライベート・クラウドをホストしていて、プライベート・クラウドのデプロイとパブリック・クラウドのデプロイで同じポリシーを使用する場合に特に大きな利点が得られる。このようにポリシーを同期するには、クラウド・サービス・プロバイダーのポリシー・ストア（205）でのポリシー仕様と、クラウド利用者のポリシー・ストアでのポリシー仕様との間で、170、250、225、および 227 による調停が必要になる。

9.4 クラウド・サービス・プロバイダーのサービス・ストア（210）

クラウド・サービス・プロバイダーのサービス・ストア（210）は、仮想マシンおよびサービスにクラウド・ストレージを提供する。本書では、特定のサービスを提供するように構成した仮想マシンのコレクション（集まり）をサービスと定義している。たとえば、電子メール・システムは、SMTP ゲートウェイ、IMAP4 サーバー、POP3 サーバー、ストレージ、DNS サーバーなどで構成されている。トラフィックの増減や不平衡にサービスが適切に反応するように、ロード・バランシングとロード・シェディングの規則をサービスに設定することもできる。

従来から、監理者（172）はサービス・カタログ（230）やテンプレート・カタログ（245）を使用して、クラウド・サービス・プロバイダーのサービス・ストア（210）に構成済みになっているサービスと仮想マシンを探し出している。仮想マシンやサービスの構成は、サービス・マネージャー（235）を通じて実行するが、このサービス・マネージャーも仮想マシンやサービスを新規に定義する機能を監理者（172）に提供している。

現在のプライベート・クラウドの使用状況では、プライベート・クラウドと複数のパブリック・クラウドの両方で使用できるサービスと仮想マシンを、クラウド利用者が自身専用として定義できるようにすることが有利になっている。この機能は、CS（クラウド・サービス）利用者のサービス・ストア（110）としてアーキテクチャーに示されている。番号 170 で示されている同期機能は、サービス・マネージャー（235）が提供する機能とサービスを通じて利用者のサービス・ストア（110）をプロバイダーのサービス・ストア（210）に同期するメカニズムを提供する。同期によって仮想マシンとサービスを追加することもできれば、削除することもでき、そのたびにサービス・カタログ（230）が更新される。クラウド利用者の関係者のうち、仮想マシンとサービスで意図したユーザー（すべての一般ユーザーまたはメンバーシップ・リストで制限したユーザー）のみがそれらにアクセスできるように、サービス・カタログ（230）とサービス・マネージャー（235）はいずれも ID で保護されている。このアーキテクチャーには、クラウド利用者のサービス・ストア（110）と ID プロバイダー（160）との関係も示されている。法規制へのコンプライアンスと監査上のその他の懸案の面から、ID および ID ごとに適用されるポリシーがますます重要になっているので、デジタル署名とデジタル ID で仮想マシンとサービスを保護することが必要である。これらのデジタル ID、および多くの場合の各種の暗号要素は ID プロバイダー（160）から得られる。信頼関係（165、265）があることから、このように ID を適用した仮想マシンとサービスはクラウド・サービス・プロバイダーのインフラストラクチャー（具体的にはデプロイマネージャー（300））で認識可能である。これにより、ポリシーとアクセスを ID で規制でき、改ざんを認識できる。

9.5 クラウド・サービス・プロバイダーの請求/コンプライアンス・イベント・ストア（215）

クラウド・サービス・プロバイダーの機能をサポートしているアーキテクチャーには、重要な機能要素として、クラウド・サービス・プロバイダーの請求/コンプライアンス・イベント・ストア（215）がある。仮想マシンとサービスはクラウド・サービス・プロバイダーのインフラストラクチャー（305）の中で機能しているので、サービス・モニター（310）で監視を実行して、クラウド・サービス・プロバイダーの請求/コンプライアンス・イベント・ストア（215）に適切な請求イベントと規制コンプライアンス・イベントを記録することが重要である。クラウド・サービス・プロバイダーは、これらのイベントを使用して、月末の請求資料、クラウド・サービス・プロバイダーからの請求を顧客が検証するための監査資料、および規制コンプライアンスに関連する監査証跡を提供する。

クラウドを利用している企業は、クラウド・リソースの使用状況を追跡し、クラウドのプロセスとリソースで発生する監査イベントをリアルタイムで監視する方法を求めている。アーキテクチャーでは、これは CS 利用者の請求/コンプライアンス・イベント・ストア（115）として示されている。このイベント・ストアは、通知マネージャー（240）を通じてクラウド・サービス・プロバイダーで請求イベントとコンプライアンス・イベントを監視するようにしたプル・モデルまたはプッシュ・モデルによって提供され、監理インターフェース（250）を通じて管理される。通知マネージャー

1221 (240)もリアルタイムのイベント・フローを監視/評価機能(180)に提供可能であり、これによってクラウド利用者の
1222 業務がクラウドの評価を監視し、規制コンプライアンスに発生する可能性のある逸脱をリアルタイムで捕捉できる
1223 ようになる。監視/評価機能(180)も、イベント・ストア(115)に収められているデータから得られた法的な評価に対
1224 応できる。監視/評価機能(180)によるすべての評価で、イベントによってコンプライアンスからの逸脱やコスト目
1225 標の超過が発生するかどうかが、クラウド利用者のポリシー・ストア(105) に保存されているポリシーによって判
1226 断される。同様に、サービスレベル目標(SLO)を使用して、クラウド利用者とクラウド・サービス・プロバイダーが
1227 合意したサービスレベル・アグリーメント(SLA)が遵守されていることを検証できる。

1228 ### 9.6 クラウド・サービス・プロバイダーのデプロイ(300)

1229 最後に、このアーキテクチャーには、クラウド・サービス・プロバイダーのインフラストラクチャー(305)での仮想マ
1230 シンとサービスのデプロイが示されている。これは、デプロイ・マネージャー(300)により、まずアクセス上の問題
1231 を解決し(305)、続いてアクセス要求をサービスに整合させることで実行する。サービスと仮想マシンの使用がポ
1232 リシーと ID によって管理されるようにする対応は、デプロイ・マネージャー(300)でとる必要がある。このポリシー
1233 には、クラウド・サービス・プロバイダーによる内部ポリシーとクラウド利用者が指定するポリシーがある。サービ
1234 スは、監理者(172)による手動要求または監理インターフェース(250)を通じた手続き上やポリシー上の指定に
1235 従って、クラウド・サービス・プロバイダーのインフラストラクチャー(305)にデプロイされ、同様に最終的にはクラ
1236 ウド・サービス・プロバイダーのインフラストラクチャーからプロビジョニング解除される。

1237 ## 10 結論および次のステップ

1238 Open Cloud Standards Incubator の現在のフェーズは終了している。このインキュベーターの作業から、IaaS
1239 クラウドについてユースケース、データ・アーティファクト、およびアーキテクチャー上の検討事項をまとめた 3 点
1240 の文書が得られている。この成果物は、各種標準の策定に使用できるように適切な DMTF 作業部会とアライア
1241 ンス・パートナーに移管される。一方、Open Cloud Standards Incubator は、次に取り組むべき分野の検討に入
1242 っている。

1243 ### 10.1 標準策定の手順

1244 インキュベーターの成果物を取り入れて DMTF 標準仕様を作成するには、対象領域をモデル化し、インフラスト
1245 ラクチャーとインターフェースの詳細を設計する作業部会が必要である。インキュベーターでの検討により、次の
1246 2 つの作業部会設置が要請されている。

1247 • Platform Management Subcommittee の下に置く Cloud Management Model Working Group
1248 (CMMWG)

1249 • Infrastructure Subcommittee の下に置く Cloud Management Interface Working Group(CMIWG)

1250 DMTF の理事会がこれらの適切な作業部会を承認した後、その趣意表明が
1251 http://www.dmtf.org/about/committees/に掲載される予定である。

1252 ### 10.2 検討すべき重要な拡張機能

1253 このセクションでは、DMTF クラウドの柔軟性、相互運用性、および取得の容易さの改善に向けて、さらに必要な
1254 作業について説明する。以下では、残存している作業を項目別にまとめている。設置される 2 つの作業部会には、
1255 その検討作業の一環として何らかの方法でこれらの作業に取り組むことが期待される。

1256 #### 10.2.1 インターフェースの拡張性

1257 インキュベーターのインターフェースは、クラウドで利用できるインターフェースよりも対応範囲が「狭い」。たとえば、
1258 この両方が必要な利用者はどのような対応をとるであろうか。両方を並行して使用するだろうか。それとも、

DMTF インターフェースの範囲にある実装によってより詳細なインターフェースが得られるように、拡張可能な DMTF インターフェースを確保するだろうか。

10.2.2 テンプレートの拡張性

新しい種類のエンティティーをカタログに追加する手段およびカタログにある既存のエンティティーを別のエンティティーに結合する手段の実現が必要になる可能性がある。たとえば、クラウド利用者が既存のサービスの前面にファイアウォールを追加しようとする状況はどのように表現すればいいだろうか。インターフェースには、既存のテンプレートをコピーしてその一部を変更する機能、クライアントの末端にあるメタデータ・ファイル（OVF など）にテンプレートをダウンロードしてその一部を変更し、クラウドにアップロードして戻す機能などが必要である。

10.2.3 管理インターフェースの粒度

業界での実装で実現できるクラウド・インターフェースは、大別して以下の 3 グループに分類できる。

- 総合的な関係、サービス契約、カタログ管理などの面を担当する監理グループ
- サービス・インスタンスやそのライフサイクルに関連する内容など、ワークロードを扱うワークロード管理グループ
- VM や論理サーバー、NIC、ストレージ・ボリュームなどの個々のアーティファクトの管理面を担当するリソース管理グループ

インキュベーターのインターフェースは、監理グループを表し、さらに一定の範囲のワークロード/グループを表している。リソース管理グループを表すアーティファクトは扱っていない。

10.2.4 セキュリティー・モデルの粒度

セキュリティー・モデルでは、いっそうの明確化を必要とする可能性がある。その例として、クラウドの参照アーキテクチャーに対する主要なセキュリティー・カテゴリー（ID とアクセスの管理、データ保護、セキュリティー・イベントの管理、およびソフトウェア、プラットフォーム、インフラストラクチャーのコンプライアンス保証）の対応関係がある。さらに、サービス・プロバイダーによってクラウド利用者のポリシーと要件がどのように満足されるかという点について、より細分化した可視性と制御を指定できるオプションをクラウド利用者に提供するために、専用のインターフェースを定義することが必要になる可能性もある。

10.3 アライアンス・パートナーのフレームワークとの統合

Open Cloud Standards Incubator は、分野固有の実装を提供するアライアンス・パートナーに注目し、DMTF クラウドの実装がアライアンス・パートナーと相互運用できることを期待している。その一例として、SNIA で作業中のストレージ・クラウド標準がある。DMTF クラウド・インターフェースと SNIA などのアライアンス・パートナーが作成したインターフェースを接続するための指針を提供する活動が必要である。リソースとモデルの適合性の見地からの作業およびプロトコルの見地からの作業の両方が必要になることもあり得る。

11 クラウド・インキュベーターの今後

Open Cloud Standards Incubator に関しては、さらに作業を必要とする 2 つの領域が見出されている。その 1 つはクラウドの連携であり、もう 1 つはプラットフォーム・アズ・ア・サービス（PaaS）である。インキュベーターの趣意は、これらの領域でいっそうの作業を展開するように拡大されることが推奨される。

Annex A
（参考情報）

メッセージ交換プロトコルの例

この付録では、XML-RPC 指向のメカニズムおよび REST 指向のメカニズムを使用して、9.1 で紹介したさまざまなメッセージ交換パターン（MEP）を実現する例を提供する。

A.1 一方向/プッシュ MEP：RPC の例

A.1.1 クライアントの要求：ユーザーの年齢の設定

```
POST /MyWebService.Contos.com HTTP/1.1
   Content-Type: text/xml
      Content-length:
      <?xml version=”1.0”?>
         <methodCall>
            <methodName>Users.SetUserAge</methodName>
            <params>
               <param>
                  <name><string>Brian Young</string></name>
               </param>
               <param>
                  <value><i4>23</i4></value>
               </param>
            </params>
         </methodCall>
```

A.1.2 サービスの応答

```
   HTTP/1.1 200 OK
   Host: contoso.com
   Content-Type: text/xml
   Content-length:0
```

A.2 一方向/プッシュ MEP：REST の例

A.2.1 クライアントの要求：ユーザーの年齢の設定

```
   PUT /uri_of_brain_young
   Host: contoso.com
   Content-Type : application/vnd.cloud.com.User+xml
   Content-Length: xxxx
   <xml>
      <user>
         <age value=”23”/>
      </user>
```

```
    </xml>
```

A.2.2 サービスの応答

```
    HTTP/1.1 200 OK
    Host: contoso.com
    Content-Type: text/xml
    Content-length:0
```

A.3 一方向/プル MEP:RPC の例

一方向/プル MEP は、MEP を実行するたびに異なる特定の選択指定で問合せた結果が「プル」で得られるメッセージになるのではないという点で、要求応答(双方向)同期 MEP とは異なる。むしろ、メッセージのキューからの読み取りにより近い形態である。多くの場合、プル・メッセージはチャネル ID またはキューID(メールボックス ID と同等)のみを認識している。通知済みの「イベント ID」を使用するものもある。これは、クライアントが後工程で 1 回だけ使用することが必要な ID である。

プルを実行するクライアントは、事前にこのイベント ID またはチャネル ID を把握しておく必要がある。イベント ID またはチャネル ID は、特定のクライアント・メソッドを呼び出すサービスから動的に得ることができる。ここの例では、クライアントが呼び出しを発行する前か、発行する呼び出しの中で、クライアント側の URL を指定できることが前提となっている。

A.3.1 サービスが開始した通知

```
<!--- In this variant, an "event ID" is initially notified to the client by the
service. The event ID is to be used for subsequent pulling. ---!>

    POST /MyAddress.Client.com HTTP/1.1
    Content-Type: text/xml
    Content-length:

    <?xml version="1.0">
       <methodCall>
          <methodName>notifyEvent</methodName>
          <params>
             <param><eventID><i4>1234</i4></eventID></param>
             <param>
                <eventDescription><string>"disk low"</string></eventDescription>
             </param>
          </params>
       </methodCall>
```

A.3.2 クライアントの応答

```
    HTTP/1.1 200 OK
    Host: client.com
    Content-Type: text/xml
    Content-length:0
```

A.3.3 クライアントの要求:プル・イベント

```
<!--- Pull the Event Request using a one-way/pull method. ---!>
```

```
   POST /MyWebService.Contoso.com HTTP/1.1
   Content-Type: text/xml
   Content-length:

   <?xml version=”1.0”?>
   <methodCall><methodName>eventGetInfo</methodName></methodCall>
      <params>
         <param>
            <eventID><i4>123456</i4></eventID>
         </param>
      </params>
   </methodCall>
<!--- Pull the Event Response ---!>
```

A.3.4 サービスの応答

```
   HTTP/1.1 200 OK
   Host: contoso.com
   Content-Type: text/xml
   Content-length:

   <?xml version=”1.0”?>
   <methodResponse>
         <params>
            <param>
               <maxDisk><kilobytes>512000000</kilobytes></maxDisk>
            </param>
            <param>
               <usedDisk><kilobytes>501000000</kilobytes></usedDisk>
            </param>
         </params>
   </methodResponse>
```

A.4 一方向/プル MEP:REST の例

A.4.1 サービスが開始した通知

```
   POST /MyAddress.Client.com/Events
   Content-Type : application/vnd.cloud.com.Event+xml
      <event>
         <eventURI>/Events/1234</eventURI>
         < ventide>1234</ ventide>
         <eventMessage>disk low</eventMessage>
      </event>
```

A.4.2 クライアントの応答

```
   HTTP/1.1 200 OK
   Host:Client.com
   Content-length:0
```

A.4.3 クライアントの要求

```
GET /MyWebService.Contoso.com/Events/1234
Accept :application/vnd.cloud.com/Event+xml
```

A.4.4 サービスの応答

```
HTTP/1.1 200 OK
Host: contoso.com
Content-Type : application/vnd.cloud.com/Event+xml
Content-Length:xxx

<?xml version="1.0"?>
   <event>
      < ventide>1234</ ventide>
      <eventMessage>disk low</eventMessage>
      <eventDetail>
         For Volumn abc: 501000000 out of 512000000 are used
      </eventDetail>
   </event>
```

A.5 双方向/同期 MEP:RPC の例

A.5.1 クライアントの要求:フィルターによるユーザーの列挙

```
POST /MyWebService.Contoso.com HTTP/1.1
Content-Type: text/xml
Content-length:
<?xml version="1.0"?>
   <methodCall>
      <methodName>Users.EnumerateUsers</methodName>
         <params>
            <param>
               <filter><string>*B*</string>
            </param>
         </params>
   </methodCall>
```

A.5.2 サービスの応答

```
HTTP/1.1 200 OK
Host: contoso.com
Content-Type: text/xml
Content-length:

<?xml version="1.0"?>
   <methodResponse>
      <params>
         <param>
            <name><string>Betty White</string></name>
         </param>
```

```
            <param>
               <name><string>Brian Young</string></name>
            </param>
         </params>
      </methodCall>
```

A.6 双方向/同期 MEP：REST の例

A.6.1 クライアントの要求：フィルターによるユーザーの列挙

```
   GET /Users?name=[*B*]
   Host:MyWebService.Contoso.com
   Accept:application/vnd.cloud.com/Users+xml
```

A.6.2 サービスの応答

```
   HTTP/1.1 200 OK
   Host: contoso.com
   Content-Type : applicationapplication/vnd.cloud.com/Users+xml
   Content-Length:

   <?xml version="1.0"?>
      <users>
         <user>
            <name>Betty White</name>
         </user>
         <user>
            <name>Brian Young</name>
         </user>
      </users>
```

A.7 双方向/プッシュ・プッシュ MEP：RPC の例

```
   <!----- Issue API call to format volume ----!>
```

A.7.1 クライアントの要求：ボリュームのフォーマット

```
   POST /MyWebService.Contoso.com HTTP/1.1
   <?xml version="1.0">
      <methodCall>
         <methodName>formatVolume</methodName>
         <params>
            <param>
               <volumeID><string>"some volume"</string></volumeID>
            </param>
            <param>
               <callbackURL><string>"http://MyAddress.Client.com"</callbackURL>
            <param>
         </params>
      </methodCall>
```

A.7.2 サービスからの最初の応答

```
HTTP/1.1 200 OK
Host: contoso.com
Content-Type:text/xml
Content-Length:

<?xml version=”1.0”?>
   <params>
      <param>
         <jobID><i4>1234</i4></jobID>
      </param>
      <param>
         <eventDescription>
            <string>”volume format Initiated”</string>
         </eventDescription>
      </param>
   </params>
```

A.7.3 サービスが開始した進捗通知

```
POST /MyAddress.Client.com
Content-Type: text/xml
Content-Length:

<?xml version=”1.0”?>
   <methodResponse>
      <params>
         <param><jobID><i4>1234</i4></jobID></param>
         <param><maxDisk><kilobytes>512000000</kilobytes></maxDisk></param>
         <param><usedDisk><kilobytes>0</kilobytes></usedDisk></param>
      </params>
   </methodResponse>
```

A.7.4 サービスに対するクライアントの応答

```
HTTP/1.1 200
Host: contoso.com
Content-Length:0
```

A.8 双方向/プッシュ・プッシュ MEP：REST の例

A.8.1 クライアントの要求：ボリュームのフォーマット

このクライアントは、要求を送信する前にコールバック URI を登録済みである。

```
PUT /Volumes/MyVolumeuri
Host: MyWebService.Contoso.com
Content-Type: application/vnd.cloud.com.Volume+xml
Content-Length: xxxx

```

```
<xml>
   <volume>
      <format>desired_format</format>
   </volume>
</xml>
```

A.8.2 サービスからの最初の応答

```
HTTP/1.1 200 OK
Host:Contoso.com
Content-type: text/xml
Content-length:0
```

A.8.3 サービスが開始した進捗通知

```
POST /MyAddress.client.com/
Content-type : applicationapplication/vnd.cloud.com.Event+xml
Content-length :
< ?xml version="1.0" ?>
<xml>
   <event>
      <resource
       uri="/Volumes/MyVolumeuri"
       type="application/vnd.cloud.com.Volume+xml"
       status="READY"/>
      <detail>Reformatting the volume from aaa to bbb</detail>
   </event>
</xml>
```

A.8.4 クライアントの応答

```
HTTP/1.1 200 OK
Host: Client.com
Content-length:0
```

クライアントは、このイベントで返されたリソース URI を使用して、GET メソッドで目的のボリュームの詳細を取得できる。

A.9 双方向/プッシュ・プル MEP：RPC の例

A.9.1 クライアントの要求：ボリュームのフォーマット

```
<!----- Issue API call to format volume ----!>

POST /MyWebService.Contoso.com HTTP/1.1
Content-Type: text/xml
Content-length:

<?xml version="1.0">
   <methodCall>
      <methodName>formatVolume</methodName>
```

```
         <params>
            <param><volumeID><string>”MyVolumeID”</string></volumeID></param>
         </params>
      </methodCall>
```

A.9.2 サービスの応答

```
   HTTP/1.1 200 OK
   Host:Contoso.com
   Content-Type: text/xml
   Content-Length:
   <!--- receive jobID to poll --!>

   <?xml version=”1.0”?>
      <methodResponse>
         <params>
            <param><jobID><i4>1234</i4></jobID></param>
         </params>
      </methodResponse>
```

A.9.3 クライアントの要求:ジョブの状態の確認

```
   POST /MyWebService.Contoso.com HTTP/1.1
   Content-Type: text/xml
   Content-length:
   <!-PULL with jobID as the parameter ---!>

   <?xml version=”1.0”?>
      <methodCall>
         <methodName>pollJob</methodName>
         <params>
            <param><jobID><i4>1234</i4></jobID></param>
         </params>
      </methodCall>
```

A.9.4 サービスの応答

```
   <!- get back progress information ---!>
   HTTP/1.1 200 OK
   Host: contoso.com
   Content-Type: text/xml
   Content-Length:

   <!xml version=”1.0”?>
   <methodResponse>
      <params>
         <param><jobID><i4>1234</i4></jobID></param>
         <params<percentComplete><i4>74</i4></param>
      </params>
   </methodResponse>
```

A.10 双方向/プッシュ・プル MEP：REST の例

A.10.1 クライアントの要求：ボリュームのフォーマット

```
PUT /Volumes/MyVolumeuri
Host: MyWebService.Contoso.com
Content-Type : application/vnd.cloud.com.Volume+xml
Accept : application/vnd.cloud.com.Volumn+xml, application/vnd.cloud.com.Event+xml
Content-Length: xxxx

<xml>
   <volume>
      <format>desired_format</format>
   </volume>
</xml>
```

A.10.2 サービスの応答

```
HTTP/1.1 200 OK
Content-type : application/vnd.cloud.com.Event+xml
Location: /Events/ThisEventUri
Content-Length:xxxx

<?xml version=”1.0”?>
   <xml>
      <event uri=”/Events/ThisEventUri”>
         <resource
          uri=”/Volumes/MyVolumeuri”
          type=”application/vnd.cloud.com.Volume+xml”
          status=”REFORMATTING” percentage=”0”/>
         <detail>reformatting volume from aaa to bbb</detail>
      </event>
   </xml>
```

A.10.3 クライアントの要求

```
<!---------- Poll for the status of the job ---!>

GET /Events/ThisEventUri
Host: MyWebService.Contoso.com
Accept : application/vnd.cloud.com.Event+xml,
```

A.10.4 サービスの応答

```
http/1.1 200 OK
Content- type: application/vnd.cloud.com.Event+xml
Location: /Events/ThisEventUri
Content-Length:xxxx
<?xml version=”1.0”?>
   <xml>
      <event uri=”/Events/ThisEventUri”>
```

```
            <resource
             uri=”/Volumes/MyVolumeuri”
             type=”application/vnd.cloud.com.Volume+xml”
             status=”REFORMATTING” percentage=”74”/>
            <detail>reformatting volume from aaa to bbb</detail>
          </event>
      </xml>
```

Annex B
（参考情報）

ポリシーの検討

この付録では、クラウドのポリシー・モデルがどのようなものになるかについて説明する。これは規範的なものではなく、あくまでも説明にすぎない。

B.1 ポリシー・モデル

ポリシー・モデルは図 B-1 のように表現できる。

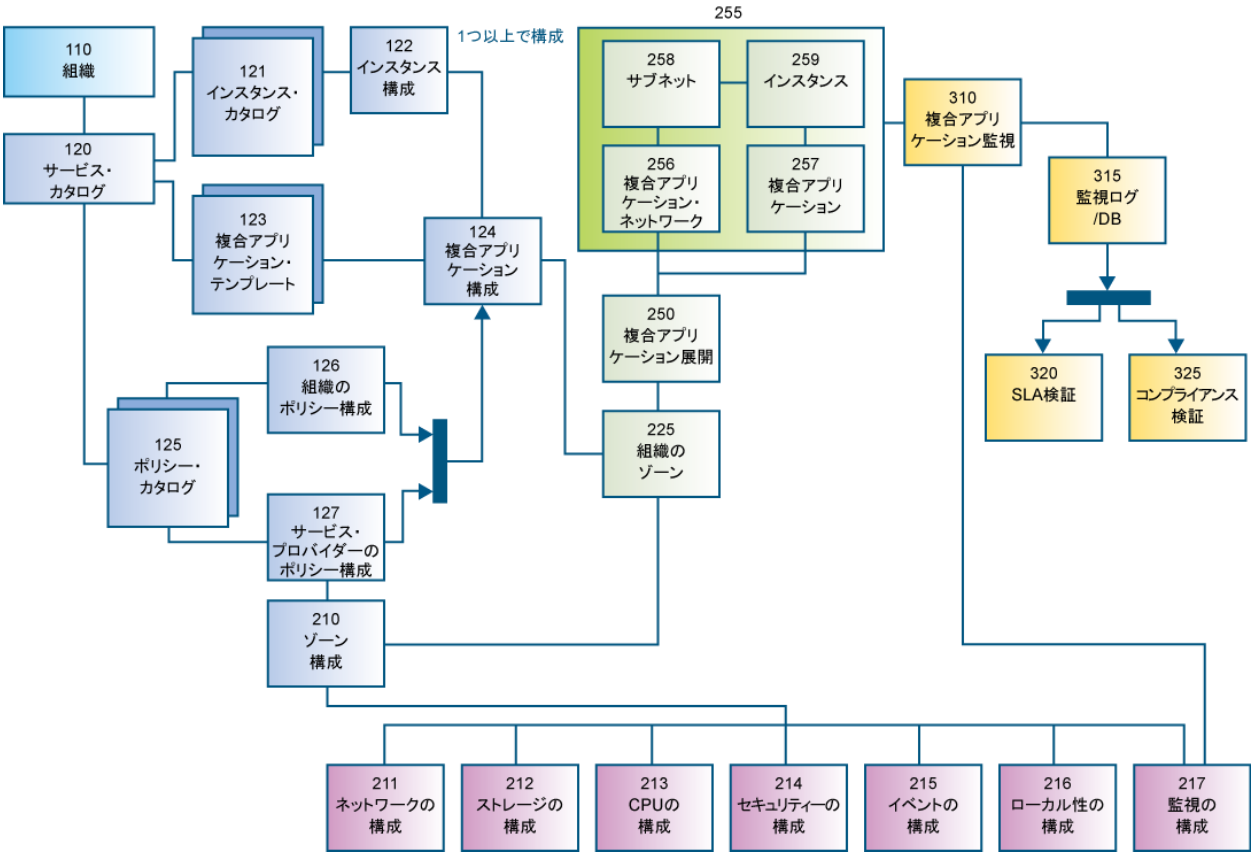


図 B-1 - ポリシー・モデル

B.1.1 組織の観点から見たポリシー・モデル

ここでは、サービス・プロバイダーのクラウドを使用することを目指している組織（110）を考える。この組織（110）で使用するサービスは、サービス・プロバイダーとの契約により、サービス・プロバイダーのカタログのうち、組織が承認したものにあるサービスに限定されている（組織とサービス・プロバイダーが取り交わした契約の中での取り決め）。サービス・プロバイダーのカタログは、インスタンス・カタログ（121）、複合アプリケーション・テンプレート

1693 （123）、およびポリシー・カタログ（125）で構成されている。インスタンス・カタログ（121）は個々のインスタンス
1694 （122）のコレクションであり、VM リポジトリーを形成している。

1695 複合アプリケーション・テンプレート（123）は、使用することを目的として組織が定義したインスタンスとテンプレー
1696 トのコレクションである。これらのテンプレートは、API によるアップロード、個々のインスタンス・グループからの作
1697 成、サービス・プロバイダーのクラウドへのデプロイ、または手動でのアップロードが可能である。

1698 ポリシー・カタログ（125）は、組織のポリシー構成（126）およびサービス・プロバイダーのポリシー構成（127）から
1699 成っている。サービス・プロバイダーのポリシー構成（127）は、IaaS のインフラストラクチャーの使用とデプロイを
1700 管理するサービス・プロバイダーの各種ポリシーで構成される（たとえば、ネットワーク（211）、ストレージ（212）、
1701 CPU（213）、セキュリティー（214）、イベント（215）、ローカル性（216）、および監視（127）の各構成）。

1702 ここで組織はサービス・カタログ（120）にサービスのリポジトリーを構築する。これは、インスタンスの仕様（インス
1703 タンス（122）へのポインターなど）と構成および運用上の規定（SLO の仕様など）を含んでいる。この組織では、
1704 サービスが運用に入ったときにそのサービスに関して組織がガバナンス面で期待する内容を定義したポリシー
1705 （126）を事前に規定している。120、121、123、および 126 のコレクションは組織の CMDB の一部と見なすこと
1706 ができる。

1707 ### B.1.2 サービス・プロバイダーの観点から見たポリシー・モデル

1708 サービス・プロバイダーは、IaaS クラウド全体を構成するゾーンごとに運用上の特性と構成を規定する。これは、
1709 ネットワークの構成（211）、ストレージの構成（212）、CPU の構成（213）、セキュリティー境界の構成（214）、イベ
1710 ントの構成（215）、ローカル性の構成（216）、および監視の構成（217）の各ゾーン構成から成るコレクションとし
1711 てのゾーン構成（210）で表される。これらの構成のそれぞれは、インフラストラクチャー・ポリシー適用の構成を指
1712 定するうえで必要な、ハードウェアと相互接続の詳細な構成を提供する。

1713 サービス・プロバイダーのポリシー構成（127）は、ゾーン構成（210）のコレクションに対するガバナンスを提供す
1714 る（複数のゾーンが存在するので、ゾーンとその相互作用を管理するポリシーも複数存在する）。210 および 127
1715 のコレクションは、サービス・プロバイダーの CMDB の一部と見なすことができる。

1716 複合アプリケーションの構成プロセス（124）は、123、126、127、および 210 の情報を使用して、複合アプリケー
1717 ションのゾーン構成（225）に一貫性のある定義を提供する。複合アプリケーションの構成プロセス（124）では、組
1718 織（110）とサービス・プロバイダー双方の期待する内容が適用されるように、126 および 127 のポリシーを平準
1719 化する必要がある。ポリシーに対するこれらの期待内容を実現できない場合は、特定の複合アプリケーションの
1720 構成プロセス（124）が作成されず、組織（110）とサービス・プロバイダーの両方に対して例外の発生が表明され
1721 る。

1722 得られた複合アプリケーションのゾーン構成（225）は、サービス・プロバイダーが持つ 1 つ以上のインフラストラク
1723 チャー（255）にデプロイ（250）できる。このようなインフラストラクチャーとして、256、257、258、および 259 があ
1724 る。ここには「メディア」も含まれるが、図 B-1 では明記していない。

1725 デプロイ結果は監視され（310）、その経過がログに記録される（315）。SLO を通じた SLA の検証（320）および
1726 規制当局に対する運用上のコンプライアンスの検証（325）のプロセスが提供される。SLA の検証（320）およびコ
1727 ンプライアンスの検証（325）は、組織のデータセンター、クラウド・サービス・プロバイダー、またはそれらの組み
1728 合わせで運用できる。

1729 デプロイ可能な複合アプリケーションの構成を一貫性をもって定義する手順は、次の例のようになる。

1730 1) すべての 10 GB ネットワークの相互接続はプラチナ顧客のみが利用可能であることを、サービス・プ
1731 ロバイダーがポリシーの構成（127）で注記している。

2) ここで検討対象とする組織は、このサービス・プロバイダーから標準パフォーマンス・パッケージを購入した顧客である。標準パフォーマンス・パッケージの指定では、この組織が利用できる相互接続は 100 MB ネットワークおよび 1 GB ネットワークのものに限定されている。

3) 複合アプリケーションの構成を生成するとき、この組織で利用できるあらゆるゾーン構成から 10 GB ネットワークの相互接続がすべて削除される。このように、10 GB のネットワーク接続を備えたゾーンにこの組織が参加していても、その接続を利用することはできない。この組織がデプロイするインスタンスでは、10 GB のネットワーク・インターフェース・カードはどれも使用できないからである。

4) この組織がその顧客ステータスをプラチナ顧客のステータスにアップグレードすると、10 GB ネットワークの相互接続はゾーンの構成から除外されないので、どの複合アプリケーションでも、アプリケーション・テンプレートやインスタンスなどの変更を必要とせずに、より高い帯域幅を使用できるようになる。

5) 同様に、プラチナ顧客である契約期間が終了すると、顧客ステータスは元の標準ステータスに戻るので、高いネットワーク帯域幅は利用できなくなる。

6) 組織がその組織ポリシー（126）の中で、特定のサービスは 10 GB ネットワークに接続する必要があることを規定していることも考えられる。顧客の種類が「標準」のステータスであると、サービス・プロバイダーのポリシーと組織のポリシーとの間に矛盾が存在することになる。この矛盾は、その解決に向けてサービス・プロバイダーと組織の双方で認識される。

7) 複合アプリケーションの構成（124）の際に、組織とサービス・プロバイダーの間でポリシーが平準化され、複合アプリケーションのデプロイ（250）の際にその平準化が検証されるので、構成に基づくポリシーの解決（顧客の種類の変更など）によって所定のあらゆるデプロイで機能の追加と削除が可能である。

B.1.3 ポリシーの例

表 B-1 - は、ポリシーの種類を分野別に分類した例である。

表 B-1 - ポリシーの例

ポリシーの分野	例
ユーザーおよび ID	ID：連携 認証 認可
運用上の属性	アクセス制御 ユーザーのプロファイル 鍵/証明書の管理 パッチの適用 境界 仮想化のセキュリティー 監視、インシデント対応、および通知 セキュリティーで保護したワークロード移行
法令およびコンプライアンス	電子的証拠開示 データの管轄 データの保持および破棄 データの場所 プライバシー

ポリシーの分野	例
ネットワークおよび接続	境界ファイアウォール・テンプレート（ネットワーク境界のファイアウォール規則を組織で規定可能）: • アクション：許可/拒否 • TCP/UDP/IP プロトコル • 送信元の IP • 送信先の IP • ポート • ログ
	レイヤー2 の ACL テンプレート（ネットワーク層の ACL 規則を組織で規定可能）: • アクション：許可/拒否 • TCP/UDP/IP プロトコル • 送信元の IP • 送信先の IP • ポート
	IDS/IPS テンプレート（ネットワーク IDS/IPS 署名を組織で規定可能） • アクション検出/防止 • 署名 • 習熟期間
	ネットワークの分離（ネットワーク・リソースの物理的分離および論理的分離の実現）
	転送のセキュリティー：TLS、VPN
ストレージ	ブロック、ファイル、キュー、データベース、キー・ストアなどのすべての形態のストレージに、ストレージの規則、制約、およびポリシーを適用する。次に例を挙げる。 • ゾーン（組織のストレージに論理的分離を提供） • データ分類（「公開」または「機密」のデータ分類を組織に提供） • アクセス/暗号化（保存データ（Data at Rest）のアクセス制限または暗号化を組織で要求可能） • 分離 • 場所 • 保持（保持期間を組織で定義可能） • バックアップおよび復元（データのバックアップと復元のポリシーを組織で規定可能）
計算リソース	親和性 複合アプリケーションの制約 ウイルス保護 パッチの管理 優先順位 可用性（DoS 攻撃に対する耐性など） 場所および配置 マルチテナントでの分離 プラットフォーム（ハードウェア/ソフトウェア）の一貫性と保証 ポリシーの範囲（適用可能な部分）

ポリシーの分野	例
セキュリティー	DPI ファイアウォール規則（ネットワークとの関係）を使用することで、顧客はそれぞれの階層対階層ファイアウォール規則をサービス・プロバイダーのサーバー階層ファイアウォール規則に変換できる。 • アクション：許可/拒否 • プロトコル：TCP、UDP、ICMP、IP • 送信元の IP • 送信先の IP • ポート • ログ • 脅威の管理（ネットワークとの関係） • ログの管理（監査との関係） • 法令/規制の管理 • 分離（親和性との関係） • 監査
アラーム	しきい値 チケット処理
監視	イベント アラート しきい値
場所	特定の場所（法令/規制に関連） 可用性

Annex C
（参考情報）

変更履歴

バージョン	日付	説明
1.0.0	2010-06-18	DMTF 参考情報として公開