エンタープライズ・ドメインの独立性を重視した透過的なサービス連携の実現

中村 航一　石井 陽介　諸富 聡

Transparent service integration platform with special respect to enterprise domain independence

Koichi Nakamura, Yohsuke Ishii and Satoru Morotomi

グローバルに点在する拠点や関連企業などのドメインでは，全体で標準化されたガバナンス下で，ドメインをまたがったリアルタイム，かつ安全なシステム連携が必要となる．一方，各ドメイン内でIT管理や開発プロセスが最適化されているため，それらを活かし効率を維持しつつ，ドメイン外のシステムやサービスの管理を共存させることが要求される．著者らは，各ドメインの自由度や責任を最大限に維持し独立性を保持するため，ドメインをカプセル化し大部分は隠ぺいしつつ，その間をリアルタイム，かつシームレスに安全性を確保して連携するためのドメイン・ゲートウェイ・ソリューション・パターンを新規に考案し，お客様プロジェクトにて実装して，その有用性を実証した．これにより，システムの柔軟性や迅速性が向上することによる経営判断の精度向上に加え，効率的なシステムの集約や分散が可能になり，開発，維持・運用にかかる要員やコストが改善されるなど，全体最適も可能になる．

In enterprise domains, such as local offices or suppliers in regions, it would require standardized, secure, efficient, and real-time integration across the domains. On the other hand, each domain will have their own optimized IT management and development process which best suits their requirements, and it is required to establish connectivity with systems and services outside its domain, while such local optimizations are utilized to the maximum extent. As a solution to this scheme, the team has invented a new solution pattern named Domain Gateway Solution Pattern, which encapsulates each enterprise domain to allow for maximum independence, while enabling safe, real-time, and seamless integration. The Domain Gateway Solution Pattern enhances business decision reliability by providing additional system flexibility and agility, and also accelerates overall efficiency of system development, maintenance, and operations through flexible aggregation and distribution.

Key Words & Phrases：グローバル連携，フェデレーテッド ESB，ゲートウェイ・パターン，ガバナンス，セキュリティー，SOA
Global Integration, Federated ESB, Gateway Pattern, Governance, Security, SOA

1. はじめに

グローバルに点在する各地域の拠点や関連企業などのドメインには鮮度が高く経営判断に有用なデータが埋もれているが，ビジネスに十分に活用できていない［1］．これは広くビジネスを展開するエンタープライズにおいて，経営判断の迅速化や精度向上の観点で重要な問題である．ドメインに埋もれているデータを活用するためには，ドメインをまたがったシステム連携を実現する必要があるが，おのおのに異なる方針でシステムの構築・運用が最適化されている“ドメインの独自性”と，ドメイン間で接続ルールを標準化した方法でシステムを連携する“ドメイン間の接続性”の相反する要件を両立しなければならない．これまで，複数のドメイン間でシステムを連携する場合，Federation パターンを採用し，中央で一元管理する方式が採用されてきた［2］．この方式ではトップダウン的に各ドメイン内の細部まで標準化する必要があるため，異なる方針で構築されたシステムを連携させることが困難であった．

そこで，SOA（Service Oriented Architecture）における ESB Federation という考え方をベースに，接続性・サービス管理・セキュリティー管理の観点で各ドメインをカプセル化し，標準化された接続ルールを決めてゲートウェイ［3］［4］を介して連携していくソリューション・パターン（Domain Gateway Solution Pattern（以下,Domain G/W パターン））を考案した．この方法であれば，カプセル化した中は，各ドメインの自由に任せることができるため“ドメインの独自性”は保たれ，ドメイン間で公開し連携する部分だけを全体で標準化して連携する“ドメイン間の接続性”も実現できる．本論文では，あるお客様プロジェクトでの Domain G/W パターンの実装をベースに，その重要性，新規性，有用性，可能性を論ずる．

提出日:2011年9月20日　再提出日:2012年3月14日

2. グローバルSOA基盤システムの構築

2.1 プロジェクト概要

ビジネスの俊敏性と柔軟性の向上のため，点在する拠点間でリアルタイムにアプリケーションを連携する SOA 基盤を構

築した. この基盤では, “ドメインの独自性” を維持しながら, ドメイン間の連携において一貫性のあるサービス連携, サービス情報管理, セキュリティー管理をそれぞれフェデレーテッドESB, フェデレーテッド・サービス・レジストリー, フェデレーテッド・セキュリティーにより実現している.

2.2 ビジネス要件

お客様からの明示的な要件を以下に示す.

- ビジネス・ニーズ
  - ▶コスト削減と市場拡大への迅速な対応
  - ▶鮮度が高いデータを活用した業務効率化
  - ▶ Globally Integrated Enterprise（GIE）の実現
- システム要件
  - ▶世界中の拠点をつなぐシステム基盤の構築
  - ▶グローバル・レベルでの情報連携
    - サービスの連携・再利用を促進
    - 独立的で安全なセキュリティーの実現
  - ▶各拠点の主体性を重視したサービス管理
    - 各拠点の管理責任の明確な分担

ただし, これら要件の裏に隠れて, お客様の企業文化に根付いた特別な条件が存在していた.

- 企業文化に根付いた暗黙的な条件
  - ▶各拠点は完全に独立しており, 対等の連携が必要
  - ▶現場主導が深く根付いており, トップダウン的な展開には激しい抵抗があり不可
  - ▶全体最適のために, 各拠点の業務やプロセスに非効率を強いることは不可

これら暗黙的な条件に気付かず（あるいは無視して）, 明示的な要件だけを満たすよう構築してしまうと, まったく使われないシステムになってしまう. 実際, 全拠点を統一したやり方で連携するという試みは過去に失敗していた.

ビジネス・ニーズに応えるために, 明示的なシステム要件だけではなく, 暗黙的な条件も満たすシステムを設計・構築していくという試行錯誤の中で Domain G/W パターンを考案した.

3. Domain G/Wパターン

3.1 コンセプト

拠点間をつなぐため, 各アプリケーションで個々に連携させると, 以下のようなさまざまな弊害が出てしまう.

- サービスが, ドメイン内にあるか, ドメイン外にあるかを意識して連携する必要がある.
- 拠点数を N とすると, N*(N-1) マッピングが発生してしまい連携数が爆発する.
- 連携ごとにルーティングや安全性の確保が必要になる.

結局, 接続性やセキュリティー管理などを標準化することになり,各拠点（ドメイン）の独自性の維持が困難になる（図 1）.

そこで, 各ドメインに ESB による Gateway（以下, G/W）を配置し, これを通して連携することで, 上記弊害を回避し以下のメリットを出すことができると考えた（図 2）.

- 各アプリケーションは, サービスの存在がドメイン内か外かを意識せず連携する（両ケースとも ESB に接続）.
- 各ドメインをできる限りカプセル化し内部を隠ぺいすることができる（出入り口を極力絞る）.
  - ▶ドメイン間連携部分のみ標準化
  - ▶各ドメインで, 全体標準とドメイン独自仕様の間の違いを変換して吸収
- 各ドメインの管理責任範囲を明確化できる.

この連携を Domain G/W パターンとし, お客様プロジェクトでその妥当性を検証した. 以下に 3 つの主要コンポーネントの機能概要を説明し, 図 3 に Logical Architecture を示す. おのおののコンポーネントは, 3.3 より詳細に説明する.

- フェデレーテッド ESB：ESB のフェデレーション（連合）により, ドメイン間でシームレスなアプリケーション連携を実現
- フェデレーテッド・サービス・レジストリー：ドメイン間でサービス再利用のための情報管理を実現
- フェデレーテッド・セキュリティー：安全なドメイン間情報連携（ドメイン間の認証, 認可）を実現

3.2 Domain G/W パターンの新規性

従来の G/W パターンは, ドメイン間のサービス連携にのみ注目し実装されてきた. Domain G/W パターンでは, ドメイン間のサービス連携に加え, サービス情報管理, 認証・認可も実装できる点が革新的である. 企業が必要とする IT ガバナンスやセキュリティーの要件にも対応できるアーキテクチャーとして, より実践的といえる.

さらに, Domain G/W パターンによる連携を実現することで, ドメインの独立性を重視し, 各ドメインの局所最適化を保持しつつ, ドメイン間を対等な関係でシームレスかつ安全にリアルタ

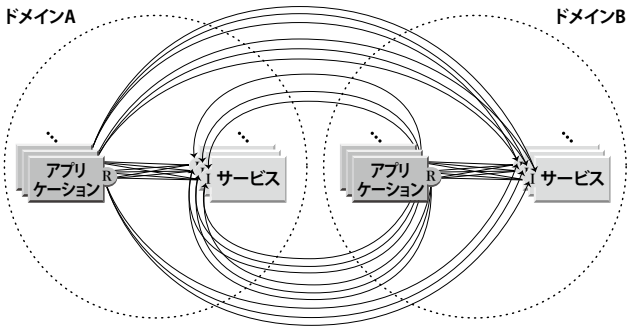

図1. アプリケーションが個々に連携する場合

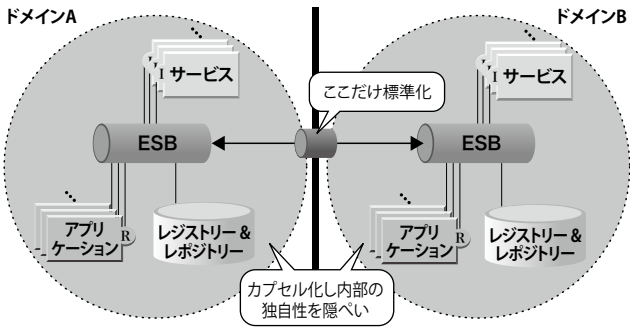

図2. ESBによるG/Wを通して連携する場合

イム連携することができる.

3.3 フェデレーテッド ESB

Domain G/W パターンでは，各ドメインの ESB による G/W を経由してシームレスな連携を行うフェデレーテッド ESB を実装した．それには，連携のためのルーティングを正確に行う必要がある．たとえば，呼び出し側のドメイン（図 3 のドメイン A）の G/W では，呼び出すサービスが同じドメイン内にある場合は，そのサービス自体にルーティングするが，他のドメイン（図 3 のドメイン B）内の場合は，正確にそのドメインの G/W にルーティングする必要がある．そのような連携のためのルーティング情報（以下，Endpoint）を管理する必要があるが，ESB 内に静的に定義してしまうと，変化に弱いシステムになってしまうため，サービス・レジストリーに外出し管理し，問い合わせによる動的ルーティングを行うようにした．また，連携のためのセキュリティー操作（認証，認可，Security Token の変換（詳細は 3.4 を参照））も同様にアクセス／アイデンティティー管理に外出しし，より変化に強いシステムにした．

このように外出しした機能を G/W から呼び出すことで，連携に必要な操作を一元管理することができるため，ドメインの独自性を最大限確保したシームレスな連携が可能になる．Domain G/W パターンの重要なアイデアの 1 つである．

3.4 フェデレーテッド・サービス・レジストリー

Domain G/W パターンでは，各ドメインに1つの管理用ガバナンス・サービス・レジストリー（以下，ガバナンス・レジストリー）と，開発／テスト／本番環境に 1 つずつ計 3 つの実行時参照用ランタイム・サービス・レジストリー（以下，ランタイム・レジストリー）を設置して，ドメイン間のサービス情報連携はガバナンス・レジストリー間で公開を行い，それぞれのランタイム・レジストリーへの配信は各ドメイン内のガバナンス・レジストリーから行う方式を考案した点がユニークで重要なポイントになる．サービス情報公開時に Endpoint 情報を動的に変換する仕組みも考案しフェデレーテッド ESB を実現した．

エンタープライズのお客様では，アプリケーションやサービスの開発／検証／本番稼働といったそれぞれの段階で使用する環境として，開発／テスト／本番という複数のシステム環境が存在する．各環境に，その環境の G/W 用 ESB から外出しした Endpoint を保持し，動的ルーティングの問い合わせに対応するランタイム・レジストリーを配置する（図 4）．

正確なルーティングのためには，各環境のランタイム・レジストリーに必要な Endpoint を必要な時にコピーする必要があるが，それを実施するためガバナンス・レジストリーを使用する．ガバナンス・レジストリーでは，各サービスのライフサイクル（例：提案→承認→開発→テスト→本番稼働→リタイア）の管理を行い，必要な時に必要な情報を必要な環境へ配信する（Promotion 機能:図4内の各プロモーション）．例えば，提案したサービスが承認されて開発フェーズに遷移する時には開発環境に開発用の Endpoint を，そして開発からテストへフェーズの遷移時にはテスト環境にテスト用の Endpoint を Promotion する．これらにより，各環境で Endpoint 問い合わせが有効になり動的ルーティングが機能し始める．

Domain G/W パターンでは，各ドメインにそのドメイン用のガバナンス・レジストリーを 1 つ配置するトポロジーを考えた．一般的には，ガバナンス・レジストリーはシステム全体で 1 つ配置し，すべてのサービス情報を一元的に管理する．例え

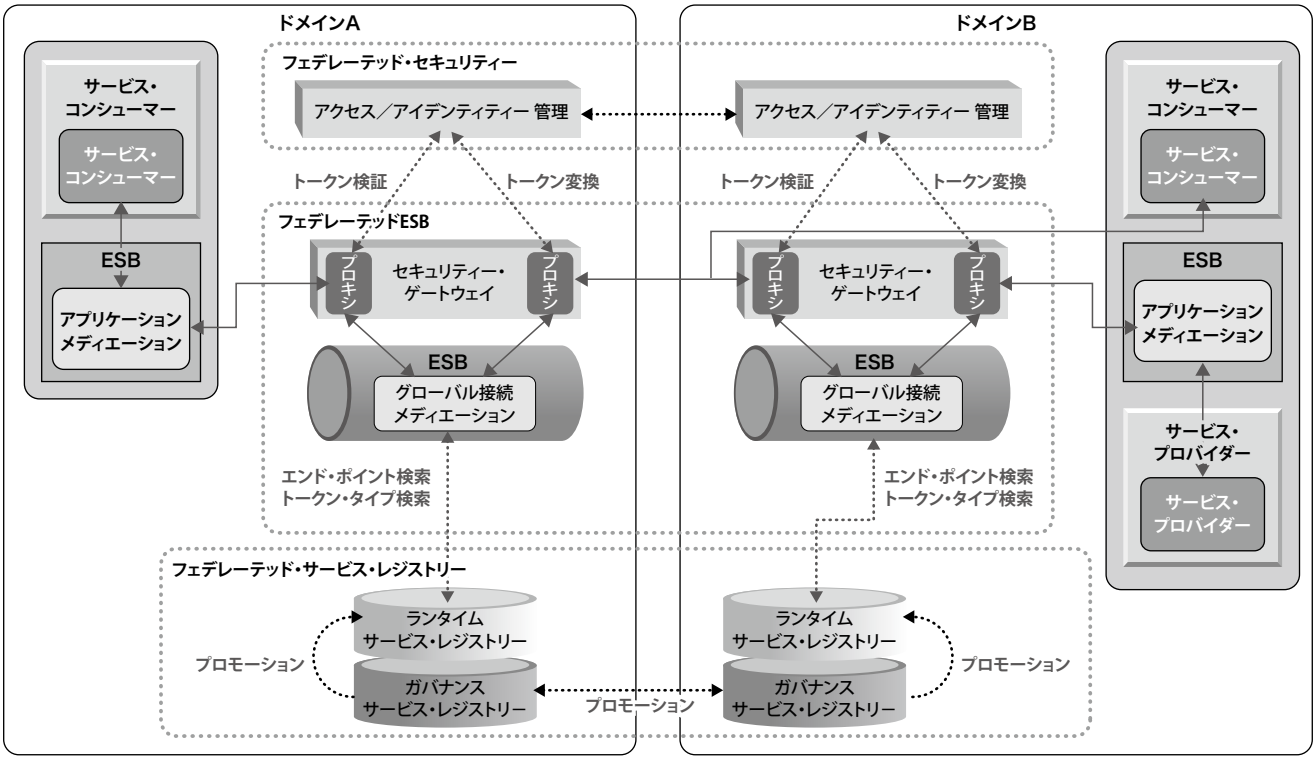

図3. Logical Architecture

ば，図4のドメイン B にガバナンス・レジストリーを配置し，ドメイン B だけでなくドメイン A のランタイム・レジストリーへの Promotion もそれで管理する．しかし，これでは各ドメインの自由度は低くなってしまい，責任の範囲がはっきりしなくなる（ドメイン A のランタイム・レジストリーで問題が発生した場合，ドメイン A 内の問題なのか，ドメイン B からの Promotion が影響したのかはっきりしない，など）．各ドメインに配置されたガバナンス・レジストリー間でサービス情報を公開し（図4の“拠点間のプロモーション”），各ドメインでライフサイクル管理する Domain G/W パターンでは，以下のように問題点を解決することができる．

- 各ドメインの責任範囲を明確にする．
  - ▶連携の出入り口をガバナンス・レジストリーに絞り，ランタイム・レジストリーを他のドメインからは触らせない．
  - ▶Endpoint情報は,各ドメインの責任でガバナンス･レジストリーからランタイム･レジストリーへPromotionする．

各ドメインのガバナンス・レジストリー間でサービス情報の公開を行うにあたり，“各ドメインで管理される Endpoint は，同じサービスのルーティング情報であっても異なる情報になる”，という点を考慮しなければならない．例えば，図3のドメイン B がサービスを提供し，ドメイン A から呼び出しを行う場合を考える．ドメイン B のサービス連携（実行時）のための唯一の入り口は G/W であるため（詳細は 3.2 を参照），ドメイン A からのリクエストはドメイン B の G/W にルーティングされ，その後ドメイン B の中でサービスにルーティングされる．そのため，ドメイン B で管理される Endpoint はサービスそのものをポイントする必要があるが，ドメイン A で管理される Endpoint は，サービスではなくドメイン B の G/W をポイントする必要がある．それを解決するため Domain G/W パターンでは，ドメインのガバナンス・レジストリー間のサービス情報公開において，ルーティング情報の単なる Promotion ではなく，必要な Endpoint の自動置換を行い Promotion する手段を考案し実現した（知的 Promotion）．先の例では，ドメイン B からドメイン A へのサービス情報公開において，サービスそのものをポイントしている Endpoint を，ドメイン B の唯一の入り口となる G/W をポイントする Endpoint に自動置換して Promotion する．

3.5 フェデレーテッド・セキュリティー

Domain G/W パターンでは，オープン・スタンダードの WS-Security に基づき Security Token を SOAP ヘッダーに組み込んでサービスを連携する．ここで各ドメイン独自の Security Token と標準化された Security Token を G/W 上で変換する方式を考え，ドメインの独自性を保持したまま安全な連携を行うための認証・認可を実現した．

一般的には，全ドメイン間でシームレスに安全なサービス連携を行うには，標準化した同一の Security Token による連携が必要になる．しかし，各ドメインの自由度を上げるためには，ドメイン内でその標準化した Security Token の使用を強要することはできない．そこで Domain G/W パターンでは，ドメイン間の Security Token のみ標準化し（ex. SAML（Security Assertion Markup Language）[5]），送受信する双方のドメインの G/W でその標準と各ドメインの独自仕様（ex. Username Token, LTPA（Lightweight Third Party Authentication）[6]，etc）の間の違いを変換し連携する方法を考案した．変化に強くするため，セキュリティー機能はアクセス／アイデンティティー管理に外出しし,G/W から WS-Trust（オープン･スタンダード）を使用して呼び出す．

図５に一例を示す．ドメイン間の標準として SAML（認証

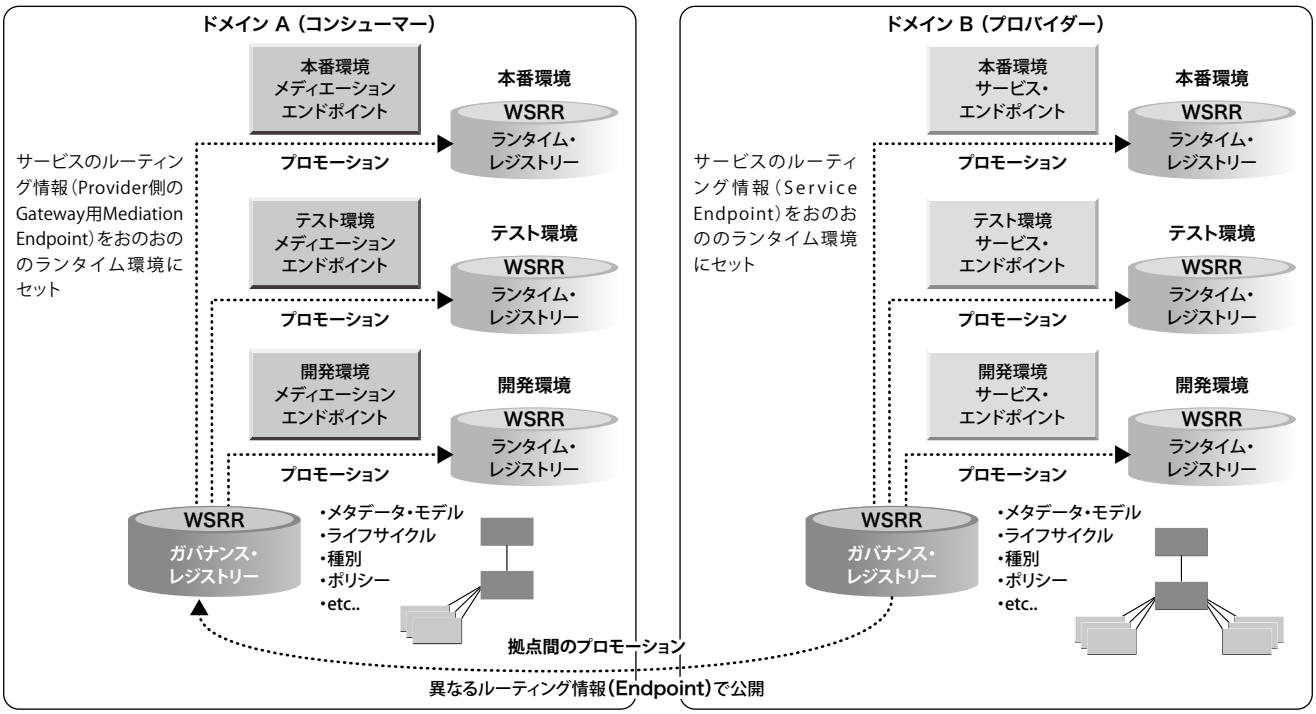

図4. サービス情報管理

のため Signed SAML: 認証については後述）が定義され，ドメイン A とドメイン B の独自仕様がそれぞれ Username Token とLTPA Token である場合を考える.ドメイン間の連携において，ドメイン A の G/W では Username Token から SAML へ変換されリクエストがルーティングされる．リクエストを受信したドメイン B の G/W では，SAML から独自仕様の LTPA Token に変換されサービスが呼び出される.

安全性の確保のため，ドメイン連携における認証は必須となる．Domain G/W パターンでは，以下の認証を考慮している.

- 各ドメインを認証：個々のユーザーは各ドメイン内で認証済みという前提のもと，信頼するドメインからのリクエストであることを認証.
  ▶ドメイン間で Certification のみを交換
  ▶電子署名した Security Token で連携し認証
- 各ユーザーを認証：認証されていないユーザーからのリクエストも想定し，すべてのリクエストでユーザーを認証.
  ▶ドメイン間でユーザー情報を共有

図5の例はドメインを認証するケースで，ドメイン A の Certification で Sign した SAML を送信し，ドメイン B でそれを認証している.

G/W での認可の使用は，アプリケーションが選択できるようにした．現在 Domain G/W パターンで提供できる認可は，WS-Trust や製品の制限により,サービス･レベルの認可である．オペレーション・レベルの認可は開発チームと検討中であるが，データ・レベルのような各アプリケーションに特化した細かいアクセス制御は，個々のアプリケーションで対応すべきと考える.

以上により，カプセル化したドメイン内部の独自仕様に影響を与えることなく，ドメイン間の安全な連携が可能になり，条件と仕様を満たすことができる.

4. Domain G/Wパターンの実装と検証

4.1 グローバル SOA 基盤システムの実装

グローバルにビジネスを展開しているお客様プロジェクトで Domain G/W パターンを実装し，要件を満たして安定稼働を続けていることで，その重要性，有用性を実環境にて証明することができた.その実装の要点と検証した結果について示す.

お客様プロジェクトでの実装は以下になる.

- 連携するドメイン：2 つのドメイン（拠点）間連携を対象
- ドメイン間の標準：SAML をドメイン間の標準 Security Token とし，ドメインの独自仕様（両拠点共に Username Token）との変換を行い連携
- 認証：各ドメインで個々のユーザーの認証は実行されるため，信頼するドメインからのリクエストかどうかを認証（各ドメインを認証）
- 認可：サービス・レベルの認可を G/W で制御

図３（Logical Architecture）に示した各コンポーネントは，以下の製品で実現した.

- フェデレーテッド ESB：
  ▶ WebSphere Enterprise Service Bus
  ▶ WebSphere DataPower
- フェデレーテッド・サービス・レジストリー：
  ▶ WebSphere Service Registry and Repository
- フェデレーテッド・セキュリティー：
  ▶ Tivoli Federated Identity Manager
  ▶ Tivoli Access Manager for e-business

4.2 Domain G/W パターンの検証（有用性／有効性）

Domain G/W パターンが，2.2 で説明したお客様要件にいかに応えているかを説明する．本当に使ってもらえるシステムにするため，暗黙的な条件を満たしシステム要件に応えること

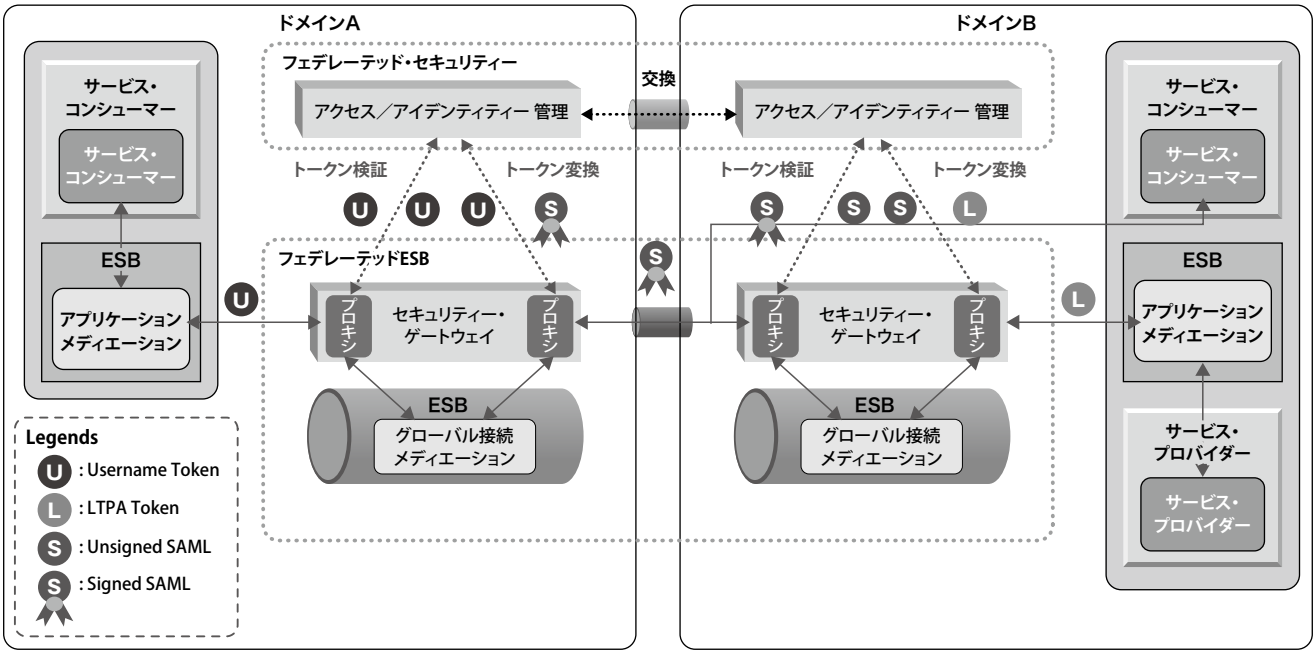

図5. セキュリティー管理

が重要なポイントになる.

フェデレーテッド ESB とフェデレーテッド・セキュリティーを実装することにより，各ドメインをカプセル化してその中は自由に任せた連携を可能にし，現場主導で独立した各拠点の業務やプロセスへの影響を最小限にして，各拠点対等につなぐという条件を満たすと同時に，ドメイン間を安全にリアルタイム連携するというシステム要件を実現することができた．さらに，フェデレーテッド・サービス・レジストリーと知的 Promotion の実現により，おのおののドメインにてランタイム・レジストリーの管理が行えるようになるため，各拠点の管理責任の明確化と主体性を重視したグローバル標準のサービス管理というシステム要件に応えることができた.

各ビジネス・ニーズには，以下のように応えた.

- コスト削減と市場拡大への迅速な対応：大幅な改修を伴わずにドメイン間連携が可能になり，再利用性が高まることによるコスト削減や新規ドメイン（拠点）の迅速な連携が実現可能となった．さらに，各ドメインが透過的に連携できるようになり，散在していた機能の集約や災害対策のための分散など，全体最適化によるコスト削減が可能となった.
- 鮮度が高いデータを活用した業務効率化：既存のバッチ・ファイル転送や DB 共有に加え，リアルタイム連携が可能となった.
- Globally Integrated Enterprise（GIE）の実現：地域のビジネスの独立性を持ったグローバルビジネスの実現に貢献した.

4.3 Domain G/W パターンのパフォーマンス

Domain G/W パターンは，サービスの直接呼出しと比較すると，G/W を経由する動的ルーティングや標準と独自仕様の変換などのオーバーヘッドがかかるため，パフォーマンス劣化が大きな懸念点であった．第一フェーズでは，本基盤を通して連携するアプリケーションが既に拠点間で直接呼出しにより本番稼働していたため，大幅なパフォーマンス劣化は許されなかった．プロジェクトでは早い段階から本番環境と同等の環境を用意し，パフォーマンス検証を実施した．そのアプリケーションで一般的なリクエスト（50K byte 程度）で検証したところ，直接呼び出しでは平均応答時間が約 0.6 秒に対し，本基盤を通しても約 0.7 秒となり，問題ないことが確認できた．大幅な劣化を防ぐことができたのは，採用製品のキャッシング機能（特に動的ルーティングのための Endpoint 検索結果を一定時間キャッシングする機能）の効果が大きい.

4.4 お客様の評価

お客様では，企業文化として現場主導が深く根付いているため，期待に反して柔軟性・迅速性向上や全体的な最適化にはつながっていなかった．Domain G/W パターンを採用することで，各拠点の自主性とグローバルな接続性を両立でき，効率よく最適につながることに対するお客様満足度は非常に高く，第一フェーズを成功裏にサービス・インできた．今後グローバル基幹システムを支えるインフラ基盤として，連携拠点やアプリケーションを追加して拡張する予定である.

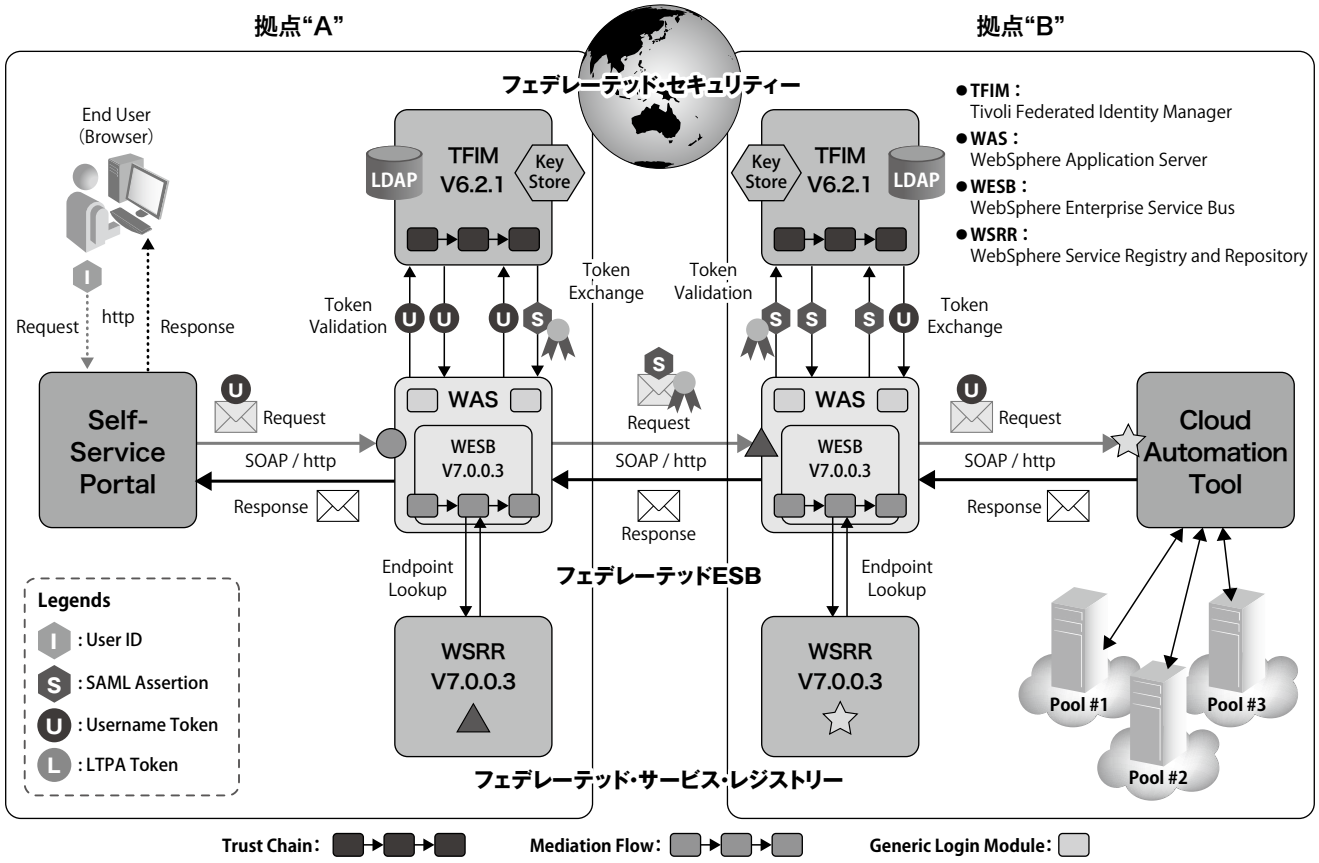

図6. プライベート・クラウドへの適用イメージ

5. Domain G/Wパターンの可能性

5.1 エンタープライズでの再利用性

このお客様に限らず，事業形態が複雑化するエンタープライズでは，Line of Business（LOB）や地域，関連会社ごとに特性が異なるため，各ドメインの独自性を維持したままでの連携が重要視される．このような要件において，Domain G/W パターンは非常に重要な考えであり，幅広くグローバル企業のコア・アーキテクチャーとして再利用可能である．実際，Domain G/W パターンは他のお客様により採用の検討が行われている．今後さまざまな分野でのドメイン連携のソリューション展開として，適用範囲を広げることが大いに期待できる．

5.2 クラウド・システムへの適用

普及が進むクラウド・コンピューティングにも Domain G/W パターンは適用できる．大規模なクラウド環境をグローバルに構築するケースでは，リソース・プールを Web API で制御する方式が一般的である［7］．例えば，日本から米国，欧州，アジアなどの拠点にあるリソース・プールを制御するために Web API を呼び出して仮想マシンの作成や削除を行う．Web API を認証・認可などセキュリティーに配慮して呼び出す仕組みに加え，仮想マシンの監視や課金情報を交換するための連携基盤を Domain G/W パターンを適用して構築することができる．

図 6 に，Domain G/W パターンのプライベート・クラウドへの適用イメージを示す．

6. おわりに

事業形態が複雑化するエンタープライズにおいて，LOB や地域・関連会社など特性が異なるドメインの独自性を維持したままでの連携を実現することは大きな課題であったが，Domain G/W パターンを考案し，それを解決した．本論文では，お客様プロジェクトでの実装をベースに，その重要性，新規性，有用性，可能性を述べた．Domain G/W パターンは，特定の市場や分野に特化したものではなく，あらゆる業界に適用可能であるため，社会へのインパクトは大きく，幅広い採用が見込まれる．

謝辞

Domain G/W パターンの実装をサポート頂きました，お客様プロジェクト・チームの皆さまや WAS, TFIM 開発チームの皆さまに深謝いたします．

参考文献

［1］ “IT 人材白書 2011” IPA（2011）．
［2］ IBM developerWorks: “Guidelines for choosing an ESB” http://www.ibm.com/developerworks/jp/websphere/library/event/impact2007/2.html（2007）．
［3］ IBM Redbooks: “Patterns: Integrating Enterprise Service Buses in a Service-Oriented Architecture” http://www.redbooks.ibm.com/abstracts/sg246773.html（2005）．
［4］ IBM developerWorks: “ESB and Connectivity Patterns” http://www.ibm.com/developerworks/wikis/display/esbpatterns/ESB+and+Connectivity+Patterns（2009）．
［5］ Security Assertion Markup Language（SAML）V1.1 http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SAMLTokenProfile.pdf
［6］ Lightweight Third Party Authentication（LTPA）．http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp?topic=%2Fveicinfo%2Feicarconcept_security_ltpa.htm
［7］ IBM developerWorks: “Automate development and management of cloud virtual machines” http://www.ibm.com/developerworks/cloud/library/cl-automatecloud/（2011）．

日本アイ・ビー・エム株式会社
ソフトウェア事業．WebSphere 事業部
シニア・マネージング・コンサルタント

中村 航一 Koichi Nakamura

［プロフィール］
1991 年，日本 IBM 入社．大和ソフトウェア開発研究所にて，ホスト端末エミュレーターの開発，リアルタイム・コラボレーション製品の開発，WebSphere 製品の開発及びサービス・コンサルタントに従事．2011 年，ソフトウェア事業に移り，WebSphere 技術営業に従事．
jl25377@jp.ibm.com

日本アイ・ビー・エム株式会社
ソフトウェア事業．WebSphere 事業部
第二クライアント・テクニカル・プロフェッショナルズ
IT スペシャリスト

石井 陽介 Yohsuke Ishii

［プロフィール］
2003 年，日本 IBM 入社．アプリケーション開発支援サービス事業にて，主に大手製造系のお客様を中心に Java 開発の標準化やグローバル SOA 展開のコンサルティングおよび導入・運用支援を担当．2011 年，ソフトウェア事業に移り，WebSphere 技術営業に従事．
yoishii@jp.ibm.com

日本アイ・ビー・エム株式会社
システム製品事業．テクニカル・セールス
アドバイザリー・アーキテクト

諸富 聡 Satoru Morotomi

［プロフィール］
1993 年，日本 IBM 入社．2003 年よりアーキテクトとして中央省庁，製造業のお客様を中心に担当し，連携基盤システム，統合運用管理システムなど IT インフラの設計および構築を幅広く経験．現在はシステム製品事業にてプライベート・クラウドのテクニカル・セールスを担当．
morotom@jp.ibm.com