クライアントサーバー型アプリケーションのサービス化

Building Client-Server Application as a Service

曽 根　信 一 郎

要　約　クライアントサーバー型アプリケーションをサービス化して，iDC 上からネットワークを通じて提供するニーズが高まっている．サービス化する際には，利便性の向上，コストの低減などのサービス化要件を満たすために，「アプリケーションの仮想化」や，「サーバーのマルチテナント化」を検討する．仮想化に当たっては，サービス化するアプリケーションとの適合性，その起動と認証などについて考慮し，マルチテナント化に当たっては，テナントの分離方法やパフォーマンスと可用性，セキュリティなどを考慮しなければならない．その結果，仮想化やマルチテナント化が行えない場合は，それを前提としたシステム構成を検討する必要がある．事例では，複数の考慮点からの具体的な検討結果を示した．

Abstract　We have a need for providing the client-server type application as a kind of service over the network from the iDC (Internet data center). When building those applications as network-accessible services, it is necessary for us to consider “the application virtualization” and “server utilization through multitenancy” to meet requirements for such as the improvement of convenience, and the cost reduction. As for the application virtualization, we need to consider the fitness for the application turned into the accessible service, including its start-up and authentication, for the multitenancy, we need to consider how to isolate individual tenants each other, as well as performance, availability, security, and etc. As a result, if it turns out that both virtualization and multitenancy can not be applied to the situation, we have to examine the system constitution assuming inapplicability. In the example, I show the concrete result of the study from the plural viewpoints.

1. は　じ　め　に

インターネットを経由してハードウェアやソフトウェアなどのコンピュータリソースを利用するクラウドコンピューティングは，リソースを自社で所有するよりも，初期投資や運用・保守のコストを抑えることができる．クラウドコンピューティングでアプリケーションを提供することを SaaS（Software as a Service）という[1]．SaaS で提供されるアプリケーションは，一般には Web アプリケーションとして提供することが多い[2]．

一方で，従来からワークフローなど一般の業務アプリケーションや，CAD ソフト，物流ソフトなどの特定業務向けアプリケーションは，クライアントサーバー型アーキテクチャで構築されている．これらの既存アプリケーションには，優れたものも多く，Web 化せずクライアントサーバー型のまま提供することで，低コストで早期にサービスを立ち上げるニーズが高まっている．

本稿では，クライアントサーバー型アプリケーションをサービス化する際のさまざまな検討事項と考慮点について述べる．まず，2 章でアプリケーションのサービス化要件と，それを満たす技術である「アプリケーションの仮想化」と「サーバーのマルチテナント化」について述べ，

これらの技術を用いたシステム構成について説明する．次に，3章と4章ではこれらの技術を適用する際に考慮すべき点を記述する．最後に5章では事例を通じて具体的な検討結果を示す．

2. サービス要件を満たす技術およびシステム構成

本章では，サービス化の要件とそれを満たす技術について述べ，その技術を適用した場合の効果とシステム構成を示す．

2.1 アプリケーションのサービス化要件

アプリケーションをサービスとして提供する場合，サービスのレベル，システム構成などさまざまな要件を満たさなければならない．

主な要件には次のものがある．

1） 利用者の利便性の確保
2） サーバー台数削減による運用コストの低減
3） データセンター（iDC）からのネットワークを介したサービス提供
4） 複数顧客へのサービスの同時提供
5） 利用者端末とiDC間，およびiDC内部のセキュリティの確保
6） パフォーマンスや可用性などの品質の確保

また，クライアントサーバー型アプリケーションのサービス化では，納期とコストを考慮して，アプリケーションには修正を行わないという要件が加わる．

2.2 サービス化要件に応える技術

サービス化要件に応えるためには「アプリケーション仮想化」技術を利用し，「サーバーのマルチテナント化」を行う．それぞれの概要と，要件に対する効果について述べる．

2.2.1 アプリケーション仮想化

アプリケーション仮想化（以降，仮想化）とは，利用者端末やOSからアプリケーションを切り離して仮想化し，管理，運用できるようにする技術である．表1に，代表的な仮想化ミドルウェア製品と2種類ある仮想化の実装方式との対応をまとめた．

一つめの実装方式は，シンクライアント方式である．「サーバーベースドコンピューティング」とも呼ばれ，クライアントプログラムをAPサーバー上で稼働させ，そのGUIを利用者端末のビューアに表示する．利用者はこのGUIを通じてサーバー上のクライアントプログラムを遠隔操作することができる．このときのサーバーとソフトウェアの構成を図1に示す．

表1 仮想化製品と実装方式

製品名	シンクライアント方式	ストリーミング方式
Citrix XenApp™（旧Metaframe）	○	○
Microsoft® Application Virtualization	○	○
GO-Global®（米国GraphOn社）	○	×
Symantec™ Workspace Streaming	×	○

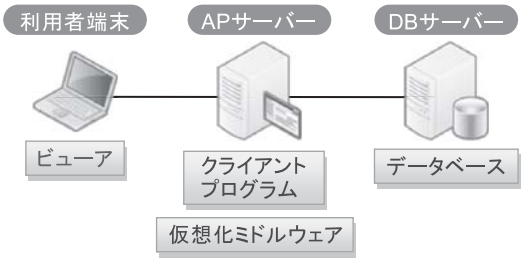

図 1　仮想化適用時のサーバーとソフトウェア構成

二つめの実装方式であるストリーミング方式は，クライアントプログラムを実行時にサーバーから配信し，利用者端末上で実行する方式である．この方式は 2006 年頃に登場した比較的新しい技術である．以降では，利用実績の多いシンクライアント方式について述べることとする．

仮想化を適用することにより，サービスの利用者や管理者，運用者にとって利便性が高くなる．利用者はアプリケーションを端末にインストールする必要がないため，利用者端末の OS とアプリケーションの互換性や，既にインストールされているアプリケーションの影響などの心配が軽減される．また運用者は，バージョンアップやパッチの適用，障害時のログの回収などをすべてサーバー側で行うことができる．

2. 2. 2　サーバーのマルチテナント化

サーバーのマルチテナント化とは，1 台のサーバーを複数の顧客企業（テナント）で利用できるようにすることである．クライアントサーバー型アプリケーションをサービス化する際には，AP サーバーと DB サーバーのそれぞれに対してマルチテナント化を検討する必要がある．検討した結果，図 2 で示す 3 種類となった．①はマルチテナント化を行わず，AP サーバーと DB サーバーはシングルテナントとなる．②は DB サーバーのみマルチテナント化，③は AP サーバーと DB サーバーを両方ともマルチテナント化した構成である．

マルチテナント化のメリットは，サーバー台数を減らし，運用コストを低減することである．サービスを提供する際に必要となるサーバー台数を比較すると，AP サーバーと DB サーバー

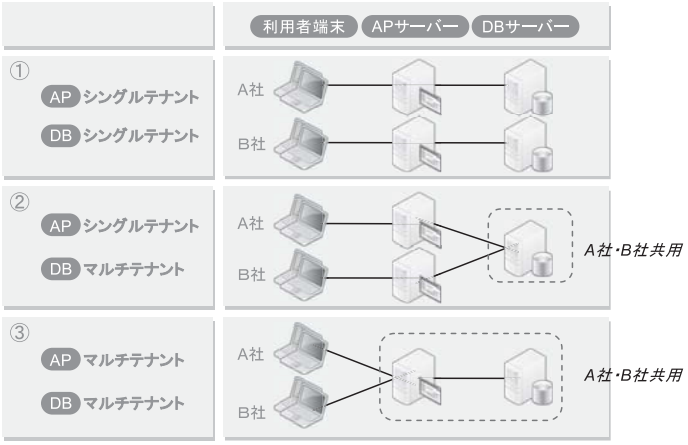

図 2　AP サーバー，DB サーバーのマルチテナント化

を両方ともマルチテナント化した場合（図2の③）に最も台数が少なくなるが，その反面，テナント間を意識して運用設計，セキュリティ設計を行う必要があり，サービス化する際の難易度が非常に高くなる．

2.3　サービス化要件を満たす最適なシステム構成

クライアントサーバー型アプリケーションをサービス化する際に，サービス化要件を満たす最適なシステム構成を図3に示す．この構成では，仮想化の技術を適用した．利用者端末にクライアントプログラムを導入することなくアプリケーションを利用することができるので，利用者の利便性が向上している．また，iDCに配置するAPサーバーとDBサーバーは共にマルチテナント化しているため，全体のサーバー台数が減り，運用コストを抑えることができる．利用者端末とiDC間の接続は，インターネット上で情報を暗号化して送受信するプロトコルであるSSLを利用して，セキュアな接続を確保している．

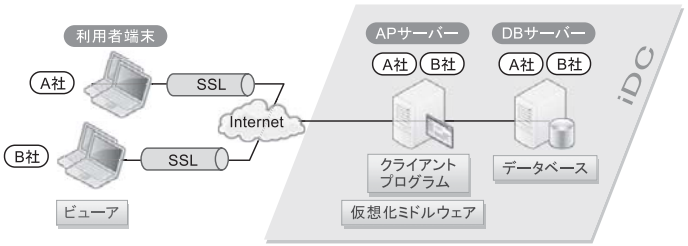

図3　サービス化要件を満たす最適なシステム構成

3.　仮想化における考慮点

本章では，仮想化を行う上で注意すべき点を述べると共に，アプリケーションの起動と認証方法の検討，およびセキュリティを考慮したネットワーク構成について記述する．

3.1　仮想化と対象アプリケーションの適合性

サービス化するアプリケーションは，その特性によって仮想化できない場合がある．仮想化の可否を判断するときに必要となる四つのポイントについて述べる．

1）　複数プロセスの起動

複数の利用者が同時にアプリケーションを利用する場合，APサーバー上では複数のクライアントプログラムが同時稼働する．このため，仮想化するアプリケーションは複数プロセスが起動した場合でも問題がないことを事前に調査する必要がある．

2）　ファイル出力の競合

複数プロセスの同時起動が可能であっても，ログファイルのようなクライアントプログラムが書き込むファイルのファイル名が固定され，さらにロック処理も行っていない場合，ファイル出力の競合が発生する．このとき，後からファイルを開こうとしたプロセスでは，プログラムがエラーとなってしまう．このケースは，アプリケーションを修正することなく仮想化を適用することは難しい．

3）　動作環境

通常APサーバーのOSはWindows Serverなどのサーバー OSとなることが多い．しかし，クライアントプログラムはWindows　XPなどのクライアントOS上でのみ動作保障されていることが多い．そのため，クライアントプログラムがサーバー OS上で動作保障されているかを，事前に確認する必要がある．

4）　リソースのパフォーマンスへの影響

複数の利用者が同時にアプリケーションを利用する場合，APサーバー上では複数のクライアントプログラムが同時稼働する．クライアントプログラムの種類によってはリソースを大量に消費したり，仮想化ミドルウェアの制限により，CPUやメモリなどのパフォーマンスが低下したりすることが想定される．そのような場合には，十分なパフォーマンス検証を実施し，サーバーのスケールアップや利用者の同時アクセス数の制限などを検討する必要がある．

以上の四つのポイントを検討した結果，仮想化を適用することができないと判断した場合には，図4のようにクライアントプログラムを各利用者端末にインストールし，利用者端末とiDCのDBサーバーが直接通信するシステム構成となる．

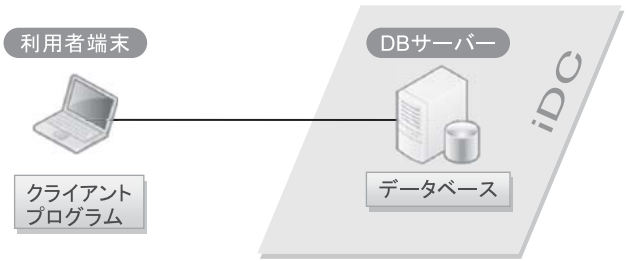

図4　仮想化を適用しないシステム構成

3.2　アプリケーションの起動と認証

仮想化した環境でクライアントプログラムを起動するには，APサーバー上の仮想化ミドルウェアに直接接続して起動する方法と，公開されたWebページ経由で仮想化ミドルウェアに接続して起動する方法がある．前者は接続に専用のプログラムを必要とするのに対し，後者は一般的なブラウザを使って接続ができるため，利用者にとって利便性が高い．また，後者の方法ではWebページの閲覧を利用者に限定することで，利用者以外が仮想化ミドルウェアに直接アクセスするのを防ぐことができるため，セキュリティを高めることができる．後者の方法でクライアントプログラムを起動する手順を以下に示す．

①利用者はWebサーバーで公開されたページにアクセスし，利用したいアプリケーションを選択する．

②仮想化ミドルウェアはWebサーバーからのリクエストに応じ，選択されたクライアントプログラムを起動する．

③起動されたクライアントプログラムはアプリケーションのログイン画面を表示する．

それぞれの手順において，図5のようにユーザー認証を検討する必要がある．①では，ページの閲覧を利用者だけに限定するため，Web サーバーへのアクセス認証を実施する．②では，仮想化ミドルウェアは AP サーバー上のユーザープロファイルを指定してクライアントプログラムを起動するので，ユーザーを識別するためのログイン認証が必要となる．更に③では，アプリケーションにログインする必要がある場合，認証が発生する．

図5　アプリケーションの起動と認証

利用者の利便性を向上させるために，認証と併せて，シングルサインオン（SSO）について検討する必要がある．ここで述べる SSO とは，図5の AP サーバー側の認証を連携するものであり，利用者端末へのログイン認証は含まない．③でログインが必要となる場合は，アプリケーション独自認証であることが多いため，アプリケーションを修正せずに①②③すべての認証を連携するのは難しい．しかし①と②の連携は，Web サーバー認証の実装を工夫することで実現可能である．例として，①と②で Windows 認証を使用し，①で入力した資格情報を②の認証時に渡す実装をすることで SSO が実現できる．

3. 3　利用者端末から iDC へのネットワーク接続

利用者端末から iDC 内のサーバーに接続する方法には，セキュリティの高い順に「専用線接続」「VPN 接続」「SSL 暗号化による internet 接続」「暗号化なしの internet 接続」などがある．これらの接続方法を決定する際は，セキュリティの要件に加えて，仮想化ミドルウェアの制約も考慮に入れなければならない．仮想化ミドルウェアがリバースプロキシ[*1]に対応している場合は，図6の左図のように AP サーバーと DB サーバーを内部セグメントに配置することができる．リバースプロキシに対応していない場合は，図6の右図のように AP サーバーは DMZ セグメントに配置しなければならない．そのためセキュリティ要求が高いサービスについては，接続方式を専用線や VPN などにする必要がある．3. 1 節の図4のように仮想化を適用しない構成の場合も，クライアントプログラムとデータベースの通信は暗号化されていないことが多いため，専用線や VPN などのセキュリティの高い接続方法を検討する必要がある．

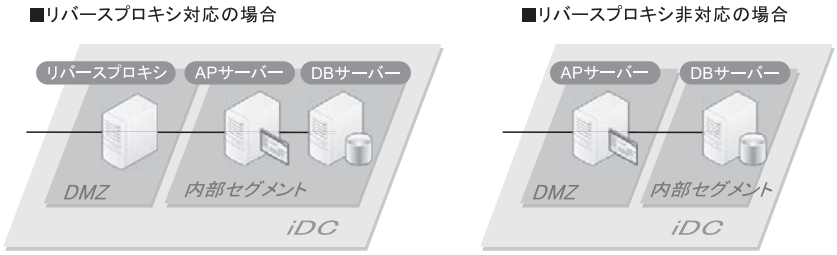

図6　iDC 内のセグメントとサーバー配置

4.　マルチテナント化における考慮

マルチテナント化を検討する上では，アプリケーションの制約やセキュリティを考慮しなければならない．本章では AP，DB それぞれのサーバー毎にマルチテナント化の検討を行う．

4. 1　AP サーバーのマルチテナント化

マルチテナント化する際には，テナントの分離方法を検討すると同時に，サービスのパフォーマンスや可用性を考慮する必要がある．また，提供するアプリケーションの制約によっては，マルチテナント化せずにシングルテナントを採用しなければならない場合もある．

1）　テナントの分離方法

AP サーバーのマルチテナント化には，テナント間で一つのプログラムを共有するシングルインスタンス方式と，テナント毎にプログラムを用意するマルチインスタンス方式がある．既存のクライアントサーバー型アプリケーションは，マルチテナントを意識して構築されていないため，一つのプログラムで複数のテナントに対応することは難しい．また，マルチインスタンス方式で AP サーバーをマルチテナント化する際には，複数のプログラムをインストールすることでレジストリやファイルの競合が起きないことを確認する必要がある．

2）　パフォーマンスと可用性

AP サーバーで複数のテナントを稼働させる場合には，クライアントプログラムの同時起動によるパフォーマンスの低下について検討する必要がある．パフォーマンスの低下が懸念される場合には，サーバーのスケールアップや同時アクセス数の制限による対応が効果的だが，シングルテナントのまま，同時に起動するインスタンス数を抑える対策も有効である．

また，サーバーの可用性についても検討する必要がある．たとえば A 社ユーザーのオペレーションによって AP サーバーに障害が発生すると，最悪の場合同じサーバーで稼働している B 社ユーザーにも影響が及ぶことになる．他テナントへの影響を最小限にするためには，AP サーバーをシングルテナントとして構築しなければならない．

3）　セキュリティ

マルチテナント化すると，一つのサーバー内に複数のテナントが同居するため，クライアントプログラムが出力したファイルに他テナントのユーザーがアクセスする危険性など，サーバー内のテナント同士のセキュリティリスクは高くなる．対策の一つとして，サーバー上のファイルやディレクトリなどの共有リソースへのアクセスを，テナント毎に制御することが挙げられる．

4. 2　DB サーバーのマルチテナント化

DB サーバーのマルチテナント化には，図 7 で示すように二つの方法がある．一つめはテナント間で一つのインスタンスを共有し，インスタンス内部にテナント毎のデータベースを構築する方法（図 7 の【A】）であり，二つめはテナント毎にインスタンスを用意する方法（図 7 の【B】）である．多くの既存アプリケーションでは，データベースとの接続情報を設定ファイルで管理している．設定ファイルにインスタンス名とデータベース名を指定できる場合は

【A】と【B】のいずれの方法も選択することができるが，インスタンス名のみ指定できる場合は【B】の方法に限られる．また，データベース名とインスタンス名が指定できない場合には，マルチテナント化することはできない．

AP サーバーの場合はアプリケーションの制約やセキュリティの要件によりマルチテナント化できないケースが多いが，DB サーバーの場合は比較的容易にマルチテナント化できるケースが多い．

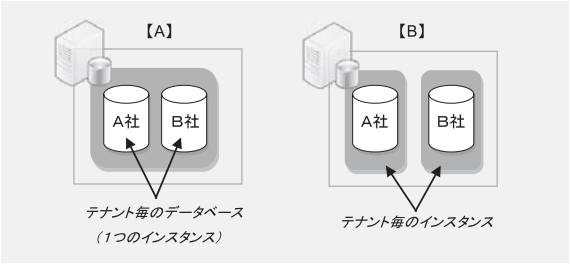

図 7　DB サーバーのマルチテナント構成

5.　アプリケーションサービス化の構成例

これまでに，2 章ではアプリケーションをサービス化する際のシステム構成について説明し，3 章や 4 章ではアプリケーションの制約やその他の考慮によって構成を変更することについて説明した．本章では二つの事例を挙げて，具体的な要件や制約を基にシステム構成を決定する手順を紹介する．

1）　セキュリティを重要視した構成例

セキュリティを確保するために，SSL で暗号化されたインターネット通信を利用した．テナント構成については，AP サーバーはマルチテナント化するとサーバー内のテナント同士のセキュリティが低くなってしまうおそれがあったため，シングルテナントとした．DB サーバーはマルチテナント化し，インスタンスを分けることでセキュリティを確保した．

アプリケーションと仮想化の適合性検証では，このアプリケーションは比較的動作が軽く，同時接続ユーザー数も 10 人程度であり，同一サーバーのリソースで対応できると判断して仮想化を採用した．表2に構成を決定するための条件を，図8に最終的に選択した構成を示す．

表 2　サービス化の条件

要件	ネットワーク セキュリティ	優先1
	テナント間の セキュリティ	優先1
	利便性	優先2
	運用コスト	優先3
制約	パフォーマンス	問題なし
	APサーバーの マルチテナント化	テナント間のセキュリティ確保できない
	DBサーバーの マルチテナント化	APの設定ファイルで対応可能

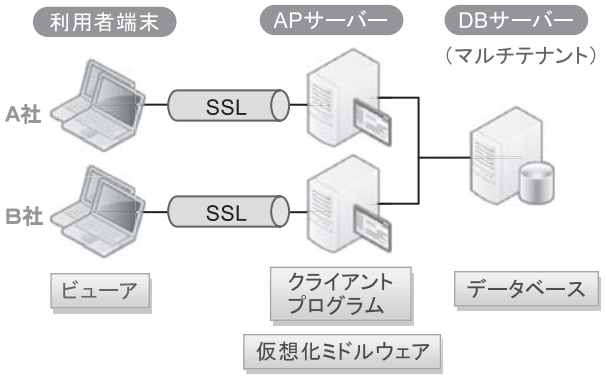

図 8　選択した構成

2)　マルチテナントが適用できない構成例

アプリケーションの制約により，仮想化およびマルチテナント化が適用できない場合がある．アプリケーションと仮想化の適合性を検証したところ，AP サーバー上で同時に五つ以上のプロセスが起動したときにパフォーマンスが著しく低下することが分かった．このアプリケーションは一つのテナントで少なくとも 10 人以上のユーザーが同時使用することが求められているため，仮想化しないこととした．それに伴って，利用者端末からの接続方法はセキュリティを高めるために VPN を利用した．

表 3　サービス化の条件

要件	運用コスト	優先1
	利便性	優先2
	セキュリティ	優先3
制約	複数プロセス起動	問題なし
	ServerOS上での動作	問題なし
	パフォーマンス	問題なし
	APサーバーの マルチテナント化	1サーバー 5テナント
	DBサーバーの マルチテナント化	1サーバー 5テナント

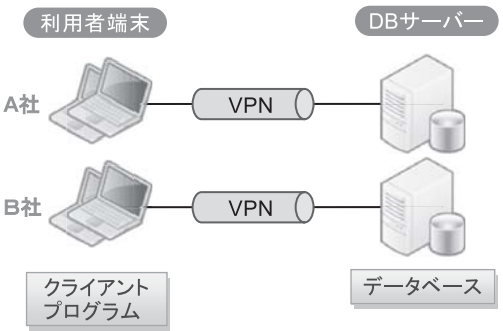

図 9　選択した構成

テナント構成は，運用コストを下げるためにDBサーバーをマルチテナント化すべきであるが，クライアントプログラムが固定のデータベース・インスタンスにしか接続できない仕組みになっているため，DBサーバーもシングルテナントとした．表3に構成を決定するための条件を，図9に最終的に選択した構成を示す．

6. おわりに

本稿ではクライアントサーバー型アプリケーションを複数の顧客に対してサービス化する方法について説明した．このアプローチは，特定顧客に対するサービス化（プライベートクラウド）にも流用可能である．アプリケーションをサービス化する上で重要なことは，本稿で挙げた検討ポイントに従って十分に検証することだと考える．これらを実現するためには様々な障壁を乗り越えなければならないが，本稿で示したガイドラインがその一助となれば幸いである．

＊1　クライアントとサーバーの中間に位置し，クライアントからのリクエストを中継して，サーバーにリクエストを送信する代行サーバー

参考文献　［1］「SaaS向けSLAガイドライン」，経済産業省，2008年1月，http://www.meti.go.jp/press/20080121004/20080121004.html
［2］　太田智朗，「クラウドコンピューティング概況と日本ユニシスの取り組み」，ユニシス技報，日本ユニシス，Vol.29 No.1 通巻100号，2009年5月，P65～78

執筆者紹介　曽根　信一郎（Shinichiro Sone）
2003年日本ユニシス(株)入社．自動車部門にてPLMソリューションのシステム適用に従事．2008年より共通利用部門に所属し，ICTサービスの企画開発に携わる．