「クラウド・コンピューティング時代の SAM の考え方」

平成 23 年 6 月

一般財団法人日本情報経済社会推進協会

はじめに

本冊子は、一般財団法人日本情報経済社会推進協会が財団法人 JKA の補助金を受けて実施した平成 22 年度情報化推進に関する調査研究等補助事業「IT サービスマネジメントの利活用によるシステム構築・運用環境の改善に向けた調査研究」事業の一環として作成したものである。

近年、クラウドコンピューティングの利用が、ビジネスの世界にとって必要不可欠な時代となってきた。しかしながら、セキュリティやライセンスのコンプライアンス、TCO の問題、ICT ガバナンス等、ユーザー側がどのように実現するかは喫緊の課題となっている。

クラウドコンピューティングもソフトウェアの利用方法の一形態と考えられるため、クラウドコンピューティングという新しい形態で何をどう管理すればよいのか、どのような点に留意する必要があるのかをソフトウェア資産管理（Software Asset Management：SAM）の側面から調査研究することとした。

本冊子は、「クラウド・コンピューティング時代のSAMの考え方」について解説するとともに、SAMの観点からクラウド・コンピューティング利用の留意点を取りまとめたものである。本冊子が企業・団体におけるソフトウェア資産管理に携わる方々のお役に立てば幸いであり、ソフトウェア資産管理の普及促進に資することが期待される。

平成 23 年 6 月

一般財団法人日本情報経済社会推進協会
ソフトウェア資産管理評価検討委員会

目次

1. はじめに

今日，クラウド・コンピューティングという言葉が紙面を賑わさない日はない。クラウド・コンピューティングでは，複雑な ICT のお守りをすることなく，コストを最小限に抑え，直ちにビジネスを立ち上げることができると言われている。利用者にとってはバラ色の世界であるが，本当にそうなのであろうか？確かに個人で利用しているとき，いわゆる，BtoC では，Web ブラウザーさえ動作すれば，大量のドキュメントや写真を保管してくれるし，メールや知り合いとのコミュニケーションも可能だ。簡単なワープロや表計算，プレゼンテーション資料までできてしまう。移動先でも簡単に利用でき，移動中のスマートフォンからもアクセスできる。更に，個人の PC よりは信頼おける管理をしてくれる。使わない方が損である。

このような便利でスマートなコンピュータの使い方をビジネスの世界でも利用できないかという思いは当然起こることであろう。30 日限定の評価ライセンスで CRM（Customer Relationship Management）の SaaS を利用してみた。Web から簡単にメンバーのアカウントを登録し，要件の定義を行う。画面のカスタマイズも簡単に行える。後は，評価ライセンスを正式なものに変更するだけ。この間，情報システム部門に依頼することは何もない。しかもビジネスの規模が拡大すると自動的にコンピュータリソースは拡大される。

昔，EUC（End User Computing）という言葉が流行した。これは，理由の一つとして，自分たちのやりたいことになかなか情報システム部門が対応してくれないので自分たちで作るというものであったが，1990 年代の PC の進展とともに普及に加速がかかった。早く，ビジネスを立ち上げられるという意味ではいいことなのだが，企業にとってみれば，どこでどんなシステムが導入されているのか？ 新しく全社的なシステムを導入したいが，互換性の保証は？ TCO（Total Cost of Ownership）はどうなっているか？ ウィルス対策は？ ソフトウェアの不正コピーで訴えられることはないのか？･･･など非常に厄介な状況に陥ってしまった。2000 年代になり，情報セキュリティやガバナンスが厳しく問われるようになると，このような状況は，企業の存続をも危うくする大問題になってくる。

今，クラウド・コンピューティングという非常に有効な仕組みがビジネスの世界に入ってきた。セキュリティやライセンスのコンプライアンスや TCO の問題，企業における ICT のカバナンスをどうするかを適切に考え対応しておかなければ，これまで以上に大変なことになることが予想される。

クラウド・コンピューティングは，どの層までのサービスを受けるかで，SaaS（Software

as a Service)，PaaS（Platform as a Service)，IaaS（Infrastructure as a Service)，DaaS（Desktop as a Service）などと分類して説明されることが多い。ビジネスに使われるということから，セキュリティや品質面の安心感を求めて，雲の向こうのサービスを自分の組織だけが利用する形態も提唱されている。これを通常のクラウド・コンピューティングと分類して，プライベートなクラウド・コンピューティングと呼んでいる。因みに従来のものは，パブリックなクラウド・コンピューティングという。分類は色々だが，クラウド・コンピューティングでは， ICT リソースの物理的な構成にとらわれず（仮想化)，ユーザーの要求に応じて迅速に割り当て提供される（プロビジョニング)。また，ソフトウェアの費用を抑えるために， 或いは，柔軟性を確保するために OSS（Open Source Software ）を利用することも多くなる。

よく，クラウド・コンピューティングを導入すれば，ＩＣＴの運用管理がなくなるとかＳＡＭが楽になると言われるが，ビジネスの分野では，ユーザーの資源として管理しなければならない範囲が増えれば増えるだけ，サービスを提供するソフトウェアや関連する資産の管理が発生し，先に述べたように，より一層困難になる場合も多い。一般的な企業や団体に適用され始めたクラウド・コンピューティングで，何をどう管理してゆけば良いのかを，改めて考察することは非常に有意義であると考えた。

本書では,「クラウド・コンピューティング時代のＳＡＭの考え方」をまとめる。即ち，ICT を利用する現代の企業或いは団体組織が，クラウド・コンピューティングや仮想化技術，OSS（Open Source Software）を利用する際にどのようなマネージメントが必要になるのかを SAM の延長として捉える。

先ず 2 章では，本書で記述する鍵となる用語について解説する。3 章では，クラウド・コンピューティングでポイントとなるクラウドの形態と仮想化の技術，OSS について，それぞれの一般的な説明とサービス事例を解説する。 4 章では，SAM の観点からクラウド・コンピューティングで特徴的な留意点を SaaS と DaaS に着目し記述する。ここでいう DaaS は，Desktop as a Service を指しているが，特に取り上げたのは，今後，投資，運用，セキュリティのどれをとっても有望と思われる DaaS の世界での SAM が問題になりそうだからである。5 章では，そもそもクラウド・コンピューティングを可能としている仮想化技術に着目し，SAM をどうすべきかについて考察する。また，クライアントで利用されている仮想化 OS での SAM の考え方についても触れる。6 章では，組織が OSS を活用する場合における SAM としての留意点をまとめた。

クラウド・コンピューティングとか仮想化，OSS で SAM に取り組んだ書籍はあまり例がない。本書がこの分野での SAM に関心のある諸兄にお役に立てれば幸いである。

2. 用語の解説

2.1 仮想化

サーバーやストレージ, ネットワークなどの ICT リソースを物理的な構成にとらわれず, 論理的に構成する技術。

2.2 プロビジョニング

ネットワークやシステムリソースなどをユーザーの要求に応じて迅速に割り当てサービスを提供すること。

2.3 ハイパーバイザー

仮想化を実現するための制御プログラムで仮想化 OS とも呼ばれ，ハードウェア上で直接動作し，その上で複数の OS が動作する。

2.4 セキュリティ・アプライアンス

不正アクセスとかウィルスなどの外部の脅威から企業の IT 環境を守るための機器。

2.5 クラウド

クラウド・コンピューティングの略称。

2.6 クラウド・コンピューティング

ネットワークを介して，仮想化されたネットワーク，ハードウェア，OS 及びミドルウェア，アプリケーションソフトウェアなどの IT リソースを効率的なプロビジョニングにより利用すること。

2.7 クラウドサービス

クラウド・コンピューティングにより提供されるサービス。

2.8 パブリッククラウド

多種多様な企業や組織，個人といった不特定多数の利用者を対象に広く提供されるクラウドサービス。

2.9 プライベートクラウド

サーバー仮想化とリソースのプロビジョニングを効率的に利用した企業のファイヤーウォールの内側に存在する個別企業ないしはグループ企業のクラウドサービス。

2.10　オンプレミス

ハードウェアやソフトウェアを自社で調達して運用する方式。

2.11　ハイブリッド型

クラウド・コンピューティングを行うときに費用とセキュリティを最適にするため「パブリッククラウド」,「プライベートクラウド」,「オンプレミス」を併用して目的とするサービスを実現する形態。

2.12　SaaS（サース）Software as a Service

アプリケーションをネットワーク経由でクラウド・コンピューティングのサービスとして提供する形態。

2.13　PaaS（パース）Platform as a Service

アプリケーション開発・実行基盤をクラウド・コンピューティングのサービスとして提供する形態。

2.14　IaaS（イアース）Infrastructure as a Service

CPU やメモリー，ディスク，OS 等，データセンターに準備されている共用のシステムリソースをクラウド・コンピューティングのサービスとして提供する形態。通常はハイパーバイザーを含んでいる。

2.15　DaaS（ダース）Desktop as a Service

データベースの機能を提供するものとしての Database as a Service やデータストレージの機能を提供するものとしての Data Storage as a Service を DaaS と呼ぶこともあるが，本書では，サーバーの仮想化 OS 上に複数のデスクトップ環境をクラウド・コンピューティングのサービスとして提供する形態を言う。技術的には，仮想 PC 方式のシンクライアントシステムのサービスに相当する。

2.16　OSS （Open Source Software）

「ソフトウェア著作者の権利を守りながらソースコードを公開することを可能にする」というオープンソースの概念に基づき，ソフトウェアのソースコードが無償で公開され，改良や再配布を行うことが誰に対しても許可されているソフトウェアのこと。

2.17　グリーン・コンピューティング

IT の利用で環境に与える負荷を削減するために IT のライフサイクル全般で取り組む諸

活動またはその考え方のこと。

2.18　クライアント仮想化 OS

クライアント自身で動作する仮想化 OS で，通常は，クライアントの OS（ホスト OS と呼ぶ）上で動作し，他の複数の OS（ゲスト OS と呼ばれる）を動作させる。代表的な例として以下のものがある。

・　VMware の VMware Workstation
・　マイクロソフトの Virtual PC

2.19　シンクライアントシステム

ユーザーが使うクライアント端末に最低限の機能しか持たせず，サーバー側でアプリケーションソフトやファイルなどの資源を管理するシステムのアーキテクチャ全般のこと。

2.20　シンクライアント

シンクライアントシステムで使われる機能を絞り込んだ専用のクライアント端末のことで一般的には以下の 4 つの実装方式がある。

・　ネットワークブート方式
・　サーバベース方式
・　ブレード PC 方式
・　仮想 PC 方式

本書で取り扱っている DaaS でのシンクライアントは，仮想 PC 方式を想定している。

2.21　ネットワークブート方式

シンクライアントの実装方式の一つで，サーバー側に OS やアプリケーションを置き，実行時にサーバーから端末に転送し実行する方式。

2.22　サーバベース方式

シンクライアントの実装方式の一つで，アプリケーションの実行など全ての処理をサーバー上で行い，端末側は遠隔操作端末としての役割のみを担う方式。

2.23　ブレード PC 方式

シンクライアントの実装方式の一つで，デスクトップ環境 1 台毎に用意したブレードをネットワーク越しに操作する方式。

2.24　仮想 PC 方式

シンクライアントの実装方式の一つで，　サーバーOS で仮想 OS を実行させ，複数のブ

レード PC のように見せてネットワーク越しに操作する方式。

2.25　SLA（Service Level Agreement）

サービス品質に対する利用者側の要求水準と提供者側の運営ルールについて明文化したものである。

2.26　SLM（Service Level Management）

サービスに関わるルール，プロセス，体制などの改善により高品質なサービスを維持し，サービスレベルの要求水準とサービス内容を利用者の事業上の要求の変化に対応させるための継続的な運営・管理手法である。

2.27　EULA（End User License Agreement）

ソフトウェア利用許諾契約またはソフトウェア使用許諾契約が書かれた契約書のこと。ソフトウェアの著作権者が、そのソフトウェアを利用するユーザに示す利用条件を表している。

2.28　ISMS（Information Security Management System）

個別の問題ごとの技術対策の他に、組織のマネージメントとして、自らのリスクアセスメントにより必要なセキュリティレベルを決め、プランを持ち、資源を配分して、システムを運用するための仕組み。組織が保護すべき情報資産について、機密性、完全性、可用性をバランス良く維持し改善することが ISMS の基本コンセプト。

2.29　マッシュアップ（Mash up）

元来音楽用語ではあるが，複数の Web サービスの API を組み合わせ，あたかも一つの Web サービスのようにすること。

2.30　XML（Extensible Markup Language）

文書やデータの意味や構造を記述するためのマークアップ言語であり，マークアップ言語作成にも使われる汎用的な仕様。

2.31　TSV（Tab Separated Values）

データをタブ文字で区切って並べたファイル形式。ファイルの拡張子は規程されていない点がカンマ区切り形式の CSV と違う。

2.32　VDI（Virtual Desktop Infrastructure)

デスクトップ OS 環境を，サーバ・ファームの仮想マシン上に実装・集約するための基盤

システム。

2.33　JavaEE

Java による大規模サーバアプリケーション構築のためのフレームワーク・API の総称。J2EE のこと。

2.34　SOA（Service Oriented Architecture）

業務の構成単位に合わせて構築したソフトウェアの機能をサービスとしてネットワーク上に公開し，これらを相互に連携させることによりシステム全体を構築する設計手法。

2.35　ITIL（IT Infrastructure Library）

IT サービスマネジメントのベストプラクティスで、IT サービスマネジメントのデファクトスタンダード（事実上の標準）と呼ばれる。1980 年代から英国で IT サービスを効率的に管理・運用していくための方法論の模索として整理され、実際の IT サービス運用のノウハウ等が集積されたライブラリで、一連の書籍群から構成される。

2.36　CMDB（configuration management database）

IT インフラの各構成品目に関連するすべての詳細，及びそれら構成品目間の関係を含むデータベース。

2.37　GNU（GNU's Not UNIX）

UNIX 互換のソフトウェア環境を全てフリーソフトウェアで実装することを目標とするプロジェクト，およびそのソフトウェア全体を指す。

2.38　プロプライエタリ・ソフトウェア (proprietary software)

ソフトウェアの使用、改変、複製を法的・技術的な手法を用いて制限しているソフトウェアを指す。法的制限手法としては、著作権や特許権及びそれに基づくソフトウェアライセンス許諾といった方法がある。技術的制限手法としては、バイナリ実行コードのみをユーザーに提供し、ソースコードは公開しないというソフトウェア流通の方法がある。

2.39　コピーレフト (copyleft)

著作権（copyright）に対する考え方で、著作権を保持したまま、二次的著作物も含めて、すべての者が著作物を利用・再配布・改変できなければならないという考え方である。

2.40　サイロ型システム

アプリケーションやデータが縦割りに独立した部門最適型のシステム。

3. 仮想化，クラウド，OSS の現状

本章では，SAM における仮想化，クラウド，OSS 利用時の留意点について検討するに当たり，第２章で定めた「仮想化，クラウド，OSS」の定義に従い，それぞれの現状説明とサービス事例について解説する。

表 3-1　仮想化，クラウド，OSS の用語定義（第２章より）

#	用語	本稿における定義
1	仮想化	サーバーやストレージ, ネットワークなどの ICT リソースを物理的な構成にとらわれず，論理的に構成する技術。
2	クラウド	ネットワークを介して，ハードウェア，OS 及びミドルウェア，アプリケーションソフトウェアなどの IT リソースを使用する，IT サービスの利用形態。
3	OSS	ソースコードを無償で公開し，誰でもそのソフトウェアの改良，再配布が行なえるようにしたソフトウェア。

3.1.　仮想化

3.1.1. 仮想化の定義

仮想化（英：Virtualization）とは，様々な IT リソースの物理的特性を，そのリソースと相互関連するシステム，アプリケーション，エンドユーザーから隠蔽する技法である。この技法により単一の物理リソースを複数の論理リソースに見せかけたり，逆に複数の物理リソースを単一の論理リソースに見せかけたりすることができる[1]。

具体的には，１台のサーバーをあたかも複数台のサーバーであるかのように論理的に分割し，それぞれに別の基本ソフト（OS：Operating System）やアプリケーションソフトを動作させる「サーバー仮想化」や，複数のディスクをあたかも 1 台のディスクであるかのように扱い，大容量のデータを一括保存したり耐障害性を高めたりする「ストレージ仮想化」などが実用化段階に入っている。

仮想化という用語自体の歴史は古く，1960 年代から広く用いられている。初期のコンピュータはメモリー上に 1 つのプログラムしか呼び出せなかった。そのため１つのプログラムがリソースをすべて占有していた。これでは高価なコンピュータの利用効率が上がらない。そこで１台のコンピュータを複数のユーザーで同時利用するための研究・開発が始ま

[1] 出典：Virtualization 101: Technologies, Benefits, and Challenges; 04/17/2007, Enterprise Management Associates; White paper

り，タイム・シェアリング・システム（TSS：Time Sharing System）や仮想記憶などの技術が登場した。これらはコンピュータの限られた物理リソースを論理的に分割して，ユーザーごとに割り当てる（多重化）と共に，ユーザープログラムから物理リソースを隠蔽して，より使いやすい論理的な利用環境を提供することを目的としていた。これらの技術はその後，オペレーティングシステム（OS：Operating System）へと発展し，現在では OS の標準機能として，我々が普段特に意識せずに利用しているものも多い。

このように，コンピュータ発展の歴史は仮想化の積み重ねともいえる。その後も様々な仮想化技術が登場しているが，いずれの場合も共通する目的は「カプセル化によって実装技術の詳細を隠蔽すること」である。近年ハードウェアの性能向上と共に新たな仮想化技術が登場し，この歴史ある用語・概念が再び注目されている。

3.1.2. 仮想化の分類

「仮想化」という用語は，現在コンピュータ以外の分野においても様々な文脈で用いられている。そのため仮想化を「複雑な実装を隠蔽し，単純化されたユーザインタフェースを提供するもの」と定義するだけでは議論が混乱する恐れがある。そこで本稿ではまず現在実用化されつつある仮想化技術を大きく，サーバー仮想化，ストレージ仮想化，ネットワーク仮想化，クライアント仮想化（デスクトップ/アプリケーション仮想化），の４つに分類した上でそれぞれの基本概念を解説し，その後「ソフトウェア資産管理における仮想化技術利用時の留意点」の検討対象となる仮想化概念を絞り込むことにした。

表 3-2　仮想化の分類

#	分類
1	サーバー仮想化
2	ストレージ仮想化
3	ネットワーク仮想化
4	クライアント仮想化／デスクトップ仮想化

3.1.3. サーバー仮想化

サーバー仮想化とは，単一のサーバーをあたかも複数のサーバーであるかのように論理的に分割し，それぞれに別の基本ソフトウェア（OS：Operating System）やアプリケーションソフトウェアを動作させる技術である。具体的には，単一サーバー内に OS からアプリケーションまで含めた分割領域を複数設定し，それぞれの領域が独立して動作するものである。この分割領域をそれぞれ「仮想サーバー」と呼ぶ。仮想サーバーは単一のサーバー上で動作しながら，お互いに悪影響を及ぼすことがないように設計・設定されているので，たとえある仮想サーバーの OS やアプリケーションソフトウェアがダウンしても，他の仮想

サーバーは稼動し続けることができる。サーバー仮想化を一言で表現すると「複数の物理サーバーをそのまま単一のサーバーに移行する技術」と言える。(図 3-1)

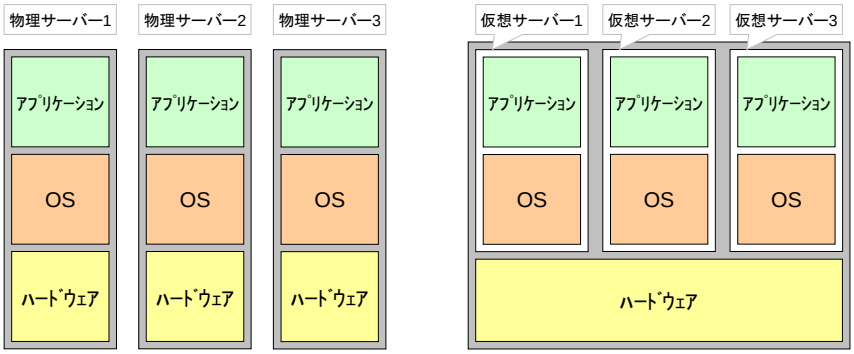

図 3-1　サーバー仮想化の基本概念

サーバー仮想化には，ホスト OS の上にゲスト OS を稼働させる方法，ハイパーバイザーと呼ばれるファームウェアの上に複数 OS を稼働させる方法, ハードウェア自体にシステムを分割／統合する機能を装備する方法などがある。また，グリッド・コンピューティング関連技術も，サーバーを統合化して取り扱う仮想化技術の応用といえる。サーバー仮想化技術はこれまでは独立したソフトウェアとして提供されることが多かったが，OS や CPU 製品でサポートするものも登場しつつある。

サーバー仮想化のメリットは，前述のように「これまで複数の物理サーバーで行ってきたことを単一のサーバー上で実現できる」ということである。ここではサーバー仮想化の導入による代表的なメリットを 3 点挙げる。

(1)　サーバー統合によるコスト削減効果

多数の物理サーバーを保持・運用している組織では，各サーバーはそれぞれの処理能力は余裕を持ってキャパシティ設計されているため，全てのサーバーが常時フル稼働している訳ではない。組織によっては，極端に利用率が低いサーバーも存在する。これを少数の大型サーバーに統合・置き換えることで単純にサーバーの台数を減らすことができる。

統合先となる大型サーバーは従来の小型サーバーよりも堅牢な上位機種が必要となる。

しかしサーバーの利用に関するコストでは運用管理のための人件費が大きな割合を占めている。そして人件費はサーバーの台数に比例して増加するとも言われている。従って大型サーバーへの買い替え費用を考慮しても，多数の小型サーバーを個別に購入，維持運用するコスト（特に人件費）を削減できる効果は大きいと考えられる。

更にサーバーを多数稼動させることは大量のサーバー電力消費と発熱（空調電力消費）につながる。このような環境負荷を軽減し，グリーン・コンピューティングを実現することも，サーバー統合すなわちサーバー仮想化の副次的効果と考えることができる。

(2) プロビジョニングによる安定稼動効果

サーバー仮想化技術によって仮想サーバー同士がリソースを共有できるようになれば，異なるアプリケーション間でリソースを融通し合ったり，予備のリソースをプール（集約管理）したりできる。特定の仮想サーバーの処理負荷が高まった時，この予備のリソースを追加で割り当てることで，サーバーの稼動を止めずに特定の仮想サーバーの性能をアップさせることができる。このような処理負荷の増減に応じたITリソースの動的再配置による最適化はプロビジョニングを呼ばれ，システムの安定稼働やディザスタ・リカバリのソリューションとして有効である。

(3) コンピューティング環境整備の短工期化

あるアプリケーションソフトウェアを新規に導入したり，開発環境や検証環境を整備したりする際に，サーバーの調達・設置に長時間を要することがある。仮想化サーバーを利用することにより，必要なときに必要なサーバー環境を即座に設定し，役割が終われば即座に撤収することもできるので，新規アプリケーションソフトウェアの導入コストの削減や，開発・検証作業効率の向上策として大変有効である。

3.1.4. ストレージ仮想化

ストレージ仮想化とは，複数のストレージ装置を論理的に集約する技術である。基本的にはサーバー仮想化の一領域とも考えられるが，ストレージの構成要素はサーバーほど複雑ではないため，むしろサーバー仮想化（サーバー統合）よりも先行して普及している。特に企業のサービスの拡大などに伴ってデータが増えるにつれて，企業システムではストレージ装置の台数や容量も急速に増大している。近年このデータやストレージ装置の増加がシステム構成を複雑化し，コストを押し上げる大きな一因となりつつある。

ところで，サーバー仮想化が基本的には「複数の物理サーバーで行ってきたことを単一のサーバー上で実現する」こと，つまり「サーバー統合」とほぼ同義であるのに対し，ストレージ仮想化には，統合と分割という二つの意味があるので，注意しなければならない。

現在主流となっているストレージの仮想化は，複数の物理ストレージ装置を抽象化して仮想的に単体のストレージ装置（ストレージプール）に見せるという考え方である（図 3-2 左）。

もうひとつの考え方は，単一の大型物理ストレージ装置を仮想的に複数のストレージ装置に見せるというものである。（図 3-2 右）これはサーバー仮想化（サーバー統合）と同じ狙いであり，多数のストレージ装置を管理・運用する負荷を削減するために既存のストレージ群を 1 台の大型ストレージ装置に統合することを目的としている。

両者はあたかも正反対の行動に見えるが，「物理な実装を隠ぺいして仮想的なユーザインタフェースを提供する」という点では仮想化の定義に合致している。それぞれのユーザーの目的が「小型ストレージをあたかも大型ストレージのように利用したいか」あるいは「大型ストレージをあたかも小型ストレージのように利用したいか」という点で異なるだけである。

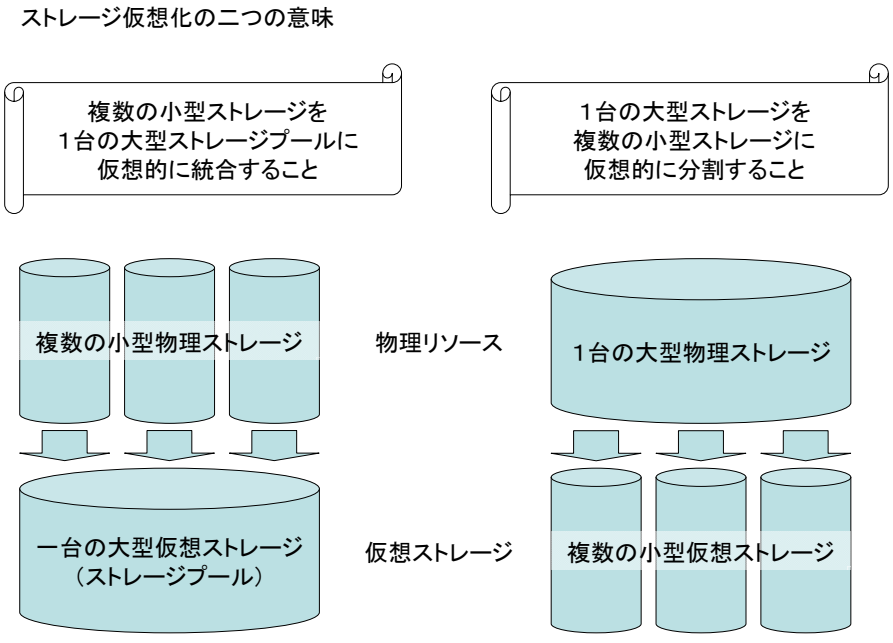

図 3-2　ストレージ仮想化の基本概念

3.1.5. ネットワーク仮想化

ネットワーク仮想化とは，ネットワーク機器を個別に用意するのではなく大型サーバーの内部に仮想的にルーター／スイッチ，セキュリティ・アプライアンスといったネットワーク機器を準備し，柔軟に組み合わせて使うものである。サーバー仮想化のネットワーク

機器版とも考えられる。システム変更の効率化が最大のメリットである。

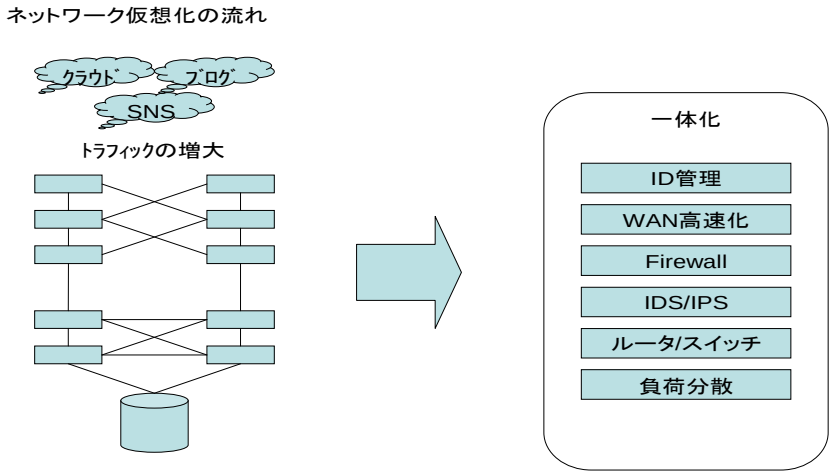

図 3-3　ネットワーク仮想化の基本概念

3.1.6. クライアント仮想化／デスクトップ仮想化

クライアント仮想化（またはデスクトップ仮想化）とは，仮想化技術を利用してクライアント PC の運用管理を効率化するための技術の総称である。

クライアント PC の運用管理の負荷は PC 台数の増加に比例して増大する。そこでソフトウェアの実行をサーバー側で行ない，クライアント PC はユーザインタフェースとしての役割のみに徹するという考え方が生まれた。これによりクライアント PC の OS やアプリケーションソフトウェアをサーバー側で一元管理できるようになる。仮想化の対象をアプリケーションソフトウェアまでとして PC 側の OS を利用するものをアプリケーション仮想化と呼び，OS まで含めて全てサーバー側で実行するものを OS 仮想化と呼ぶ。

クライアント仮想化の主なメリットは以下の通りである。
(1)アプリケーションやデータを集中管理できる
(2)サーバー側でポリシー設定を行うことでユーザー別にアクセス制御ができる。

例えば直営社員と派遣社員で利用可能なアプリケーションを区別することができる。最近では外注業者を社内に常駐させてシステム開発を委託する場合に，アプリケーション仮想化を利用する例が増えている。あるいは旧式のアプリケーションソフトウェアと最新のアプリケーションソフトウェアを同じクライアント PC 上で動作させることもできる。

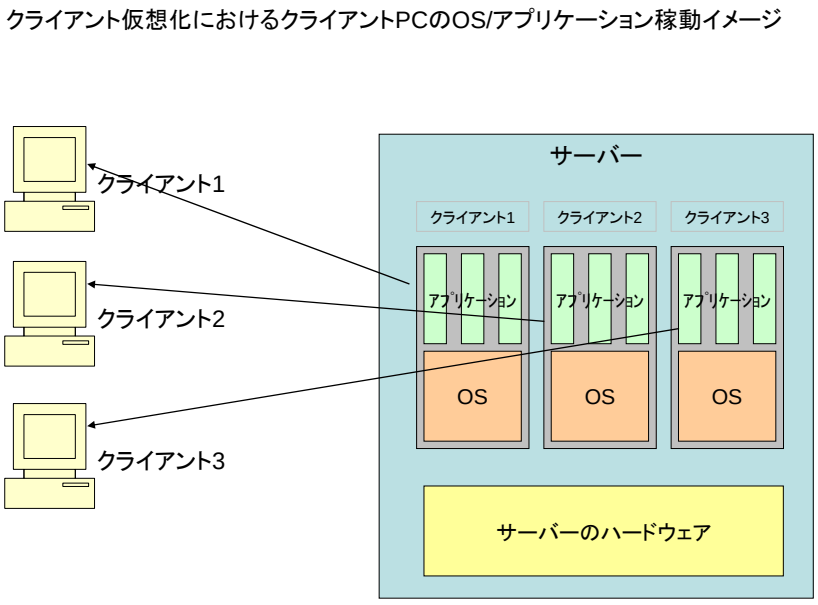

図 3-4　クライアント仮想化の基本概念

3.1.7. 仮想化とソフトウェア資産管理

以上の通り，代表的な「仮想化」概念について解説した。これらの仮想化技術利用時におけるソフトウェア資産管理の留意点については第 4 章以下で詳説するが，ここでは若干の課題提示を行いたい。

(1)一般的な注意点

仮想サーバーのソフトウェアのライセンス料は一般的に仮想サーバー単位で発生する。仮想サーバーを設定する毎にそれぞれにライセンス料が必要になる。

(2)仮想サーバー追加・削除時の注意点

なお，仮想サーバーは容易に追加・削除が可能なので，その分ライセンス管理が甘くなりがちである。サーバー仮想化を行う際は，追加・削除という変更管理に合わせて，ライセンス変更も連動した管理体系が求められる。

(3)ソフトウェアベンダーの対応

サーバー仮想化が定着しつつある中でソフトウェアベンダー各社も柔軟なライセンス体系を導入しつつある。CPU 課金体系を大別すれば，物理サーバーの CPU 数，仮想サーバーに割り当てられた CPU 数のいずれかをカウントすることになる。仮想サーバーに割り当てられた CPU 数によって課金される場合，仮想サーバーの特徴である CPU 数の動的配置

への対応が課題となる。

仮想サーバーにおけるCPUライセンスの課金イメージ

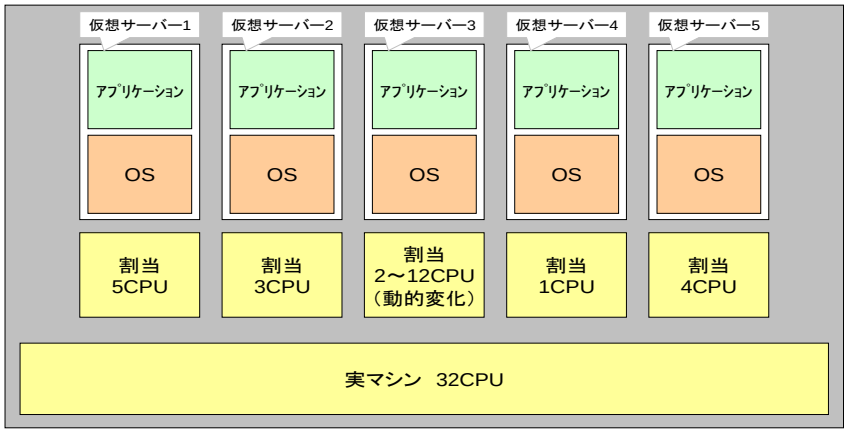

図 3-5　サーバー仮想化におけるライセンス課金問題

参考：代表的なソフトウェアベンダーのライセンス体系

・Microsoft

Windows Server 2008：ライセンスの種類により実行可能な仮想マシン数が定められている（Windows Server 2008 Standard は最大 1 つ，同 Enterprise は最大 4 つ，同 Datacenter は無制限）

SQL Server Standard：仮想マシンで割り当てられたプロセッサに応じて課金

SQL Server Enterprise：物理マシンに搭載されているプロセッサ数で課金

Exchange Server や SharePoint Server には仮想環境向けのライセンス体系は用意されていない。

・Redhat Linux

最大 2CPU をサポートする「Red Hat Enterprise Linux」は，ゲスト OS として Red Hat Enterprise Linux を 4 つまでサポート

CPU 数無制限の「Red Hat Enterprise Linux Advanced Platform」では，ゲスト OS として Red Hat Enterprise Linux Advanced Platform を無制限にサポート

この範囲であれば，追加費用なしで Red Hat Enterprise Linux を仮想マシン上で利用できる。

・Nobel

SUSE Linux Enterprise Server を 1 つ購入することで，同一物理サーバー上でゲスト OS として SUSE Linux Enterprise Server を無制限に利用可能

・Oracle

Soft Partitioning ポリシーに従い，物理マシン上に搭載されているすべてのプロセッサ＆コア数をライセンス対象とする。仮に 1CPU を割り当てた仮想マシン上でオラクル製品を利用する場合でも，物理マシンの全 CPU 分のライセンスが必要になる。

・Symantec

Symantec Backup Exec は物理マシン単位で課金されるライセンス体系となっている。製品そのものが仮想環境に対応しているので，ライセンスも考慮されたものとなっている。

ウィルス対策製品はノード数に応じたライセンス体系となっており，仮想マシンであるかどうかは区別されていない。そのため仮想マシンの数だけカウントされることになる。

3.2 クラウド

3.2.1. クラウドの定義

クラウド・コンピューティング（Cloud computing）とは，ネットワークを介して IT リソース（ハードウェア，OS 及びミドルウェア，アプリケーションソフトウェア等）を使用する IT サービスの利用形態である。クラウドとは雲という意味であるが，これは情報システムのネットワーク構成図を描く際に，インターネット全体という意味で「雲」の絵を慣習的に使用することに由来するとされている。利用者がインターネット（雲）の先に用意されている IT リソースの実装の詳細を知らないまま（あるいは気にせずに）それらのリソースを利用することから，「クラウド・コンピューティング」と呼ばれるようになった。（以下「クラウド」と称す。）

クラウドでは，ハードウェア，OS 及びミドルウェア，アプリケーションソフトウェアなど全ての IT リソースが提供される。インターネット回線とブラウザーがあれば，ユーザーはこれらのリソースをネットワーク経由でいつでもどこでも調達できる。またシステム構築やシステム運用管理の専門知識や技術がなくても，IT を手軽に利用できることが特徴である。クラウドで提供されるリソースの所有権・一次使用権は一般的に提供者（サービスプロバイダー）側にあり，費用もサービス単位の月額制あるいは従量課金制が採用されているものが多い。顧客企業側から見ると「IT の所有から利用へ」というパラダイムシフトが起きており，社会的にも大きな注目を集めている。

3.2.2. クラウドの分類

クラウドには向先別の分類と提供するリソースの階層別の分類がある。主な分類は下記の通りである。本稿ではまず向先別分類を解説し，続けて階層別分類について解説する。

表 3-3　クラウドの分類

分類法	分類
向先別分類	パブリッククラウド
	プライベートクラウド
	ハイブリッド型
提供するリソースの階層別分類	SaaS（サース）Software as a Service
	PaaS（パース）Platform as a Service
	IaaS（イアース）Infrastructure as a Service HaaS（ハース）Hardware as a Service
	DaaS（ダース）Database as a Service / Desktop as a Service

3.2.3. パブリッククラウド

パブリッククラウドとは，インターネットを経由して不特定多数の利用者に向けて提供される IT サービスであり，一般的に「クラウド」という場合はこのパブリッククラウドを指すことが多い。パブリッククラウドとは，次項のプライベートクラウドと対比する場合の用語と考えることもできる。

パブリッククラウドを提供する事業者は自社で大規模なデータセンターを保有し，サーバー仮想化技術によってクラウド環境を構築している。また提供される IT 基盤にオープンソースのソフトウェアを積極的に活用して，コスト削減を図っていることも特徴である。

表 3-3　代表的なパブリッククラウドサービス　（抜粋）

サービス分類	サービス名	事業者名
SaaS	Salesforce.com	セールスフォース・ドットコム（米国）
	Google Apps	グーグル（米国）
	Microsoft Office 365	マイクロソフト（米国）
PaaS	Force.com	セールスフォース・ドットコム（米国）
	Google App Engine	グーグル（米国）
	Microsoft Windows Azure Platform	マイクロソフト（米国）
IaaS / HaaS	Amazon Web Services	アマゾン（米国）
	Rackspace Cloud Server	ラックスペース（米国）

DaaS ※	Amazon Simple DB	アマゾン（米国）
	Microsoft SQL Azure Database	マイクロソフト（米国）

※DaaS: Database as a Service

3.2.4. プライベートクラウド

プライベートクラウドは，インターネットやイントラネット等を経由して特定の利用者（顧客企業等）に向けて提供される IT サービスである。顧客企業等はデータセンターのサーバー上で Web アプリケーションソフトウェアを稼働させ，企業内の利用者がイントラネット等を通じて IT サービスを利用する。

プライベートクラウドは，クラウド技術を活用しつつ，あくまで IT リソースは顧客企業等が自社の管理下に置いている点が特徴である。パブリッククラウドと比較して，自社のポリシーに基づいたセキュリティマネジメントや IT サービスマネジメント（サービスレベル管理を含む）を反映しやすいことから，大企業や官公庁などで利用が拡大している。

プライベートクラウドは，クラウド技術を活用した新時代のアウトソーシングサービスであると考えることもできる。一方，IT リソースを自社保有あるいは自社管理下に置かれるため，そもそもプライベートクラウドはクラウドか否か，というような議論がなされる場合もある。いずれにせよ，（顧客企業等の）利用者が IT の設計・構築・設定・運用・保守などの作業から解放され，ネットワーク経由で IT をサービスとして必要な時に必要なだけ利用できるという点から，クラウド的なアウトソーシング形態であると考えられる。

3.2.5. ハイブリッド型

現在のようなクラウドへの移行期においては，前述のパブリッククラウドとプライベートクラウドの混合型が採用される場合も多い。わが国ではシステムインテグレータが提供するハイブリッド型クラウドの提供事例が増えている。

表 3-4　代表的なプライベートクラウドサービス

サービス名	事業者名
Biz Cloud	NTT データ（日本）
Enterprise Private Cloud	日本 IBM（日本）
Cloud ISLE	ビットアイル（日本）
たよれーる	大塚商会（日本）

3.2.6. クラウドの階層別分類

クラウドでは提供するリソース別に階層があり，それぞれ「XaaS: Xxxxx as a Service」

の形で呼ばれている。

(1) SaaS（Software as a Service：サース）

クラウド・コンピューティング・サービスの中でアプリケーションソフトウェアを提供するものである。基本的にはASP（Application Service Provider）と同義語と考えられる。

SaaSという名称は，2004年頃に米国Salesforce.com（セールスフォース・ドットコム社）が営業支援用ソフトウェアをインターネット・サービスとして提供した頃から広まり始めたとされる。従来型のASP事業者によるサービスとは異なり，同社のSalesforceサービスはカスタマイズ性に優れており，自社のアプリケーションとのデータ連携等も自由にできたため，急速に普及していった。

現在米国では，営業支援分野に限らず，グループウェア，セキュリティ管理，財務会計，HR（人事/給与/勤怠），タレントマネジメントなどの各方面でSaaSが急速に普及し始めている。

SaaSで提供されるアプリケーションソフトウェアは，パッケージソフトウェアと同様，自社でコントロールできる部分は少ない。提供されるサービス内容，サービスレベルが自社の業務に適合していることを確認することが前提となる。

表3-5　代表的なSaaS

サービス名	事業者名
Salesforce.com	セールスフォース・ドットコム（米国）
Google Apps	グーグル（米国）
Microsoft Office 365	マイクロソフト（米国）
Lotus Live	アイ・ビー・エム（米国）
SilkRoad Technology	シルクロードテクノロジー（米国）
Cybouz	サイボーズ（日本）
Zoho Business	大塚商会（日本）

(2)PaaS（Platform as a Service：パース）

クラウド・コンピューティング・サービスの中で，アプリケーションソフトウェアが稼動するITプラットフォームを提供するものである。具体的には，データセンターにあらかじめ用意されたハードウェア，OS，ミドルウェア，フレームワークなどのリソースが，インターネットを経由して従量課金制のITサービス形態で提供される。

PaaSを利用することで，顧客は必要なIT基盤を即座に，しかも必要な期間だけ利用することができる。例えば，アプリケーションソフトウェアの開発業務にPaaSを利用すると，開発用機器の調達工期が不要となるため，開発プロジェクトの工期（特に立ち上げ準備期間）を短縮することができる。自社データセンターを保有しないSaaS事業者が，サービス

提供基盤として他社の PaaS を利用する例も多い。

表 3-6　代表的な PaaS

サービス名	事業者名
Force.com	セールスフォース・ドットコム（米国）
Google App Engine	グーグル（米国）
Microsoft Windows Azure Platform	マイクロソフト（米国）
Lotus Notes	アイ･ビー･エム（米国）

(3)IaaS（Infrastructure as a Service：イアース）

クラウド・コンピューティング・サービスの中で，ハードウェアやネットワーク回線等の IT インフラストラクチャーを提供するものである。顧客はインターネット経由で IT インフラにアクセスし, その上に自ら OS やアプリケーションソフトウェアをインストールして利用する。

以前は HaaS（Hardware as a Service）と呼ばれていたが，近年ネットワーク等を含む IT インフラの提供がなされることが多くなったため，IaaS（Infrastructure as a Service）という用語が広く使用されるようになった。

従来型のホスティングサービスと異なる点は，仮想化技術を活用して仮想インフラを提供する点である。仮想化技術によりユーザーはサーバーやストレージの実装の詳細を気にせずに，IT インフラのサービスを利用できるようになった。

表 3-7　代表的な IaaS / Haas

サービス名	事業者名
Amazon Web Service	アマゾン（米国）
Rackspace Cloud Server	ラックスペース（米国）
ニフティクラウド	ニフティ（日本）
オンデマンド仮想システムサービス	富士通（日本）

(4)DaaS（Desktop as a Service：ダース）

クラウド・コンピューティング・サービスの中で，デスクトップ機能を提供するものである。シンクライアント，VDI（Virtual Desktop Infrastructure）などの仮想デスクトップを IT サービスとして提供する。ユーザーは Web ブラウザーがあれば，インターネット経由でどこでも自分の仮想デスクトップを操作できる。

仮想デスクトップでは，管理者がサーバー上に複数の仮想マシンを稼働させており，これらの仮想マシンの処理結果画面のみを端末に転送するという仕組みである。端末側はキーボードやマウスなど入力装置の操作情報のみをサーバーに転送する。

DaaS がもたらすメリットは，端末にデータを保存しないためセキュリティや事業継続性の確保が容易である，端末のセキュリティパッチやソフトウェア資産管理などについてサーバー側で一括管理ができる，などの点が挙げられる。

なお，同じ DaaS という略称で表現されるクラウドサービスとして，Database as a Service（データベース管理システムを提供するクラウド）や，Data Storage as a Service（ストレージ装置を提供するクラウド）がある。前者は PaaS，後者は IaaS の一種と考えることができる。

表 3-8　代表的な DaaS（Desktop as a Service）

サービス名	事業者名
Amazon Simple DB	アマゾン（米国）
Microsoft SQL Azure Database	マイクロソフト（米国）
Xen Desktop	シトリックス（米国）
VMware Virtual Desktop Infrastructure	ヴイエムウェア（米国）
ワークプレイス LCM サービス	富士通（日本）
スマートクラウド・デスクトップ	NTT コムウェア（日本）
IIJ GIO	IIJ（日本）

3.3 OSS

3.3.1. OSS の定義

OSS（英：Open Source Software）とは，「ソフトウェア著作者の権利を守りながらソースコードを公開することを可能にする」というオープンソースの概念に基づき，ソフトウェアのソースコードが無償で公開され，改良や再配布を行うことが誰に対しても許可されているソフトウェアのことである。

一般的にソフトウェアのソースコードは知的財産として秘匿され，他社に使用許諾される場合はライセンス料が徴収される。そのためソースコードへのアクセスは制限されることが多く，これらのソフトウェアは「プロプライエタリ・ソフトウェア」と呼ばれる。これに対して OSS は，あらゆるユーザーが良質のソフトウェアを利用可能となるように，ソースコードを共有の知的財産として扱い，修正や改良を重ねていくことでより良いソフトウェアに育んでいくことを志向している。

The Open Source Initiative（OSI）では，「オープンソースの定義」（The Open Source Definition）と呼ばれるライセンス文書を策定，公開しており，当ライセンスに準拠していると認められたソフトウェアに対して「OSI 認定マーク」を与える活動を行っている。

表 3-9　オープンソース・ライセンスの要件（出典：The Open Source Initiative）

#	内容
1	自由な再頒布ができること
2	ソースコードを入手できること
3	派生物が存在でき，派生物に同じライセンスを適用できること
4	差分情報の配布を認める場合には，同一性の保持を要求してもかまわない
5	個人やグループを差別しないこと
6	適用領域に基づいた差別をしないこと
7	再配布において追加ライセンスを必要としないこと
8	特定製品に依存しないこと
9	同じ媒体で配布される他のソフトウェアを制限しないこと
10	技術的な中立を保っていること

3.3.2. 代表的な OSS

OSS として提供されている代表的なソフトウェアとしては，リーナス・トーバルズによって最初に開発された UNIX 互換のオペレーティングシステム（OS）である Linux を挙げることができる。Linux は OS としての改良が進められ，ハイエンドサーバーを含む幅広い領域で，高い市場占有率を獲得するに至っている。また Linux ディストリビューションと呼ばれる商用パッケージソフトウェアへの応用も活発に行われている。

Linux の他にも，Web サーバーの Apache，データベース管理システムの MySQL，プログラミング言語の Java，統合開発環境（IDE）の Eclipse，スクリプト言語の Perl，PHP，Python，Web ブラウザーの Firefox，といったソフトウェアが，OSS として提供されている。近年では，IT ベンダーが OSS のコミュニティに参加し，開発活動に参画する例も増えている。

表 3-10　代表的な OSS

#	分野・領域	ソフトウェア名称
1	オペレーティングシステム	Linux
2	Web サーバソフトウェア	Apache
3	データベース管理システム	MySQL
4	プログラミング言語	Java
5	統合開発環境（IDE）	Eclipse
6	スクリプト言語	Perl
7	同上	PHP

8	同上	Python
9	Web ブラウザー	FireFox

3.3.3. クラウド育ちの OSS

近年，クラウド・コンピューティングの世界で新型 OSS が広く活用され始めている。代表的なものを以下に示す。過去の OSS がそれぞれの分野の代表的な商用ソフトウェアをキャッチアップしようと機能強化してきたのに対し，これらの新型 OSS の特徴はクラウドで使われている技術を基にしているという点である。

表 3-11　社内向けクラウド構築のために活用できるソフトウェア（出典：IPA）

#	分野・領域	ソフトウェア名称
1	仮想化機構に関するソフトウェア	Oracle VM VirtualBox KVM Xen Citrix XenServer
2	システム監視・管理に関するソフトウェア	Groundwork Monitor ZABBIX Hinemos Nagios Xymon Virt-manager oVirt Eucalyptus Proxmox Virtual Environment ConVirt OpenNebula Nimbus
3	利用者向け基盤認証ソフトウェア	OpenSSO Shibboleth Higgins SimpleSAMLphp OpenDS OpenLDAP

4	分散処理基盤	Hadoop SkyNet Gfram CouchDB HBase Hypertable Voldemort Cassandra

3.3.4. OSS のライセンス概念

OSS のライセンス条件は様々であるが，一定の条件の下でソフトウェアの使用，複製，改変，（複製物または二次的著作物の）再頒布を認めている他，以下の 2 条件はほぼ共通している。

⑴無保証であること

オープンソースの性質上，ソフトウェアやその二次的著作物は元の著作者でも制御しきれない形で流通し，元の著作者がそこから直接に利益を得ることは難しい。そのため，ソフトウェアは「有用であるとは思うが無保証である」と謳っている。つまり，著作者は，そのソフトウェアについて，予期した動作をする／しないの保証をしない。また，その動作の結果何らかの損害をもたらしたとしてもそれを保障しないものと定めている。

⑵著作権表示を保持すること

オープンソースは一定の条件内で自由な利用を認めるものであって，著作権を放棄するものではない。むしろ，「一定の条件」を守らせるための法的根拠は原著作者の著作権に求められる。そのため，多くのライセンスは適切な形でソースコードや付属文書に含まれる著作権表示を保持し，つまり二次的著作物を作った者が自分で 0 から作ったように偽らないことを定めている。

ソースコードを伴わないバイナリ形式のみでの配布を認めているライセンスでは，その際にも付属文書に著作権表示を記載するように定めているものもある。

4. クラウド・コンピューティング環境での SAM の考え方

4.1 業務アプリケーションを SaaS の形態で利用する場合

オンプレミス型と比較して企業 IT において様々なメリットが期待される SaaS。「所有」から「利用」に情報システムのスタイルが変化することにより多くのメリットがあるとして大きな期待が膨らんでいるが，多くの場合は「俊敏性の向上」が最大の価値提案となっている。SaaS を採用すれば，極端な話し契約したその日からアプリケーションを使い始めることができる。SaaS によるサービスインのスピードはとても魅力的だ。

しかし，その一方では様々な課題点なども挙げられている。例えば米国のメディアなどでは，「SaaS はコンプライアンスの地雷原」や，「SaaS：CIO にとってのコンプライアンスの悪夢」などと報道されている。なぜなら，SaaS を利用することで「俊敏性」や「スピード」を得ることはできるが，ユーザー企業がシステムを「所有」せずに「利用」しているからといって，今まで存在した法的責務から開放されるわけではなく、むしろ法的責任をわかりにくくしてしまうものだからだ。

ユーザー企業は「サービス」だから，情報システム部門が関与することなく「利用部門」がサービス提供者と契約するだけでシステムの利用を開始できてしまう。しかし，そこにはコンプライアンスリスクや，コストの問題など様々な課題が潜んでいることを忘れてはならない。

例えば，

- サービス提供者へアプリケーションの製造，管理，管理義務の対象となる情報の保管を委譲しており、情報セキュリティのレベルはサービス適用者に委ねられていること
- サービス提供者側の問題により発生する法的なコンプライアンスリスクを回避するための予防的戦略の立案
- サービス提供者のコンプライアンス対応や報告を監督するための適切な制御機能を契約において交渉するためのアディショナルな管理コスト
- 契約の複雑化と対応コスト

などが考えられる。

また，SaaS という市場自体が立ち上がりの時期であることから，経験値の低いサービス提供者の存在もある。経験値の高いサービス提供者であれば様々なシナリオに対するプロセスの定義なども周到な準備がなされているだろうが，そうでない場合，問題などに対する対処が定義されていないため，サービスが長期間にわたって停止されたり、情報セキュ

リティの脆弱性によるセキュリティ事故に会うことも考えられる。たとえば、データアクセスに対する適切なプロセスが欠落していることにより，個人情報や取引情報などの重要データへのバックドアが開放されてしまうことにもなりかねない。SaaS 導入のメリットは大きい。しかし，ソフトウェアを所有せずサービスとして利用するからといって，まったくソフトウェア資産管理や IT 資産管理から開放されるというわけではない。

それではどのような点に留意して SaaS 導入をするべきなのか？ 本章では以下の観点から留意点をまとめたので参考にしてほしい。

- 組織
- 契約（コンプライアンス，SLA，EULA）
- セキュリティ
- 運用管理（SLA／SLM の運用）

4.1.1. 組織

業務アプリケーションを SaaS で利用する場合であっても単純に SAM の対象外とはしない方が良い。

業務アプリケーションが SaaS で提供されている場合，サービスを利用する利用部門（契約部門）は，「SAM の業務アプリケーションを SaaS で利用する場合の規程」に則ってサービスを利用し，管理者はその運用状態が規程に則って利用されているかどうかを管理しなければならないからである。

以下の図に組織の役割の一例を示す。

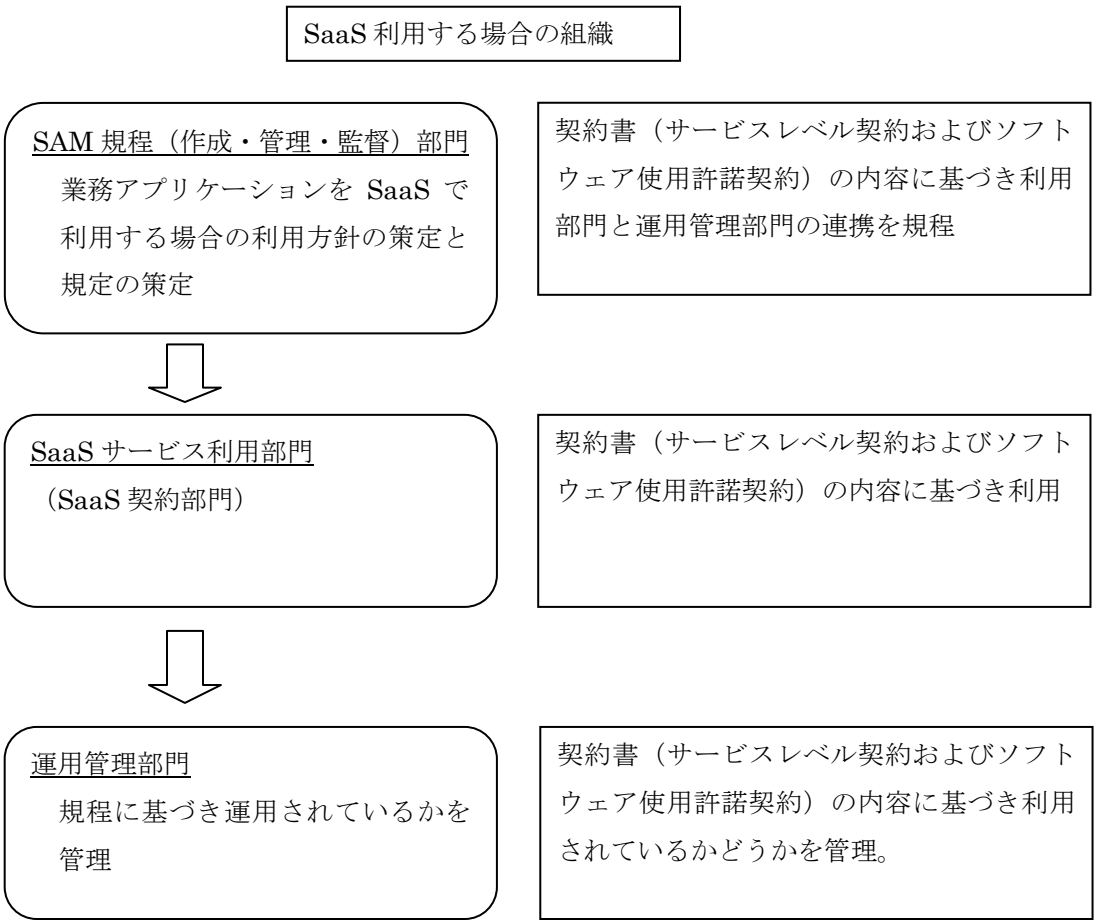

図 4－1 SaaS 利用する場合の組織

（1）SAM 規程部門

SaaS の利用においては当然 SAM の組織管理プロセスに則って方針，プロセスおよび手順が規程されなければならないが，「利用」するサービスとして提供される SaaS の性質から，サービス利用部門が利用可能なサービスの範囲，サービス提供者としての適格性の基準，SLA（Service Level Agreement：サービスレベル合意）で網羅すべき項目および課題，SLM（Service Level Management：サービスレベル管理）などが定義されていることが望ましい。

また，サービス利用部門が個別に契約できる利用可能なサービスの範囲と，企業による包括契約により利用されるべきサービスなども規定されていることが望ましい。サービスの利用によるメリットを最大化するためにはデータ連携などが必要に応じて行われるシステム連携を考慮しなければ，サービスがあちこちで「孤立」するシステムの乱立という結果を招くことになる。「孤立」するシステム利用は，結局は無駄なコストの上昇を招き，スピードだけを求めてコスト削減のメリットを享受することが不可能となる。

（2）サービス利用部門

スピード，コスト，利便性などのバランスを考慮し，規程に則り利用サービスを決定することが望ましい。SLA／SLM の成功には利用部門の参画が必須であるため，早期に関係各部門との共通認識を形成することが肝要である。

（3）運用管理部門

SaaS の実行環境となるクライアント環境を提供し，SLA／SLM の運用管理の実働部門として，規程に則りサービス利用部門の利便性を損ねることなく、サービスインのスピード, コストなど SaaS のメリットを利用部門が享受すべく支援する体制とプロセスを形成することが肝要である。

SaaS を利用する場合，サービス利用部門は「うちの部門がサービス契約をするのだし，コスト負担しているし，契約も部門契約だから他部門に干渉されずに進められる。運用管理も SaaS 提供者が行うわけだし，情報システム部門にお世話になることもない」と考えてしまうことが多い。この考えが企業内に蔓延するとサーバー統合以前のサイロ型システムが乱立し，孤立したシステムにデータが重複して散在し，結果としてデータ整合性の取れない，全体最適化が不可能で情報統制がとれないコスト高な情報システムの利用を推進することになる。

例えば，企業の顧客管理システムを各部門において自部門で使い勝手が良いと考える SaaS の利用を進めていくと，結果としてデータ整合性の取れない顧客データが部門毎に存在することになる。また，SaaS 提供者との契約を全社的な内部統制ポリシーやセキュリティ，法令順守などコンプライアンスに対応を考慮せずに進めた場合，結果として企業として法的責務を果たせずに市場における信頼の失墜などの原因となることも考えられる。

これらのことから，SaaS 提供者との契約は社内の規程に則り，全体最適化を前提とした内部統制ポリシーやセキュリティ，法令順守など，業務影響分析（BIA），リスクアセスメントなど考慮されたかたちでサービス契約，SLA（Service Level Agreement：サービスレベル合意または契約）が行われ，システム利用の効果測定を SLM（Service Level Management）により実施し，SLA の改善を行うライフサイクル管理が実施されることが望まれる。

4.1.2. 契約（コンプライアンス，SLA，EULA）

SaaS 提供者との契約時に留意すべき点は多い。SaaS 提供者が提示するサービス契約や SLA では利用者の利益や法的責務への考慮が不足している場合も考えられるので、十分な

検討や，網羅すべきポイントを規程にまとめ、利用部門や関係各部門の合意と共通認識の上，契約交渉を行うことが望ましい。また，ソフトウェアがサービスとして提供されているとしても，ソフトウェアコンポーネント（フォント，常駐プログラム，アプリケーションの一部機能を担うソフトウェアコンポーネント）などがクライアント PC へインストールされ、提供される場合がある。この場合は，コンポーネントの使用許諾契約が，サービス契約とは別途提供されることも考慮したほうがよい。ソフトウェアコンポーネント毎に使用許諾契約がある場合は，これら契約に基づく運用が必要となるので運用管理部門の関与は不可欠となる。これらを怠った場合，SaaS というサービス提供の形態であってもソフトウェア使用許諾契約違反として損害賠償請求されることも考えられる。少なくとも以下の点に留意し契約することが望まれる。

- ✓ SaaS 提供者の選定（提供者に依存するセキュリティ対策と継続性）
- ✓ システム間連携
- ✓ カスタマイズ
- ✓ サービス終了時のデータ移行
- ✓ ソフトウェアコンポーネントの使用許諾契約
- ✓ 財務情報，営業機密情報
- ✓ SLA：サービスレベル合意（契約）
- ✓ サービスサポート

（1）SaaS 提供者の選定（提供者に依存するセキュリティ対策と継続性）

SaaS の利用は，自社データを外部に預けるということであり，SaaS 提供者のセキュリティレベルにデータの安全性を完全に依存することである。また，サービス利用の継続性が，SaaS 提供者の存続性と等しいので，提供者の選択は慎重に行わなければならない。

SaaS 提供者を選定する際の留意点としては SLA（Service Level Agreement）や企業の財務諸表，セキュリティポリシー，データセンターの堅牢性，インターネット接続回線，ハードウェア，ソフトウェアなど基盤，アプリケーションや Web システムとしてのセキュリティ，脆弱性診断の報告書など加味し慎重に問題がないことを確認し、契約しなければならない。もちろん，重要な業務や機密性の高い情報を処理するサービスと，比較的機密性の低い情報を処理するサービスでは選定条件を分けて検討する。

以下に継続性における選定条件の一つとして，SaaS 提供者の提供基盤の構成例をあげる。

① 建物施設を所有し，インフラストラクチャーのハードウェア，ソフトウェアを所有し，アプリケーションを所有。全ての構成要素を所有した状態でサービス化を行い，

ユーザー企業へサービスを提供している。

② 建物施設を利用し（ハウジングの状態），インフラストラクチャーを所有，アプリケーションを所有，これらを組み合わせてサービス化を行い，ユーザー企業へサービスを提供している。

③ 建物施設を利用し，インフラストラクチャーを利用し（PaaS 利用の状態），アプリケーションを所有し，これらを組み合わせてサービス化を行い，ユーザー企業へサービスを提供している。

④ 建物施設を利用し，インフラストラクチャーを利用し，アプリケーションを利用し，これらを組み合わせてサービス化を行い，ユーザー企業へサービスを提供している。

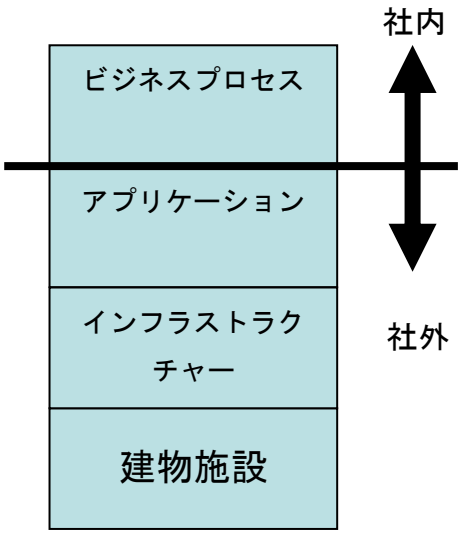

図 4－2 基盤のレイヤー

①から④の順に SaaS 提供者の制御範囲が制限される。また，建物施設やインフラストラクチャー自体が海外にある場合は，何らかの問題が発生した際に，その対応に時間を要し，かえって運用コストが高くなるという可能性も考慮する必要がある。

①の提供者の例としては，電力系，通信系などそもそも IDC 事業を経営の主軸とし，インフラを所有している事業者があげられる。

②の提供者の例としては，大手 IDC 事業者の施設をハウジングしきた準大手のサービス事業者があげられる。PaaS や IaaS などのサービスを提供している場合もあるだろう。

③の提供者の例としては，現在増加している PaaS などを利用して自社開発したアプリケーションの SaaS 展開を図るアプリケーション開発事業者があげられる。

④の提供者の例としては，今後増加すると考えられる PaaS 上にアプリケーション開発事業者から OEM 提供を受けサービスを提供するサービス事業者があげられる。

SaaS で提供されるアプリケーションを分類すると以下のようにまとめられる。

（ア）基幹系，情報系，顧客管理系

（イ）セキュリティ系，運用管理系，データ保存系

（ウ）業務支援系

独自カスタマイズやシステム間連携が必要で，高いデータの機密性や継続性が求められるシステムであれば SaaS 提供者の制御範囲に制限が少ないほうが最終的なコストや SLA の柔軟性が高いと考えられる。一方で，中小企業や（イ）の分類のような、独自カスタマイズやシステム間連携が限られた範囲で足りるシステムであれば、SaaS 提供者の制御範囲が制限されていても問題とならない範囲が存在するだろう。ユーザー企業独自のカスタマイズへの対応やシステム間連携を高いデータ機密性と同時に求める場合は，アプリケーションを所有していなければカスタマイズや機能対応はコスト高となる。具体的には，アプリケーションを OEM 提供で受けてサービスを提供している事業者であれば，カスタマイズはアプリケーションを製造しているソフトウェアメーカとの交渉となり，実際の開発もユーザー企業とソフトウェアメーカの間にプロジェクトマネジメントとして関与することとなり，複雑でコスト高な対応となることが考えられる。技術情報などもアプリケーション開発元であれば，独自カスタマイズやシステム間連携に必要となるソースコードレベルでの理解のある技術者が対応することもあり得るが，開発元でなければユーザー企業の要件を反映させるための技術的要件定義に時間を要することも考えられる。また，著作物にたいする著作権や使用権などの交渉も開発元と直接交渉できるか，そうでないかでは大きく工数に影響することが考えられる。具体的には，例えば交渉相手が開発元ではなく，開発元が海外の企業であった場合などは，著作物に対するカスタマイズ部分の著作権や使用権の交渉には海外にある開発元の許可や，適法もその開発元企業が存在する国の知的財産法に則って契約書を英語にて取り交わさなければならないなど工数を増加させることも考えられる。

（２）システム間連携

システム間連携を行うには，連携部分の作りこみが発生するため，自社でシステム開発者を用意するか，開発を外部委託する必要がある。

簡単なカスタマイズであれば SaaS 提供者が API（Application Programming Interface）を提供し，連携が可能な場合もあるが，既存システムや他社 SaaS アプリケーションとの連携が可能かどうかは技術要件を確認しなければならない。

SaaS アプリケーションの API は，各サービス提供者が独自に定義しているが，連携を意識した共通の API やデータ構造を持たない限り異なる SaaS 提供者のシステム連携を行うことは困難である。システム間連携が必要となるシステムを SaaS で利用する場合は，

十分に連携の実現可能性，費用対効果，開発コストなどを考慮して判断するべきである。今後はデータを XML 化し，Web Service 連携によるシステム連携の可能性を提供する SaaS 提供者も現れると考えられることから，SaaS 提供者との契約時に提供者がどこまでシステム間連携をサポートするか，技術情報の提供範囲，今後の対応予定なども事前に確認するべきである。

（3）カスタマイズ

今日の SaaS は，パラメータ化などにより，プログラムの改修を行わず設定変更レベルの作業でカスタマイズが可能である。

しかし，設定変更レベルでユーザー企業が期待する全ての機能を実現できるわけではない。したがって，将来的な利用用途や業務拡大で必要となる機能が事前にわかっている場合は，提供されるカスタマイズ機能で実現可能かどうかを検討しておく必要がある。ユーザー企業独自のカスタマイズが発生する場合は，実現性やコスト，著作権，サービス移行時の再利用性など SaaS 提供者と事前に協議して合意しておかなければならない。

カスタマイズは結果的に多額のカスタマイズ費用や特別に保守費用やメンテナンス費用がひつようとなることが考えられる。加えて，レスポンスの低下，SaaS 提供者の通常保守や機能拡張の際に問題が発生する場合もあり注意が必要である。

また，カスタマイズの内容はしっかりと文書化しておく必要がある。サードパーティのコンサルティング会社を利用する場合は，プロジェクトのオーナーシップや，責任の所在など明確にしなければならない。SaaS 提供者の乗り換えに伴うコストや，開発作業のリスクを共有することにコンサルティング会社が同意するかを事前に確認する必要もある。選択肢としては，SaaS 提供者のソフトウェアのカスタムフロントエンドを作成することで，自社でカスタマイズを行い，その成果を保持することができる。

この場合，自社でカスタマイズする部分に自社のノウハウが含まれて，市場における差別化や企業優位性となるプロセスが実装されたりする。SaaS はサービスインのスピードやコスト面でのメリットがあるが，実装されたノウハウが SaaS 提供者のノウハウとして吸収され，将来のバージョンアップに利用され競合他社へも一般的なサービスとして提供されてしまうという危険性を含んでいる。ユーザー企業内での使用者数が少ない場合は，オンプレミスのシステムより SaaS のメリットが勝ることになるが，それでもノウハウの流出が企業優位性を損なう可能性を秘めている場合は，あえて SaaS を選択しないということも考慮する必要があるだろう。もちろん，ある程度の流出を想定しながらも SaaS 提供者に対して，SaaS システムに実装するカスタマイズ部分の著作権に関する契約により保護し，必要であれば損害賠償訴訟により一般公開を妨げるという選択肢も考えられる。

（4）サービス終了時のデータ移行

SaaS の利用を終了する際に，SaaS 提供者のシステムに蓄積されているデータから新たなシステムまたは異なるSaaS提供者のシステムにデータ移行を行う必要がある。SaaSの導入検討の際に次期システムへの移行を具体的に計画することは困難であるため，移行に必要と考えられるデータの権利（利用期間中に入力したデータや，入力データから得られる集計結果，加工されたデータなどを契約解除時に再利用する権利，SaaS 提供者のデータ消去処理のプロセス），機能面での出力可否（CSV，XML）などデータ出力の対応状況など事前に確認する必要がある。

（５）ソフトウェアコンポーネントの使用許諾契約

SaaS を利用する場合でも全てのソフトウェアがブラウザーで提供されるとは限らない。アプリケーションによってはネットワークの負荷やクライアントにおける処理パフォーマンスを考慮して一部機能をソフトウェアコンポーネントとしてクライアント PC の環境にインストールするものや，適宜ダウンロードされメモリー上で利用されるもの，テンプレートやフォントなど利用頻度が高いのでローカルハードディスクに保存されるものなどが考えられる。これらのソフトウェアコンポーネントは，各コンポーネントが SaaS 提供者の著作物として EULA（End User License Agreement：使用許諾契約）がインストール時に結ばれることが多い。これら契約も，サービスの利用者の利用形態を考慮して SaaS 提供者と合意できるかの可否も含め検討する必要がある。

（６）財務情報，営業機密情報

SaaS を利用して財務情報や営業機密情報を扱う場合は，SaaS 提供者が国内法規（商法，会社法，税法，労働法など）に則っているかなど事前に確認しなければならない。また，上場企業が SaaS 型の財務関連のシステムを導入する場合は，金融商品取引法の適用を受けるため，以下の要求事項が考えられる。

１．提供する財務関連のシステムが会計規則の要件を満たしていること
２. IT 全般統制や財務関連システムの IT 業務処理統制に対する経営者評価や監査人監査への協力を提供者が受け入れること
３．日本公認会計士協会の監査基準委員会報告書第 18 号に準拠した監査報告書の提供

これら国内法令対応は必須であり，SaaS 提供者が国内法令対応を疎かにしていると，業務を複雑にするばかりでなく，データ保全，説明責任という観点からも事故を発生させやすいということを認識しなければならない。法令改正に対する対応のスピードや，今後の対応予定など契約時に確認しなければならない。

（７）SLA：サービスレベル合意（契約）

SLA（Service Level Agreement）は，提供されるサービスの範囲・内容・前提事項を踏まえた上で「サービス品質に対する利用者側の要求水準と提供者側の運営ルールについて明文化したのも」である。SaaS によるメリットへの期待値が膨らむ一方，SaaS は様々な課題を抱えているのも事実である。

「SaaS を利用すればソフトウェアの管理や，運用管理の責任を一切免れることができる」，「コストを削減し，短期で導入可能で，レスポンスなどパフォーマンスの高いシステムを利用できる」など過度な期待は，トラブルへと発展する恐れがある。

SaaS 提供者と利用者双方にとっては，適切な SLA の締結が重要であり，定めたサービスレベルを定期的に測定，分析，評価することにより継続的にサービス改善を実現することが必要である。

SaaS を利用している際に発生したトラブルがすべて提供者の責任であるとは限らない。自社所有システムであればインシデント管理などにより原因分析に必要な情報を管理しているが，SaaS として提供されている場合，提供者がどこまでを管理対象の範囲としているのか，利用者への報告対象としているのかにより，利用者が知りえる範囲が決定されてしまう。

何がトラブルなのか，セキュリティ事故なのかの判断基準は企業によって異なり，ポリシーや事業影響度分析（BIA）やリスクアセスメントにより判断される。SaaS 提供者の基準と利用者の基準は，利用者が基準としているポリシー，事業影響度分析，リスクアセスメントの基準に則った合意が SLA に明示されていなければ期待する報告が SaaS 提供者からは提供されないことも考慮しなければならない。

また, 利用者がもとめているすべてについて SaaS 提供者が対応できるわけではないので，それを前提に利用者の責任と提供者の責任を明確にし，SLA に反映させておかなければならない。

しかし，SLA を交渉・管理できる能力を有する情報システム部門を持たない中小企業においては，SaaS 提供者があらかじめ用意している標準的な SLA を締結する際に，本章の確認事項や留意点を十分に確認するとともに, 信頼できる SaaS 提供者の選定を行うことが肝要である。

（8）サービスサポート

SaaS 提供者のサービスはアプリケーションの機能だけの提供にとどまらず，アプリケーションを利用するためのサポートも重要なサービスの一つである。この観点から, SaaS 提供者が IT サービスマネジメントのプロセスを正しく運用できるのか，というサービス運用能力も契約時に見極める必要がある。ヘルプデスクの設置，迅速なトラブル対応などの体制やプロセスについては ITIL に取り組んでいるか，また JIS Q 20000:2007 認証を取得しているかなどについても確認を行う必要がある。

4.1.3. セキュリティ

SaaS 提供者の選定には，安全性の観点から JIS Q 27001：2006（ISO/IEC27001:2005）の要求事項を基本としたセキュリティ対策の実施状況を確認することが重要である。更に，Web 脆弱性検査など，第三者による安全性検証試験／セキュリティ診断を定期的に実施し，その結果をユーザー企業に対して公開していることを前提条件と考えるべきである。サービスの継続性，信頼性の高さを判断する基準としても，これらの対応を含む，プライバシーマーク付与認定，ISMS 認証取得，情報セキュリティ監査制度の利用などを行っているユーザー企業においては，必要に応じて利用者の基準に応じた監査を行うことができるかどうかも重要な判断要素となる。以下の点について留意点を記述する。

- ✓ 機密性
- ✓ 完全性
- ✓ 可用性
- ✓ データ保護
- ✓ アカウント管理

（1）機密性

SaaS ではデータは外部に委託され，他社のデータと同じデータベース上で管理されている。したがって，データベースのセキュリティ上の懸念事項については自社のデータベースセキュリティ以上にデータ機密性の高さに応じた SaaS 提供者のセキュリティ管理能力の考慮が必要である。

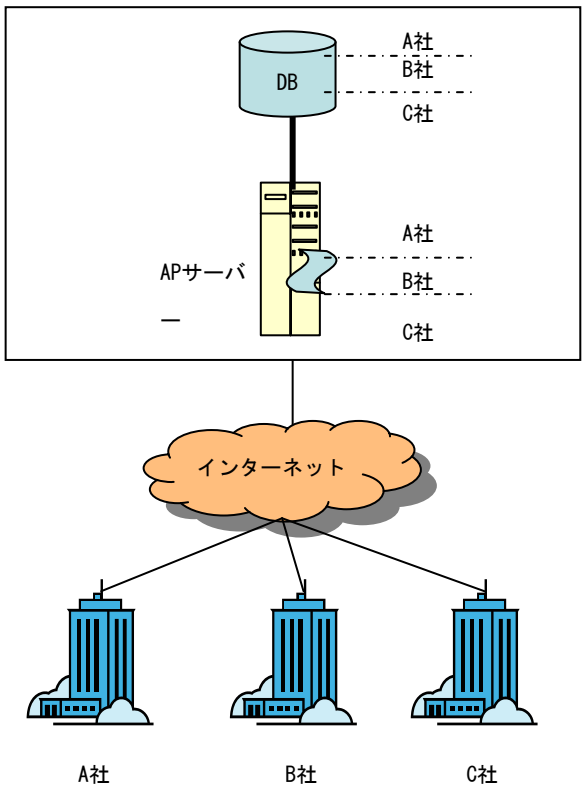

図 4－3 SaaS 型サービス ： マルチテナントにおけるデータベース

例えば，SaaS に使用されているデータベースの製品，バージョン，セキュリティパッチの適用ポリシー，システム構成など，必要なセキュリティ対策が行われているかどうか自社システムのセキュリティポリシーとの比較など行い機密性に問題がないかどうか確認しなければならない。

また，データベースに蓄積されているデータの種類や特性，蓄積される情報の種別，情報管理に関するアクセス制御の内容なども明確にしておく必要がある。SaaS で提供されているユーザー権限だけでは，今まで自社で運用してきたアクセス制御や情報の閲覧権限の制御など厳格な情報管理ができなくなる可能性がある。社内ポリシーに則った情報統制など，内部統制にかかわるコンプライアンス構築の一環として社内ポリシーの運用にそぐわない SaaS であれば実運用に耐えることができない。

Web アプリケーションである SaaS は，機密性の確保のため通信として HTTPS を用いることが要求される。しかし，通信路の保護だけでは十分ではなく，預託したデータを記録したハードディスク，光学メディア，USB メモリーなどの記憶媒体の管理状態やデータベースへの直接アクセスによる情報漏えいを防ぐための適切なアカウント管理など，データ保護の管理策についても確認する必要がある。

（2）完全性

データが漏えいしなくても改ざんされることで信頼性が損なわれる。また，消去されれば業務の継続ができなくなるなど，重大な問題が発生するため，預託データの完全性，整合性検証について対策が施されていることを確認する必要がある。

情報システムの効率的利用にはデータの再利用が不可欠であるため，データをダウンロードして加工できるなどの手段が提供されていなければならない。

ダウンロードしたデータは独自のデータフォーマットではなく，標準的な CSV, TSV, XML などで提供され，再利用性の高いデータでなくてはならない。

（3）可用性

SaaS の特徴として柔軟なカスタマイズ機能や，マッシュアップによる機能の融合などがある。この際に的確な処理が行われているか，データの受け渡しの正確性についても確認する必要がある。また，マッシュアップなどで複数の SaaS 連携を行うサービスを利用する場合は，それぞれの役割と責任範囲を明確にしておかなければならない。

SaaS のサービスの利便性としてネットワークさえ繋がっていれば利用できるという点があるが，低コストを追求しすぎるとサービス継続性の低いサービスであったり，サービスの停止だけでなく，復旧に要する時間も特定できないという問題が発生する可能性もある。国外の SaaS 提供者で国内拠点ではないサポート窓口の場合，時差によりコミュニケーションに時間を要し，問題の詳細説明を外国語で行わなければならないなど，大きな負担としてかえってコスト高になることも考慮する必要がある。

（4）データ保護

重要な業務や機密性の高い情報を扱うサービスの場合，インターネットを経由することから暗号化通信が必須となる。サービス提供されるシステムがユーザー認証時だけでなく，HTTPS 通信や VPN に対応していることや，データの格納形態（分散化，暗号化の有無）の確認，障害時の復旧範囲（復旧できるデータとできないデータの種類），復旧に要する時間，データに関わるサービス提供者のプロセスや要員の数の最小化，アクセスできるデータの範囲などに関して SaaS 提供者に確認し，事前に SLA などで定義しておく必要がある。

SSL が使用される場合は SSL3.0 および TLS1.0 に限定し，脆弱性のある SSL2.0 は使用しないなどのポリシーを定めることが望ましい。

（5）アカウント管理

SaaS はインターネットで提供されているサービスであるため，ユーザー企業が利用するログインページへ誰でもアクセスすることができる。なりすましなどにより攻撃者が

不正ログインを試みるリスクが存在する。このような脅威に対して，連続したログイン失敗時の処理方法（一定期間ログインできなくなる制限処理や，証跡（ログ）の保存，ユーザーの運用管理者への通知，回復手順など）や，パスワードの桁数，使用可能文字種類，有効期限，履歴管理などユーザー企業の規程やセキュリティポリシーに適用可能かどうかを確認し，SLA などで定義しておくことが望ましい。

4.1.4. 運用管理

SaaS 型サービスであれば運用管理は SaaS 提供者が基本的には提供する。しかし，だからといって運用管理の責任がまったく無くなるわけではない。システムの運用面では工数は大幅に削減されるが，SaaS 型だから増える管理の項目も存在する。

ここでは SaaS 形式と自社所有のオンプレミス型とを比較して情報システム部門が調達・導入・運用管理において留意すべき点を検討する。

（1）導入のための事前検討

SaaS 型サービスであっても，業務分析，業務設計／見直しなどはオンプレミス型同様必要である。上流工程の設計を外部に委託する場合は，コンサルティング費用などもオンプレミスと変わらないコストが発生する。

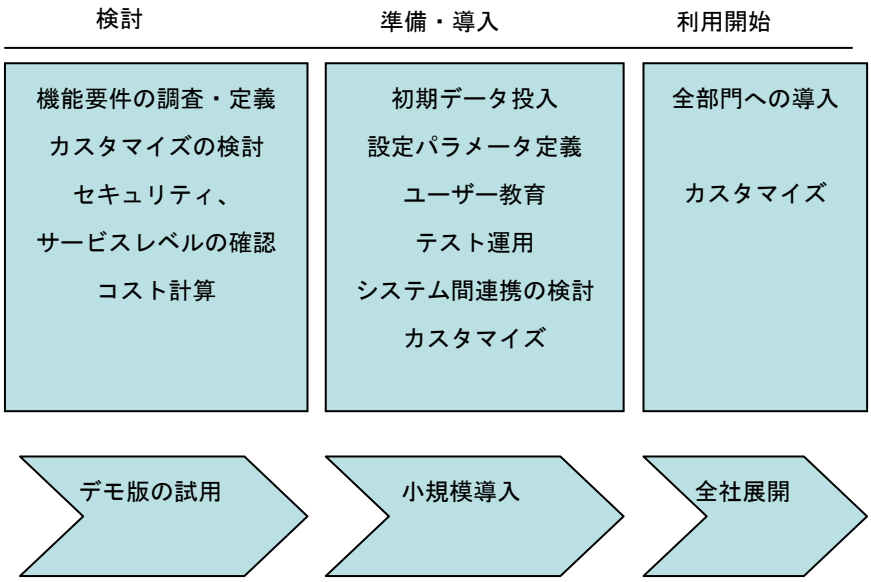

図 4－4　SaaS 型サービス導入フロー

（2）アプリケーションのカスタマイズ

利用部門の要望により利用にあった仕様を要件定義書にまとめ，SaaS 提供者が提供する汎用カスタマイズなどパラメータの設定で対応可能か否かを検討する。表示項目，表示項目名や参照データのレベル，参照アクセス権の設定など自社の情報統制のポリシーに則って運用可能かを確認する。

また，パラメータ設定では対応不可能である独自のカスタマイズが必要と判断される場合は，SaaS 提供者が提供する API を利用する範囲でカスタマイズの開発が自社情報システム部門の開発リソースで対応可能かどうかを調査する。この場合，カスタマイズ部分に自社ノウハウや差別化，企業優位性などが含まれるプロセスを実装する場合，これらカスタマイズ部分が実装される SaaS 提供者のシステムにおいて，SaaS 提供者のスタッフがどのレベルまでの情報アクセスができるのか，あるいは実装することで SaaS 提供者のスタッフによりノウハウが吸収される恐れがある場合は，その防止策を技術的に実現できるのかなどを検討することが望ましい。規程作成部門と SaaS 提供者とのサービス契約の内容に自社ノウハウを含むカスタマイズのコンポーネント部分の著作権保護など盛り込むなどを検討することが望ましい。

また，独自カスタマイズを実装した場合の SaaS 提供者から提供されるサービス（例えば保守，機能バージョンアップなど）が通常通り提供可能か，独自カスタマイズにより別途，機能バージョンアップ時にも同様の独自カスタマイズ部分の差異を吸収するための作業が発生するのかなども考慮することが望ましい。

（3）システム初期導入

ネットワーク環境やクライアント環境が SaaS 型サービスの仕様要求に対応しているかなどを確認し、準備する。対応 OS や，ブラウザーの種類，バージョン，クライアントのローカルにインストールしなければならないコンポーネントの有無，実行に必要となるメモリー，適宜ダウンロードされ実行されるオンメモリのコンポーネントの有無や Java Runtime Edition，.NET フレームワークのバージョンなどが考えられる。

ローカルにインストールされるコンポーネントがある場合は，それらの使用許諾契約書に則った運用ポリシーの策定。例えばフォントやテンプレートなど，インストールされたクライアントからコピーされて当該 SaaS サービスの利用者ではないクライアント PC 上で実行された場合は使用許諾契約違反となり損害賠償請求の対象となることから，使用範囲と利用者管理を行うことが望ましい。

また，ネットワーク負荷を考慮して SaaS サービスのソフトウェアコンポーネントの一部がクライアントに常駐する形式でインストールされる場合，そのダウンロードのサイズや常駐のサイズなど管理し，多数のユーザーが同時にダウンロードしてネットワーク負荷が突然ピークに達しないような計画を立てることも考える必要がある。同時に常駐コンポーネントなどは，常駐するクライアントのメモリーを圧迫する可能性があるので利用者のメモリーの空き容量や CPU の処理能力なども考慮することが望ましい。

ネットワークおよびクライアントのセキュリティ対策の実施はオンプレミス型同様の対応が必要となる。

システム間連携が必要となる場合は，連携部分の作りこみや，連携データの整合性の設計や計画なども必要となる。

（4）ユーザー教育

利用者の操作および情報モラルの教育はオンプレミス型同様の教育が必要となる。SaaS 型サービスはどこからでもアクセスできることから，部外者の誰でもユーザーアカウントの情報さえ入手できれば，なりすましでデータへのアクセスが可能となってしまう。アクセス権限が高いユーザーほど高いモラルやセキュリティの意識を持たなければ，データの消失，漏えい，改ざんなどオンプレミス型以上にセキュリティリスクは高くなる。

（5）クライアント管理

セキュリティの観点から，一般的には OS はもちろん、クライアントにインストールされているソフトウェアを最新のパッチが当たっている状態に維持することが望ましい。クライアントの管理はオンプレミス型同様の管理が求められる。見落とせないのは SaaS サービスによっては提供されるクライアント常駐コンポーネントの実行環境のサポートと，常駐コンポーネントのイメージ管理やバージョン管理，展開などインストールの管理などである。具体的には，例えばクライアントシステムのセキュリティ管理を行うためのクライアント常駐コンポーネントを，クライアント PC にインストールする場合などは，常駐コンポーネントとなるソフトウェアのシステム要件にみあった実行環境（例えばシステム要件に.NET Framework のバージョンや，JRE のバージョンの指定がある場合など）のサポートを情報システム部門が対応しなければならなくなる。さらに，常駐コンポーネントとしてソフトウェアが提供される場合は，これらの在庫管理プロセスも，通常の SAM 同様，情報システム部門が対応しなければならない。

また，当該 SaaS サービス利用者ではないユーザーによる不法なコンポーネントの利用管理も必要となる。例えば，フォントなどが提供されるソフトウェアを SaaS 型で利用しているユーザーが，フォントのファイルをコピーして利用権のないユーザーへ提供して，それが使用された場合に，ソフトウェア使用権を持たないユーザーの不正な使用として検知し管理する仕組みなどが望まれる。

（6）データメンテナンス

SaaS 提供者のサービス内容にもよるが，システムによっては、サービス管理のプロセスを ITIL に則ってポリシーにあったデータのベースラインを保存しておく必要がある。

具体的には，例えば IT 管理の一部のシステム（IT 資産管理システム，セキュリティ管

理システムなど）を，SaaS 提供者によるサービスとして利用し，そのデータを社内にある IT 統合管理システムと連携させているような場合は，SaaS で提供されている IT 管理システムのデータベースが，直接（SOAP などプロトコルを利用した Web サービスによる直接連携により），社内の IT 統合管理システムの構成管理データベースと連携してベースラインを保存している場合を除いては，別途，変更管理に必要となるベースラインを適宜，バッチ処理などによりデータとして保存し，標準的なデータフォーマット（例えば，CSV，TSV，XML など）でエキスポートしたデータを社内の IT 統合管理システムの必要なデータベースへインポートしたりする必要がある。

また，システムのデータにも依存するが，データの整合性や鮮度，正確性を維持するためのデータメンテナンスなどが必要な場合も考えられる。

具体的な例としては，顧客管理システムのマスターデータの管理や，人事管理システムや人事情報を利用する資産管理システムや，職務や職責に紐付けられるセキュリティポリシー管理などリアルタイムのデータ連携が行われない場合はバッチ処理でのデータの洗い替えなどが挙げられる。

（7）ヘルプデスク

SaaS サービスとして提供されるアプリケーションの基本操作に関するヘルプ対応要員の要否も検討することが望まれる。データの再利用のためのダウンロードや，再利用の際の制限や規制などに対応する担当者も必要に応じて定めることも望まれる。

（8）セキュリティ

アプリケーションが SaaS サービスとして提供されている場合には，SaaS 提供者がアプリケーションの稼働環境となるプラットフォームのレイヤーからアプリケーションのレイヤーまでのセキュリティを確保しなければならない。だからと言って，SaaS 提供者のセキュリティが完全なものであるという保証はないため，SaaS 提供者との SLA により，SLM（本項（9）参照）を実施し，利用部門やユーザー企業が要求するセキュリティレベルが維持されているかどうかを定期的に評価する仕組みを持つことが望ましい。セキュリティ事故などが発生した場合には，どこに責任があるのかを明確に切り分けることが可能なレベルの情報が収集でき，改善に必要な要件やプロセスの提案ができることが望ましい。

一般的には、クライアントの管理はオンプレミス型と同様に必要となる。

（9）SLM（Service Level Management：サービスレベル管理）

また, SLM についてはサービス管理の (ITSM) のベストプラクティスである ITIL や, その国際標準である ISO/IEC 20000, 国内標準である JIS Q 20000-2 では, SLM を「サービスレベル管理とは，“サービスレベルを定義，合意，記録及び管理するため”の継続

的なプロセス活動である」と定義している。SLM においては，利用者と SaaS 提供者が協力して問題を確認し，根本原因の分析やプロセスの変更などを通じて問題の再発を防ぐ継続的な問題解決が重要となる。

SaaS サービスの運用管理において重要なことは SLA に基づいた SLM（Service Level Management：サービスレベル管理）の実施である。SLA では契約時にサービスレベルの定義が行われ，サービスレベル測定のための項目が設定される。SLA で定められた条件を SLM によって管理できるようにする。万一トラブルが発生した場合には、業務に与える影響を考慮した上で、優先順位を定め，優先順位が高いものを中心に定義する。あまり多くの項目を管理しようとすると，管理負荷の増加やコストの上昇を招くことにもなるため，システムの重要度に則って、必要最小限に抑えた効率的な運用を行うことが望ましい。重要なことは，サービスの内容が決まり，サービスレベルが設定され、利用者にサービスが提供されてからそのサービス契約が終了するまでの間，SLM は継続的なプロセス監視活動として利用者（ユーザー／事業部門）と運用者（情報システム部門）およびサービス提供者（SaaS サービス提供業者）のすべての関係者によって参画，実施されなければならない活動であるということである。

SLM の一般的な目標としては以下の項目が挙げあれる。

① 提供されるサービスのレベルを定義，文書化，合意，モニタ，測定，報告およびレビューすること
② サービスに対して具体的で測定可能な目標値が策定されるようにすること。
③ 提供されるサービスの品質に対する利用者の満足度をモニタし改善すること
④ サービス提供者と利用者が，提供されるサービスのレベルに対して、双方に解釈の差異が生じないような定量的な目標を持つこと

SLM の成功には，計画段階からの利用部門の参画が必須である。SLM 運営組織を立ち上げる際には利用部門の主要関係者を配置し，SLA／SLM の重要性について早期に共通認識を形成することが望ましい。

4.2 クライアントが仮想化デスクトップサーバー(DaaS)を利用する場合

前章のサーバーの仮想化で述べられているメリット（サーバー統合によるコスト削減効果，プロビジョニングによる安定稼動効果，コンピューティング環境整備の短工期化）により，サーバーの仮想化は，様々なコンピュータ/ネットワーク技術で広く用いられるようになっている。

その一つの形態として，サーバー上に集約されたデスクトップ環境をクライアントに配信することで，仮想的にクライアント PC 環境を提供する技術があり，VDI(Virtual Desktop

Infrastructure)や仮想化デスクトップサーバーと呼ばれている。

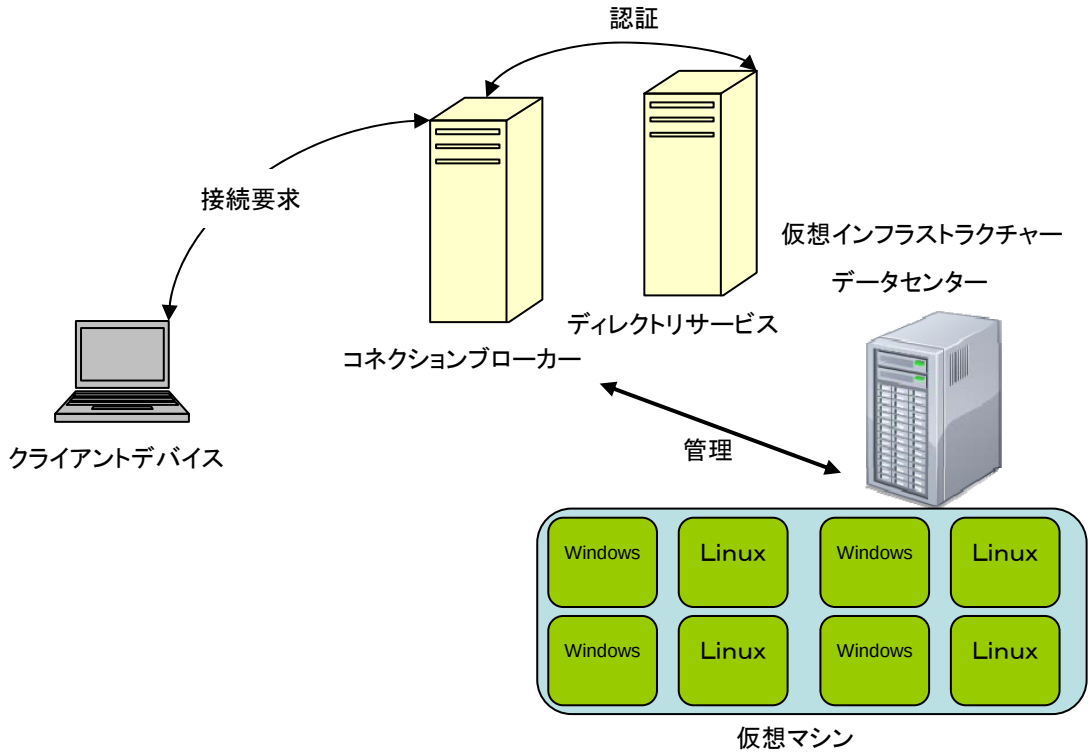

図 4－5　DaaS 型システム構成のイメージ

大きく分けて以下のキーコンポーネントから構成される。

- クライアントデバイス
  仮想マシンにリモート接続する。専用端末としてシンクライアントを使う場合や既存のパソコンが使われる。
- コネクションブローカー
  クライアントデバイスから接続要求を出したときに，仮想マシンを割当る。
- 仮想インフラストラクチャー
  仮想マシンを動かすインフラストラクチャーである。仮想化製品で仮想化されたサーバー，ストレージによって構築される。
- 仮想デスクトップ
  仮想インフラストラクチャー上で動作する仮想マシンを指す。Windows，Linux などのクライアント OS が使われる。

DaaS(Desktop as a Service)とは，この VDI をサービスとして利用者に提供するもので，コンピュータの稼働率，所有/運用に対するコスト低減，セキュリティの向上といった観点

で，DaaS の導入を進める企業が増加している。

ただ，デスクトップ環境がサーバー上に集約されたからといって，これまでの SAM に関する課題を考えなくてよくなったのではなく，サービスを提供してくれるソフトウェア及び関連する資産について，所有と責任の範囲を整理し，何よりも導入する企業，団体組織にとって何が本当に重要なのかを改めて整理する必要がある。

これまで,クライアント PC 上にあったデスクトップ環境をサーバー上に集約することで,運用面での課題や，インターネットからアクセスしたユーザーの認証，サーバー上に集約したクライアントデスクトップ環境で利用されているソフトウェアの利用許諾管理といった，新たな課題も出ている。

以下，どのような点に留意して DaaS を導入するべきなのかについて，以下の観点からまとめたので参考にしてほしい。

・組織
・契約
・セキュリティ
・運用管理

4.2.1. 組織

本ケースの場合，IT コストの低減という目的で，全社統一的に利用する場合や，情報漏えい等のセキュリティリスクへの備えとして，営業部門等，社外への PC 持ち出しが多い部門や，特定の利用者等に利用を制限する場合等，いくつかの利用形態があると考えられる。

SAM の管理者は，こうした様々な利用形態を検討したうえで，運用部門(仮想化デスクトップサーバーの導入)もしくは，社外のサービス提供業者(DaaS 導入)と，会社全体もしくは部門単位で導入する際の，方針，プロセスおよび手順を定めておく必要がある。

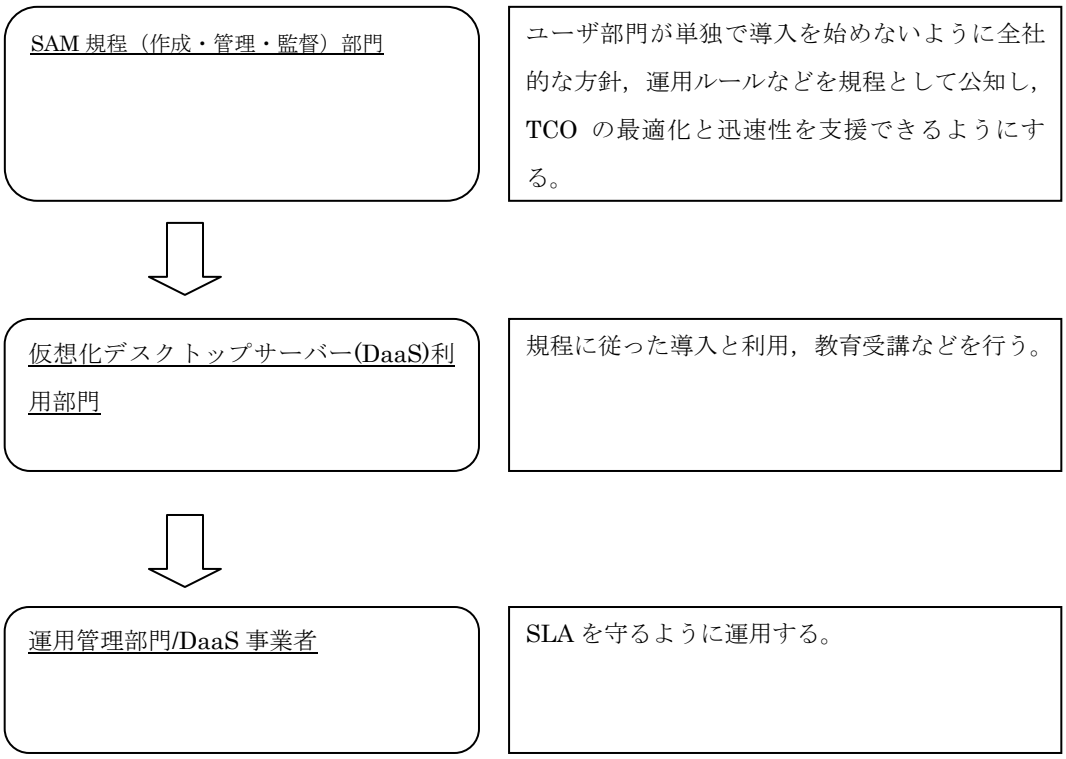

図 4－6 DaaS 利用する場合の組織

（1）SAM 規程部門

プライベートな DaaS を構築するのか，DaaS 業者のものを利用するのかを含め，全社的な方針や規程を定め，組織内に公知する。規程は，導入から利用，教育，ヘルプデスク，破棄などのライフサイクル全般について明確にし，全社的な戦略に合致し，TCO や迅速性に貢献できるものであることが望ましい。

（2）仮想化デスクトップサーバー(DaaS)利用部門

自部門のサービスに対する要求を明確し，SAM 規程部門/運用管理部門と交渉し，サービスレベルを決定する。決定した規程に従って利用する。

（3）運用管理部門/DaaS 事業者

SAM 規程部門/利用部門と合意したサービスレベルを維持するように運用管理する。

4.2.2. 契約

仮想化デスクトップサーバーや DaaS を導入する場合，そのサービス自身のライセンスとそのデスクトップサーバー上で利用するアプリケーションのライセンスについて以下の点を考慮することが望ましい。

- ✓ 仮想化デスクトップサーバー環境のライセンス契約
- ✓ アプリケーションソフトウェアのライセンス契約
- ✓ 管理台帳の整備，棚卸

（1）仮想化デスクトップサーバー環境のライセンス契約

仮想化デスクトップサーバーを社内導入する場合は，新規に必要になるのは，サーバー上にデスクトップ環境を集約し，クライアントに配信するためのソフトウェアのライセンス契約がまず，必要である。

代表的なものとしては，VMware View4，Citrix XenDesktop，Microsoft VDI といった製品があげられる。

また，これらの製品を導入するだけでなく，VDI 上でクライアント OS を動作させるための仮想 PC の OS ライセンス契約も別途必要であり，現在利用されているクライアント OS のサポート契約，企業向け包括契約の内容を確認しておく必要がある。

DaaS を導入する場合についても，ほとんどのサービス事業者は，OS を搭載しない仮想マシンとしての提供にとどまるため，利用する企業側で，仮想 OS のライセンス契約をクライアント OS 提供企業と個別に契約する必要がある。なお，一部サービス事業者の中には，サービス料金に，仮想 OS のライセンス契約料金を上乗せして分割支払可能にしているところもある。

（2）アプリケーションソフトウェアのライセンス契約

仮想化デスクトップサーバー上にインストールし，各企業で利用するアプリケーションソフトウェアのライセンスも，利用者側で購入し，契約をする必要がある。また，ソフトウェアによっては，こうした仮想化 OS 上での動作を保証していない製品もあるので注意することが望まれる。

また，現行物理 OS 上で利用しているアプリケーションソフトウェアのライセンスを仮想デスクトップサーバー・DaaS 導入に伴い，仮想 OS 上での利用に移行させる場合についても，利用許諾説明書等に記載されている契約条件等を確認し，正しいアップデートパスで移行が出来るよう対応する必要がある。

（3）管理台帳の整備，棚卸

仮想化 OS を利用する場合，デバイスとインストールされた OS，アプリケーションソフ

トウェアの数が必ずしも一致しないため，少なくとも，ハードウェア，導入ソフトウェア，保有ライセンス，ライセンス関連部材の管理台帳の整備が必要である。

また，DaaS サービスを利用する場合，自社内にデータを持たないため，サービス提供者側にあるデータを定期的にチェックする仕組みが必要である。現在展開されている仮想化 OS の数と種類,仮想 OS 上に導入されているソフトウェアの数と種類,ハードウェアの構成といった情報について,インベントリツール等で,即時確認できる手段を持つサービス事業者を選定し,棚卸情報のレポート契約を結ぶことで，現在利用されているソフトウェアの管理ができる仕組みについて合意を行っておくなどの対応をする必要がある。

例 マイクロソフトの仮想化 OS を利用する場合

<table>
<tr><td colspan="3">ソフトウェアライセンス</td></tr>
<tr><td>サーバーOS ライセンス</td><td>サーバー数に比例</td><td>導入台数による</td></tr>
<tr><td>仮想化ソリューションのライセンス(導入ソフトウェア等)</td><td rowspan="2">デバイス数に比例</td><td>ソリューションにより変動</td></tr>
<tr><td>マイクロソフト仮想化ライセンス</td><td>SA 契約契約デバイスは不要
(EA 契約を結んでいる場合等)</td></tr>
</table>

4.2.3. セキュリティ

DaaS 事業者の選定およびサービス内容について，安全性の観点から以下の点を考慮しなければならない。

- ✓ アカウント管理，履歴管理
- ✓ 機密性
- ✓ データ保護

（1）アカウント管理，履歴管理

仮想化デスクトップサーバーを社内に展開する場合，並びに DaaS により，サービス事業者からデスクトップ環境を利用する場合の双方とも，クライアントからの認証アクセスについて，パスワードの漏えい，ユーザーの成りすまし等のセキュリティリスクについて，十分，配慮する必要がある。

ユーザーがログインした際の履歴管理を行っておくことで，不正なユーザーによるアクセス時の操作履歴を無効にし，元の正常状態に戻す等の対応を行うことができる。

（2）機密性

DaaS では，企業ごとに，物理サーバーを用意する専有型と，複数の企業とで物理サーバーを共有して利用する共有型のサービスが存在する。

専有型の場合は，データの機密性としては問題がないが，共有型の場合，他社と同一サーバー上にデータが共存することになるため，DaaS 提供者のセキュリティ管理能力の考慮が必要となる。

（3）データ保護

仮想 PC の画面を端末に転送し，端末側の操作内容を仮想 PC に送信するプロトコルとしては，通常，Microsoft の RDP(Remote Desktop Protocol)が用いられる。Win2000 以降の RDP は基本的に RSA RC4(SSLv3 などで使用されるストリーム暗号)で暗号化されており，現状の暗号化レベルのまま利用するのであれば，通信のデータ保護としては問題ないものと考えられる。

4.2.4. 運用管理

仮想デスクトップサーバーの導入にあたり,運用管理部門,DaaS 提供事業者は,運用管理について以下の点を考慮しなければならない。

- ✓ サービスレベル管理
- ✓ 導入,展開
- ✓ 運用,保守
  - ✧ 仮想化デスクトップサーバー環境のチューニング
  - ✧ サーバー障害時のシステム復旧
  - ✧ OS，アプリケーションのメンテナンス
  - ✧ ポリシー管理
  - ✧ ヘルプデスク
- ✓ 廃棄,撤去

（1）サービスレベル管理

DaaS サービスを提供する事業者とは，提供される仮想デスクトップサーバーのサービス内容について，サービスレベルについて合意しておく必要がある。事業者側から提供される DaaS の年間稼動率や，障害が発生したときのバックアップ系への切り替え時間，アプリケーションの障害が発生した場合のマスターデータからの切り戻し，ユーザーデータの復旧などについて，どういったレベルで運用管理をしておくか，あらかじめ取り決める必要がある。

（２）導入,展開

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門は,仮想デスクトップサーバーの導入,展開にあたり,VDI 環境の構築およびシンクライアント端末の展開を実施する必要がある。

（３）運用,保守

・VDI 環境のチューニング

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門は,サービス提供後の,利用部門の状況を確認し,HDD 容量,メモリサイズ等が適性になるようチューニングを実施する必要がある。

・サーバー障害時のシステム復旧

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門の物理サーバーに障害があった場合でも，利用部門側の業務に影響がないよう，対応できる必要がある。物理サーバーに障害があった場合，予備系のサーバー群に切り替え，次接続のクライアント端末からのアクセスはこちらに変更し，現在接続中のクライアントが更新していたデータについては，バックアップをとっていた環境から復旧するなどの対応を行う必要がある。

・OS，アプリケーションのメンテナンス

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門は，提供している OS，アプリケーションに不具合があった場合，直ちにマスターイメージから正しい環境に戻す操作を行う必要がある。また，OS，アプリケーションにパッチが出た場合，付帯サービスとしてアップデート保守サービスを請け負っている場合には，これらパッチの適用作業も行う必要がある。

・ポリシー管理

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門は，提供している OS，アプリケーションについての,追加,更新,削除等の変更管理について,利用部門と合意しておく必要がある。一般の事務部門であれば,利用者のアクセス権を制限し,利用部門側で勝手に変更できないようにするなど,利用者のアクセスと変更についてのポリシーを定め,適切な利用環境が保全されるよう対応をする必要がある。

・ヘルプデスク

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門は，提供している OS，アプリケーションについて,利用部門からの問い合わせがあった場合の窓口業務を実施し,合意したサービスレベルに基づいた対応を行う必要がある。

（4）廃棄,撤去

DaaS および仮想化デスクトップサーバーを提供する事業者および運用管理部門は，提供している OS,アプリケーションについて,データの廃棄,撤去作業を安全に実施する必要がある。データ廃棄に伴い,利用者が契約済みの OS およびソフトウェアのライセンスについて,変更があった場合は,ライセンスの戻し処理等も合わせて行う必要がある。

また,DaaS の利用中止,サービス変更に伴い,シンクライアント端末の撤去が必要な場合は,これに対応する必要がある。

5. 仮想化における SAM の留意点

5.1 サーバー仮想化環境における SAM の留意点

5.1.1. サーバー仮想環境における構成管理の必要性

今までの「システム」と呼ばれる単位が「サーバーハードウェア上に存在する一つの OS に紐づくアプリケーション」というハードウェア，OS，アプリケーションが 1 対 1 対 1 のサイロ型のシステムであったのに対し，クラウド環境においては，サーバーは統合され，仮想化される。

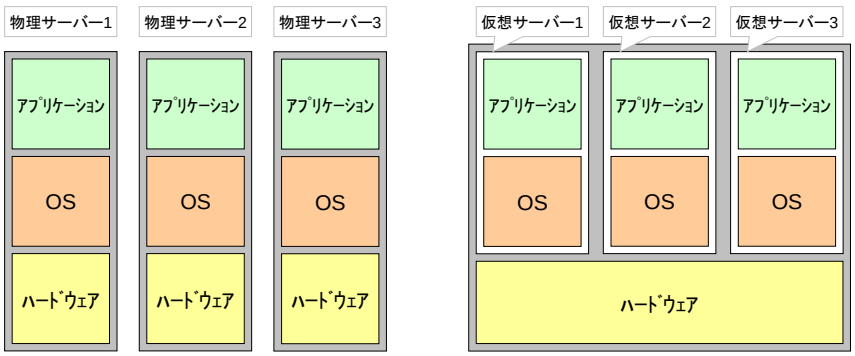

図 5－1 サーバー仮想化の基本概念

統合され仮想化されたサーバー環境は，例えば 4 つのコアを持つ CPU が複数個，一つのサーバー筐体の中に存在し，仮想化層のハイパーバイザー上に複数個の OS が存在し，OS 上には例えば JavaEE などのミドルウェア，そしてそれぞれのミドルウェア上に個別のアプリケーションが存在する環境が考えられる。

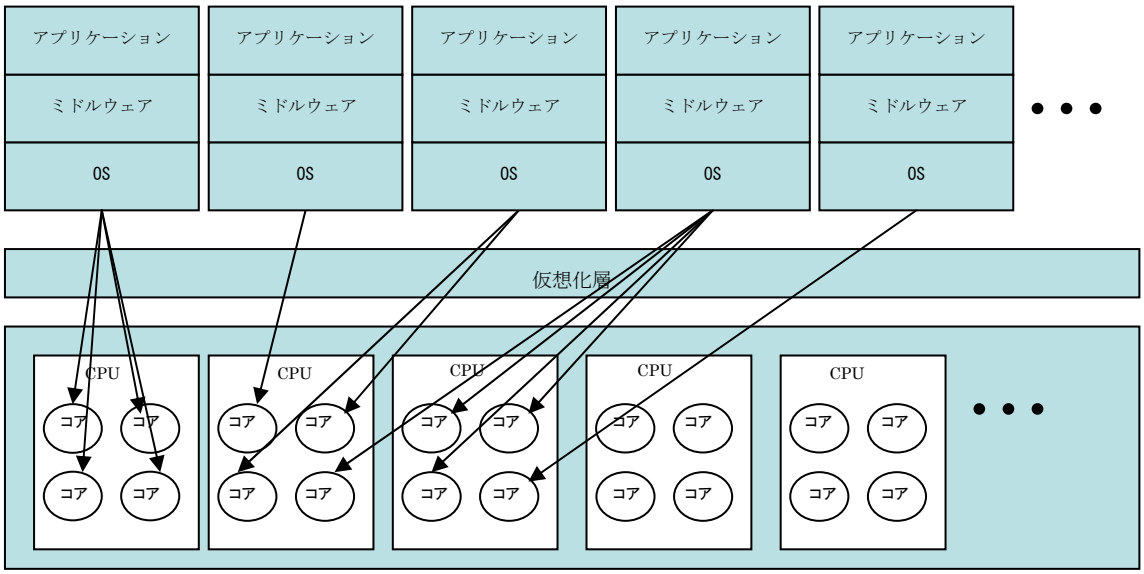

図 5－2　サーバー仮想化環境における OS と CPU コアの関係

このような環境でのソフトウェアは,

(1) 仮想化層を構成するソフトウェア（例：ハイパーバイザー）
(2) OS
(3) ミドルウェア
(4) アプリケーション

これら全てのソフトウェアにおいての使用許諾契約（EULA：End User License Agreement ）やライセンス契約の内容に基づいて利用されているかどうかを常に管理しなければならない。

具体的には，ハイパーバイザーや OS，ミドルウェア，アプリケーションのそれぞれの使用許諾条件において，その条件に物理的な CPU 数やコア数，サーバー数での制限などへの考慮が考えられる。

また，クラウド環境では柔軟なプラットフォームが求められるため，OS が使用する CPU コア数をポリシーベースの運用により動的に変化させることが考えられる。その場合，OS に紐づくミドルウェアのライセンス契約が CPU コア数による制限があった場合は，契約しているコア数を考慮にポリシーを設計しなければならないし，使用するコア数を常に監視し，管理しなければコンプライアンス違反を犯してしまうことも考えられる。

クラウド環境は，SOA 化（Service Oriented Architecture：サービス指向アーキテクチャ）により俊敏性やスケーラビリティなど，特にミッションクリティカルな基幹システム

の運用では優先度に応じて優先的に CPU リソースが割り当てられる。その場合，IT アーキテクトは基幹システムの可用性を重視するあまり，そのシステムを構成しているインフラストラクチャーのソフトウェアライセンス契約の内容までの順守に運用設計の考慮を怠ることもある。あるいは，設計者は「後は運用チームがなんとかしてくれるだろう」と運用者にゆだねてしまうことも考えられる。

柔軟な環境を構築すれば，当然，システムの優先度に応じた CPU リソースの再配置がどこかのタイミングで発生する。全くスケールアウトや動的なプロビジョニングを行わないシステムであればクラウドとは言えないのだから，当初は予定になくとも，いずれは CPU リソースの再配置が行われるという前提で，ライセンス契約の内容を加味したポリシーを設計し管理しなければならない。

具体的には，例えば「アプリケーション A」 と「アプリケーション B」という異なるアプリケーションが稼動しているのは，「ミドルウェア C」という同じミドルウェア上だった場合，当初「ミドルウェア C」が使用する CPU コア数の契約が 6 個と仮定する。「アプリケーション A」はミッションクリティカルな基幹システムで優先度が高いので，必要に応じて優先度の低い「アプリケーション B」に割り当てられている CPU コアを一つだけ「アプリケーション A」に動的に再配置できるポリシーを設計したとする。この場合は，動的なリソース再配置の結果，「ミドルウェア C」が使用する CPU コア数の合計が 6 個と再配置前と変わらないので，「ミドルウェア C」のライセンス契約である「CPU コア数 6 個までの使用許諾ライセンス」の条件に違反がないので問題はない。(図 5-3)

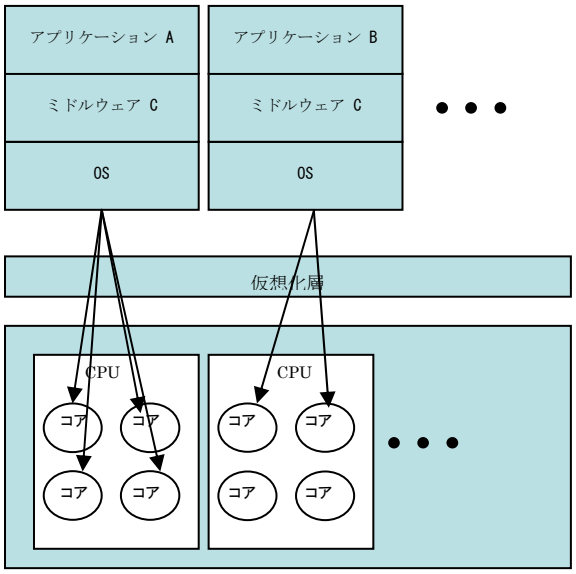

図 5－3 仮想環境におけるアプリケーションと CPU コアの関係

ところが，「アプリケーション A」の CPU コアを「アプリケーション B」ではない「アプリケーション X」に割り当てられた CPU コアを 2 個，再配置したとすると。(図 5-4)

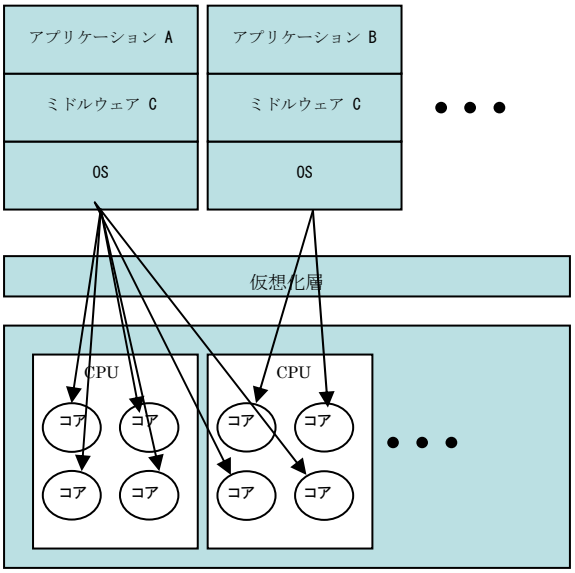

図 5−4 アプリケーションに対する CPU コアの再配置

この場合，「ミドルウェア C」が使用する CPU コア数の合計が 8 個となり，ライセンス契約において「CPU コア数 6 個までの使用許諾ライセンス」となっているので，2 個の CPU コアライセンスをオーバーして使用していることとなる。これは，ライセンス契約違反となりコンプライアンス違反が発生することになる。

ビジネス環境はめまぐるしく変化する。気が付けば「アプリケーション B」の優先度が「アプリケーション X」の優先度に勝るようになり，「アプリケーション A」は，さらに優先度が増したので追加可能な CPU コア数を増加させた，などという状況は発生するだろう。運用ポリシーを再検討するなど，何らかの変更が発生する場合は，物理的な影響分析だけではなく，その変化にともなうソフトウェアライセンス契約も考慮しなければならない。

このようにサーバー仮想化環境では今までのサイロ型のシステムでは無かった新たな考慮点が出現するのである。特に複雑さを隠蔽し自動化のシステムを提供するハイパーバイザーなどを使用する場合は，自動化され動的に変化する運用環境のソフトウェア資産管理を今までのサイロ型システムの管理同様に人手だけに頼って管理することは不可能なのは明らかである。

もちろんそのような事は織り込み済みで，それらを考慮した運用管理の手法が IT サービ

ス管理では既に提唱されている。

例えば，IT サービス管理（ITSM）のベストプラクティスである ITIL（IT Infrastructure Library）や，その国際標準である ISO/IEC20000，国内標準である JIS Q20000 では，構成管理データベースにより管理対象となる構成品目（Configuration Item：CI）の関係を管理するとしている。

> 2.5 構成管理データベース、CMDB（configuration management database）
> 各構成品目に関連するすべての詳細、及びそれら構成品目間の重要な関係の詳細を含むデータベース。
> （JIS Q20000-1:2007 2.5 構成管理データベース、CMDB（configuration management database）より引用）

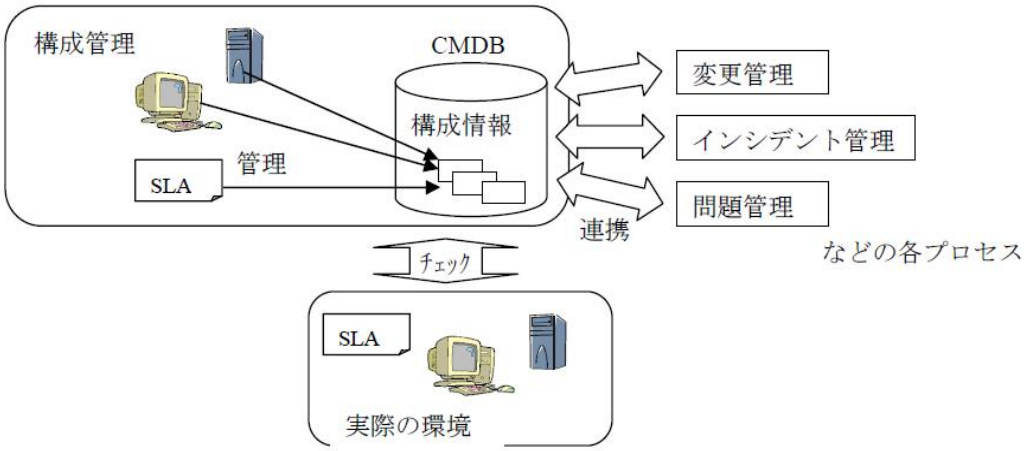

図 9-1　構成管理の概要

図 5－5 CMDB：構成管理の概要[2]

[2] JIPDEC　ITSMS ユーザーズガイドーJIS Q20000(ISO/IEC 20000)対応ー 平成 19 年 4 月 20 日出版 より抜粋。詳しくは，www.isms.jipdec.jp/itsms/doc/JIP-ITSMS111-10.pdf

統合運用管理のソフトウェア製品などではApplication Dependency（アプリケーション依存関係）の検出やマッピングのソフトウェアが存在するが，ソフトウェア資産管理においてはアプリケーションの依存関係を可視化するだけでは管理不足であり，アプリケーションやミドルウェアなど，システムを構成する全てのソフトウェアのライセンス契約を当該ソフトウェアと紐付け，そのシステムに割り当てられたCPUリソースを含む全ての構成品目を紐付けた構成管理データベースによりライセンス契約に基づいてコンプライアンス違反が発生していないかどうかを実際の環境との突合などにより，常に監視し，管理することが肝要となる。

JIS X 0164-1:2010 (ISO/IEC 19770-1:2006) において 「JIS Q 20000 規格群が定義している情報技術（IT）サービスマネジメントとよく両立するように構成されており，それを適切に支援することを意図している。」（日本規格協会 発行 ソフトウェア資産管理－第 1 部：プロセス JIS X 0164-1:2010 (ISO/IEC 19770-1:2006) より抜粋）や，「ITIL との緊密な連携をとり」などと表現されているのは，ソフトウェア資産管理を実施する上で IT サービス管理において定義される構成管理データベースや変更管理などの考慮なくして、ソフトウェア資産管理を効果・効率的に実現することは困難だからである。

5.1.2. 仮想環境におけるライセンス形態の違い

さて，前述のように仮想環境においては、仮想化されたシステムを構成するソフトウェアや、ハードウェアのリソースにより、ソフトウェアの契約に基づいたライセンスの管理が必要であることは明白である。

ただし、仮想環境やクラウド環境では全てのサーバソフトウェアのライセンス形態が複雑になるわけではない。グループウェアや ERP パッケージのようなソフトウェアはユーザー数や同時アクセス・ユーザー数によって料金が決まるライセンス形態が多く， ミドルウェアや OS は、物理サーバーの台数などの「規模」や、プロセッサの「性能」といった要素によりライセンス料が決まるものが多い。

具体的な例として以下のライセンス形態が考えられる：

（1） 仮想コア／プロセッサ数型

ミドルウェアが動作する VM（Virtual Machine：仮想マシン）に割り当てた仮想コア／プロセッサ数に応じてライセンス料が決まる。ミドルウェアのインスタンス数に関わらず，ミドルウェアが動作している VM に割り当てた仮想コア／プロセッサ数で計算する。

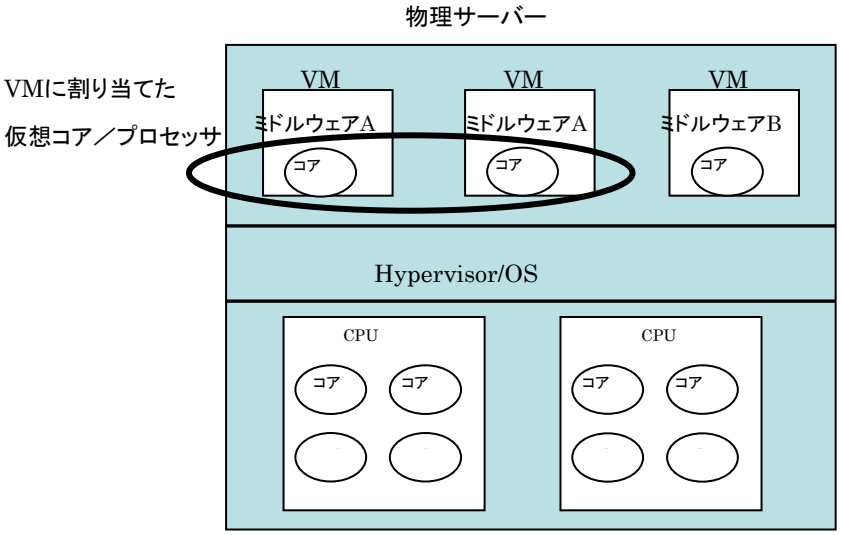

図 5－6 仮想コア／プロセッサ数型

（2） 物理コア／プロセッサ数型

ミドルウェアが配置されている物理サーバーが備えるプロセッサ数または総コア数に応じてライセンス料が決まる。ミドルウェアが配置されている物理サーバー上で有効な最大リソースのボリュームによりライセンス料が計算される。

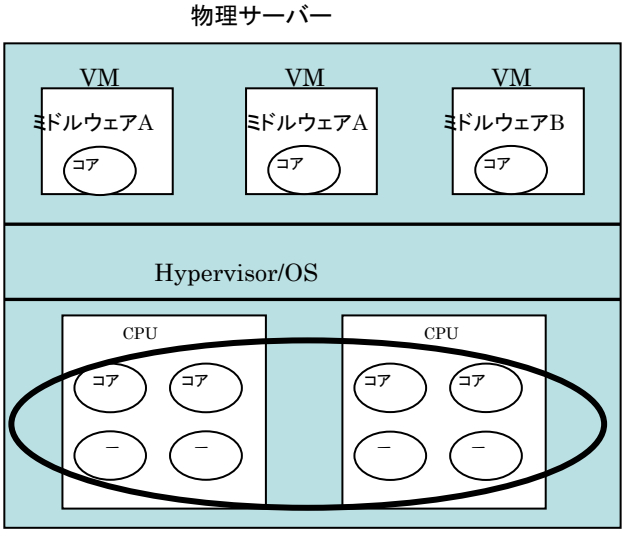

図 5－7 物理コア／プロセッサ数型

（3） インスタンス数／インストール数型

ミドルウェアの同時起動インスタンス数／VM のインストール数に応じてライセンス料が決まる。複数の物理サーバーに分散配置しても，物理コア数など物理的な割り当てリソースが変化してもライセンス料は変わらない。

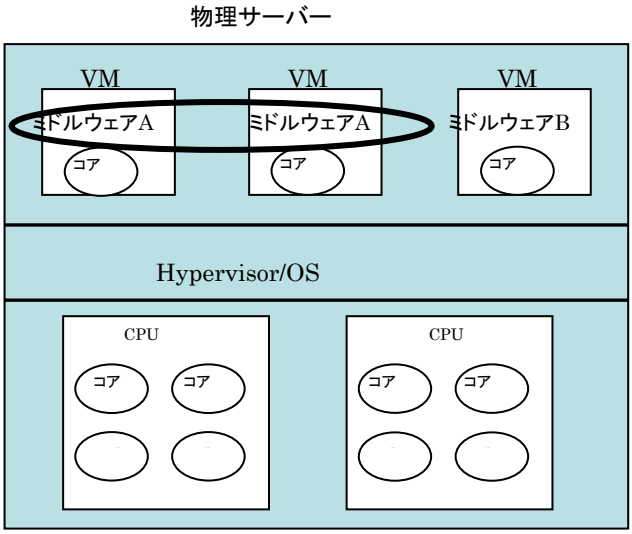

図 5−8 インスタンス数／インストール数型

仮想環境ではアプリケーション＋ミドルウェアを動作させる VM を特定の物理サーバーに集中配置することもできるし，複数の物理サーバーに分散配置させることもできる。さらに，一度配置した後，負荷分散のために再配置することもできる。こうした，集中配置，分散配置，再配置の変更があるときに注意しなければならない。

構成管理上ではリソースとの比率が変化してなくてもライセンス契約の形態によっては，配置される物理的なロケーションによりライセンス料が変化する可能性があるので，変更管理のポリシーの策定にはこれらのライセンス形態の条件を考慮して仮想コアなどの追加，インスタンスの追加を物理的に同一のサーバー上で行うのか，物理的に異なるサーバー上に展開するのかなどを考慮しなければならない。

プロセッサの性能が異なるサーバーがリソースプールに存在する場合，例えばプロセッサ性能の高い物理サーバーに配置されたインスタンスを移動する場合，VM への割り当てコア数を増やして性能の差分を補うなどするが，この場合もライセンス形態によりプロセッサ性能がライセンス料に対してどのように作用するのかを把握し管理していなければならない。

ミドルウェアのメーカーのライセンス形態によってはプロセッサ性能によってライセンス料が異なる場合もあるので，プロセッサ性能が低い場合はその分ライセンス料が割り引かれることもある。

それではIaaS などクラウド環境でアプリケーションを利用する場合はどうだろう。仮想コア／プロセッサ数型や物理コア／プロセッサ数型の場合，借りた VM が物理サーバーをまたがって再配置される可能性がある。

しかし，利用者は隠蔽されたサービス環境が一体何台の物理サーバーをまたいで提供されているかは分からない。これらの課題が後のクラウドサービス料金の不明瞭な問題とならないように，クラウドサービスの契約時に，SLA などで可用性と性能に対してのサービス料金の課金体系と，利用者がIaaS 環境で利用するミドルウェアのライセンス形態に対してクラウドサービスを提供するサービス提供者がメーカーとの間でなんらかの取り決めをしてミドルウェアの使用許諾条件を考慮した運用が可能となっているかなど確認することが望ましい。

5.2 クライアント自身がクライアント仮想化 OS を利用する場合

本ケースの利用形態（クライアント自身が，Windows などのクライアントOS の上で動作する仮想化ソフトウェア，例えば，Microsoft Virtual PC やVMware Workstation などを利用）を行っているのは，主に，技術開発部門の開発エンジニアなどが想定されるが，そうしたケースにおいては，SAM 主管部門である情報システム部門が知らないうちに，利用部門が勝手にソフトウェアをインストールして，使用しているケースが散見される。こういったケースの場合には、

・ゲストOS 毎に，当該OS，および当該OS 上で稼動するアプリケーションソフトウェアのライセンスが必要になるが，ゲストOS，および当該OS 上で稼動するアプリケーションソフトウェアの複製は，ディスクイメージのファイルをコピーするなどして，比較的容易にできてしまうため，ライセンス違反が起こりやすいこと
・デバイスとインストールされたOS，および当該OS 上で稼動するアプリケーションソフトウェアの数が一致しない状態になるため，仮想化の前にハードウェア，導入ソフトウェア，保有ライセンス，およびライセンス関連部材の管理台帳を整備していなければ，気づかずにライセンスコンプライアンスに反してしまう可能性が高くなること

などを認識し、それに対処する施策を策定しておくことが望ましい。

当該組織において，適切な管理を実施する場合に考慮すべき事項を以下の３点に絞って

まとめる。

・方針，プロセスおよび手順の策定方法例
・使用許諾条件を確認する際の考慮事項例
・管理台帳の整備，棚卸方法例

5.2.1. 方針，プロセスおよび手順

本ケースの利用形態は，全社共通の利用形態ではなく，技術開発部門などの一部の特定部門になるため，組織全体で定められている方針，プロセスおよび手順ではカバーし切れていない可能性がある。

従って，SAM 統括責任者は，使用許諾契約内容を確認のうえ，組織が適法，かつ有用な使用を推進することができるように，必要に応じて，方針，プロセスおよび手順を規程，改訂のうえ，取締役会又は同等の機関のレベルにおいて承認を得たうえで，組織内の全ての要員に伝達することが望ましい。図 5-9 に SAM 責任者，および利用者とその役割の一例を示すので参考にして頂きたい。

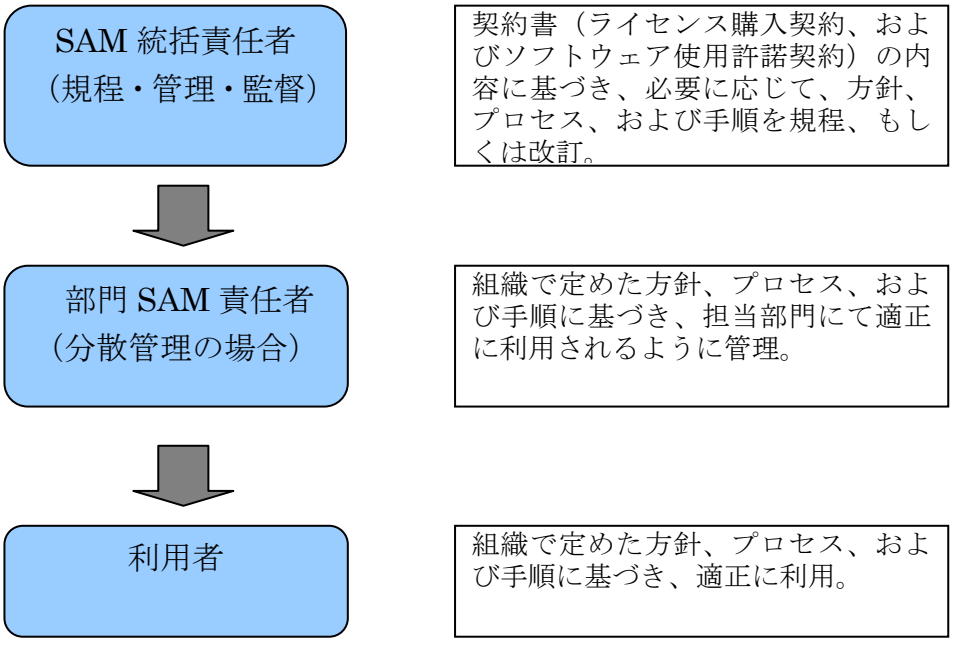

図 5-9　SAM 責任者，利用者とその役割

（1） SAM 統括責任者

契約書（ライセンス購入契約，およびソフトウェア使用許諾契約）の内容に基づき，

必要に応じて，方針，プロセス，および手順を規程，もしくは改訂のうえ，取締役会又は同等の機関のレベルにおいて承認を得たうえで，組織内の全ての要員に伝達する。また，組織全体にわたって，所定の方針，プロセス，および手順に従って，適正に利用されるように指導・管理する。

（2） 部門 SAM 責任者

契約書（ライセンス購入契約，およびソフトウェア使用許諾契約）の内容に基づき，組織で定めた方針，プロセス，および手順に従って，担当部門にて適正に利用されるように指導・管理する。

（3） 利用者

契約書（ライセンス購入契約，およびソフトウェア使用許諾契約）の内容に基づき，組織で定めた方針，プロセス，および手順に従って，適正に利用する。

5.2.2. 契約（コンプライアンス，ソフトウェア使用許諾契約）

本ケースの場合，前述の通り，ゲスト OS 毎に，当該 OS，および当該 OS 上で稼動するアプリケーションソフトウェアのライセンスが必要になるが，ゲスト OS，および当該 OS 上で稼動するアプリケーションソフトウェアの複製は，ディスクイメージのファイルをコピーするなどして，比較的容易にできてしまうため，ライセンス違反が発生する可能性が高く，その結果，損害賠償請求をされることも充分考えられる。

従って，SAM 統括責任者は，組織が適正に使用許諾を受け，かつ契約条件に従って使用することを確実にするために，該当する製品の使用許諾内容について，当該ソフトウェアベンダー，販売代理店，もしくは SAM 構築支援会社からの説明を聞いたうえで，ソフトウェア使用許諾条件を順守するように周知徹底を図ることが望ましい。

例えば，図 5-10 のように，Microsoft Windows の場合，有効なソフトウェア アシュアランス特典を取得済みのデバイスであれば，Microsoft Virtual PC や VMware Workstation などの仮想化ソフトウェアを利用して，物理 OS 環境で「Windows 7 Enterprise」（Windows 7 Professional，あるいは，その旧バージョンでも可能）を実行できるほか，同じ物理デバイス上に構築した４つの仮想 OS 環境まで，「Windows 7 Enterprise」（Windows 7 Professional，あるいは，その旧バージョンでも可能）を実行することができる。

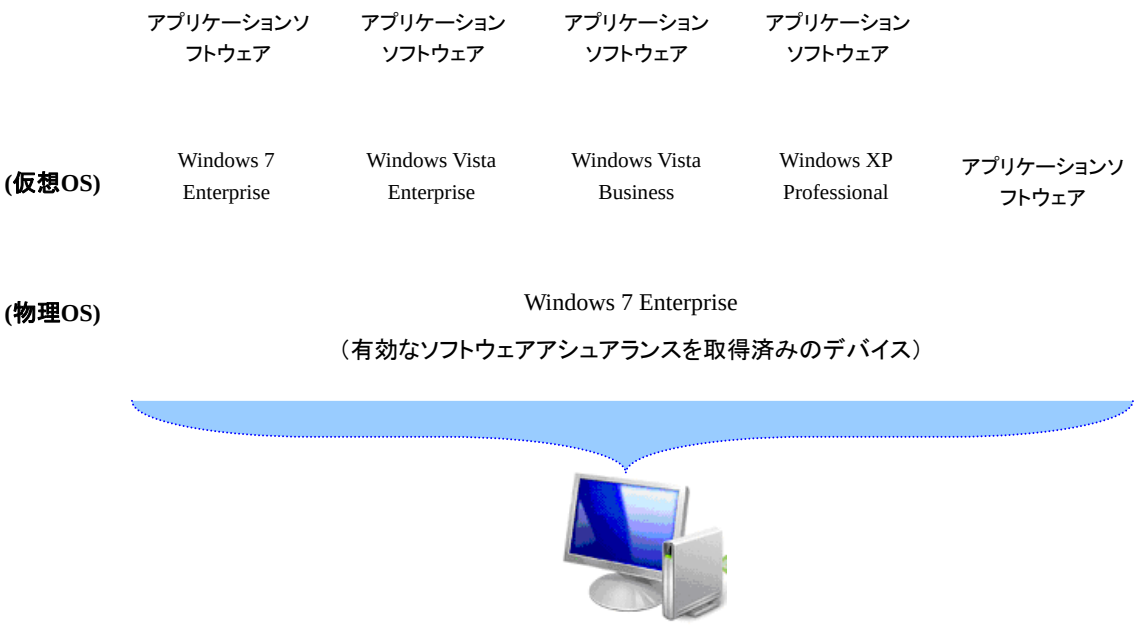

図 5-10　クライアント自身がクライアント仮想化 OS を使用する利用例

次に，物理 OS，および仮想 OS 上で稼動させるアプリケーションソフトウェアを使用する際に必要となる「ライセンスのカウント方法」，および代表的な使用許諾条件である「ダウングレード利用の可否」，「ライセンスの再割り当て」について説明したい。但し，下記の使用許諾条件の説明内容はあくまで参考情報であって，詳細については，各製品の使用許諾条件を確認する必要がある。

（1）必要となるライセンスのカウント方法

下記のいずれかに分類されることが一般的である。

①「デバイスベース」：　　1デバイスにつき，1ライセンスが必要。

②「インスタンスベース」：1インスタンスにつき，1ライセンスが必要。

また，同一製品であっても，購入方法（パッケージ製品，ボリュームライセンス等）によって，必要となるライセンスのカウント方法が異なる場合があるので注意が必要である。

例えば，Microsoft Office 2010 の場合，パッケージ製品でのライセンスのカウント方法は，「インスタンスベース」となるが，ボリュームライセンスでのライセンスのカウント方法は，「デバイスベース」となる。

一方，Adobe Acrobat X の場合，購入方法（パッケージ製品，ボリュームライセンス等）に関わらず，ライセンスのカウント方法は，「インスタンスベース」となる。

（2）アップグレード並びにダウングレード利用の可否

物理 OS，および仮想 OS 上で稼動させるアプリケーションソフトウェアに関するアップグレード並びにダウングレード利用の可否については，一般的には当該製品の購入方法によって異なることが多い。

① 購入方法が「パッケージ製品」の場合

一般的に，アップグレードは認められているものが多いがダウングレード利用は認められていないものが多い。例えば，「Microsoft Office Professional 2007」のパッケージ製品を購入した場合，使用可能なバージョンは，「2007」のみとなる。また当該ユーザーが「Microsoft Office Professional 2007」をアップグレード元ライセンスとして、「Microsoft Office Professional 2010 アップグレード」を購入した場合、使用可能なバージョンは，「2010」のみとなり、「Microsoft Office Professional 2010」を利用することが可能になる。

② 購入方法が「ボリュームライセンス」の場合

「パッケージ製品」の場合とは異なり，一般的に，ダウングレード利用が認められているため，購入したライセンスのバージョンよりも下位バージョンであれば，異なるバージョンでの利用が許諾されている。

例えば，「Microsoft Office Professional Plus 2010」をボリュームライセンスで購入した場合，購入ライセンスのバージョンよりも下位バージョンであれば，ダウングレード利用が認められているので，「Microsoft Office Professional Plus 2010」，および「Microsoft Office Professional Plus 2007 以前の旧バージョン」にダウングレードして利用することが可能である。

但し，ソフトウェアベンダーによっては，所定の申請手続きを行い，当該ソフトウェアベンダーが承認した場合のみ認めるといったケースもあるので注意が必要である。

（3）ライセンスの再割り当て

本ケースの利用形態を採用していたデバイスを廃棄する場合，物理 OS，および仮想 OS 上で利用していたアプリケーションソフトウェアについては，当該デバイスからアンインストール後，他のデバイスに再割り当てを実施することができ，再活用を図ることが可能となることが一般的である。

また，廃棄対象となる Windows プレインストールモデルに付属する Windows 製品は，当該デバイスに紐付いているので，当該デバイスとともにアンインストール後，ライセンスも廃棄しなくてはならないが，Windows の有効なソフトウェア アシュアランス特典については，再割り当て先の Windows 製品が最新バージョンであれば，ソフトウ

ェア アシュアランスの再割当権を行使して，再割り当てを実施することが可能である。

5.2.3. 管理台帳の整備，棚卸

前述した通り、クライアントの仮想化を行う場合には、仮想化の導入前に管理台帳の整備を行い、保有ライセンスと利用ソフトウェアの正確な把握をしておくことが強く望まれる。

また，定期的な棚卸作業はより重要となり、ライセンス過不足状況を正しく速やかに把握する仕組みを持つことが望まれる。

なお，管理台帳の整備については，JIPDEC SAM ユーザーズガイドの「6.4.対象資産の調査手順」を参考のうえ実施することを推奨する。詳細については，SAM ユーザーズガイドの「6.4.対象資産の調査手順」を参考にして頂きたい。

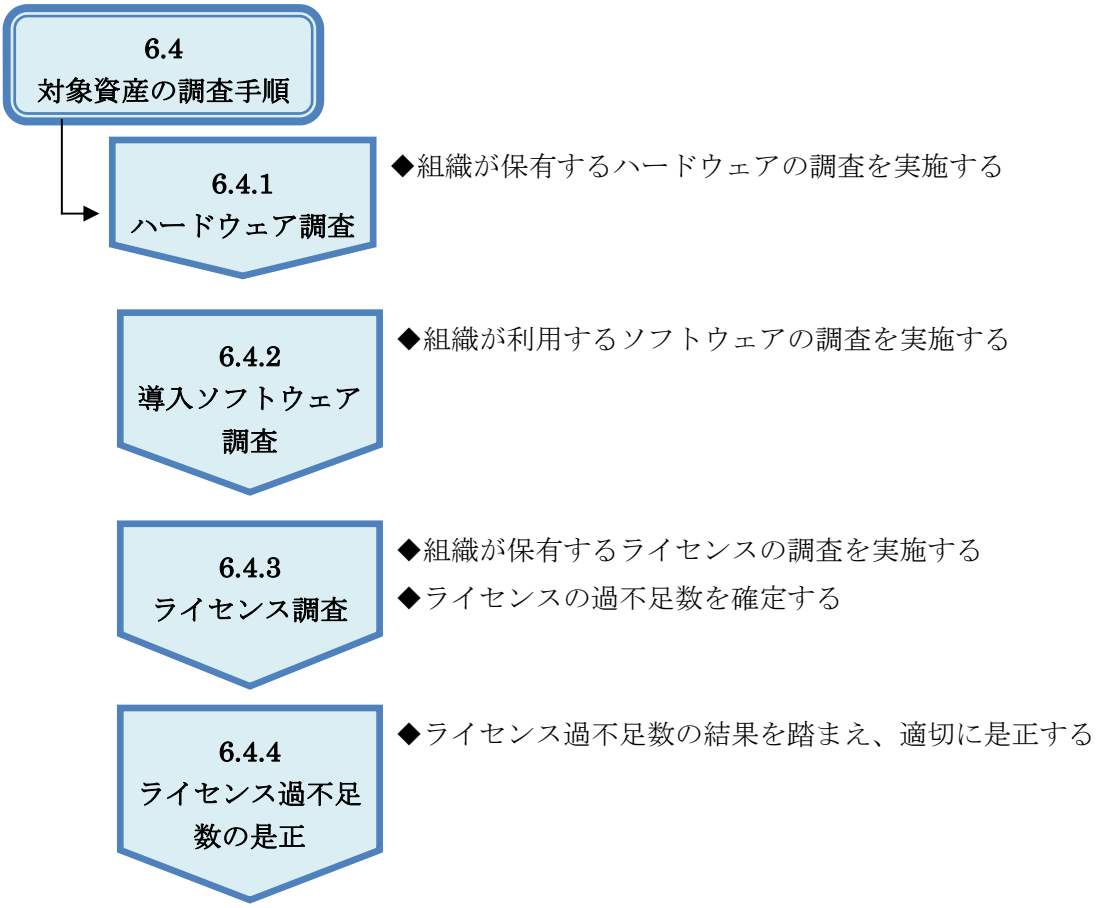

図 5-11　管理対象資産の把握の手順[3]

本ケースの利用形態を選択する場合，クライアント PC に関するライセンス管理，運用管理の負荷，およびライセンスコストは，対象 PC 台数の増加に比例して増大する。

従って, 本ケースの利用形態を選択する場合, TCO とライセンスコストの関係を考慮し，導入方法を検討すべきである。

また，上記管理工数を低減するためにも，本ケースの利用形態を検討する際には，クライアント PC の OS やアプリケーションソフトウェアをサーバー側で一元管理するクライアント仮想化を前提に検討することが望ましい。

[3] JIPDEC　SAM ユーザーズガイド「6.4.対象資産の調査手順」から引用

6. 組織が OSS を活用する場合における SAM 上の留意点

国内の調査によると，OSS を利用している企業は全体の約 7 割弱であり，OSS は多くの企業で引き続き IT ビジネスに利用されている（図 6-1）。従前は OS やサーバソフトウェア，ミドルウェアにおいて OSS が多く活用されてきたが，近年では ID 認証やシングルサインオン，オフィスソフトウェアといったアプリケーション領域での OSS の活用が本格化してきている。また，クラウド・コンピューティングの普及が OSS の利用促進につながると考えている企業は 4 割以上に達しており，仮想化技術や SaaS の提供において広く OSS が 活用されている。

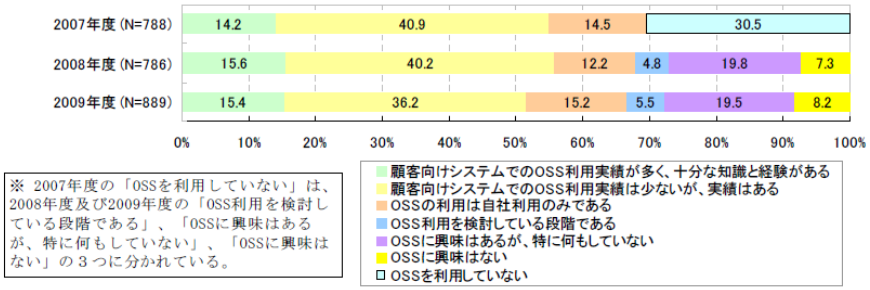

図 6-1　OSS の利用状況（2007～2009 年度）

（出典）独立行政法人 情報処理推進機構

「第 3 回オープンソースソフトウエア活用ビジネス実態調査(2009 年度調査)」

このように，わが国の組織における OSS の活用自体はもはや珍しいことではない。しかし，OSS に対する間違った観念から，OSS がソフトウェア資産管理（SAM）の管理対象ではないと誤解しているケースが多く見受けられる。本章では，今後あらゆる場面で導入が増えていくであろう OSS を活用する場合における留意点を SAM の側面から説明する。

6.1 OSS の管理の必要性

6.1.1. OSS のライセンス

多くの OSS はライセンスの下で提供されている。OSS のライセンスはプロプライエタリ・ソフトウェアのライセンスと何が違うのであろうか。その議論の前にソフトウェアと著作権について，改めて整理しておこう。

我々が通常使用しているソフトウェアの実体はコンピュータ上で動作する「コンピュー

タプログラム[4]」であり，その表現が著作権法により保護されている。この著作物であるコンピュータプログラムを実行すること（＝使用）自体は，著作物の目的でもあるため著作権で何ら制限されるものではない。ただし，「複製」，「配布（頒布）」，「譲渡」，「貸与」（＝利用）といった著作権法上の権利は著作者だけに与えられており，著作者はこれらの利用に関する権利を他人に許諾することができる。

プロプライエタリ・ソフトウェアのライセンスの多くは，上記の「利用」に関する許諾に加えて，「使用」すること自体に対する制限や条件等が盛り込まれており、一般に「使用許諾契約」と称されている。したがって、プロプライエタリ・ソフトウェアについては，「利用」するだけでなく，「使用」する際にも順守すべき事項があることに留意しておく必要がある。これらを順守するが為に，プロプライエタリ・ソフトウェアを使用する組織は適切にライセンス管理を行う必要があるわけである。

では，OSS はどうだろうか。OSS もプロプライエタリ・ソフトウェアと同じく著作権で保護された著作物であり，その権利自体は何ら変わるところはない。OSS はソースコード（＝コンピュータプログラム）を入手することが基本的に可能であり，合法的にインターネット等でソースコードを入手した後，コンピュータ上で「使用」すること自体は全く問題がない。しかし，「利用」する場合には，プロプライエタリ・ソフトウェア同様，ライセンスに従う必要があることを忘れてはならない。また，OSI が認定しているライセンスだけで 70 弱あり[5]，全てが同じ条件ではないことに留意してほしい。

ここでは，多くの組織において直面するであろう，OSS の「配布」および「複製」について，その留意点をまとめておく。

①OSS の配布

特にソフトウェア開発を行っている組織において，OSS を利用（ソースコードの改変や，ソースコードの一部あるいは全部を活用）して新たなソフトウェアを開発し提供する，あるいは OSS を機器に組み込んで販売するなどの機会があると考えられる。このような場合には，利用する OSS のライセンス条件に従って利用する必要がある。OSS を利用したソリューションを提供する上で知っておくべき概念として「コピーレフト（copyleft）」がある。

[4]電子計算機を機能させて一の結果を得ることができるようにこれに対する指令を組み合わせたものとして表現したもの

[5] http://opensource.org/licenses/alphabetical

【コピーレフト】
著作者が著作物に対する権利（著作権）を保有したまま，著作物の配布条件として，利用者に著作物を複写・改変・再配布する自由を与える一方で，複写・改変・再配布された派生物（二次的著作物）の配布者に対しても，全く同じ条件で派生物を配布することを義務付けるといった考えである。この「コピーレフト」の概念は，著作物が配布され続ける限り，制限なく適用され続けるといった特徴をもつ。
（出典）IPA,「OSS ライセンスの比較および利用動向ならびに係争に関する調査」

コピーレフトの概念に従えば，OSS のライセンスがソースコードを開示することを求めている場合，その OSS を利用して生成されたソフトウェアについてもソースコードを開示することが求められることになる。当該ソフトウェアのバイナリコードしか提供せず，ソースコードの開示請求に対して拒否意向を示した場合はライセンス違反となり，訴訟などに発展するおそれがある。実際，海外では OSS のライセンス条件を正しく認識せず利用したことから訴訟や紛争に至ったケースも存在しており，組織におけるリスクとして認識されつつある。なお，OSS のライセンスの全てがコピーレフトではないことも留意しておきたい（表 6-1）。

表 6-1　OSS ライセンスのコピーレフトによる分類

OSS ライセンスの カテゴリ・類型	①改変部分の ソースコードの開示	②他のソフトウェアの ソースコード開示	代表的な ライセンス
コピーレフト型ライセンス	要	要	GPL[6] AGPL[7]
準コピーレフト型ライセンス	要	不要	LGPL[8] MPL[9]
非コピーレフト型ライセンス	不要	不要	BSD License Apache License

（出典）IPA,「OSS ライセンスの比較および利用動向ならびに係争に関する調査」

以上から，OSS を利用してソフトウェアを開発する場合は，利用しようとしている OSS のライセンス内容を把握し，組織にとって許諾できない条件である場合には利用しないなどの対策が必要である。具体的には，ソフトウェア開発のルールに盛り込んだり，開発者に対して教育を施したりするなどが挙げられる。

[6] GNU General Public License
[7] GNU Affero General Public License
[8] GNU Lesser General Public License
[9] Mozilla Public License

②OSS の複製

では複製はどうであろうか。代表的な OSS のライセンスである「GNU General Public License(GPL)」では，複製する際の条件として以下のように示している[10]。

> *それぞれの複製物において適切な著作権表示と保証の否認声明(disclaimer of warranty)を目立つよう適切に掲載し，またこの契約書および一切の保証の不在に触れた告知すべてをそのまま残し，そしてこの契約書の複製物を『プログラム』のいかなる受領者にも『プログラム』と共に頒布する限り，あなたは『プログラム』のソースコードの複製物を，あなたが受け取った通りの形で複製または頒布することができる。媒体は問わない。*
> *（中略）『プログラム』をオブジェクトコードないし実行形式で複製または頒布することができる。ただし，その場合あなたは以下のうちどれか一つを実施しなければならない:*
> *a）著作物に，『プログラム』に対応した完全かつ機械で読み取り可能なソースコードを添付する。*
> *b）著作物に，いかなる第三者に対しても，『プログラム』に対応した完全かつ機械で読み取り可能なソースコードを，頒布に要する物理的コストを上回らない程度の手数料と引き換えに提供する旨述べた少なくとも 3 年 間は有効な書面になった申し出を添える。ただし，ソースコードは上記第 1 節および 2 節の条件に従いソフトウェアの交換で習慣的に使われる媒 体で頒布しなければならない。あるいは，*
> *c）対応するソースコード頒布の申し出に際して，あなたが得た情報を一緒に引き渡す(この選択肢は，営利を目的としない頒布であって，かつあな たが上記小節 b)で指定されているような申し出と共にオブジェクトコード あるいは実行形式のプログラムしか入手していない場合に限り許可される)。*

（出典）Open Source Group Japan

これらの条件において問題となるのは，①で述べた再配布の場合であり，組織内において OSS を複製し使用する分には問題が生じることはまずないと言える。ただし，先の GPL を始め OSI が認定しているライセンスだけで 70 弱あり，全てが同じ条件ではないことに留意してほしい。組織内で OSS を利用する場合は，少なくとも複製の条件を確認しておくべきであろう。

表 6-2　主な OSS ライセンスと複製に関する許諾条件

OSS ライセンス	代表的な OSS	複製に関する許諾条件
GPL	LinuxOS，KVM カーネルモジュ	『プログラム』に対応した完全かつ機械で

[10] 正式なライセンスは英語の原文であり、日本語訳は参考として掲載している

	ー ル，Oracle VM VirtualBox, Xen, GroundWork Monitor, ZABBIX, Hypertable など	読み取り可能なソースコードを添付すること。
BSD License	FreeBSD, Tera Term, Nimbus など	免責条項，著作権表示，ライセンスを含めること。
Apache License	Apache HTTP Server, Hadoop, CouchDB など	ソース形式，オブジェクト形式を問わず，成果物および派生成果物の複製について，無期限，世界規模，非独占的，使用料無料，取り消し不能なライセンスが許諾されている。
MPL	Firefox など	第三者が知的財産権を主張する場合を除き，複製に関して使用料無料の非独占的ライセンスが許諾されている。
LGPL	OpenOffice，KVM ユーザーモジュールなど	複製に関しては GPL と同様。

6.1.2. OSS のセキュリティとサポート

国内外には，情報セキュリティの脆弱性を公開しているサイトが存在する。その中には，exploit コード（脆弱性を攻撃するためのコード）を公開しているものがある。これは，ソフトウェア開発者に対して，脆弱性の修正を促すための考え方に基づくものである。特に，OSS はソースコードが公開されていることから，exploit コードが作成しやすい傾向にある。exploit コードが公開されてしまった場合は，ワームなどにこのコードを組み込まれる可能性もあり，攻撃されるのは時間の問題だ。

OSS は，ソースコードを公開することで多くの人々からレビューを受けることにより，その品質を高めていくことを志向している。これは，言い換えれば，多くの人々からレビューを受けていないソースコードの品質は低い可能性があるということでもある。大規模な OSS は通常「コミュニティ」と呼ばれる組織を結成しているが，OSS の品質はこの「コミュニティ」の活動状況に依存していることが多い。OSS の使用にあたっては，その動作が無保証であることを認識し，バグやセキュリティホールへの攻撃といったリスクに対しては，原則として自己責任で臨まなければならない。「コミュニティ」が解決してくれるかもしれないし，誰も解決してくれないかもしれないからだ。

したがって，OSS を使用している組織は，常に脆弱性公開サイト等をチェックし，使用している OSS に関する脆弱性情報を収集し続けることが求められる。そのためには，使用している OSS のバージョンを把握しておく必要がある。また，脆弱性を修復するためには，

OSS がどのハードウェア上で使用されているかを適切に把握しておくことが不可欠である。

表 6-3　主なセキュリティ脆弱性公開サイト

セキュリティ脆弱性公開サイト	運営団体	URL
CERT/CC	CERT Coordination Center	http://www.cert.org/certcc.html
National Vulnerability Database（NVD）	NIST	http://nvd.nist.gov/
SecurityFocus	Symantec	http://www.securityfocus.com/
OSVDB	Open Source Vulnerability Database	http://www.osvdb.org/
JPCERT/CC	JPCERT コーディネーションセンター	http://www.jpcert.or.jp/

6.1.3. 商用版の OSS

OSS がフリーウェア（無料ソフトウェア）であると誤解しているケースを見かけるが，OSS の多くが無償で公開されているだけであり，OSS は有償で提供することも可能である。オープンソースデータベースである MySQL や BerkeleyDB などはその代表例である。これらは，無償版と商用版の二つのライセンス体系（デュアルライセンス）で提供されており，商用版ではアップデートやナレッジ情報へのアクセス権などが提供されている。

当然のことながら，商用版のサービス等を無償版に適用することはできないし，サービス契約を把握しておかなければ，知らずのうちに期限切れになってしまうかもしれない。このようなソフトウェアに付随するサービス契約の管理も SAM の範疇であるため，プロプライエタリ・ソフトウェアに限らず，OSS も管理の対象に含めておく必要がある。

6.2 OSS を使用するにあたっての SAM における留意点

以上は，OSS を SAM の対象として管理していない場合，コンプライアンスやセキュリティ上の様々なリスクが顕在化するおそれがあること示唆している。OSS を使用している，あるいはこれから使用しようと考えている組織では，組織のリスク対策として OSS を SAM の対象に含めるべきであることは自明であろう。

ただし，一般的なプロプライエタリ・ソフトウェアの管理方法では，OSS を適切に管理

できないケースが存在する。これらのケースとその対処法について説明する。

①管理者権限がなくてもインストールできるソフトウェアがある

従業員によるインストール行為を制限するために, 配布している PC に対して管理者権限を与えないよう対策を講じている組織は多い。しかし，インストールあるいは使用する上で管理者権限を要しない OSS は多い。Excel のマクロなどもその一つである。

管理者権限を要しないソフトウェアの使用を統制するためには，PC から管理者権限をはく奪するだけでなく，定期的な使用ソフトウェアのモニタリングやインターネットからのダウンロードの制限等が必要であるが，それでも完全を期すことは難しい。やはり，従業員に対するソフトウェア使用についての教育および周知の徹底が不可欠である。

②インベントリ情報収集ツールで情報が取れない場合がある

OSS によっては, OS が備えているインストーラーを介さずにインストールされるものも存在する。UNIX において伝統的にソースコードが配布される形式として tarball[11]があるが，これにより OSS をインストールした場合はインストーラーを介さない。その結果，PC 上にインストールされたソフトウェアを網羅的に把握することが困難となり，使用ソフトウェアの棚卸しに支障を来すおそれがある。このような状況は，特にサーバー環境に多く見られる。

インベントリ情報収集ツールによっては OSS をファイル名で捕捉しているものもあるが，数多に存在する OSS を全て捕捉することは困難であることに留意したい。

OSS のインベントリ管理を徹底するためには，インストーラーを用いたインストール方法に限定する，これが困難な場合は OSS をインストールする際にインベントリ情報収集ツールのソフトウェア辞書を確実に登録するなど，インストール時点でのルールを厳密に策定しておくことが有効である。これに加えて従業員に対するソフトウェア使用についての教育および周知の徹底も不可欠である。

[11] UNIX の tar コマンドによりソースコードをアーカイブしたファイル

ソフトウェア資産管理評価検討委員会

「クラウド・コンピューティング時代の SAM の考え方」

作成メンバー

（敬称略）

氏名	所属
篠田　仁太郎 （委員長）	㈱クロスビート
岩下　健久	日本電気㈱
薩摩　貴人	有限責任　あずさ監査法人
高橋　快昇	富士通㈱
武内　烈	国際 IT 資産管理者協会（IAITAM）
中村　大造	ウチダスペクトラム㈱
中村　究	㈱シルクロード　テクノロジー