

ユーティリティコンピューティングと運用 変革

Utility Computing and Operation Reforming

あらまし

最近、ITの能力を水道や電気のように必要になったら必要な量だけ供給する「ユーティリティコンピューティング」のコンセプトが提唱されている。このコンセプトの極端な形態は、企業側にはコンピュータを設置せず、外部事業者が高速回線を使ってIT能力を供給するものであり、アウトソーシングの究極の姿と考えることもできる。しかし、現在のITシステムがユーティリティ形態になるためには、多数の機器やミドルウェアの統合や運用管理の自律化などまだまだ課題も多い。

本稿では、ユーティリティコンピューティングを具体化する際に直面する運用管理上の具体的な課題をあげ、これを解決する手段について考察する。

Abstract

Recently, the concept of "utility computing" has been proposed. In this concept, IT processing capacity is made available and used as required in the same way as water and electric power. Utility computing can be regarded as one of the most advanced forms of outsourcing because in some cases no additional computer hardware needs to be installed at the user's side in order to use it. However, establishing utility-like systems requires enormous efforts, for example, the consolidation of many multi-vendor devices and middleware and the realization of autonomous control of system operation. In this paper, we describe Fujitsu's approach toward solving the system operation problems of utility computing.



土屋 哲(つちや さとし)
Web&IPシステム研究センターユーティリティ研究部 所属
現在、自律コンピューティング技術、IDC運用技術についての研究開発に従事。



安達基光(あだち もとみつ)
Web&IPシステム研究センターユーティリティ研究部 所属
現在、自律コンピューティング技術、IDC運用技術についての研究開発に従事。

まえがき

半導体の高性能化・高密度化に従って、ITシステムの形態はメインフレーム、クライアントサーバ、Webシステムと分散化してきた。技術革新でハードウェア価格が安くなる一方で、こうした分散化によってシステムは複雑化し、運用管理コストが増大している。業務に要求される性能、信頼性を満たしながら、運用管理の複雑さ、コスト増大に対応する手段としてアウトソーシングが注目されているが^①、最近、これをもっと進めた形で、IT能力を水道や電気のように必要なときに必要な量だけ供給する「ユーティリティコンピューティング」が提唱されている。しかし、コンセプトが先行し、実現するに当たっての問題の議論が少ないのが現状である。

本稿では、ユーティリティコンピューティングの目指すもの、およびメリットを解説した後、その実現に当たっての具体的な障害となっている運用管理上の課題とその解決方法を述べる。

ITシステムとユーティリティコンピューティング

変革期にあるITシステム（導入から活用へ）

経済産業省が2004年に出した報告書「我が国の企業のIT化に対応する企業経営の分析」^②では、従来の部門別の生産性向上中心の考え方を脱して、「ITの活用」を重視した「経営の全体最適化」の方向に進むことを提言している。従来の「IT化」は「ITの導入」を意味し、パソコンやオンラインネットワークといった情報技術を導入して伝票処理や在庫管理などといった特定の個別業務の生産性を向上させることが目的であった。しかし、この報告書では、「ITの導入」はIT化の序章に過ぎず、経営とITシステムがネットワークを介してリアルタイムで情報を交換する「ITの活用」によって、単なる部門ごとの生産性向上にとどまらず、企業全体として「経営戦略の策定」「スピードアップ」「最適化」という新しい価値をねらうべき、としている。

では具体的に「ITの活用」とはどのようなものか、その評価項目の例を挙げると、

(1) 情報共有の活用度（業績把握）

- ・決算期のみを集計して把握
- ・部門内でオンタイムに把握
- ・企業全体でオンタイムに把握

・バリューチェーン企業群でオンタイムに把握 (2) 経営手法の活用度（経営への反映）

- ・大量生産型供給体制
- ・IT活用した需給バランスの調整
- ・顧客ニーズをすばやく企業レベル経営に反映
- ・企業の壁を越えた供給体制

となっている。つまり、ここでいう「ITの活用」とは、得られた情報を部門間で共有し、経営層がそれをオンタイムで見ながら決断を下しているか、あるいは需給バランスを細かく把握し、顧客ニーズの情報を企業全体で共有して経営判断に反映しているか、ということであり、従来のようなパソコンや電子伝票の導入による生産性向上にとどまらず「ITシステムで得られた情報を活用してリアルタイム経営に反映して業績向上につなげているか」を重要視している。

報告書では、実際に多数の企業へのアンケートが行われ、その結果から現在の日本の企業を四つのステージに分類している。

(1) ステージ1：IT不良資産化企業群

単に情報技術を導入しただけで、その活用がなされていない企業群

(2) ステージ2：部門内効率化企業群

情報技術の活用により、部門ごとの効率化を実現している企業群

(3) ステージ3：組織全体最適化企業群

情報技術を理解する経営者の決断と実行により、企業組織全体におけるプロセスの最適化を行い、高効率経営と顧客価値の増大を実現している企業群

(4) ステージ4：共同体最適化企業群

単一企業組織を越えて、情報技術により、最適なバリューチェーンを構成する共同体全体の最適化を実現している企業群

アンケートによれば、四つのステージの現在の比率は図-1のようになっており、日本の多くの企業（8割）はいまだステージ1、ステージ2という初期段階にある。同じアンケートで各企業に業績も調査しており、「業績が上向き」と答えた企業はステージ3以降の企業に多く、「IT活用度の高い企業は業績も良い」傾向にあることが分かっている。つまり、「ITの導入」から一歩進んで積極的に「ITの活用」を押し進めた企業ほど業績が好調であり、現在ステージ1、ステージ2にある8割の企業も、今後は

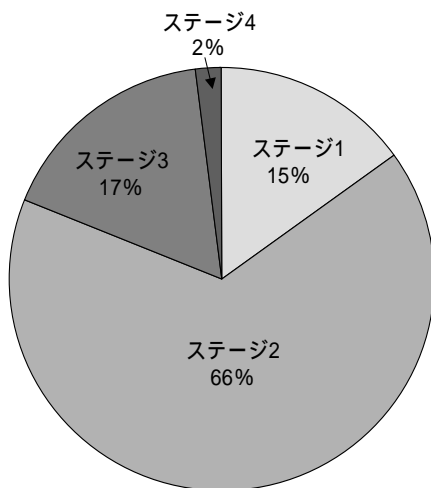


図-1 IT活用成熟度の分布
Fig.1- Distribution of IT utilization maturity.

「ITの活用」へと投資内容を変化させる、大きな変革期にあると思われる。

新しいシステム基盤

では「ITの活用」を可能とするシステムとはどのようなシステムだろうか。具体的なシステムはその企業の業種・業態によって様々であるが、いくつか一般的に共通な特徴がある。それをステージ3、ステージ4の評価項目をもとにまとめてみると次の三つになる。

(1) ITシステム間での相互情報流通

複数のITシステム間がネットワークで接続され、ITシステム間で相互に情報交換できる。

- ・ステージ3では企業の各部門のITシステム間の相互情報流通
- ・ステージ4では、(部門間ではなくて)バリューチェーンを構成する企業群(すなわち、グループ企業)の相互情報流通

(2) ITシステム規模の柔軟な変更

ITシステムの規模が要求量の変化にすばやく追従し、必要なときに必要な量の確保(品質の提供)ができること(高効率経営とバリューチェーン最適化)。

(3) ITシステムへの経営意思の適用

経営者の決断がすばやくITシステムに適用されること。

従来のITシステムは、サーバなどのシステム規模は設計時に最大能力を見積もり、一度決定したらシステム更新時まで固定であるのが普通である。新

しいITシステムではシステム規模はビジネスに合わせて変化させ、全体的な最適化を行って企業としての経営効率を高め、経営者は変化する需要をリアルタイムに把握して判断を行い、ITシステムはその判断に沿って動く。こうした動的に最適化するサイクルがあることがシステム基盤に求められるポイントである。

ユーティリティコンピューティング

最近、変化に対して動的に最適化を行うシステム基盤を「ユーティリティコンピューティング」という概念で呼ぶことが多い。なぜならユーティリティコンピューティングが実現されれば前節で示した三つの特徴を満たすことができるからである。

「ユーティリティ」とは水道や電気のように、企業活動の基盤となる資源を「必要なときに必要な分の供給を受け、使った分だけ料金を支払う」スタイルのことを指す。すなわち、「ユーティリティコンピューティング」とは、水道や電気のように「ITの能力・資源」について、必要なときに必要な分の供給を受け、実際に使用した分だけの料金を供給業者に支払う形態のことである。Sun Microsystems社のCEO(Chief Executive Officer:最高経営責任者)であるJonathan Schwartzは、過去の電力業界の変遷に例えてこれを説明している³⁾昔、電気が動力源として使われ始めたころは、各工場に電力担当役員がおり、工場の敷地に自前の発電機を設置して発電を行い、自社の使用量の見積りに合わせて「次にどういう発電・配電設備を購入すべきか」の計画を考えていたという。やがて電力供給の仕組みが整備されると、発電と送電は電力会社の仕事となり、各工場は高額な発電設備の導入を止め、代わりに電力会社から「必要なときに必要な量を購入する」スタイルに変わった。この電力業界で起こったようなことが、つぎにIT業界でも起こるというのが、最近のIT業界での一致した見方である。すなわち、CIO(Chief Information Officer:情報担当役員)がこれまで考えていたのは「次の年はどういうシステムを構築し、そのためにどういう『設備を購入』するか」であったが、今後は「次の数週間ではどういう『能力を購入』するか」に変わる、と言われている。

このような変化を可能にするのは、インターネットによってIP技術が広く普及したことでWAN回線

の価格も劇的に低下し、データ通信が同じ方式で安価に行われる基盤が整ってきたこと、また、速いサイクルで進化する計算機技術に対して、自社で計算機を購入して「所有」し、自社の要員をその設備向けに教育するよりも、変動する需要量に合わせて必要なときに必要な能力を従量制で「使用」する方が経営的に有利となることが挙げられる。

実際、今でもネットワークを使った消費者向けサービスの多くは、変動の幅が大きいし、あらかじめ需要量を見極めることも難しい。そのためには最大利用量を予想して固定的な台数を準備してもその設備が使われない時間が長く、無駄も大きい。こうした理由から、サービスの種類によってはハードウェアを購入して自前でシステムを構築・運用するよりも、サービスをアウトソーシングして、規模は需要に応じて変動させることを合理的と考える企業が既に出てきている。

ユーティリティ化に向けた課題

ユーティリティの分類と進化ステップ

ユーティリティは、資源の供給元がどこであるかで大きく二つに分類することができる。

(1) 企業内ユーティリティ (Private)

資源の供給元が企業の中にあり、企業ネットワークの中で資源供給と消費が行われる。利用規模は制限されるが、外部システムとの連携がないため、環境構築が容易でセキュリティ問題も小さい。

(2) 公共ユーティリティ (Public)

資源の供給元は企業の外にあり、外部ネットワークを介して遠隔資源の上でアプリケーションを動かす。外部事業者の大きな容量を利用することができるが、アプリケーションを動かす際に企業データのセキュリティを心配する利用者も多い。

ユーティリティというと、水道や電力との比較から企業の外の事業者からITサービスを受ける形態(公共ユーティリティ)を思い浮かべてしまうが、今のところ実際には外部事業者はまだほとんどいない。現在、外部事業者がサービスしているのは科学技術計算向けのCPUの時間貸し、といった限定された領域でわずかな例があるだけである。なぜなら一般の企業のアプリケーションは、稼働する環境を想定して一体となって開発されていて、外部事業者のリソース上ではすぐには動かないからである。企

業のアプリケーションを外部事業者のマシンで動かすには、「移植作業や構成設定のやり直し」などが必要で、これにはもう一つのシステムを構築するぐらいの手間がかかる。また、アプリケーションが動いた後、実際にはマシンの起動や停止、異常からの復旧、パッチの適用など「人でなければできない運用作業」も多数ある。こうした運用作業は利用者ごと、アプリケーションごとにやり方が違っているため、外部事業者にまかせようとする、システム運用手順の説明会を開くことから始める必要がある。

こうした現状から考えると、今後ITシステムをユーティリティ化して変革していくには、まず自分の企業の中で「企業ユーティリティ化」を行い、自分のITシステムのプラットフォーム依存性や人手への依存性をなくし、動的に最適化して動くようにすることである。これは自分のシステムのIT活用度を上げることであると同時に「公共ユーティリティ」サービスを受ける準備にもなる。

ユーティリティ化の課題

ユーティリティ化の主な課題は次の二つである。

(1) システムの仮想化

アプリケーションからリソース(CPU、ネットワーク、ストレージ)へのアクセスが仮想化されて、物理実体を意識せずに利用できること。

(2) 運用の自律化

一連の運用操作をできるだけ人手を排して自動で行い、短時間でミスなく運用管理すること。

一つ目のシステムの仮想化は、富士通のIT基盤「TRIOLE」製品などで徐々に実現されつつある。例えば最近のアプリケーションサーバは「アプリケーション配備機能」を備えるものが増え、開発サーバから運用サーバへ簡単にアプリケーション一式をコピーし、サービスの起動・終了もできる。しかし、二つ目の運用の自律化についてはまだ人による設定ファイルの書き換え、手動によるマシン操作など人手による部分が多く、自律化が進んでいない。

運用の自律化の課題

システムの仮想化は、OS、ミドルウェアといったソフトウェア進化の延長線上にあるが、これまでは運用の自律化はあまり重要視されてこなかった。しかし、ハードウェアが低価格化し、システム形態がWeb3階層のように分散化したことで、システム

を構成するノードは多数となり、運用管理の手間は膨大となり、今や主要な問題になってきている。

【構成設定の整合性維持】

Web3階層などの分散システムでは参加するマシン（ノード）が多数となる。各ノードには、ネットワークのIPアドレスからポート番号といった基本的な設定に始まり、ミドルウェアの設定からアプリケーションに依存した項目まで、数十～数百もの設定項目があり、分散型システムでは、これらをすべてのノードで整合がとれるように調整しなければシステムとして意味を成さない。しかし、サーバやアプリケーションはそれぞれ固有の目的のために作成されており、構成設定はそれぞれ機器、ミドルウェアの流儀に従って人が設定することを前提に作られているのがほとんどである。したがって、一つのノード内で整合性を取る（例えば、OSのポート番号設定を変更し、その設定に合わせてアプリケーションの構成定義を変更すること）、およびノード間や環境とノード間で整合性を取る（そのサイトのセキュリティポリシーに沿って使用するポート番号を変更し、Webアプリケーションサーバ間でポート番号の設定を合わせること）などは、人手でやるようになっていく。

従来のような固定環境においてもこのような整合性維持は運用管理者の大きな負担となっており、これがユーティリティ環境に移行した場合、企業と外部ユーティリティ事業会社、または同じ目的を持つ企業サイト間を連動させる必要に迫られ、こうした整合性維持の作業を短時間でやることは人手ではやりきれなくなる。

【環境依存で面倒な手順の記述】

システムが意図しないで停止してしまうことを計画外停止というが、その発生原因を調査してみると、最近ではハードウェアの故障が2割程度なのに対し、人為的な運用ミスが4割もある。

こうした人為ミスを防止するためには運用手順の実行をツールで行い、なるべく人手を排することが有効である。しかし、その手順の記述方法には注意が必要である。現在でも、すでに一部の運用管理者はUNIXのシェルスクリプトを使って運用手順を記述し実行しているが、こうしたスクリプトはその環境でしか使えず、別の環境を構築する際には毎回スクリプトを書き直さなければならない。というのも、

スクリプトはマシンの名前やミドルウェアが固定であることを前提に、環境に依存した書き方をしているからである。今後ユーティリティ運用になると、頻繁にマシンが変更されたり、外部事業者でアプリケーションが実行されたりするため、そうした場合にはこのスクリプトは使えない。

さらに、もし運用手順の記述ができたとしても、記述の手間が大きいと問題が残る。現在の運用操作のほとんどは人手であり、通常、こうした操作に必要な情報は文書化されて蓄積されている。しかし、現実には運用が始まった後でトラブルが発見されることが多く、こうしたトラブル対処方法が最初の文書に追記されずに運用が進行されてしまい、同じミスを繰り返すこともある。構築時になかった「現場ルール」（例えば、このメッセージは実害がないから無視しても構わない、といった細かな追加事項）は文書化されずに人に所属してしまい、その人が異動しても受け継がれず、同じ細かなミスが繰り返されることが多い。人為ミスの原因は運用開始時に準備された文書が不十分であったり、現場で運用中に手順が変更されたのに、それが伝達されていなかったりする場合が多いということである。したがって、運用修正ニーズが発生した時点ですばやく内容を追加・修正できるように、難解なプログラミング言語ではなくもっと図解的に記述し、文書よりも簡単に、そして確実に手順が実行される仕組みが必要である。

このように運用記述方式としては、環境に依存しないこと、そして記述の変更・追加が簡単であることが重要である。

運用の自律化の実現方法

ポリシー自律制御アーキテクチャ

構成要素間の整合性を維持する手段として有効なのが「ポリシー制御」である。ポリシー制御とは、利用者がサービスの要件・方針である「ポリシー」を与えると、管理機構がそのポリシーに沿ってシステムを制御する方式である。富士通が考えるポリシー制御では、ポリシーを次の4種類、お客様からのビジネス要件、サービス要件、システム制御方法、実行コマンド群に階層化し、これらを順次展開して自律制御システムの各機能ブロックに配布し、各機能ブロックが連動しながら一致した一つの目標を実現するように動作させるものになっている（図-2）。

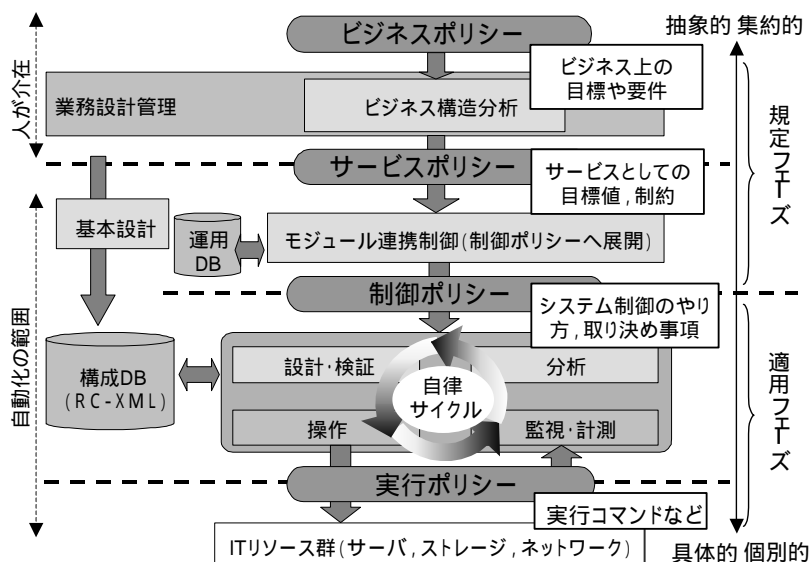


図-2 ポリシー自律制御アーキテクチャ
Fig.2-Policy based autonomic control architecture.

サービス要件は性能目標などであり、システム制御方法はIF-THENルールとして構成変更を行う条件（IF部分）と条件に合ったときに起動する論理アクション（THEN部分）のペアの形で記述する（制御ポリシー）。論理アクションとは、運用フローや操作フローといった手順の記述のことである。自律基盤製品は、監視計測、分析、設計・検証、操作の四つの機能から成り、これら四つの自律機能それぞれの制御方法として制御ポリシーを与える。自律サイクルの中で、監視・計測は状況の変化を検知し、分析は必要な対処が決定され、設計・検証で必要なシステム構成変更が選択されて、最後に操作部が実際に新しいシステム構成になるように構成変更コマンドを実行する。そのとき、論理的な変更ルールであった制御ポリシーはシステム依存の具体的な実行コマンド群（実行ポリシー）に変換されて実行される。

富士通のポリシー制御の特長の一つは、ポリシーが階層化されていて各階層ごとにルールが与えられること、そしてもう一つは（サービスポリシー以下では）ポリシーが上位から下位へ自動的に変換が行われることである。これらの特長により、高度なビジネス要件の変更の一貫性を維持しつつ、具体的なシステム構成に反映できる。例えば、下位の物理構成を変更し、新型機種を導入する場合、構成DB中の機種タイプを変更し、新しい機種の性能情報に

置き換える。性能が向上したので1台で処理できる能力は増えたので、以前のサーバなら増設が必要であった量の要求があってもまだサーバ能力は足りているから増設する必要はない。このように論理的な要件を示したサービスポリシーは同じまま、自律機能に与える制御ポリシーを新機種対応版に変換し直すことで対処でき、構成設定の整合性が維持される。

また、システム制御の自律機能群はモジュール構造となっており、顧客要件や技術の進歩に合わせて高度な判断機能を後から追加・改善していくこともできる。これにより、富士通が持つ構築・運用ノウハウを運用管理システムに蓄積し改善していくことができ、競争力を強化していくことができる。

作業プロセス管理層の追加

人為ミスを防止するための手段としては、運用作業プロセスを管理するワークフロー層を追加する。これは、TRIOLEで現在提供されつつある「自律システム基盤」製品、「統合運用管理」製品の上に「作業プロセス管理機能」を追加するものである。これによって、障害対応、パッチ適用、オペレータの監視など運用管理作業に必要な機能を統合し、作業手順やノウハウをフロー化して、どの管理作業でも利用できる運用作業環境を提供する。

【作業フローツール】

環境依存のスクリプト記述や紙の運用マニュアルの代わりにGUIを備えた作業フローツールを使い、

論理的なレベルでマニュアル情報と作業フローを一体化して記述し、物理的な構成、ミドルウェア固有の操作を排除し、環境依存情報を構成DBに登録するだけで過去の作業手順の再利用を可能にする。この「運用作業プロセスのフロー化」により、文書に記述されていない「現場ルール」をきちんと記述し、自動実行する。また、運用指示書を作業者間、システム間で共有・再利用し、間違いを防止することで、作業ミスの削減、運用マニュアルの保守や作業手順を作成・登録する工数の削減につなげる。

また、作業フローツールには、管理情報、イベント、操作、統合ビューを統合して、実行中の作業の進捗状況を記録したり、過去の作業フローの履歴も呼び出したりできるようにする。この統合化された「作業の進捗、履歴の管理」により、過去の作業に基づく継続的な改善も実現できる。

この履歴の管理については、さらに発展させて、最終的には「元に戻す」という構成変更のリカバリ(ROC: Recovery Oriented Computing)を目指している。これは、ワープロのundo機能のように、最近行った作業を覚えておき、さかのぼって作業前の状態に戻すことである。これには操作履歴を詳細に記録し、何回かの操作で削除したファイルデータなどを復活できるように別に保存し、これらをすべて操作履歴と結びつけて記録することで可能となる。

【運用シナリオのテンプレート化】

作業フローツールという登録、再利用手段のほかに、もう一つ開発を進めているのが、運用シナリオのテンプレート化である。GUIであってもゼロの状態から運用フローを作っていくのは手間がかかる

作業である。そこで一般的な運用作業についてはあらかじめいくつかの雛形(テンプレート)を準備しておき、現場ではテンプレートを組み合わせ、それに現場独自の事情を追加・変更するだけになれば、フロー記述の手間は大幅に削減できる。

そのため、現在、富士通SEの作業を調査し、現実の運用操作シナリオを詳細に分析し、その作業のテンプレート化を行っている。実際に利用されている多数の運用マニュアル、現場SEからの調査をベースに分析した結果、作業フェーズを次の四つに分けられることが分かっている。

- (1) M (Monitor: 監視)
- (2) A (Analyze: 分析)
- (3) D (Design: 設計)
- (4) E (Execute: 実施)

さらに、これら四つの作業フェーズは、いつも四つそろっているわけではない。作業によっては二つ、三つのことだけやる場合もある。そこで一つの作業に含まれるフェーズの種類(M/A/D/E)に応じて、つぎの三つの運用作業パターンに分類している(図-3)。

I MAE型: 障害対処など

監視(M)して、障害原因を分析(A)して、緊急対処を行う(E)。

II MADE型: システム変更(性能再設計)など

監視(M)して、障害原因を分析(A)し、設計の変更・実施計画を設計(D)し、変更作業を実施する(E)。

III DE型: パッチ適用など

監視、分析はなく必要な対処を設計(D)し、実

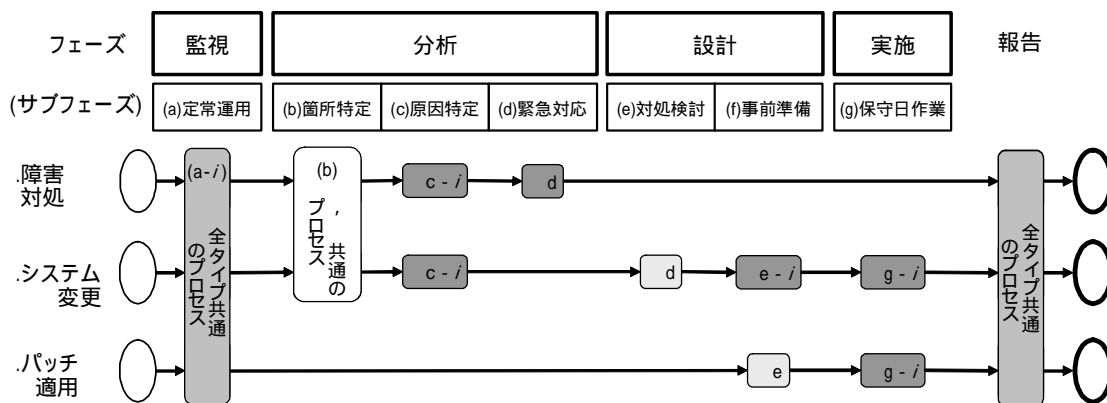


図-3 運用作業パターン(一部)
Fig.3-Operation workflow pattern (part).

施する（E）。

この分析結果をもとに、運用作業フローのテンプレートライブラリの整備を行っており、今後提供される、TRIOLEの作業プロセス管理製品の中からGUIで連結してフロー記述できる形にしていく予定である。

む す び

ビジネス環境の激しい変化への追従と業務の効率化を徹底的に進めていくと、ITシステム「所有」ではなく「使用（一時借用）」する「ユーティリティコンピューティング」に行き着く。設備を預かり運用を代行するというアウトソーシングから見ると、顧客の設備がなく、運用もすべて供給側が行う「公共ユーティリティ」は究極のアウトソーシングと言えるかもしれない。ユーティリティのコンセプトは有望ではあるが、現場で利用されるようになるにはまだ大きな問題があり、これから運用の変革を行い、多数の設定の整合性や運用作業の簡易化・ポータブル化などを解決していかなければならない。

そのためには、ポリシー制御を導入して設定値の間の整合性を維持すること、自律システム基盤の上に作業プロセス管理層を設け、紙マニュアルベース

の作業記述をフロー記述に変換して実行することが重要である。さらにフロー記述をすばやく作成し、現場の要求に合わせて改善するために、現在の運用作業を分析してテンプレート化する研究開発も並行して行っている。

ポリシー制御アーキテクチャと作業プロセス管理機能についてはTRIOLE次期運用管理基盤製品群への適用を進めており、これによって現場レベルからのユーティリティコンピューティングを浸透させていく。また今後はこれまで開発した機能を先行事例で検証し、機能の強化と確実性の向上を行う予定である。

参 考 文 献

- (1) HotWired：藤元健太郎の「ITビジネス原論」第9回 ITサービスの行方。
<http://hotwired.goo.ne.jp/original/fujimoto/041109/textonly.html>
- (2) 経済産業省：情報産業アウトルック2003。
<http://www.meti.go.jp/kohosys/press/0004086/>
- (3) CNET記事：コモディティ化の彼方に IT業界を待ち受ける「明るい未来」。
<http://japan.cnet.com/column/pers/story/0,2000050150,20081538,00.htm>