

メインフレームで稼動する既存 COBOL 資産の Windows サーバへの移行ソリューション

Micro Focus Net Express を利用した COBOL 資産移行ソリューション

メインフレームにおけるリソース管理を Windows で実現する

ProLiant Essentials Workload Management パック

マイクロフォーカス株式会社
技術部
日本ヒューレット・パッカー株式会社
IA サーバ技術部

目次

- ・ はじめに
- ・ オープンシステムへダウンサイジングするメリット
- ・ Micro Focus Net Express について
- ・ HP ProLiant Essentials Workload Management パックについて
- ・ 効果

- ・ Appendix
 - 検証シナリオ
 - 検証環境
 - 導入方法、最適化
 - よくある Q&A

理由の如何に関わらず日本ヒューレット・パッカード社およびマイクロフォーカス社の文書による許可なしに、本書の全部および一部を複写機等による複写複製、電子装置への入力などを含めて改変および転用することはできません。

日本ヒューレット・パッカード株式会社およびマイクロフォーカス株式会社は、本書の技術的もしくは編集上の間違いや欠落について、一切の責任を負いません。本書の内容は作成時点のものであり、いかなる保証を意味するものではなく、また、将来予告なしに変更することがあります。

日本ヒューレット・パッカード株式会社およびマイクロフォーカス株式会社は、製品保証書で保証する内容以外には、一切の保証はいたしません。本書の内容が、保証期間の延長や保証内容の拡張を意味することは一切ありません。

© Copyright 2003 Micro Focus. All Rights Reserved.

© Copyright 2003 Hewlett-Packard Japan, Ltd

すべての製品名、サービス名、会社名およびロゴは、各社の商標、または登録商標です。

製品の仕様・性能は予告なく変更する場合がありますので、ご了承下さい。

はじめに

世界各国では、すでに次世代の IT インフラへの移行が進んでおります。一方、日本では、多くの基幹業務を未だメインフレーム中心のシステムが支えています。メインフレーム中心のシステムでは、グローバルなサプライチェーンなど、企業間連携を行う為に大規模な変更を余儀なくされます。しかも、機能追加や仕様変更にさらに膨大なコストと時間がかかるため、市場の変化への迅速な対応が困難になっています。

今日の厳しい経済情勢の中で国際競争力をつけるには、メインフレーム中心の IT 環境からいち早く抜け出し、オープンシステムへと移行する必要があると考えられます。

マイクロフォーカス株式会社（以下 MF）と日本ヒューレット・パカード株式会社（以下 HP）は、メインフレームマイグレーション時に既存資産を最大限に活かしながら RoIT（IT 投資利用率）の最大化を追求するために共同検証を行いました。

このホワイトペーパーでは、Micro Focus Net Express にてコンパイルした既存 COBOL アプリケーションを HP IA サーバ ProLiant 上に配置し、HP の IT コンソリデーションツールの一つである ProLiant Essentials Workload Management パック（以下 WMP）を使用してワークロード管理を行うことにより RoIT の最大化を具現化していく方法を記載致します。

オープンシステムへダウンサイジングするメリット

1. システムの保守性を高める

昨今の社会・産業構造の急激な変化に伴い、ビジネスのやり方を絶え間無く変えて行くことが、企業にとって競合に勝ち抜き存続して行くための必須条件となっています。これに伴い、企業情報システムも、堅固さを保ちながら同時に変化にすばやく対応できる柔軟性を求められるようになってきました。

大企業では、20年、30年前に開発されたシステムが現在も稼動していることがあり、このようなシステムは長年の保守による変更の積み重ねのために、きわめて柔軟性の低いものになっている場合があります。

オープンシステムへのダウンサイジングにより、今まで利用することができなかった様々なミドルウェアやツールが利用可能となり、システム構築手段の選択肢は格段に増加します。

2. ダウンサイジングによるコスト削減

システム再構築を行う第2の理由はコスト削減です。

汎用機の開発・保守コスト、運用コストが、努力して削減することが非常に難しいのに対して、オープンシステムの世界でのハードウェアの価格性能比の向上スピードはめざましいものがあります。一旦初期コストを費やしてオープンシステムへダウンサイジングしてしまえば、以後10年のレンジでのコスト削減効果はきわめて大きなものになると期待されています。

この場合の付帯効果として、次のようなことも良く挙げられます：

- ・ 基幹系と情報系を統合できる
- ・ OA環境と連携できる
- ・ アウトソーシングしている運用をインハウス化できる

3. 旧環境がサポートされなくなる

汎用機のメーカーが、機種やOSのバージョンのサポートを停止するのに伴って、オープン環境へ移行するという例も多くあります。このような消極的なダウンサイジングの場合は、情報システムの機能に付加価値を与えることよりも、低コストで移行できることのほうが重要になります。

一時期、システムの西暦2000年対応に伴って、ダウンサイジングの計画を推進した企業も多くなりました。これも、このような消極的ダウンサイジングの一形態と考えられます。

Micro Focus Net Express について

1. マイクロフォーカスについて

英国ソフトウェア会社、マイクロフォーカスは 1976 年の設立以来、コンピュータの急速な発展と普及を早くから予想し、多種多様なコンピュータに対応する COBOL コンパイラおよび開発支援ツールの設計、開発、販売、保守に取り組み、常に業界をリードしてきました。

世界中でフォーチュン 500 といった金融、製造そして物流にいたる著名企業の多くのビジネスアプリケーション用 COBOL 開発環境としてマイクロフォーカス製品が採用され、事実上の業界標準 COBOL として広く認められています。

アプリケーション開発生産性の向上は無論のこと、最近では、既存資産のダウンサイジング化や WEB 化に際しての有効なソリューションとしてマイクロフォーカス製品が採用され、時代の変化に迅速に対応していくお客様のビジネスをご支援しております。常に、最新のテクノロジー環境に対応した製品を提供することで、今後も皆様のお役に立つ製品ならびにソリューションをご提供するのがマイクロフォーカスです。

2. マイクロフォーカス製品

今日のマルチプラットフォーム環境において、競争力を維持し、高めて行くために、既存のコードや技術を活用してアプリケーションを開発し、効率よく保守していくことは、もはや「あたりまえ」になってきました。

それに加え、Web サービス、XML など新しいテクノロジーを採用したシステムへの迅速な移行が必要不可欠とされています。Micro Focus 製品は、こういったエンタープライズ・アプリケーションの開発、保守、拡張、統合を強力に支援します。

「COBOL 資産とプログラマのスキルと有効活用し、情報システムを素早く構築」それが、Micro Focus Net Express です。

3. COBOL を選ぶ 7 つの理由

オープン環境下でのシステム開発においても、COBOL が最も適切なプログラミング言語である理由を解説します。

(ア) ビジネスロジックの記述力

COBOL は COmmon Business Oriented Language です。以下のような情報処理の世界で必要十分な機能が言語の文法としてサポートされています。

- ・ 順・相対・索引編成のファイルアクセス
- ・ ソートマージ
- ・ 31 桁までの十進演算
- ・ 金額編集等の印刷形式への変換
- ・ 非手続き型の帳票印刷機能

C や BASIC などでは、このような機能は言語とは別のライブラリを使用することになり、環境毎に別個のプログラムとなります。

(イ) 保守性・可読性の高さ

企業の情報システム部では、30 年前に書かれた COBOL プログラムが今も保守されて業務で活躍している例が多くあります。もちろんはじめにプログラムを書いた人はとくに担当ではなくなっています。C や Java で書かれたプログラムを、書いた人以外の人で保守することは一般に大変困難です。

COBOL ではプログラム書法が均一になり易いので、プログラマの熟達度による可読性のばらつきが少なくなることが知られています。

(ウ) システム開発方法論としての安定性

過去 30 年以上に渡って蓄積されてきたシステム開発の方法論は、アプリケーションが COBOL のような言語で書かれることを前提としています。もちろん、昨今の新しい開発パラダイムの登場により、このことは少しずつ変化して来てはいます。しかし、情報システムの構築方法はテクノロジーによって左右されるべきではなく、あくまでもシステムへの要求を満たすために選択されるべきです。この意味では、システム開発方法論は一般にいわれているほどには変化していないのが実状です。

昨今の大型のダウンサイジング事例でも、画面入出力だけは JSP、Servlet などで作成し、サーバ側のトランザクション処理やバッチ処理は COBOL で記述するという事例が増えてきています。開発案件の規模が大きくなればなるほど、実証された方法論としての COBOL を採用し、プロジェクトのリスクを低減することは重要になってきます。多くの実力のあるシステムインテグレータ、エンドユーザがこの方法を選択しています。

(エ) 高い実行性能

COBOL で書くと C で書くより実行速度が遅くなるという迷信があります。もちろんそのような例題を作成することは簡単です。しかし、実際のアプリケーションではどうでしょうか？ データ処理の世界で頻繁に行われる文字列の編集や、十進演算を考えてみます。C ではそのような機能は言語として持っていませんから、ライブラリ関数の呼び出しでプログラミングすることになりますが、COBOL ではコンパイラが直接機械命令に落とししてしまいますのでオーバーヘッドがありません。ビジネス分野では 実際には COBOL で書く方が早くなる場合のほうが圧倒的に多いのです。

(オ) COBOL プログラマの高度な資質

C 言語が書けるということは、単に C の文法を知っているということだけを意味します。ところが COBOL が書けるということは情報処理の基本を知っているということです。この意味で COBOL プログラマであることは、すでにその時点でエンドユーザと同じ言葉で話すことができる S E なのです。

加えて、メインフレームやオフコンの文化のなかで培われた、システムへの理解力や品質に対する厳格な姿勢は、COBOL プログラムの大きな財産です。

(カ) 高い開発生産性

C 言語では 1 行あたりに記述できるロジックの量は COBOL の数倍になります。このことから一見 C 言語では小さなプログラムで多くのことができるように見えます。実際、COBOL を知らない C プログラムにとって COBOL の "IDENTIFICATION DIVISION" を書くことは苦痛かもしれません。しかし、(ア) でも述べたように、逆に COBOL では 1 行の MOVE 文で済む処理が C では引数としてオペランドのアドレスと長さを渡す関数呼び出しになる場合もあります。プログラム設計からコーディング、単体試験までのトータルな生産性を考えるとき、同じ処理を実現するための開発所要時間は一般に COBOL のほうが少なくなるのです。

多くのシステムインテグレータでは、社内でステップ生産性の目安をもっていますが、一般に C のステップ生産性は 2 ~ 3 ステップ / 人時であり COBOL は 7 ~ 8 ステップ / 人時です。

また、COBOL プログラムの高い再利用性も無視できない要素です。

(キ) 将来性

情報システムのライフサイクルは 5 年から 10 年に及びます。こうした長い期間保守し続けて行かなければならないものを、いつまで存続するかわからないツールで開発してしまっても良いでしょうか。4GL や 簡易言語がその時々流行に左右されてきたことは歴史的事実です。COBOL は 30 年以上に渡って国際規格に保証されて来た言語です。一つのソフトウェアベンダーのテクノロジーに依存したのではなく、産業界全体がサポートしている言語です。国際規格自体も、最新のテクノロジーを反映して随時アップデートされて来ており、決して「過去の言語」などではありません。

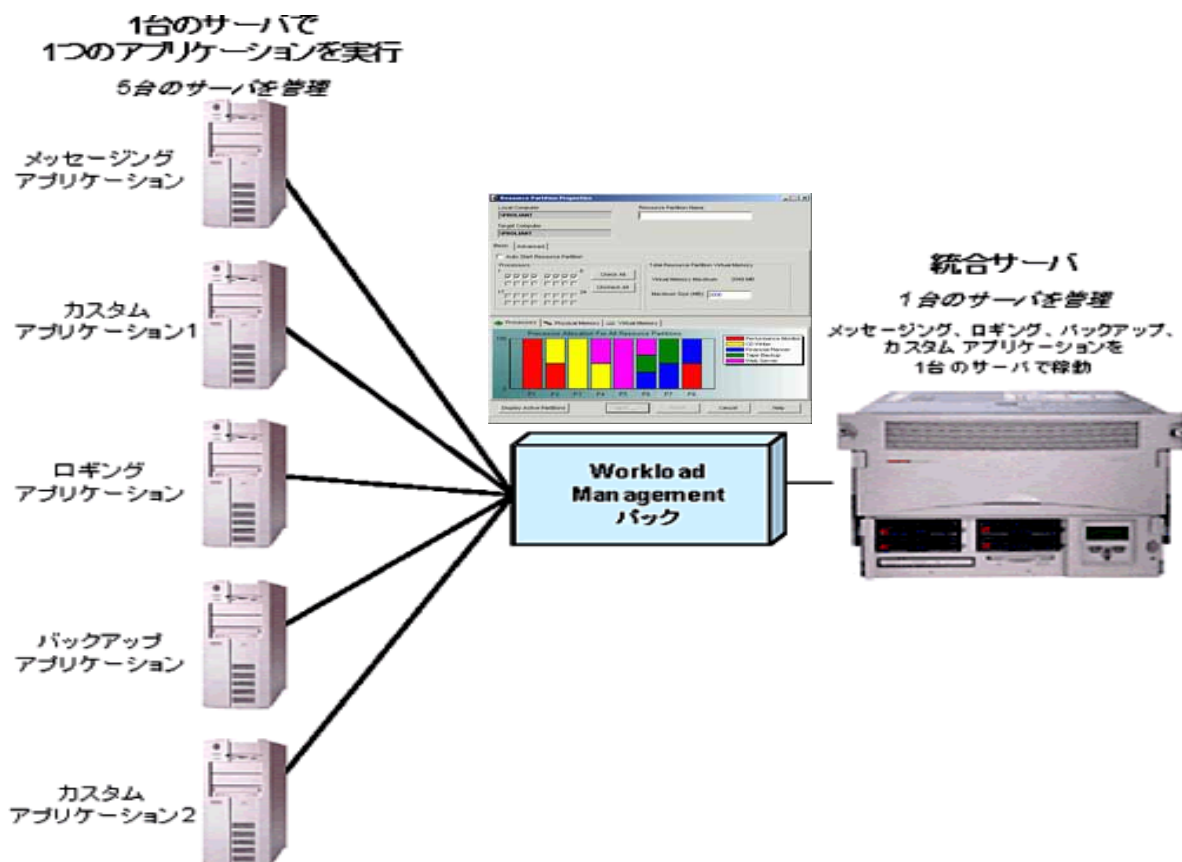
HP ProLiant Essentials Workload Management パック

ックについて

長期的なコスト削減と投資回収率の最大化を実現するためには、低価格ソリューションに投資する必要があります。

ProLiant Essentials Workload Management パックは、アプリケーションがサーバのすべてのリソースを必要とするかどうかにかかわらず新規アプリケーションのためにサーバを追加するというソリューションを見直す方法を提供できます。

新規アプリケーションのためにサーバを追加するという方法を繰り返していると、単一役割のサーバが過剰に配置され、管理費の上昇につながります。Workload Management パックを使用すると、相互に補完しあうアプリケーションを 1 台のサーバに集約でき、専用サーバで発生するアプリケーションのライセンス料金と管理費を迅速に排除して、最大利用効率を達成するサーバで投資回収率を最大化できます。加えて、システム内のリソースの割り当てを完全に制御できるので、不正な動作をするアプリケーションが統合環境の可用性を低下させる危険性を回避する事ができます。



HP ProLiant Essentials Workload Management パックの詳細な内容は、http://www1.jpn.hp.com/products/servers/proliant/essentials/wmp_sh.htmlをご参照ください。

効果

ダウンサイジングの際に、Micro Focus Net Express で作成した COBOL アプリケーションを Windows 上で稼働させた場合でも Workload Management パックを使用した事により UNIX システム等と同等以上のワークロード管理が行うことができます。

ここでは、Workload Management パックを使用してワークロード管理を行う事による効果を記載致します。

1. サービスレベルの維持

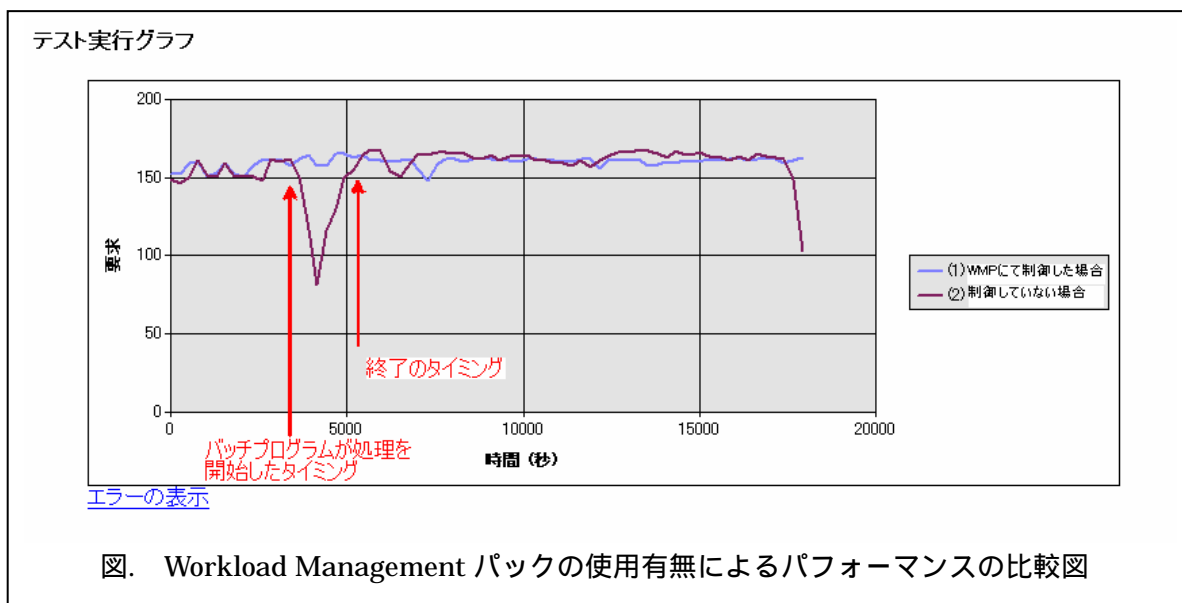
Workload Management パックでは、プロセスグループに対して使用するリソースを設定する機能があります。単一システム内に複数のアプリケーションを同居させるとアプリケーションに対して必要なリソースが割当てられないケースがあります。Workload Management パックを使用するとそのようなケースを回避してアプリケーションに対して必要なリソースを割当て、アプリケーションのサービスレベルを確保する事ができます。

ここでは、Workload Management パックを使用した COBOL アプリケーションのサービスレベルの維持について後述する Appendix での検証結果を元に記載します。

(ア) テスト概要

Appendix に記載されている検証環境において Web アプリケーションのベンチマークテストを行い、そのベンチマークテスト中にスケジュールしたバッチプログラム群を動作させる。これにより、1 システム内で複数のアプリケーションが動作し、その際に Web アプリケーションのパフォーマンスがどのように変化するかテストを行う。

(イ) テスト結果と考察



テスト結果より以下の事が確認できる。

- ・ WMP にて制御していないシステムでは、バッチプログラムを起動すると Web アプリケーションのパフォーマンスを著しく劣化させ、サービスレベルが 50%程度に下落する事。
- ・ WMP で制御したシステムでは、他のアプリケーションを動作させていたとしても Web アプリケーションのサービスレベルを維持できている事。

この結果より Workload Management パックを使用する事で、単一アプリケーションだけでなくシステム全体のサービスレベルを維持する事ができる事がわかります。これは、Workload Management パックを使用するとアプリケーションが使用する CPU、Memory を制御する事ができるからです。また、リソース割付けの自動化機能を使うとサーバリソース利用率の向上をさらに期待する事ができます。

2. システムリソース利用状況監視

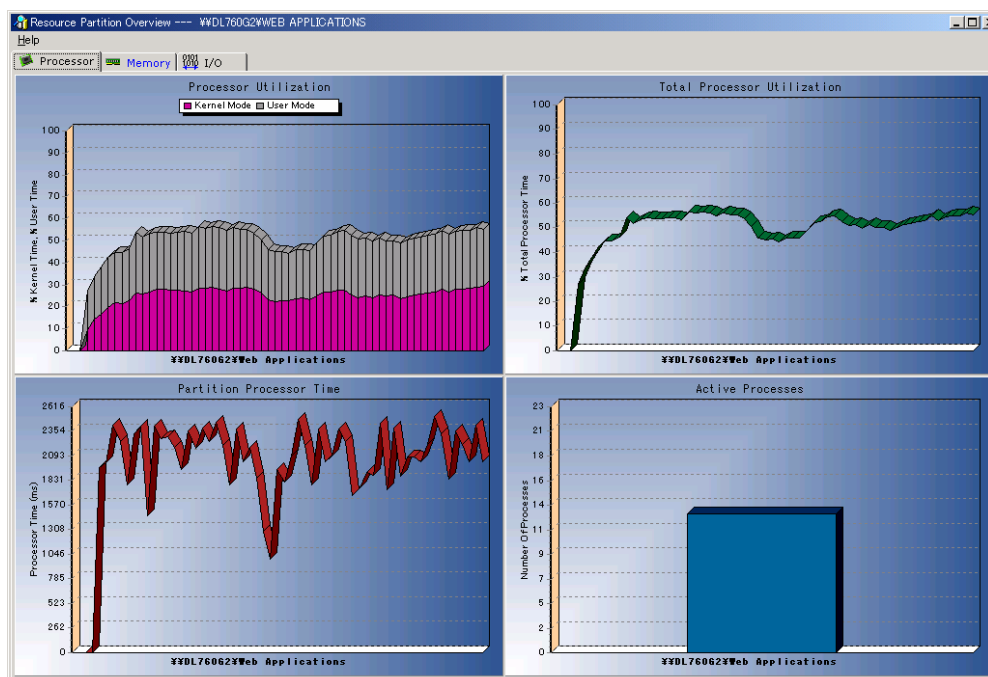
ここでは、Workload Management パックを導入したサーバのシステム利用状況の監視方法の例を記載致します。

(ア) 業務アプリケーション毎に監視

設定したリソースパーティション毎に下図のような監視機能が備わっており、Micro Focus Net Express で作成した業務アプリケーションが使用しているリソース状況を容易に確認する事ができます。

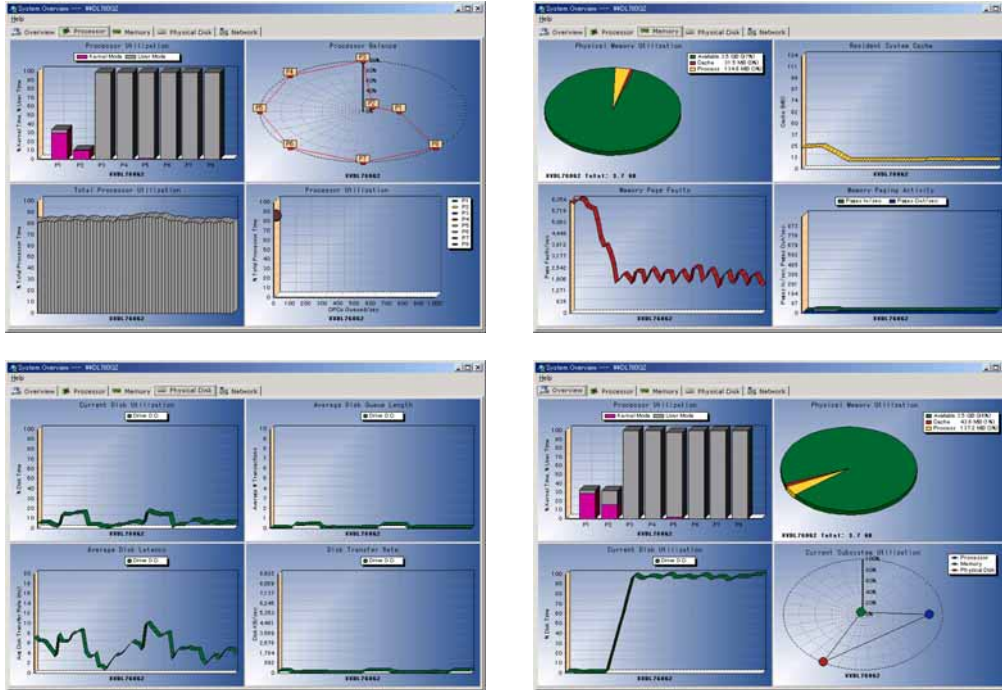
監視対象は以下のリソースです。

- ・ Processor
- ・ Memory
- ・ Disk I/O



(イ) システム全体を監視

システムの CPU、Memory、Disk、Network について下図のような監視機能が備わっており、システムの使用しているリソース状況を容易に確認する事ができます。



(ウ) ログファイルとしてテキストベースでファイルへ

アプリケーショングループ毎に設定した時間単位で使用していたリソースをファイルへ出力する事ができます。これらを定期的に保存しておく事で、システム利用率の把握、システム拡張時にシステム管理者の作業軽減に繋がります。

Machine Name: DL760G2		Partition Name: Web Applications						
Start Time	End Time	Average Processor Utilization	Average Processor Count	Average Virtual Memory	Average Physical Memory	Average IO Traffic (KB/sec)		
03:31	10/31/03 04:01	0%	4	129 MB	189 MB	0		
04:01	10/31/03 04:31	24%	4	133 MB	194 MB	0		
04:31	10/31/03 05:01	57%	4	134 MB	195 MB	0		
05:01	10/31/03 05:31	57%	4	134 MB	195 MB	0		

3. ライフサイクルの検討

今までのサーバ利用方法の多くは、システム拡張時には新しいサーバを新規に導入する必要がありました。しかし、このようなサーバの利用方法では、ビジネスの急速な変化に対応できず、RoIT（IT投資回収率）が伴わないケースも存在しました。

ProLiant Essentials Workload Management パックを使用することで1サーバに安全に導入できるアプリケーションの数に制限はなく、実際にアプリケーションのみが必要なリソースを追加するだけで、無駄にサーバを増やす必要がなくなります。その為、効率的にIT投資を回収する事が可能です。

また、上記に記載したシステム状況のログを分析する事でシステム管理者が必要なリソースを特定することが容易になります。

これによりProLiant Essentials Workload Management パックを使用してシステムを構築する事でシステムのライフサイクルに迅速に対応する事が可能となります。

Appendix

検証環境

ここでは、実際に検証を行った環境を記載致します。

1. Web/アプリケーションサーバ

(ア) DL760G2

- ・ CPU : Intel Xeon プロセッサ MP 2.0GHz x 8
- ・ Memory : 4GB 使用可能(5GB 搭載)
- ・ Disk : 36.2GB x 2 (RAID1+0)
- ・ アレイコントローラ : Smart アレイ 5i コントローラ (オンボード)
- ・ OS : Windows 2000 Advanced Server + SP4
- ・ ソフトウェア : ProLiant Essentials Workload Management パック

2. DB サーバ

(ア) DL760

- ・ CPU : Pentium Xeon プロセッサ 900MHz x 4
- ・ Memory : 4GB
- ・ Disk : 18.2GB x 4 (RAID0)
- ・ アレイコントローラ : 内蔵 Smart アレイコントローラ (オンボード)
- ・ OS : Windows 2000 Advanced Server + SP4
- ・ ソフトウェア : Microsoft SQL Server 2000 Enterprise Edition + SP3

3. 導入アプリケーション

主に、Micro Focus Net Express で作成した COBOL アプリケーションを使用。

(ア) バッチプログラム

- ・ CPU 負荷の高いアプリケーション (COBOL プログラム)
- ・ Disk I/O 負荷の高いアプリケーション (COBOL プログラム)

(イ) オンライン業務を行う Web アプリケーション

- ・ ビジネスロジックを司る COM+ アプリケーション (COBOL プログラム)
- ・ 上記 COM+アプリケーションを使用した Web アプリケーション (Active Server Pages)

4. Web アプリケーション負荷テストツール

(ア) Microsoft Application Center Test

- ・ Microsoft 社より提供されている Web アプリケーション負荷テストツール
- ・ 詳しくは下記 URL を参照の事

http://www.microsoft.com/japan/msdn/library/default.asp?url=/japan/msdn/library/ja/act/html/actml_ref_prps.asp



http://www.microsoft.com/japan/msdn/library/default.asp?url=/japan/msdn/library/ja/act/htm/actml_main.asp

導入方法、最適化

ここでは、Micro Focus Net Express で作成した COBOL アプリケーションを上記システムへ導入する例を記載します。

1. 導入前作業

本フェーズは、本番環境へのアプリケーションを安全に配置する上で非常に重要なフェーズです。以下の内容をテスト環境にて十分に検証し、その結果を元に十分に検討する事をお勧めします。

(ア) システム導入するアプリケーションを検証

アプリケーションのパフォーマンス測定を行い、要件を満たすような CPU、Memory、Disk 容量を検証します。

以下のようなワークシートを作成して本番環境へアプリケーションを導入した際にどのようにシステムリソースを利用するかを検証します。

表. 導入アプリケーション一覧ワークシート

	アプリケーション名	プロセス名	重要度	プロセス数の制限	最高 CPU 使用量	最低 CPU 使用量	メモリ使用量の制限値	稼働時間	考慮事項
1	IIS、 Web アプリケーション	inetinfo.exe、 dllhost.exe	高	特になし	100%	10%	50MB	24 時間稼働	
2	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
3	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
4	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
5	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
6	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
7	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
8	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
9	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
10	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
11	業務 COM+	dllhost.exe	高	特になし	30%	5%	30MB	未定	
12	CPU 負荷バッチ A	Sieve01.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
13	CPU 負荷バッチ B	Sieve02.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド

14	CPU 負荷バッチ C	Sieve03.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
15	CPU 負荷バッチ D	Sieve04.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
16	CPU 負荷バッチ E	Sieve05.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
17	CPU 負荷バッチ F	Sieve06.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
18	CPU 負荷バッチ G	Sieve07.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
19	CPU 負荷バッチ H	Sieve08.exe	低	特になし	100%	0%	3MB	毎日 22:00 - 0.5 時間程度	シングル スレッド
20	I/O 負荷バッチ A	cob05.exe	低	特になし	10%	0%	5MB	毎週土曜日 05:00 - 0.5 時間程度	シングル スレッド

上記資料より本番環境のワークロード管理について検討します。

1. 同名プロセスアプリケーション

COM+アプリケーション等のように同名でプロセスを起動するアプリケーションは、どのアプリケーションかの判別ができない為、一つのリソースパーティションに含めるか、もしくは、それらを判断する為の仕組みを用意する必要があります。

2. シングルスレッドアプリケーション

バッチプログラム等では、未だに多く見受けられますが、シングルスレッドアプリケーションは、複数の CPU に跨って制御を行うより一つの CPU に対して処理を割当てた場合に処理効率が向上するケースがあります。

3. ルールの設定

アプリケーションの稼働時間には、リソースパーティションを”Active Resource Partitions”へ、稼働時間外には、リソースパーティションを”Available Resource Partitions”へ変更する事でリソース利用効率をさらに向上させる事が可能です。簡単な設定は、ルールの設定により可能です。また、複雑な月次、週次処理等は、統合監視ツール等のスケジュール機能と連携してコマンドラインから構成変更を行う事ができます。

4. チューニング

システム全体でパフォーマンスが上がらない場合には、どこでボトルネックを起しているのかを判断する必要があります。また、その原因次第では、H/W、S/W チューニングやシステム構成変更等も検討する事をお勧めします。

さまざまな検討を十分に行い、以下のようなワークシートを作成する事をお勧めします。

表. リソースパーティション設定項目ワークシート (簡易版)

	リソースパーティション名	導入アプリケーション名	最高 CPU 使用量	最低 CPU 使用量	初期設定時の CPU 番号	仮想メモリ使用量の制限値	プロセス数の制限	プロセス名	稼働時間	考慮事項
1	Web Applications	IIS 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+ 業務 COM+	350%	60%	1,2,3,4	350MB	特になし	inetinfo.exe,dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe dllhost.exe	24 時間稼働	
2	sieve01	CPU 負荷バッチ A	100%	0%	5	3MB	特になし	sieve01.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
3	sieve02	CPU 負荷バッチ B	100%	0%	6	3MB	特になし	sieve02.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
4	sieve03	CPU 負荷バッチ C	100%	0%	7	3MB	特になし	sieve03.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
5	sieve04	CPU 負荷バッチ D	100%	0%	8	3MB	特になし	sieve04.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
6	sieve05	CPU 負荷バッチ E	100%	0%	5	3MB	特になし	sieve05.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
7	sieve06	CPU 負荷バッチ F	100%	0%	6	3MB	特になし	sieve06.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
8	sieve07	CPU 負荷バッチ G	100%	0%	7	3MB	特になし	sieve07.exe	毎日 22:00 ~ 0.5 時間 程度	シングル スレッド
9	sieve08	CPU 負荷バッチ	100%	0%	8	3MB	特になし	sieve08.exe	毎日 22:00	シングル

		H							~0.5時間 程度	スレッド
10	cob05	I/O 負荷バッチ A	10%	0%	4	5MB	特になし	cob05.exe	毎週土曜日 05:00~0.5 時間程度	シングル スレッド

2. 本番環境構築、アプリケーション導入

(ア) OS のインストールを行います。

詳細は以下のページよりホワイトペーパーをご参照ください。

HP ProLiant サーバの Whitepaper ページ

<http://www1.jp.hp.com/products/servers/proliant/whitepaper/index.html>

HP の Windows Server 2003 専用ページ

<http://www1.jp.hp.com/products/software/oe/windows2003/support/proliant/>

(イ) アプリケーションのインストールを行います。

3. Workload Management パック導入

(ア) 以下の手順に従って Workload Management パックを導入致します。

CDROM ドライブに Workload Management パックの CD を入れます。

右画面のような画面が立ち上がります。“Next”を押下します。

自動的に画面が立ち上がらない場合には、CDROM 内の “setup.exe” を実行します。

ライセンス許諾契約画面が立ち上がります。契約内容を十分理解した上で、“I accept the terms in license agreement”セレクトボタンにチェックして、“Next”を押下します。

ライセンス入力画面が立ち上がります。CDROM に付属されているライセンスを入力して“Next”を押下します。

インストール準備完了画面が立ち上がります。インストールする場合には、“Install”を押下します。

インストール完了画面が立ち上がります。“Finish”を押下します。これで Workload Management パックの導入は終了です。



4. アプリケーション設定方法

(ア) 導入前作業で作成したワークシートに従ってリソースパーティションを作成します。

Resource Partitioning Manager の管理ツールを起動します。

[スタート] - [プログラム] - [HP Resource Partitioning Manager]を起動します。

右画面のようにメニュー [Partition] - [Create]を押下します。



リソースパーティション作成ウィザードが立ち上がります。上記ワークシートの設定項目をウィザードに従って設定するとリソースパーティションが作成できます。

すべての設定を行うと右画面のようになります。

これでアプリケーションの設定は終了です。



よくある Q&A

HP ProLiant Essentials Workload Management パックについて

Q	どのようなライセンスが提供されていますか。
A	Workload Managementパックは、サーバ単位でライセンスされます。
Q	ProLiant Essentials Workload Management パックの中身は何ですか？
A	ProLiant Essentials Workload Managementパックは、製品名です。ProLiant Essentials Workload Management パックは、以下の内容で提供されております。 <ul style="list-style-type: none"> ・ Resource Partitioning Manager Windowsサービスコンポーネント、管理ツールから成るソフトウェア ・ ライセンスキー ・ ユーザガイド
Q	アプリケーションを1台のサーバに集約できるかどうかを判断する方法を教えてください。
A	環境を統合できるかどうかを判断するには、まず、現在アプリケーションが使用しているサーバ リソースの量を評価します。サーバ リソースを完全に利用していない複数のアプリケーションが存在する場合、これらのアプリケーションを統合できます。
Q	Resource Partitioning Managerは、VMware ESX Serverとどのように違うのですか。
A	Resource Partitioning Manager (RPM) とESX Serverは、同様の機能を提供できますが、実装方法が異なります。RPMを使用すると、単一のWindows 2000オペレーティング システム上に複数のパーティションが作成されます。VMware ESX Serverの場合は、各パーティションがそれぞれ自身のオペレーティング システムを実行するので、1台のサーバに複数のオペレーティング システムが存在します。RPMはリソースの可用性を確保したアプリケーション統合を提供しますが、ESX Serverは1台のハードウェアに複数の仮想サーバが存在する物理統合を提供します。
Q	Resource Partitioning Managerは、それ自身で何%のサーバ リソースを使用しますか。
A	Resource Partitioning Managerは、非常に少量のリソースで比類のない機能を提供しています。ほとんどの場合、Resource Partitioning Managerが使用するリソースは、パーティションが使用できるシステム リソース全体の5%未満です。
Q	Resource Partitioning Managerは、どの言語をサポートしていますか。
A	Resource Partitioning Managerは、英語のみをサポートしています。Microsoft Windows 2000の英語、フランス語、イタリア語、ドイツ語、スペイン語、または日本語バージョンを実行するサーバにインストールする必要があります。
Q	RPMは、他のソフトウェア パッケージのライセンス料金に影響しますか。
A	RPMを使用すると、複数のアプリケーションを実行するために必要なサーバの台数を減らしてライセンス料金を削減することができます。複数のアプリケーションを1台のサーバに集約すると、ベンダが規定するすべてのライセンス料金が適用されます。リソース パーティションは、ライセンス料

	金の計算では個別サーバと見なされません。
Q	ローカル コンソールを使用して、別のサイトにあるサーバのRPMパーティションを管理できますか。
A	はい。デスクトップPC、ノートブックPC、またはサーバで動作する単一のRPMコンソールを使用して、RPMパーティションを持つネットワーク上の任意の他のサーバを管理できます。
Q	Resource Partitioning Managerには、どのようなクラスタ サポートが提供されていますか。
A	Resource Partitioning Managerは、クラスタ環境で障害時のアプリケーションに「退避場所」を提供する機能を提供しています。RPMを使用すると、フェールオーバーに備えて、別のサーバにアプリケーション用のプロセッサ リソースとメモリ リソースを予約できます。また、RPMは、退避場所なしで標準的なMicrosoft Windows 2000クラスタのノードでも使用できます。
Q	「退避場所」とは何ですか。
A	退避場所とは、フェールオーバーに備えて、アプリケーション用に確保されるターゲット サーバのパーティションです。退避場所により、フェールオーバー サーバを受動スタンバイ状態に保持する必要がなく、フェールオーバーに備えてアプリケーション用のリソースが確保されます。
Q	Resource Partitioning Managerのリソース パーティションの中で任意のアプリケーションを実行できますか。
A	リソース パーティションの中では、Windows 2000で正常に動作する任意のアプリケーションやサービスを実行できます。
Q	Workload ManagementパックとMicrosoft SQL 2000データベースを使用するとき、制限事項はありますか。
A	RPMを使用すると、他のアプリケーションがSQL Serverを妨害しないので、SQL Server2000を使用したいお客様にとってRPMは最適です。この使用方法では、RPMは、SQL Serverのデータベースに内蔵されているツールを補強します。ただし、HPとMicrosoftは、SQLにはそれ自身の管理ツールがあるため、SQL Server自体をリソース パーティションで使用しないことに合意しています。
Q	リソース パーティションには、最大でいくつまでアプリケーションを追加できますか。
A	リソース パーティションに追加できるアプリケーションの数には制限がありません。Resource Partitioning Managerには、アクティブ パーティションに無制限にプロセスが生成されることを防止するために、特定のパーティション内のプロセス数を制限するユーザ設定があります。デフォルト設定は、"unlimited (無制限)"です。