HPCシステムの新たな可能性
パーソナルクラスタの考察

スケーラブルシステムズ株式会社
戸室　隆彦

HPCシステムの新たな可能性
パーソナルクラスタの考察

パーソナルクラスタの背景

- HPCマーケット
- HPCシステムの課題
- マイクロプロセッサの方向性
- TCOの重要性

HPCシステムの二極分化

- ペタスケール コンピューティング
- コモディティ コンピューティング

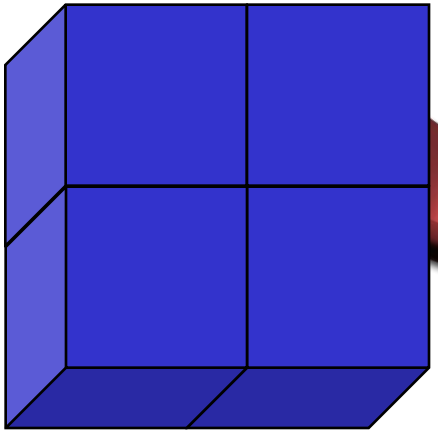
パーソナルクラスタシステム

- システムの特徴
- 並列プログラミング

まとめ

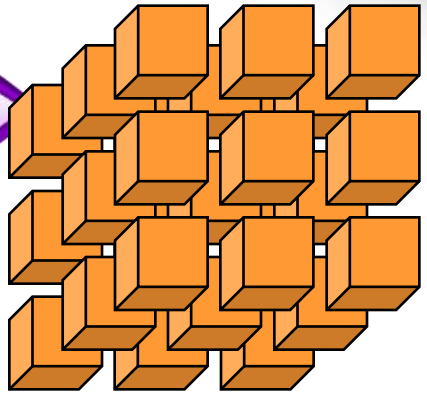
HPCプラットフォームの変遷

■メインフレーム
■スーパーコンピュータ

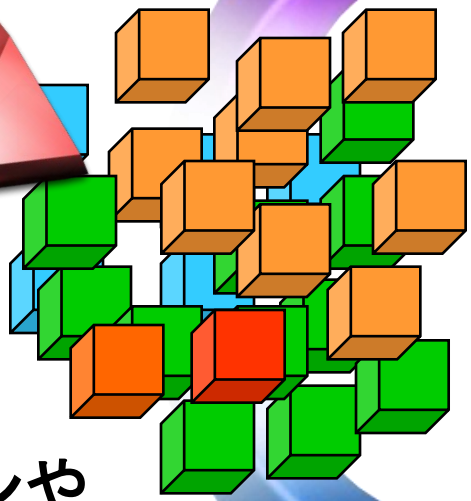


（リソースの集中と管理）

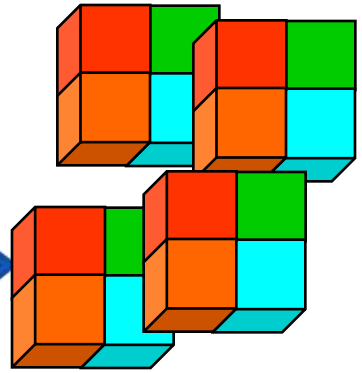
■クラスタによる
仮想コンピュータ



（分散したリソースの管理）



■ワークステーションや
サーバによる分散処理

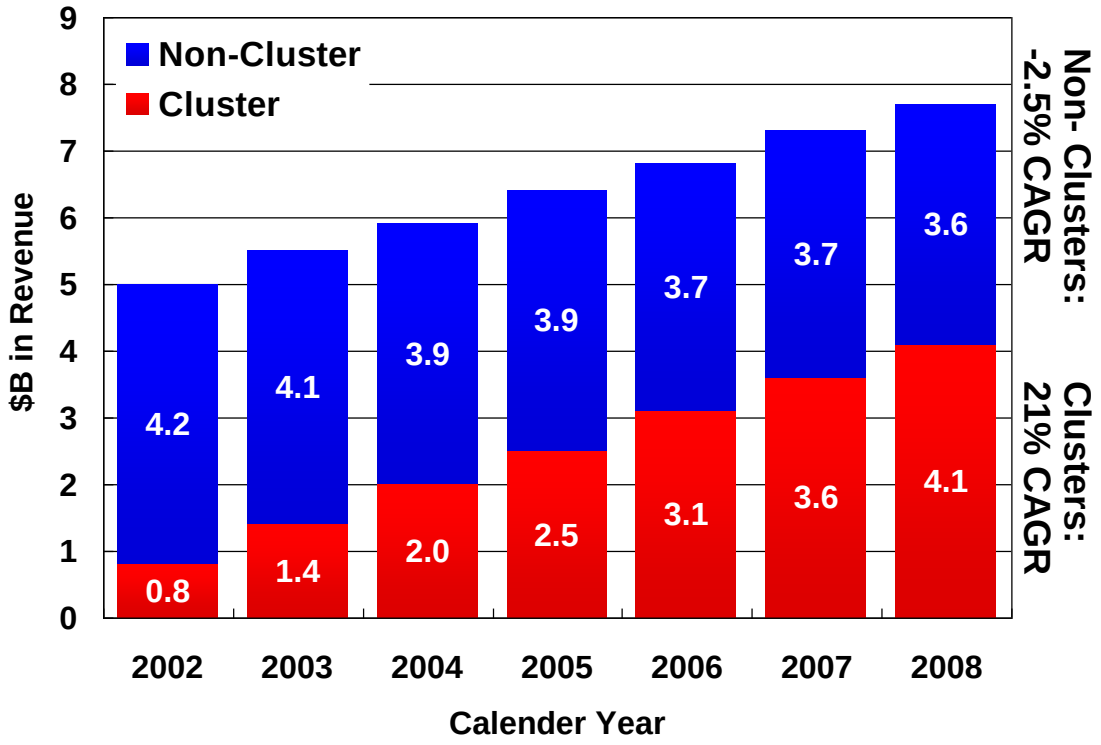


■仮想化による
サーバ・コンソリデーション

HPCマーケット

HPCマーケットでのx86サーバの売り上げが、5.9%CAGRであるのに対して、21.6% CAGRの伸びを示している （IDC)

Worldwide High Performance Computing Market



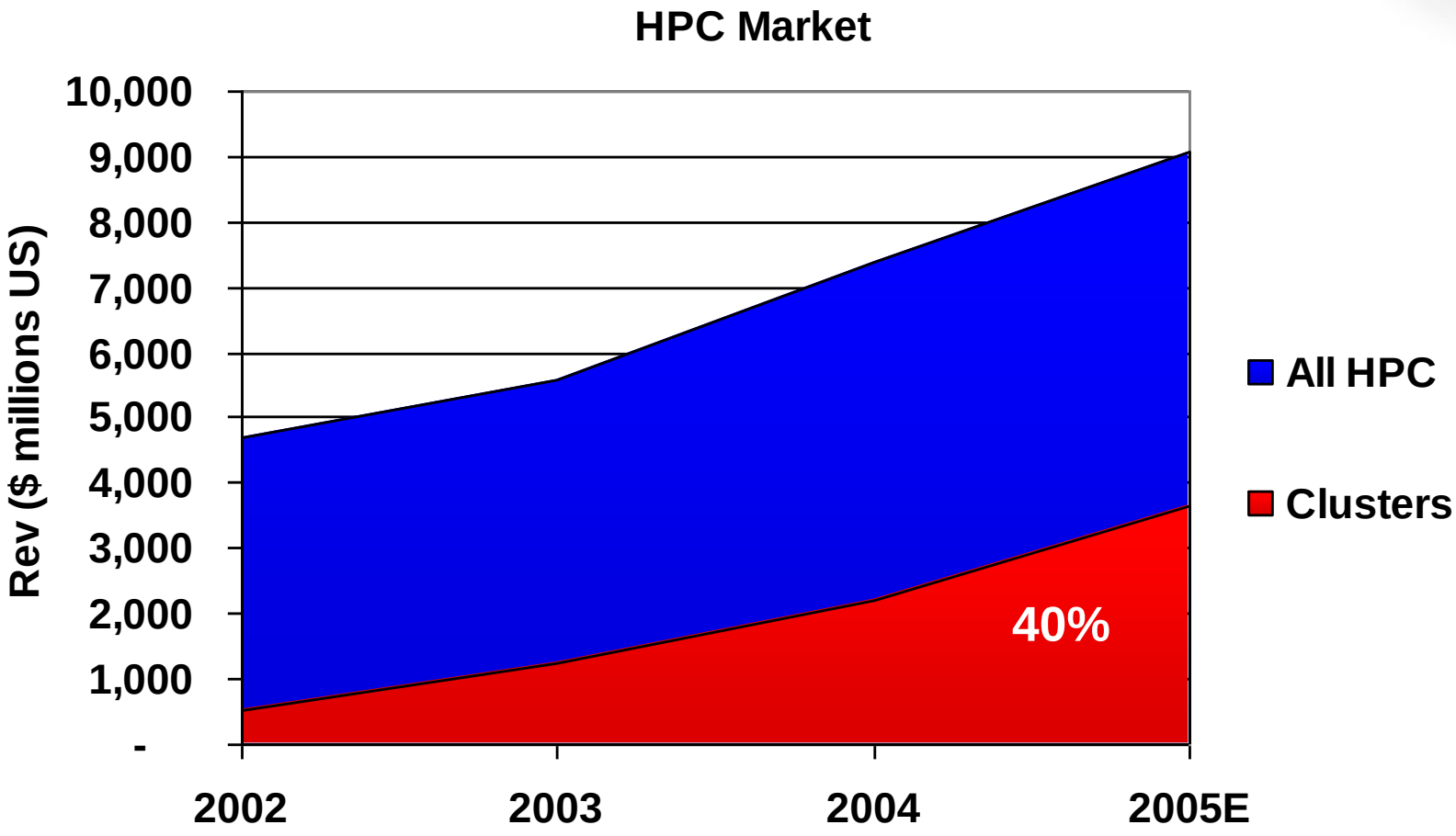
Clusters Accounted for 33% of Revenue in ‘04

部門向け（Departmental HPC、64ノード以下)クラスタシステムが、クラスタ導入の牽引（ユニット、売り上げとも）

（補足）
クラスタの出荷数の90%以上は、＄250K以下の価格レンジ
平均のクラスタのプロセッサ数は、8－16

WW HPC Server IDC Forecast

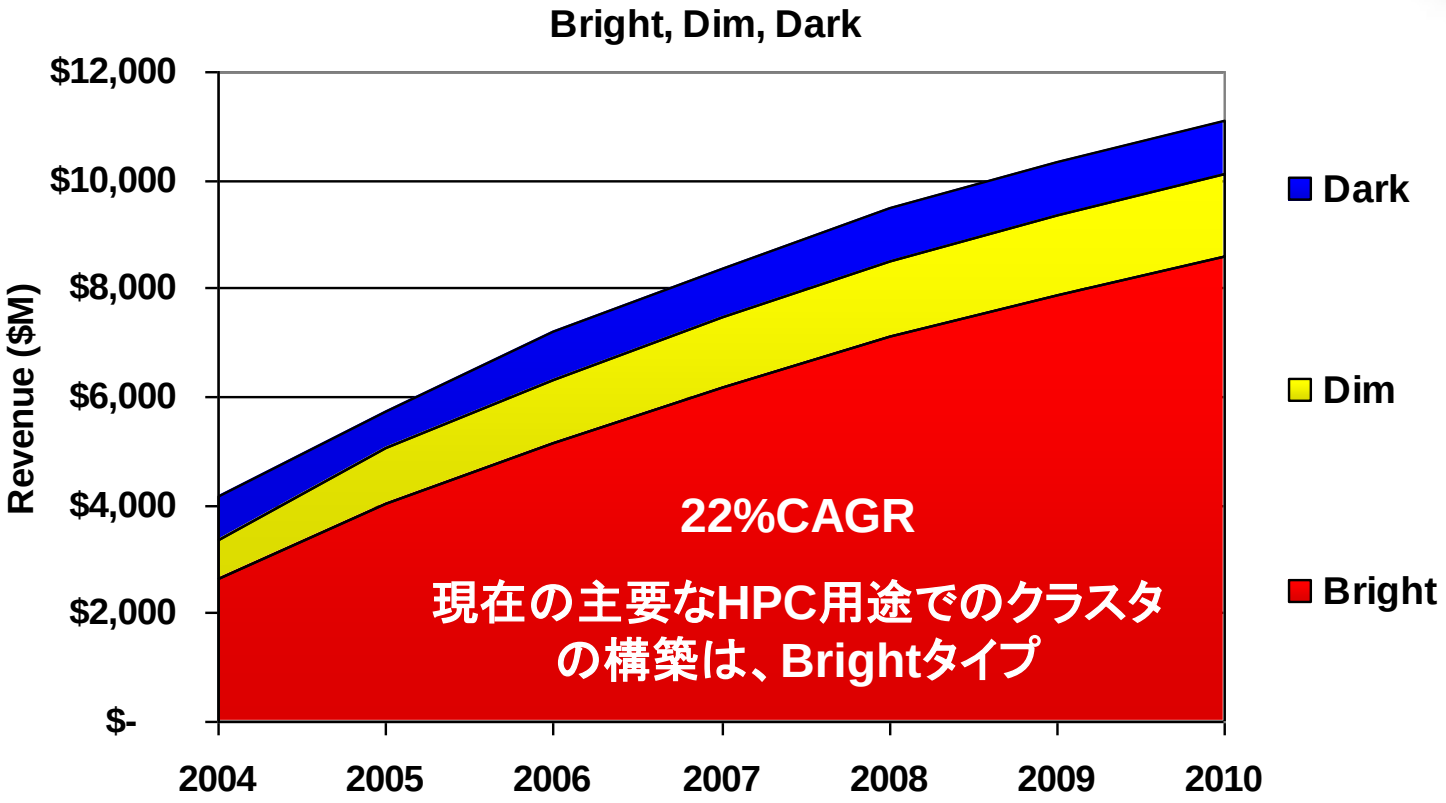
†IDC MCS: The Cluster Revolution in Technical Computing Markets (2006), IDC, Feb 2006, #



→Question:残りの60%は?

WW HPC Server IDC Forecast

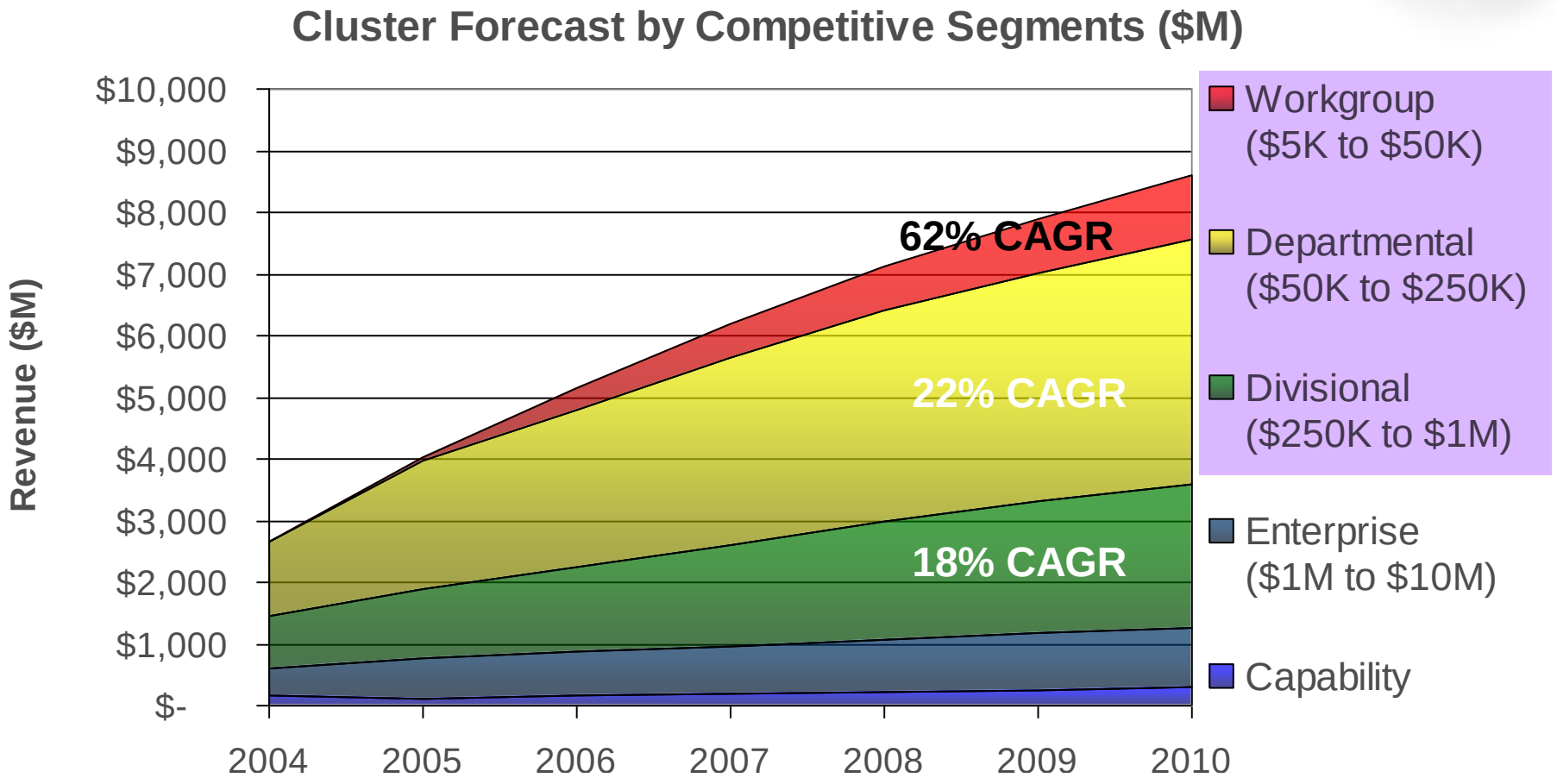
†IDC MCS: The Cluster Revolution in Technical Computing Markets (2006), IDC, Feb 2006, #



Bright Clusters: ベンダーがクラスタを構築して販売し、ノード単位でシステムをカウントするのではなく、トータルなシステムとしてカウントする→究極のBright Clusterは？
Dim Clusters: ユーザがノードを個別に購入し、クラスタを構築する

WW HPC Server IDC Forecast

†IDC MCS: The Cluster Revolution in Technical Computing Markets (2006), IDC, Feb 2006

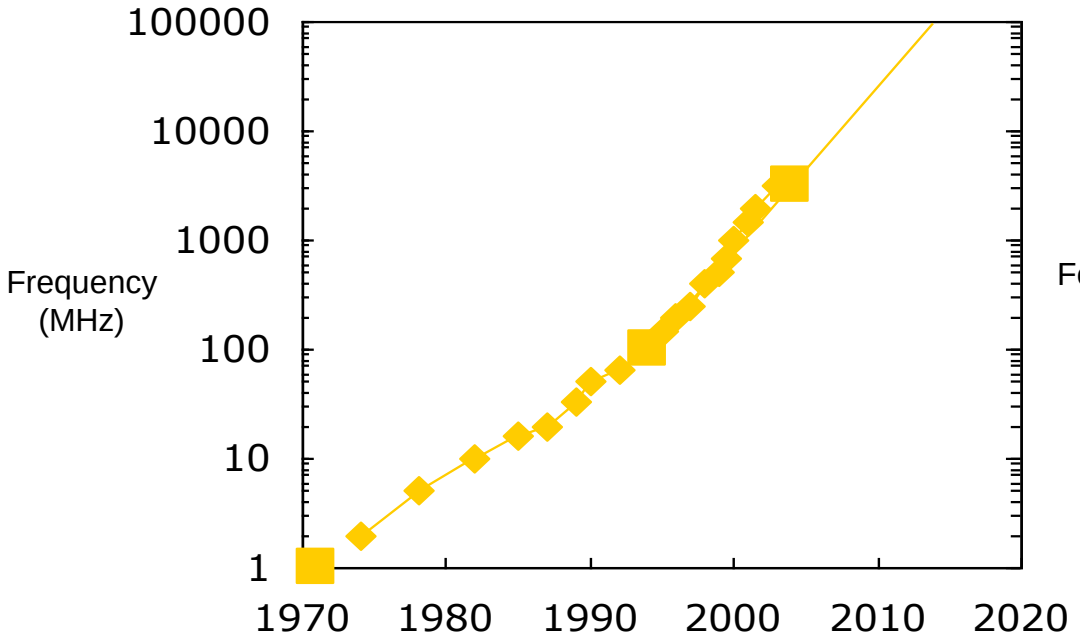


HPC マーケットでのビジネス

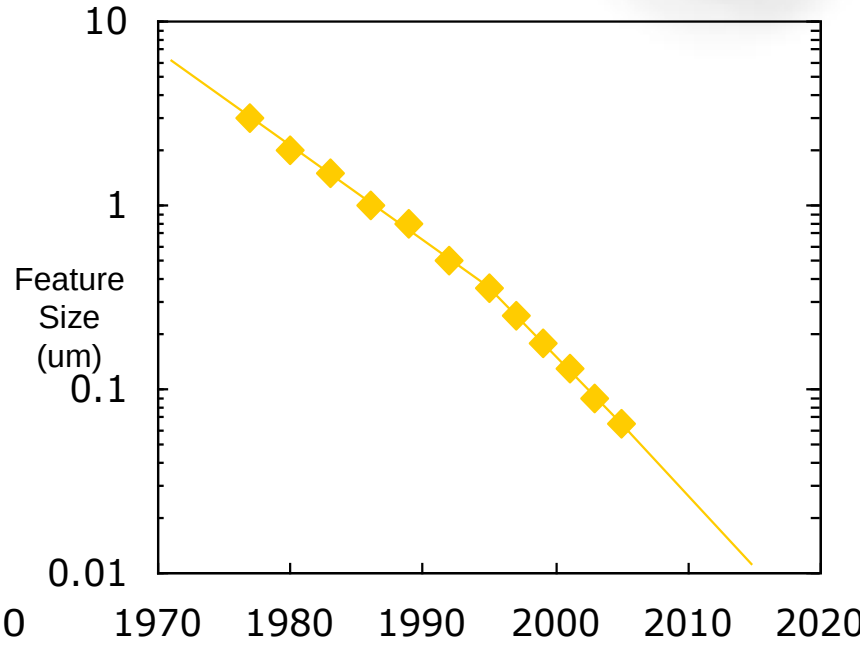
- HPC向けクラスタの伸びは堅調 （91.3% CAGR）
- ‘Departmental’と ‘Divisional’に分類されるマーケットでは、それぞれ、22% と 18% の成長を予測
- ‘Workgroup’は、62% CAGR（2005年末から1010年の間）を予想
- Bright Clusters（OEMがインテグレーションを行い、工場出荷時に既に組みあがっているクラスタ）は、22% CAGR を予想（各社がそのような計画を持つ）

歴史的な性能向上の努力

動作周波数の向上による性能向上



微細化



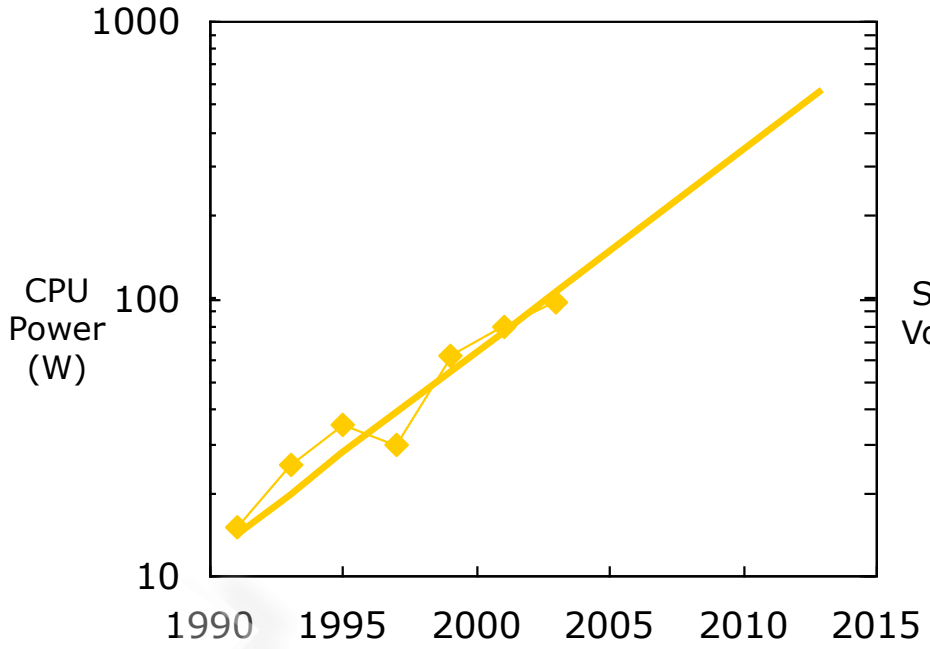
1946
メモリには、20個の
数値だけを格納可能

1971
I4004 プロセッサ
2300 トランジスタ

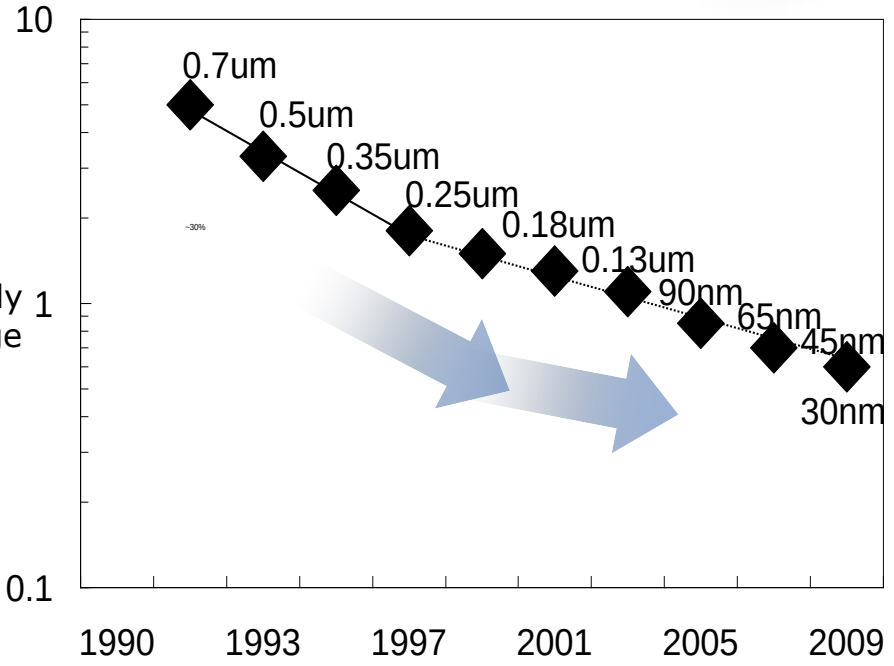
2005
65nm
+100億 トランジスタ

マイクロプロセッサの技術的な挑戦

消費電力の限界



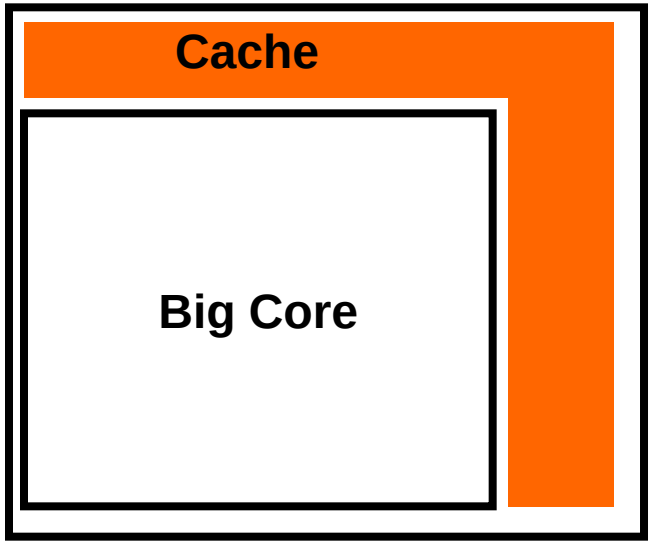
電圧



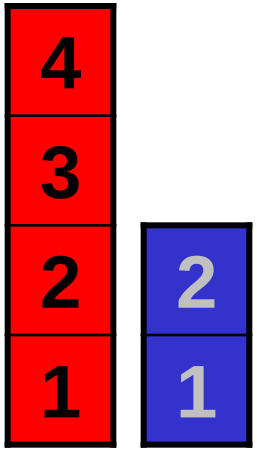
電力消費量= 静電容量 x 電圧^2 x 動作周波数

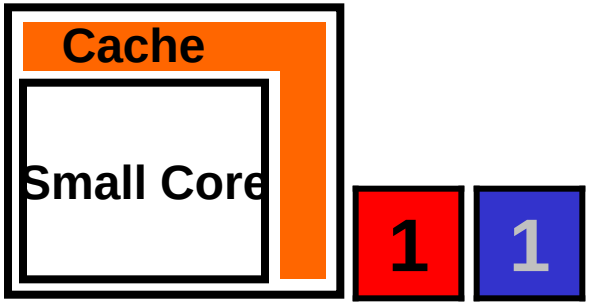
電力消費量 ~ 電圧^3

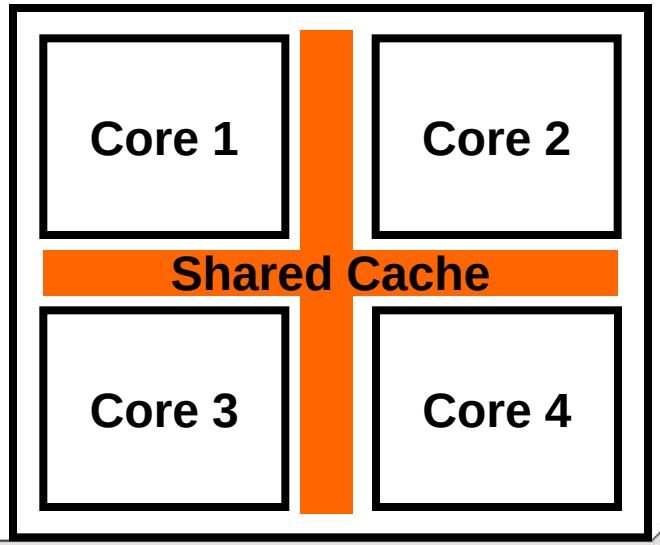
マルチコア：‘性能/消費電力’を改善

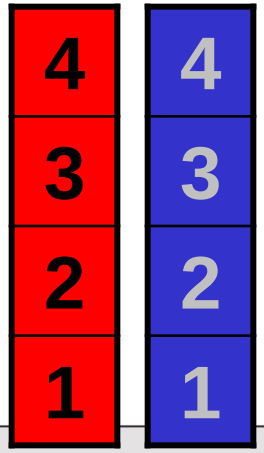


消費電力 / 性能





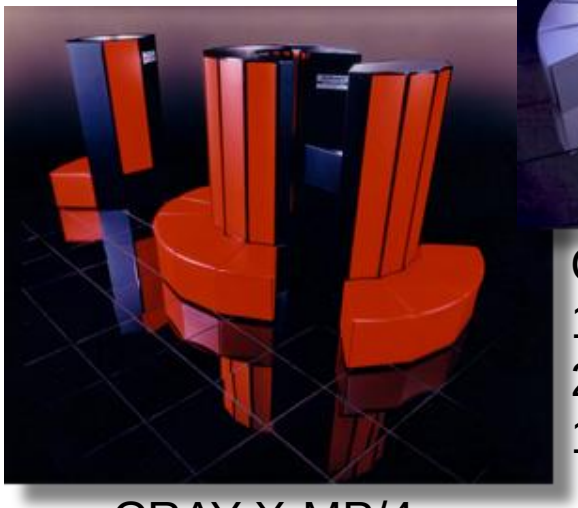




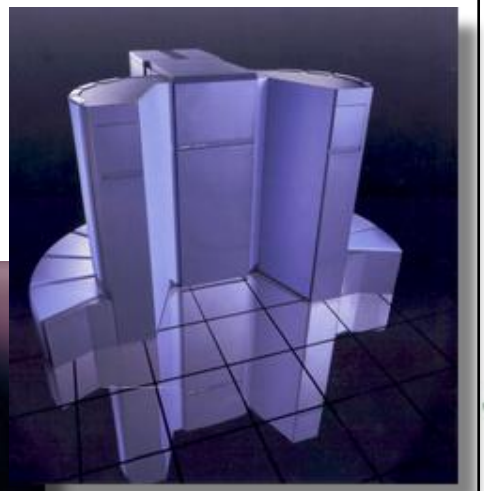
Power ~ コアサイズ
PERFORMANCE ~ √コアサイズ

HPCシステムのスケーリング

ベクトル処理
共有メモリ並列処理
……



CRAY X-MP/4
1982
800MFLOPS Peak
100MHz



CRAY Y-MP/8
1988
2664MFLOPS Peak
166MHz

6年間

Intel Xeon 5100
2006
24000MFLOPS Peak
3000MHz

Intel Xeon 5300
2006
42560MFLOPS
2666MHz

6ヶ月

60倍

HPCシステムの課題

- 代表的なHPCシステムのプラットフォームアーキテクチャ
  - クラスタシステム （1－2pノード）
  - SMPシステム （＞2pノード）
  - SMPシステムをベースとしたクラスタシステム
- HPCシステムの利用の現状
  - HPCシステムとしては、クラスタシステムが一般化している
  - SMPシステムの利点はOEM及びユーザも理解しているが、また、SMPシステムの開発、販売、導入には問題がある

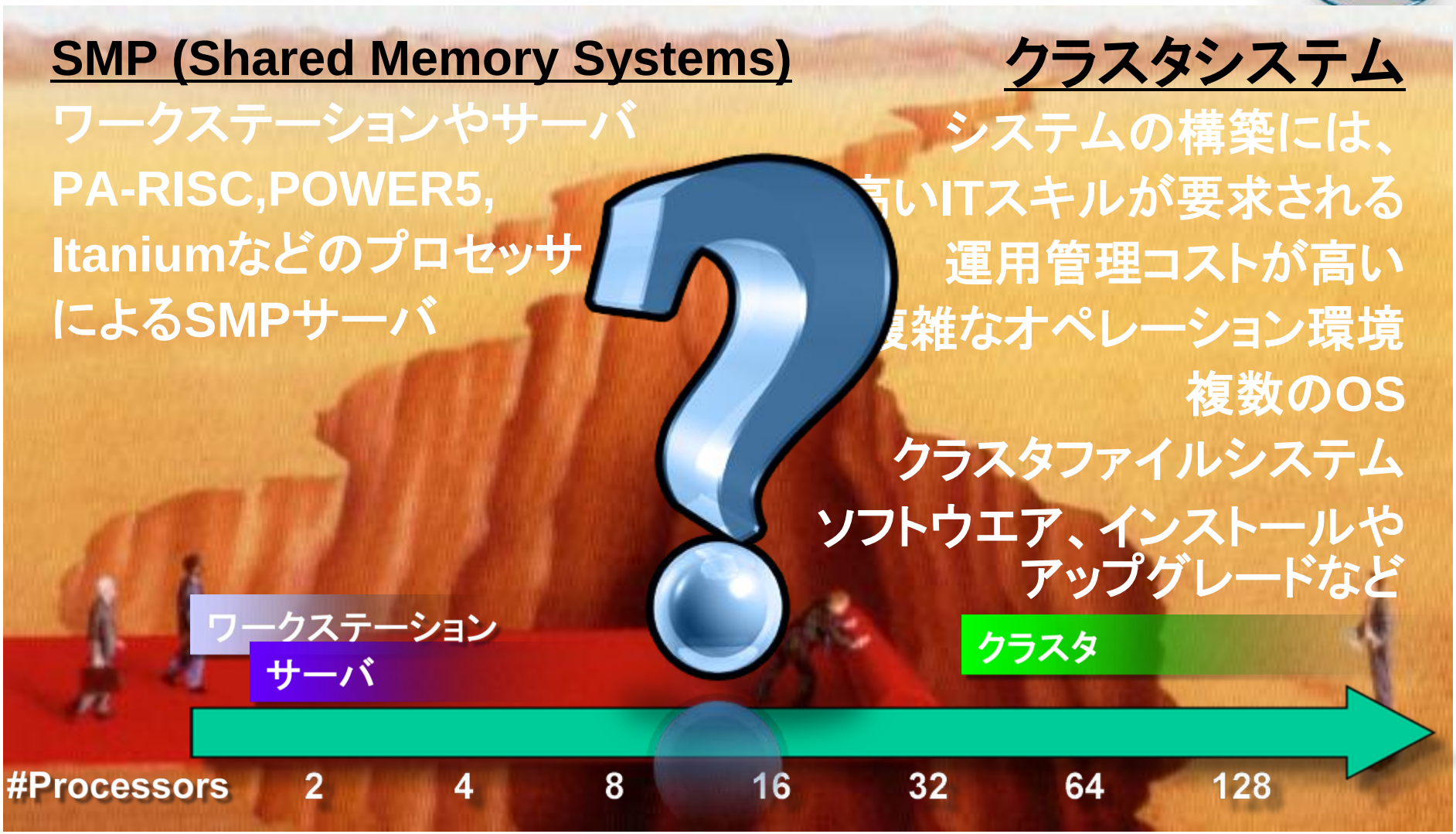
HPCシステムの問題

	OEMでの問題	エンドユーザの問題
クラスタ	• 付加価値 • ビジネスでの低マージン&価格競争	• 運用コストなどを含むTCOは劇的に低下しない • その運用管理には、かなりの経験や知識が必要 • システムの利用率及びアプリケーションの実効性能の維持
SMPシステム	• 開発コスト（ハードウエアとソフトウエア） • SMPシステム（専用システム）とクラスタシステム（一般商用システム）の互換性の問題	• 導入コスト • スケーラビリティ

クラスタシステムの利点

- ハードウエアコストの劇的な低下
- 非常に高いピーク性能のシステムの導入が可能
- 増設が容易で、必要に応じて、システムの規模の拡大が容易
- 標準コンポーネントの技術革新と性能向上
  - プロセッサの性能向上（‘マルチコア’による省電力での性能向上）
  - 高性能なスケーラブルファイルシステム（オープンソース）
  - 高速な商用インターコネクトスイッチ

HPCシステムのギャップ



HPCシステムへの要求要件

- HPCシステムの増強のニーズは高い
  - より大規模な解析
  - より多くのシュミレーション
  - より短い時間でのシュミレーションの完了
- 同時にシステムに対するコスト・パフォーマンスの要求も厳しい
  - ベンダー間での競合
  - アプリケーションのスケーラビリティ
  - より大規模なシステムの導入の希望
- 実質的には、HPCシステムとしては、「コスト・パフォーマンス」に対する要求が強い

「Fast」「Good」「Cheap」のパズル

Fast
+ Cheap
Inferior

高い性能を廉価なシステムで構築することも可能です。ただ、そのようなシステムの場合、システムの構築や利用は、必ずしも容易ではありません。

付加価値の高い、性能の高いシステムは一般には、高価です。その付加価値がユーザにとって、メリットが無ければ、コスト・パフォーマンスの悪いシステムになるだけです。

Good
+ Fast
Expensive

Good
+ Cheap
Slow

比較的小規模なシステムであれば、廉価で使い勝手の良いものを探すことは可能です。しかし、そのようなシステムでは、拡張性やより大規模なシステム構築が出来ません。

システム選択の課題

- 構築･運用管理コストの削減
- より生産性の高いシステムの構築
  - 複数の技術を効果的に組み合わせることにより解決を図る
  - 提供される機能とその価値の評価
- 生産性の定義は非常に難しい
  - ストレージも含めたトータルな解析システムの提案
  - 運用･管理の容易さ

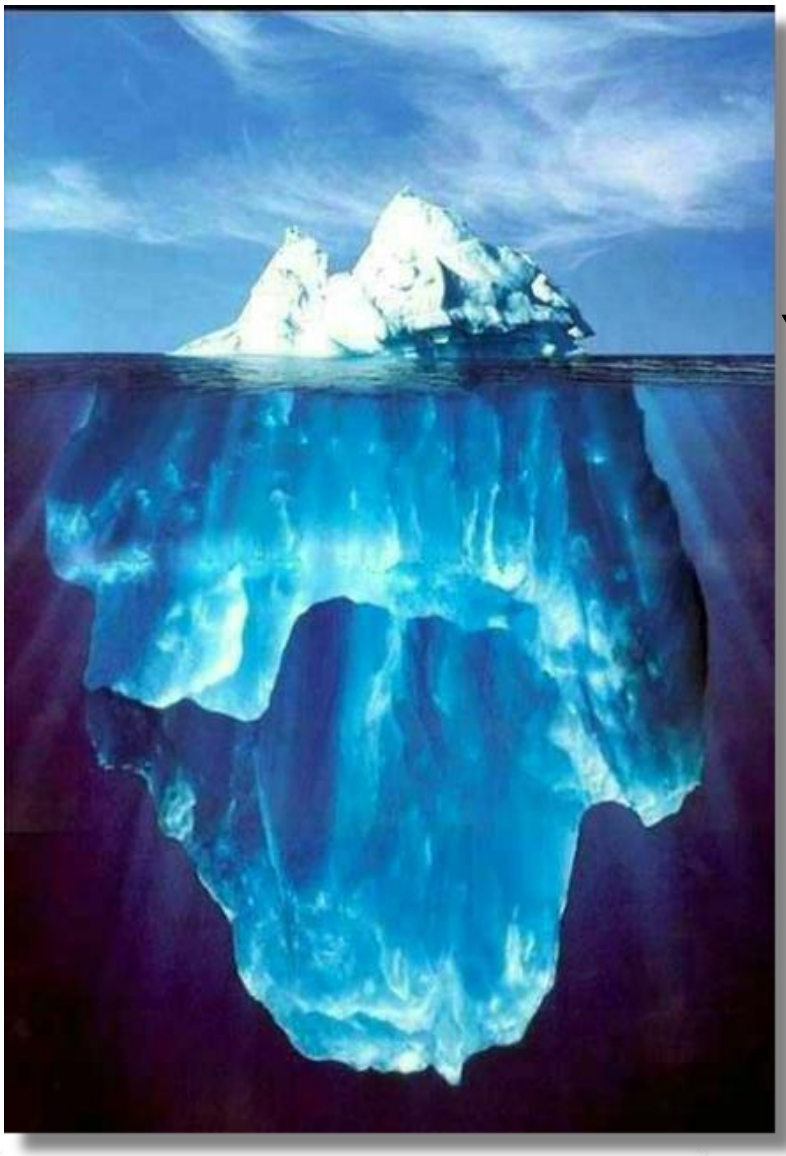
ITマネージメントの課題

- プラットフォームの内部からの保護:
  - ウイルスやワームなど悪意あるソフトウェアからの保護
- 資産管理:
  - 多くの IT 部門では、特定できない資産が問題
- オンラインおよびリモート管理・診断機能:
  - アップグレード、診断、復旧のための作業の効率化
- アプリケーション統合の困難さ:
  - アプリケーションの高度化と複雑化によって、複数のアプリケーションを組み合わせるての動作に問題
- 動的なリソース割り当て:
  - 組織内で未使用のCPUやメモリの活用

TCO : Total Cost of Ownershipの評価

- ハードウェアだけでなく、全てのコストを考慮したシステムTCOでの評価
- 運用コスト
  - フロアスペース/電力/システム管理
- インストレーションコスト
  - ノード数に大きく依存
- 購入コスト
  - プロセッサ/インターコネクト/メモリ/ソフトウェアコスト

TCO : Total Cost of Ownership

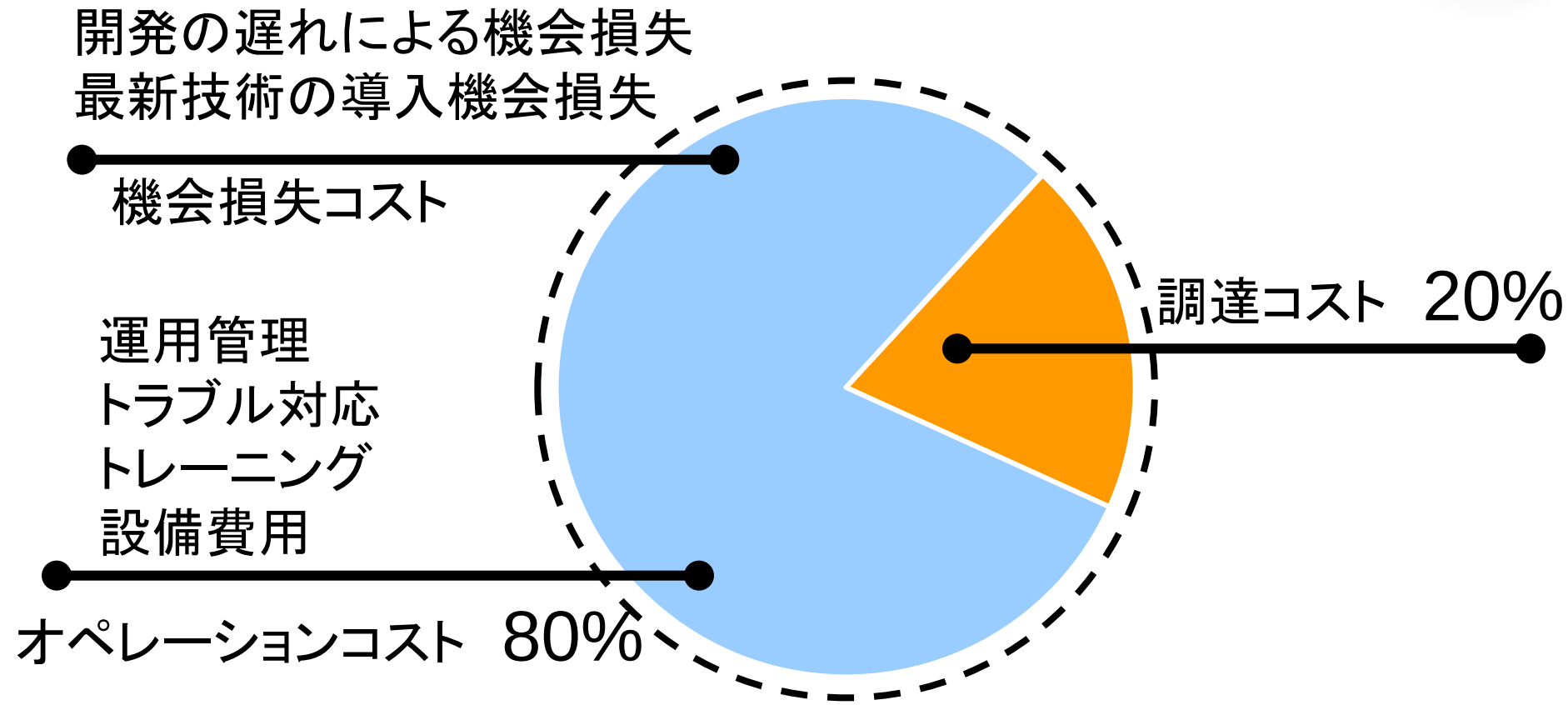


ハードウエアコストは
氷山の一角

ハードウエア導入コスト
ソフトウエア導入コスト

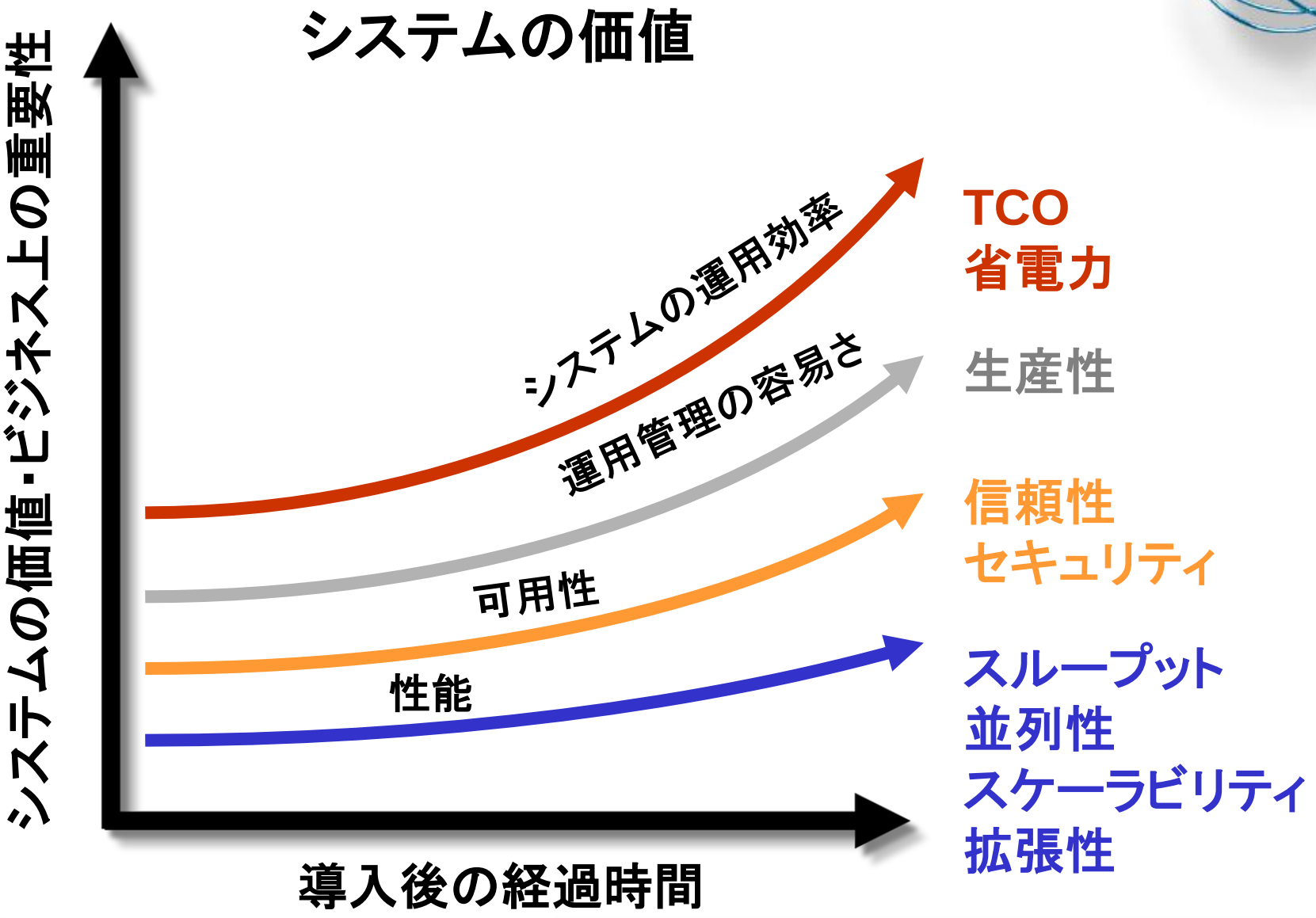
システムサポート
システム運用管理コスト
保守サービス
データマネージメント
アプリケーション開発
アプリケーションライセンス
互換性
・・・・・・・

TCO : Total Cost of Ownership

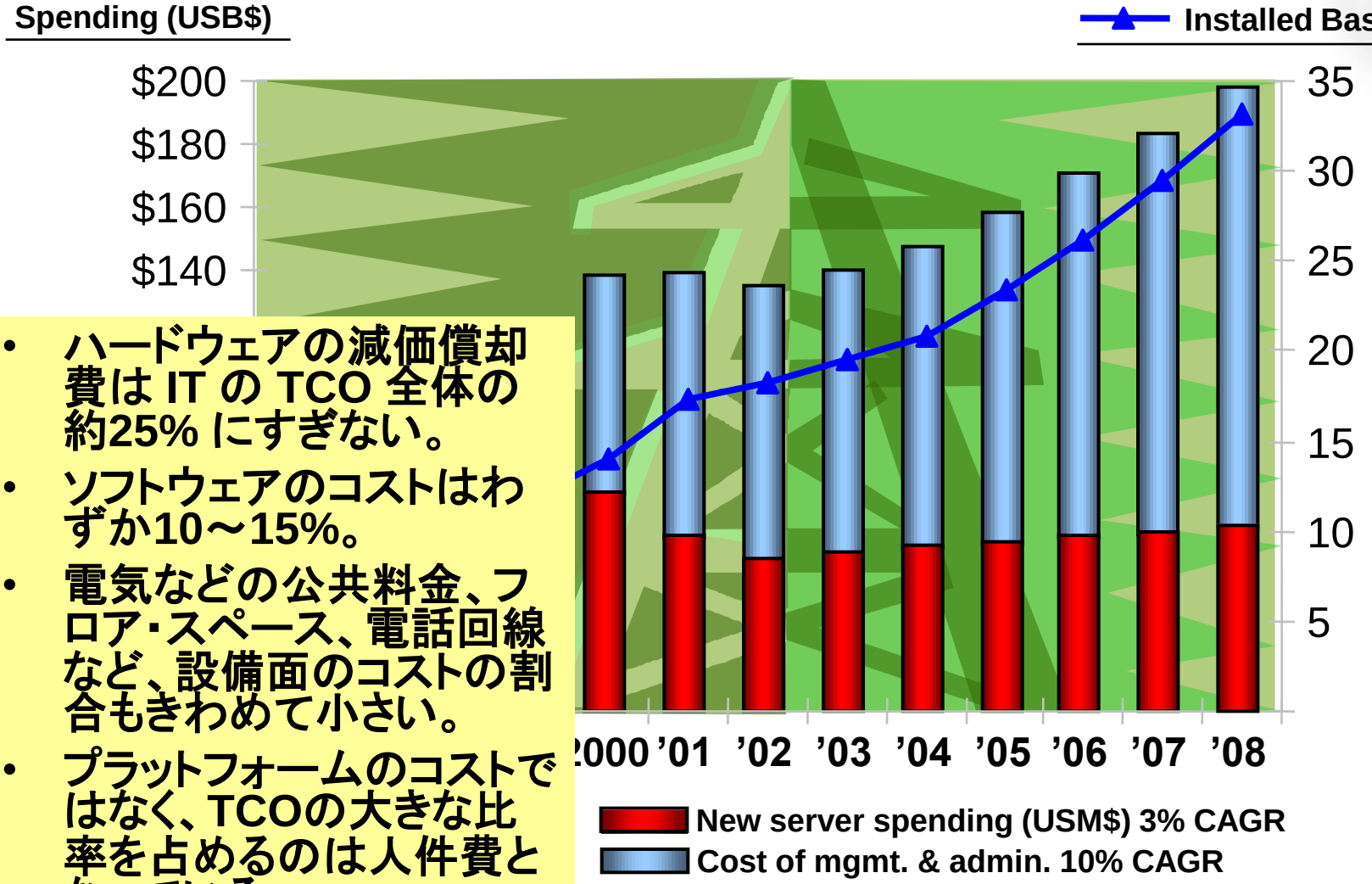


Source: Gartner Group 2005

TCO : Total Cost of Ownership

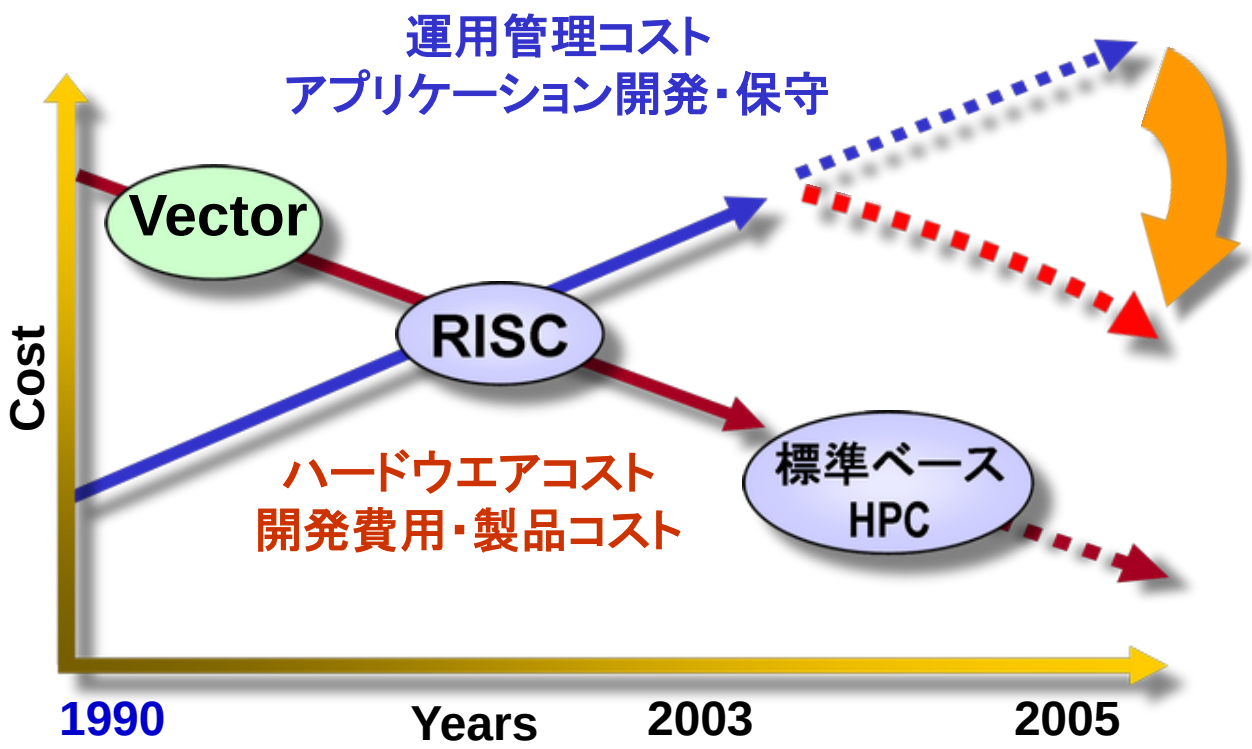


マーケットトレンド



- ハードウェアの減価償却費は IT の TCO 全体の約25% にすぎない。
- ソフトウェアのコストはわずか10～15%。
- 電気などの公共料金、フロア・スペース、電話回線など、設備面のコストの割合もきわめて小さい。
- プラットフォームのコストではなく、TCOの大きな比率を占めるのは人件費となっている。

HPCシステムでのTCO



生産性の高いプログラミング
OpenMP
クラスタOpenMP
統合クラスタ向けOS
Windows CCS
新しいシステムコンセプト
パーソナルクラスタ
スケーラブルx86システム

TCO低減の実現のための
ソリューション

HPCシステムの新たな可能性
パーソナルクラスタの考察

パーソナルクラスタの背景

- HPCマーケット
- マイクロプロセッサ
- HPCシステムの課題
- TCOの重要性

HPCシステムの二極分化

- ペタスケール コンピューティング
- コモディティ コンピューティング

パーソナルクラスタシステム

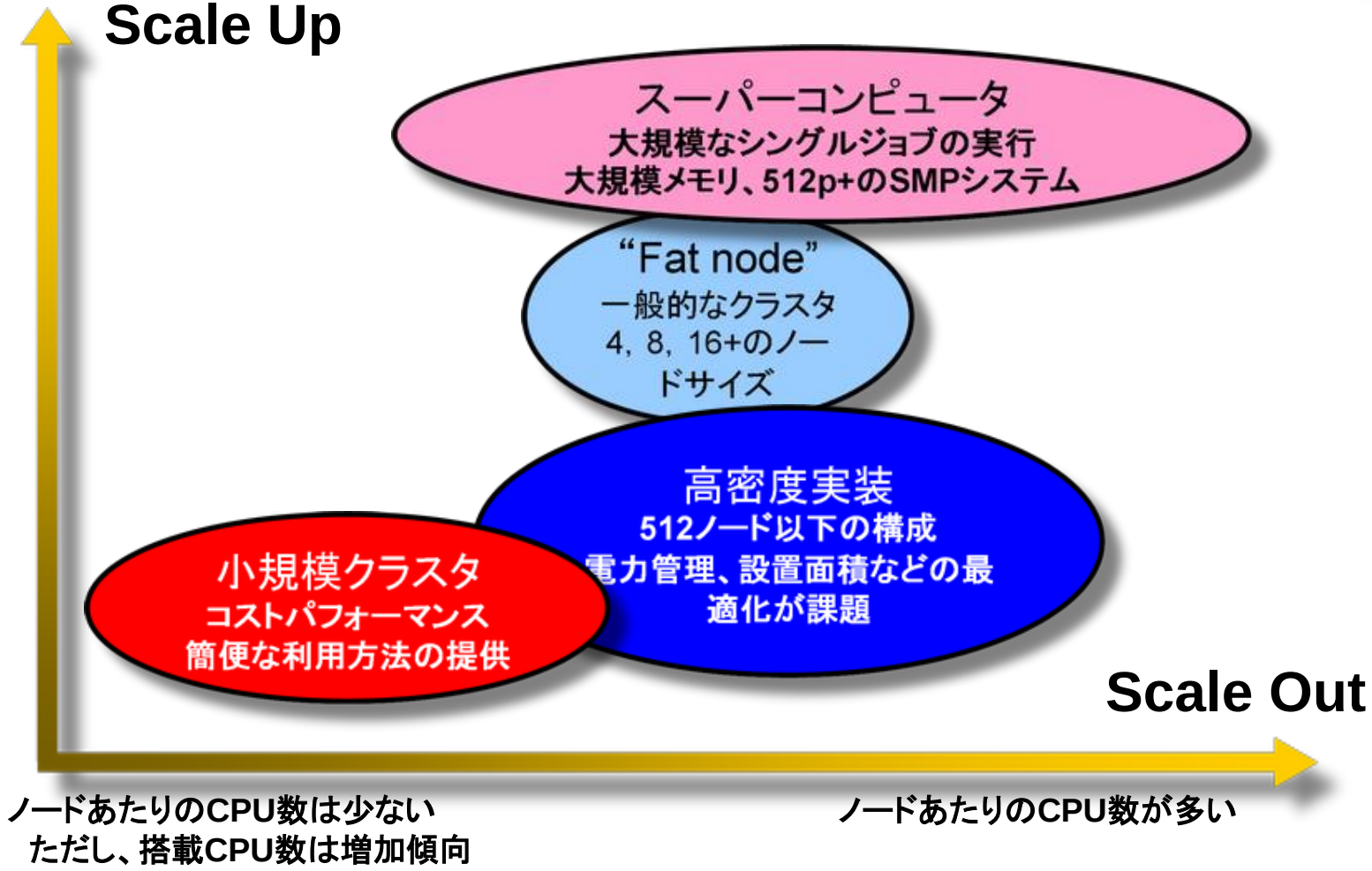
- システムの特徴
- 並列プログラミング

まとめ

HPCの二極分化


Going UP
‘Peta-Scale’
コンピューティング
• 複雑なシステム構成
• 新しいプログラミング APIの提案
• 独自のアプリケーション開発
• 商用HW/SW
• オープンソース
• パーソナルクラスタ
• 商用アプリケーション
• マルチスレッド
‘Commodity’
コンピューティング
Going DOWN

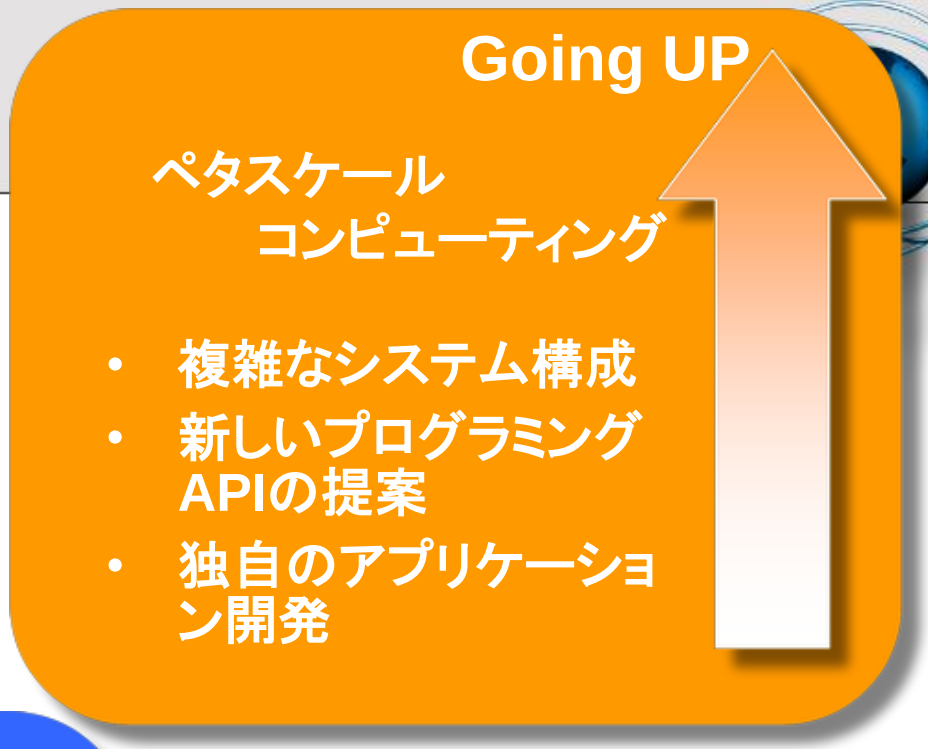
HPCシステム


Scale Up
スーパーコンピュータ
大規模なシングルジョブの実行
大規模メモリ、512p+のSMPシステム
“Fat node”
一般的なクラスタ
4，8，16+のノードサイズ
高密度実装
512ノード以下の構成
電力管理、設置面積などの最適化が課題
小規模クラスタ
コストパフォーマンス
簡便な利用方法の提供
Scale Out
ノードあたりのCPU数は少ない
ただし、搭載CPU数は増加傾向
ノードあたりのCPU数が多い

HPCの二極分化

HPCシステムの課題

- 基盤技術やコアのITテクノロジの共通化を図りながら、この極端に分極化したHPCシステムへの対応を図ることが必要となる。



HPCシステムの問題

- HPCで要求されるシステムの仕様が、大きく分極化し、この双方を一つのシステム・アーキテクチャで実現するのは、技術的に可能だとしても、経済性や生産性の点で問題がある。

HPCの定義

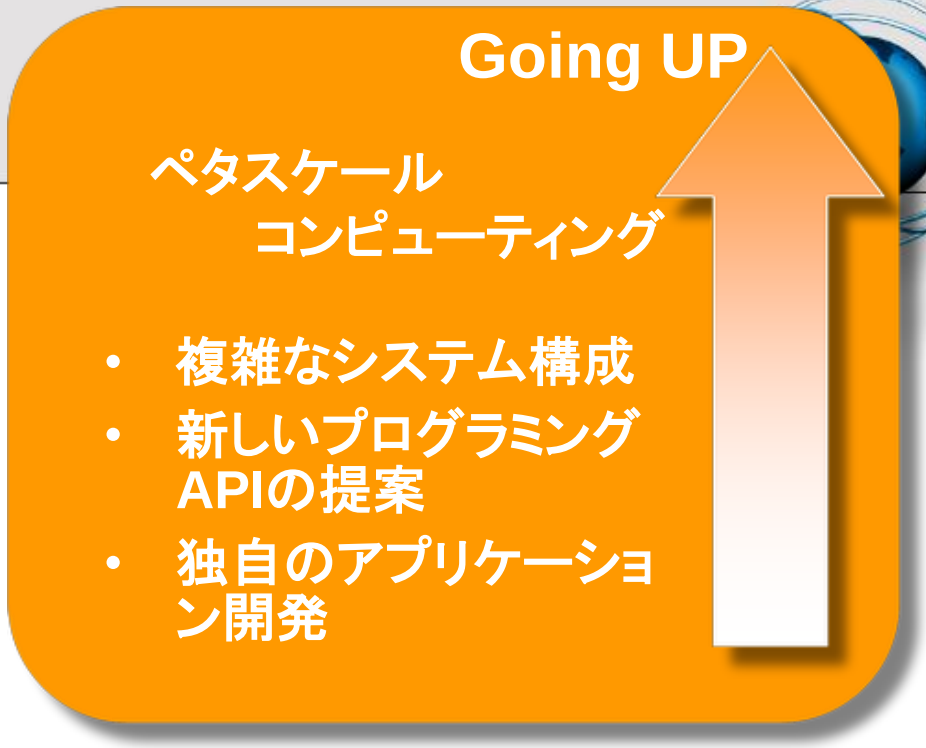
ペタFLOPS級‘
スーパーコンピュータ

- ピーク性能ではなく、アプリケーションの実効性能として、ペタFLOPSを超えるスーパーコンピュータ（ロレンス･リバモア国立研究所のHorst Simon博士の定義）

<u>ハイパフォーマンスコンピューティング</u>

- ハードウエア、ソフトウエア、開発環境など様々な技術を統合して、従来は解析出来なかった問題を十分な経済性をもって、解決すること

HPCの二極分化



- ペタスケールコンピューティング
  - 求められる基本技術と現在のHPCの主要マーケットでの要求はあまりにも差が大きい
  - コモディティのマイクロプロセッサではなく、独自のプロセッサを開発中
  - ‘複雑さ’の克服が重要

HPCの二極分化

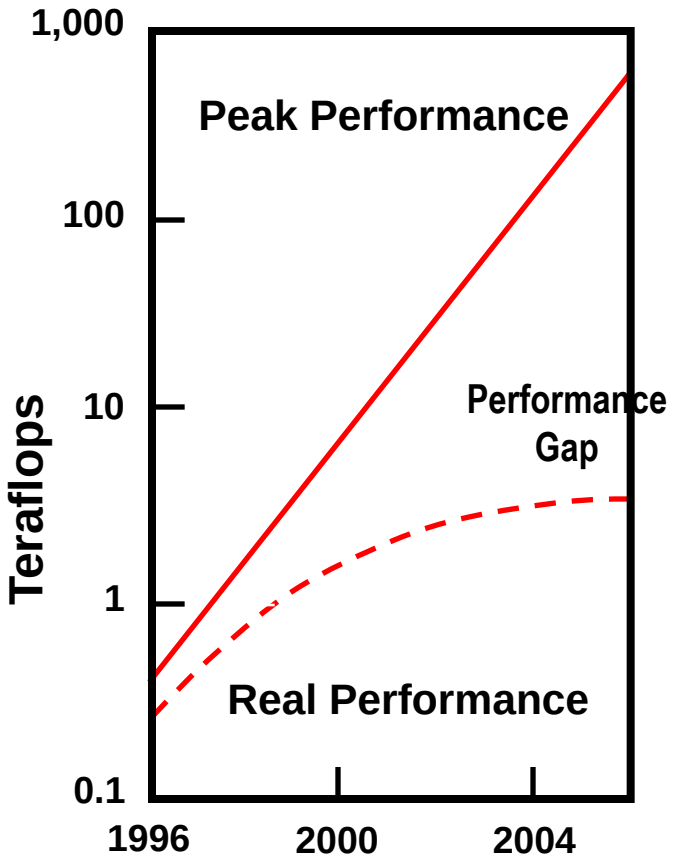
Going UP

ペタスケール
コンピューティング

- 複雑なシステム構成
- 新しいプログラミングAPIの提案
- 独自のアプリケーション開発

最も注目を集めた製品：SiCortex （www.sicortex.com ）
シングルチップに（6）MIPS64プロセッサとL2キャッシュ、メモリコントローラを内蔵し、非常に少ない消費電力での動作が可能
（IBMのBlueGeneに対抗？）

性能ギャップの拡大

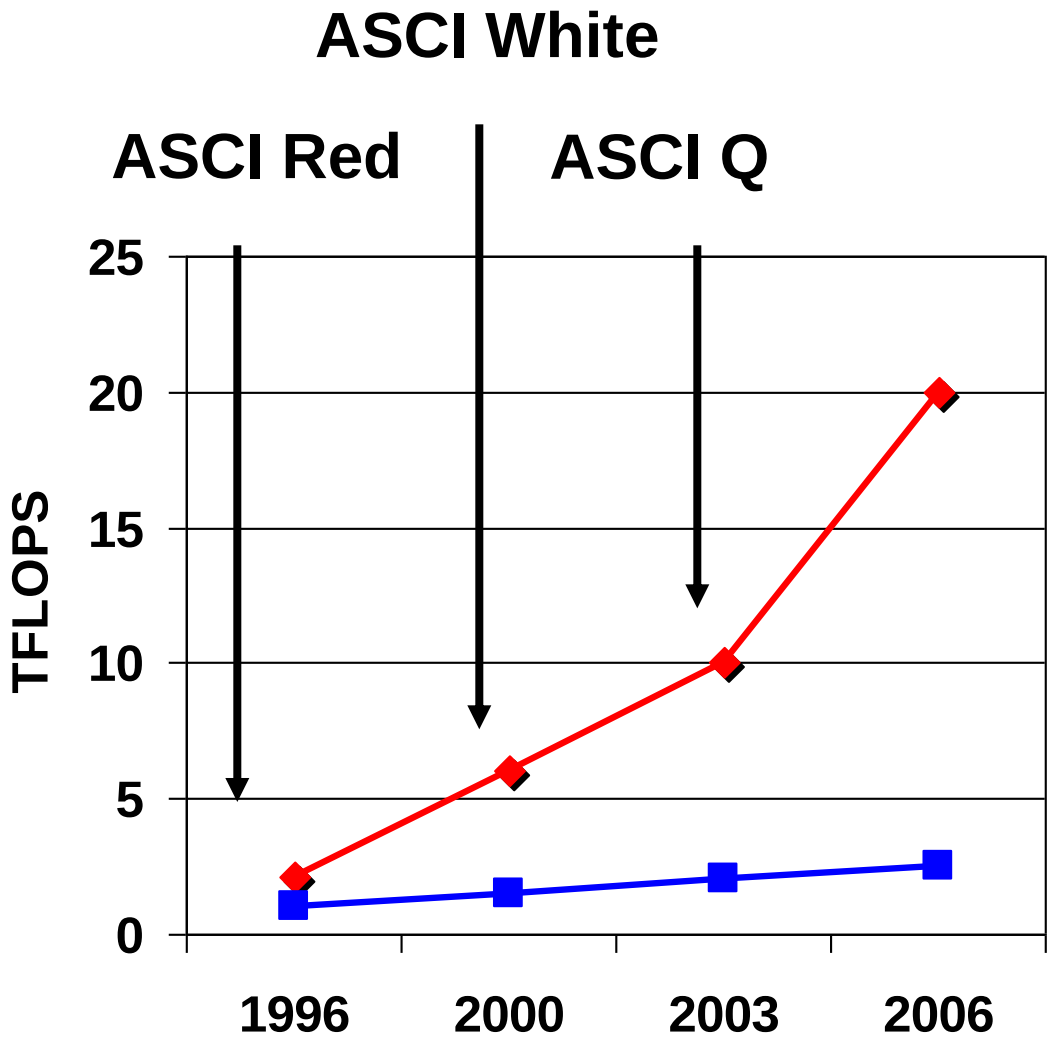


NERSC User Group Meeting June 24-25, 2004
Osni Marques and Tony Drummond
Lawrence Berkeley National Laboratory

- ピーク性能の大幅な向上
  - 1990年台は、性能の向上は、10^2のオーダーでしたが、2000年台になると10^3のオーダーで性能は向上しています。

  しかし…
  - 多くの科学技術計算用途のアプリケーションのピーク性能に対する実効性能の比率は、5-10%となっています。（1990年代のベクトル計算機は、40-50%の対ピーク性能を示していました。）
- 今、必要なのは
  - より高い実効性能を発揮することが可能な計算アルゴリズムと手法の開発とスケーラビリティの向上
  - プログラミングモデルなども含めて、スケーラブルな計算機環境の構築

性能ギャップの拡大



ソフトウエアとハードウエアの進歩は、大幅なピーク性能(Peak)の向上を可能としましたが、実効性能（SSP:Sustained System Performance)の向上には、大きな寄与がなされていないのが現状で、その格差は広がっています。

HECRTFレポートより
http://www.itrd.gov/hecrtf-outreach/

ペタスケールシステムの構築

Source: ORNL

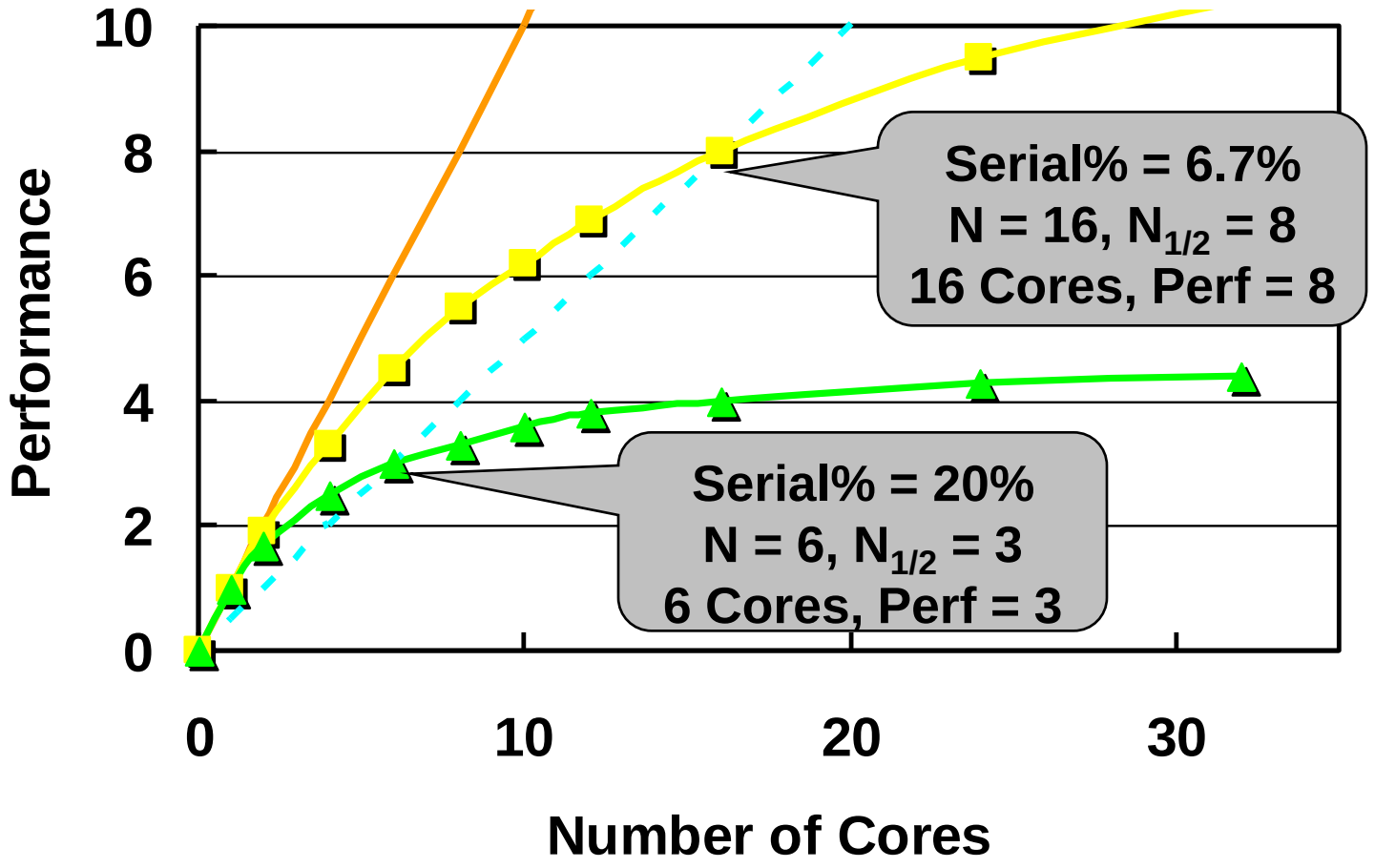
- ソフトウエア（アプリケーション、OS、プログラミングAPIなど）の課題の克服が課題
- システムの複雑さと生産性

例：

Linpack Benchmark

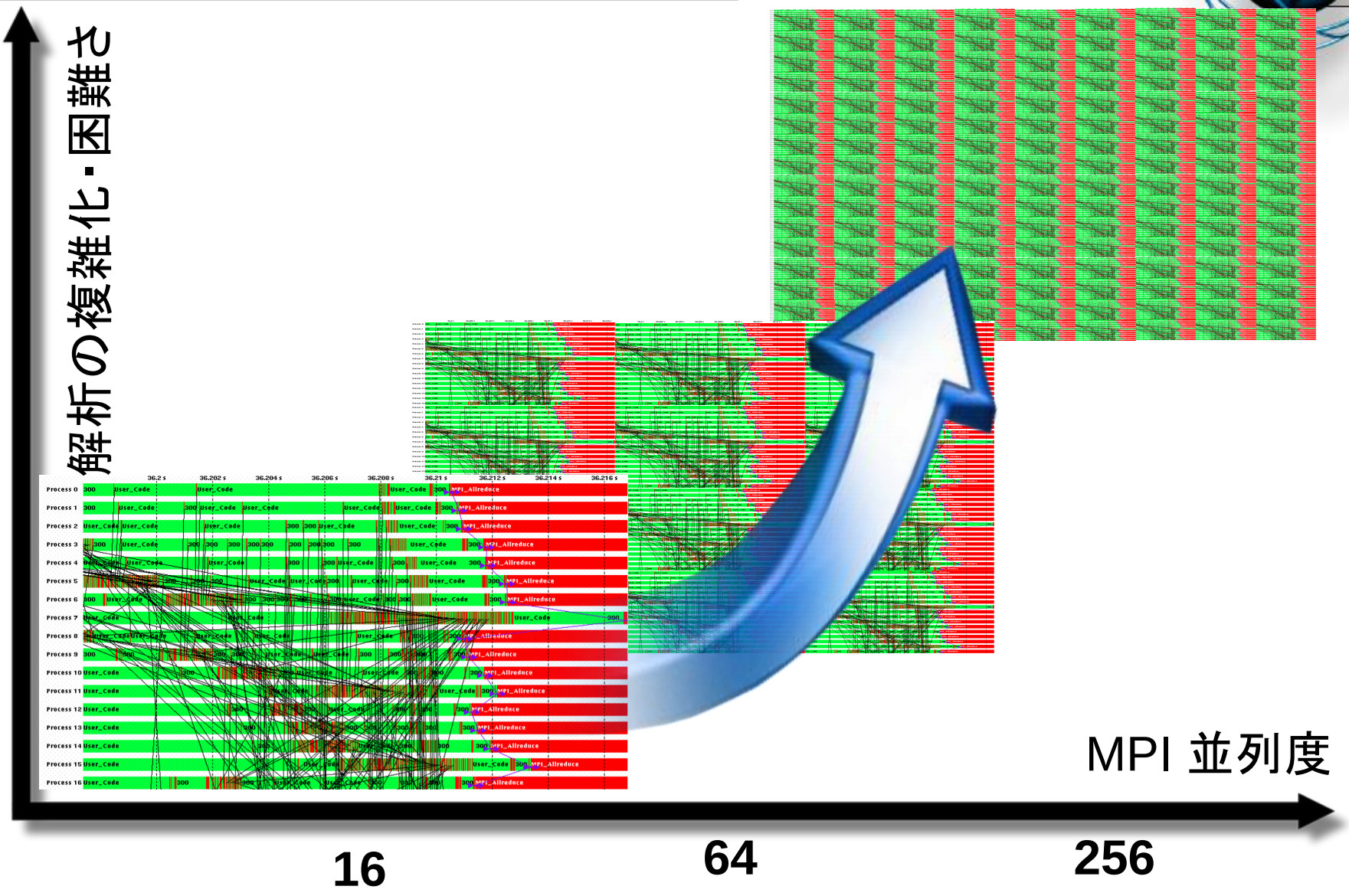
- オリジナルベンチマークプログラム　～100ライン
- HPL ベンチマークプログラム　～10,000ライン（x100より複雑？）

性能のスケーリング



並列処理でのスケーラビリティ←SWの重要性

ソフトウエア開発の困難さ



システムの信頼性

An Overview of High Performance Computing

Jack Dongarra
University of Tennessee and Oak Ridge National Laboratory

HPC Asia 2005

Reliability of Leading-Edge HPC Systems

System	CPUs	Reliability
LANL ASCI Q	8,192	MTBI: 6.5 hours. Leading outage sources: storage, CPU, memory.
LLNL ASCI White	8,192	MTBF: 5.0 hours ('01) and 40 hours ('03). Leading outage sources: storage, CPU, 3rd-party HW.
Pittsburgh Lemieux	3,016	MTBI: 9.7 hours.

MTBI: mean time between interrupts = wall clock hours / # downtime periods
MTBF: mean time between failures (measured)

次世代ハイエンドHPCインフラ

- コアとスレッド
  - より多くのスレッドを効率よく利用可能
  - マルチスレッド向け最適化
- 電力管理
  - 省電力
  - データセンター運用管理機能
- 仮想化
  - 柔軟性と優れた運用管理
  - 仮想的なシステムパーティション
- RAS
  - ハードウエアベースの自己監視/自己管理
  - ファームウエアベースのエラー履歴管理
- システム管理
  - より低いTCOを実現するための一般・標準化されたマネージメント機能

HPCの二極分化

- コモディティコンピューティング
  - ハードウエアは、コモディティなものを利用して、SWの改善、サポート、利用技術のサポート、パッケージ実装などが今後の主要マーケットでの成功の鍵となる

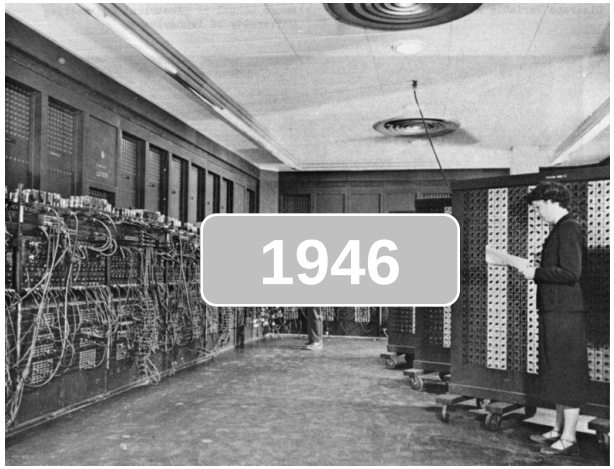
‘今日の’スーパーコンピュータ

“…現在のスーパーコンピュータ（の性能）は、将来のディスクトップで実現される….”

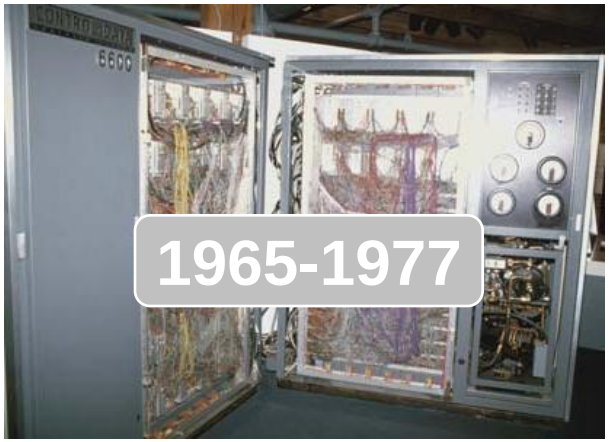
インテル社が主催する開発者向け会議「Intel Developer Forum (IDF) fall 2006」の最終日に米IntelのStephen Pawlowski氏（シニアフェロー，デジタル・エンタープライズ・グループCTO，ジェネラル・マネジャ）によるHPCに関する基調講演 より

Yesterday, Today and Tomorrow

ENIAC
20個の変数と300個の定数を記憶するメモリ



ASCI Red
最初のTFLOPSコンピュータシステム



CDC 6600
最初の商用スーパーコンピュータ

Cluster.....
デュアルコアマイクロプロセッサを搭載

PetaScale Platforms

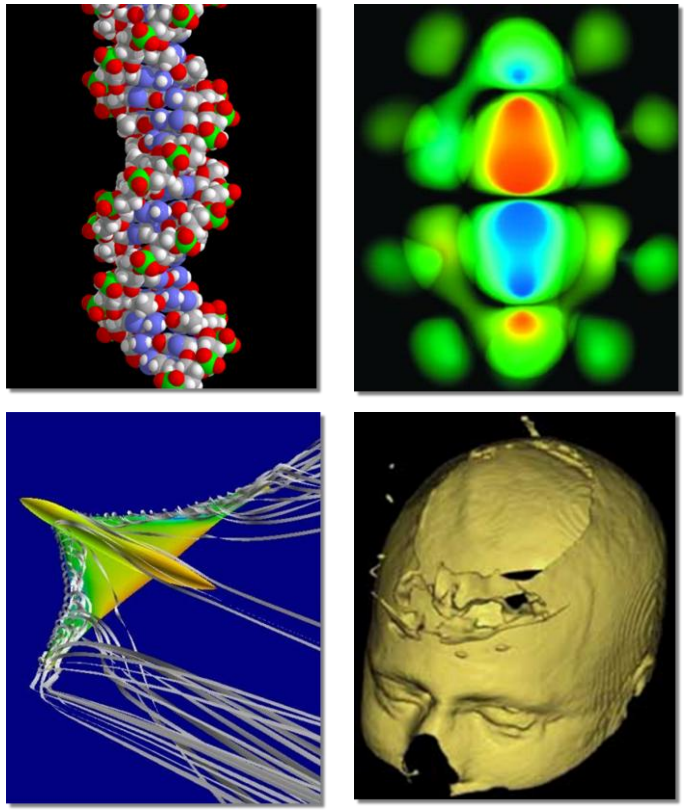
Personal Computing…

Yesterday, Today and Tomorrow

「…today‘s supercomputing problem is tomorrow’s desktop problem…」

「現在の大規模なスーパーコンピューターを必要とする非常に解析困難な問題も、将来はより強力なデスクトップ・システムによって、解析可能となるだろう・・・」

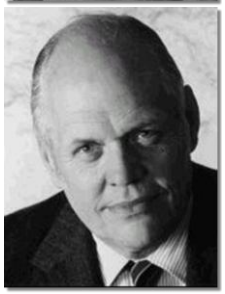
Dr. Walter Brooks, NASA



将来予測の難しさ

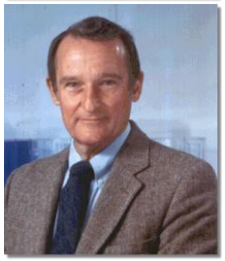
“I think there is a world market for maybe five computers.”

– Thomas Watson, chairman of IBM, 1943.



“There is no reason for any individual to have a computer in their home”

– Ken Olson, president and founder of digital equipment corporation, 1977.



“There are only about 100 potential customers worldwide for a Cray-1”

– Seymour Cray, 1977.

“640K [of memory] ought to be enough for anybody.”

– Bill Gates, chairman of Microsoft,1981.

将来予測の難しさ

「未来を予測する最良の方法は、それを創造してしまうことである」

"The best way to predict future is to invent it."

Dr. Alan Kay, President of Viewpoints Research Institute, Inc.,

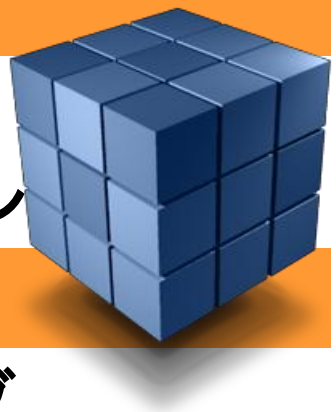
HPCシステムの新たな可能性
パーソナルクラスタの考察

パーソナルクラスタの背景

- HPCマーケット
- HPCシステムの課題
- マイクロプロセッサの方向性
- TCOの重要性

HPCシステムの二極分化

- ペタスケール コンピューティング
- コモディティ コンピューティン



パーソナルクラスタシステム

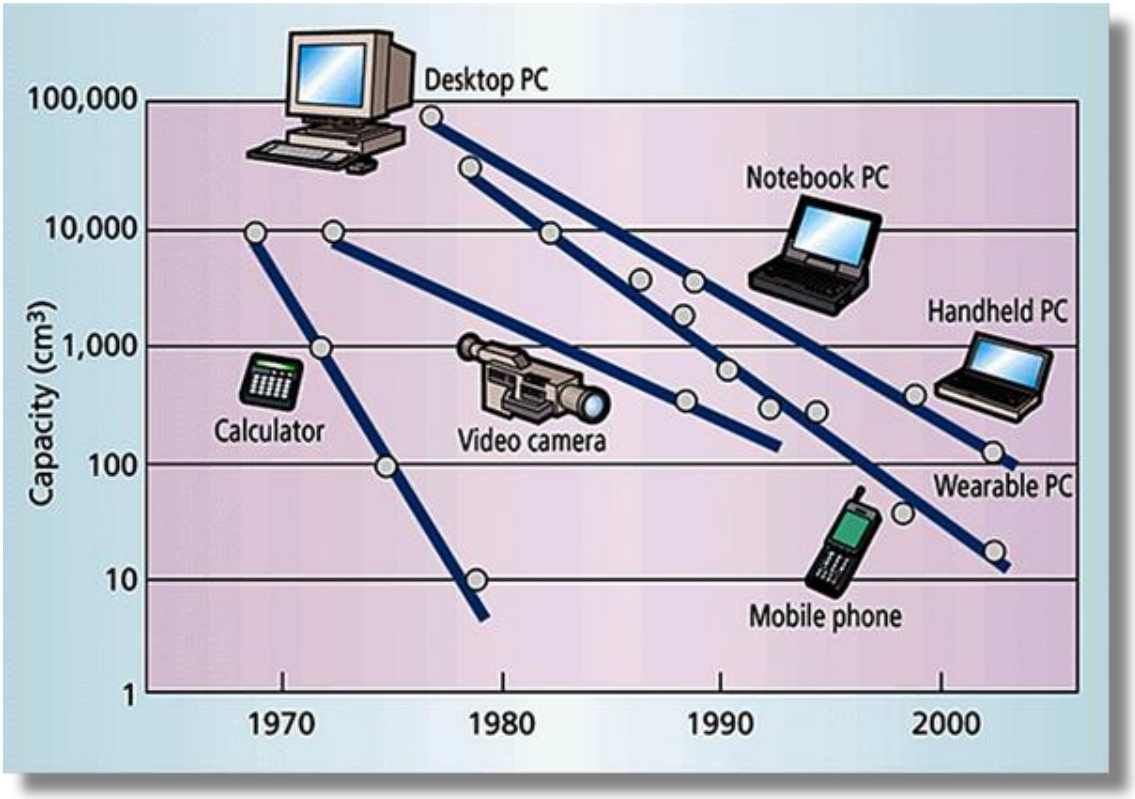
- システムの特徴
- 並列プログラミング

まとめ

パーソナルクラスタ

- 「パーソナルクラスタ」とは?
  - マイクロプロセッサのマルチコア化によって、HPCシステムの構築も変化してきます。また、プロセッサの省電力化が進み、より高密度な実装が可能となっています。このような状況で、HPCシステムもよりコンパクトで、より使いやすいものが求められています。
  - このような状況に最適なシステムとして、「パーソナルなHPCクラスタ」というコンセプトのクラスタシステムとして製品化されたものです。

多くの製品がよりコンパクト化



多くの電子機器がより消費電力が少なく、また、コンパクトな筐体に納まるようになってきています。これは、消費者の求めることでもあります・・・・

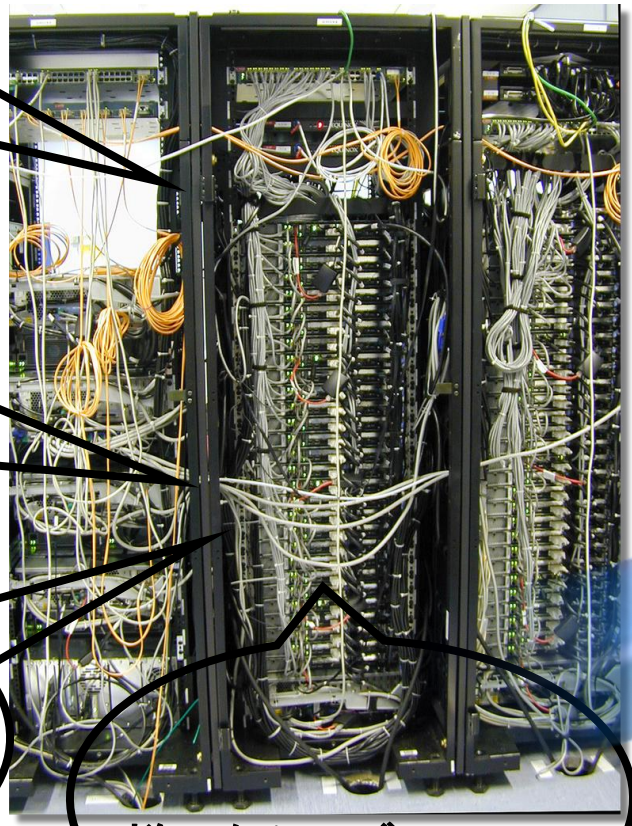
“The Cooler the Better: New Directions in the Nomadic Age” IEEE Computer Vol. 34, No. 4

クラスタ = パーソナルクラスタ

Network Switch

KVM

Terminal Server



様々なケーブル……

パーソナルクラスタ

利用用途

高性能HPCシステムを‘Turn-Key’システムとして提供

個人ユーザ向け及び部門サーバ

システム構成

8プロセッサ以上の構成

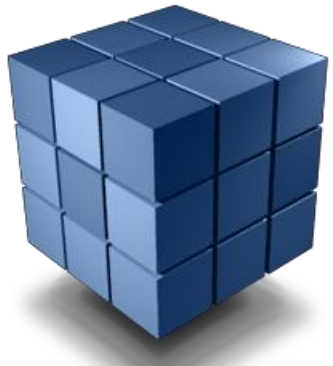
アプリケーションの動作検証保証済み

利用環境

ワークステーションのように利用可能なシステム

システム価格

‘スーパーコンピュータ’クラスの性能をビジネス向けコンピュータの価格で提供



Beowulfとパーソナルクラスタ

	Beowulf プロジェクト	パーソナルクラスタ
		
目的	遊休PCからHPCシステム（クラスタ）の構築	クラスタシステムがPC（パーソナルコンピュータ）として利用可能
システム	Hyglac-1996 (Caltech)	NEXXUS 4000 （2006）
プロセッサ	16 Pentium Pro 200 MHz	8 Intel Xeon 5160 3000MHz
インターフェイス	PCI	PCI-e
インターコネクト	100 base-T Fast Ethernet	Gigabit Ehternet 又はInfiniBand
メモリー容量	2GB	Max 128 GB
ピーク演算性能	1.25ギガFLOPS	192 ギガFLOPS

PC→クラスタ→PC

PC:パーソナルクラスタ

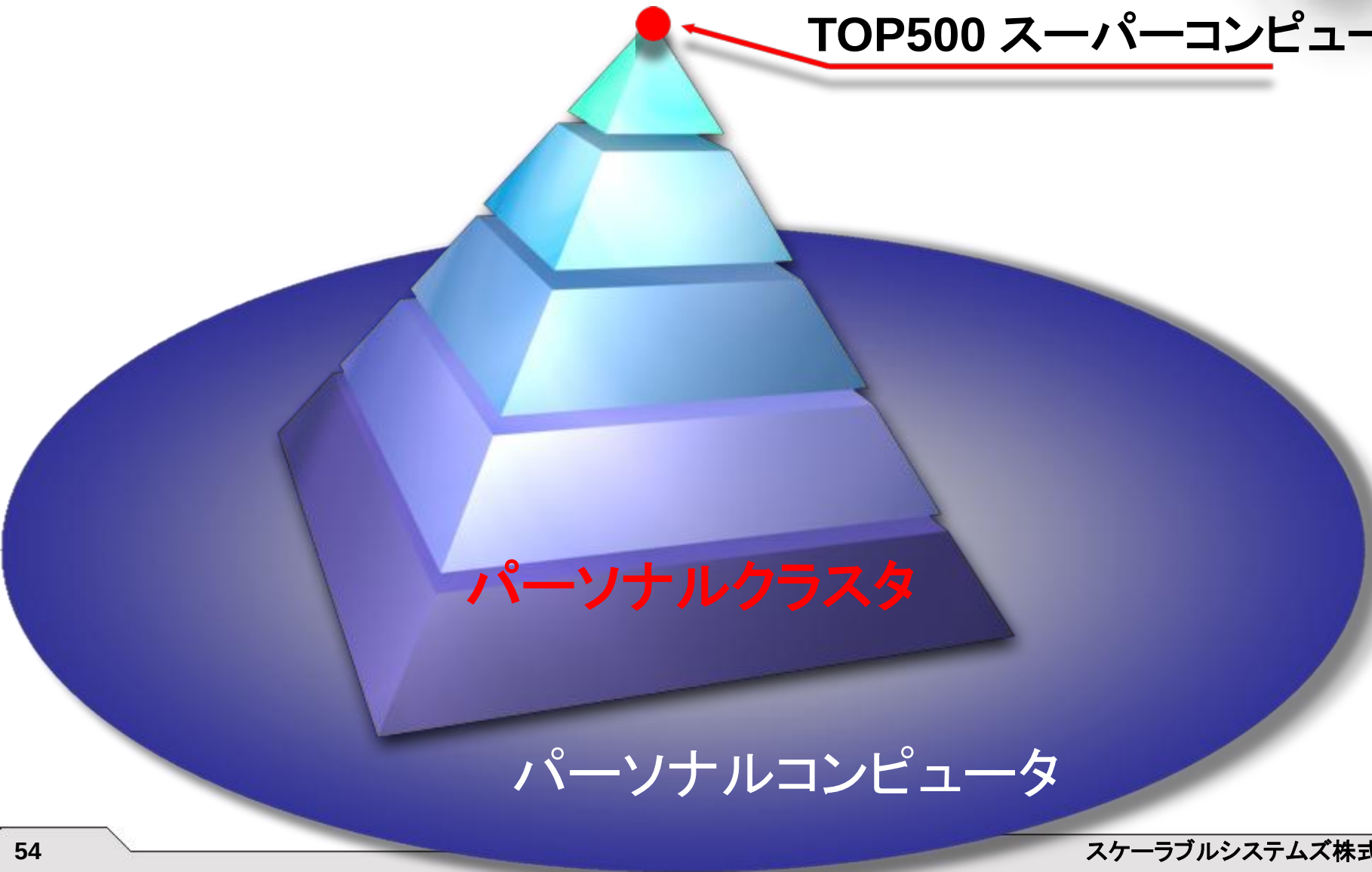
PC:パーソナルコンピュータ

Brightクラスタ

Dimクラスタ

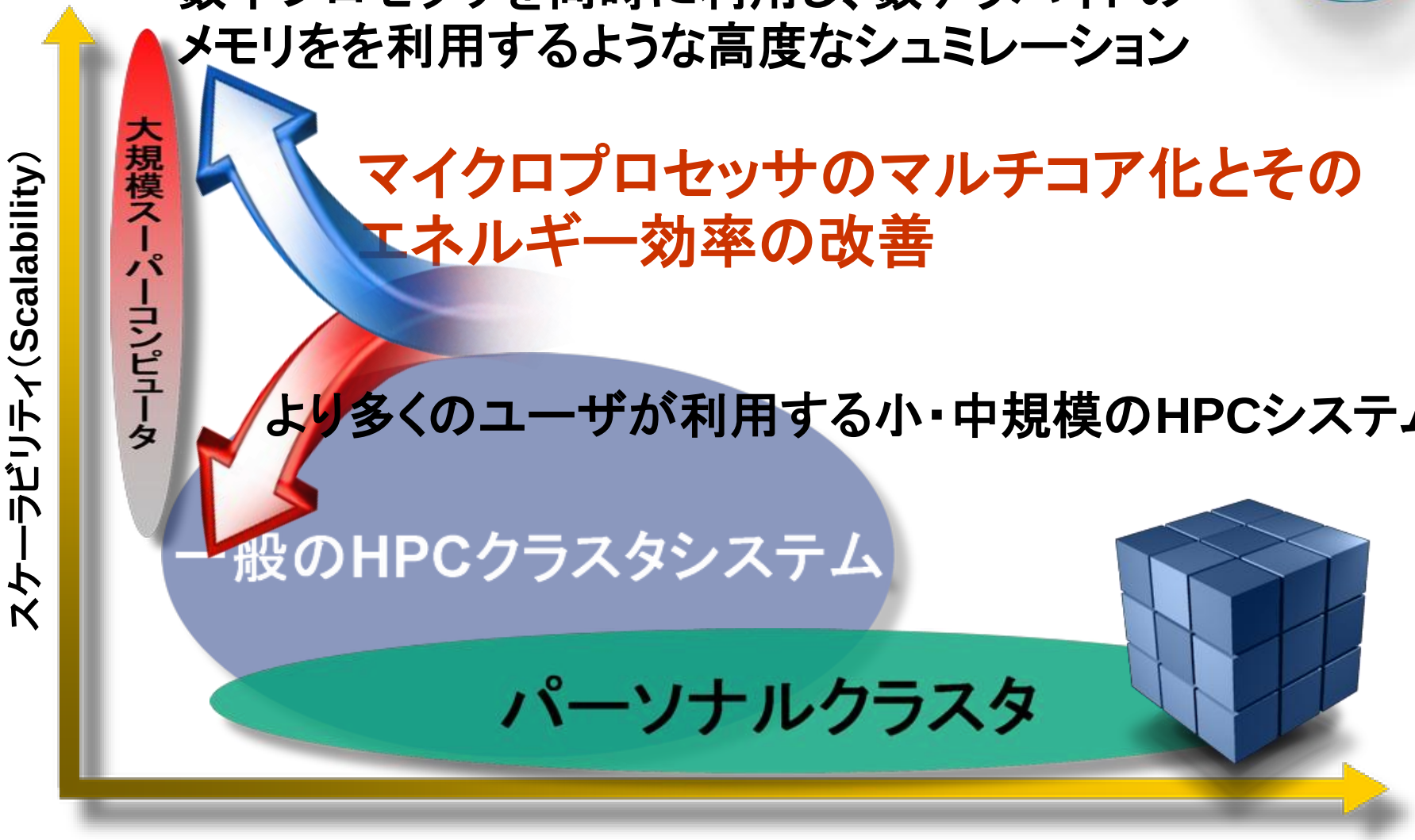
Beowulfプロジェクト

パーソナルクラスタの位置づけ



HPCシステムの利用分野とスケーラビリティ

数千プロセッサを同時に利用し、数テラバイトのメモリをを利用するような高度なシュミレーション



「Fast」「Good」「Cheap」の解

64ビットマルチコア
マイクロプロセッサ

最新チップセット

Built-in InfiniBand
インターコネクト

コンパクト&モジュール
化設計

標準ブレード

アップグレード可能

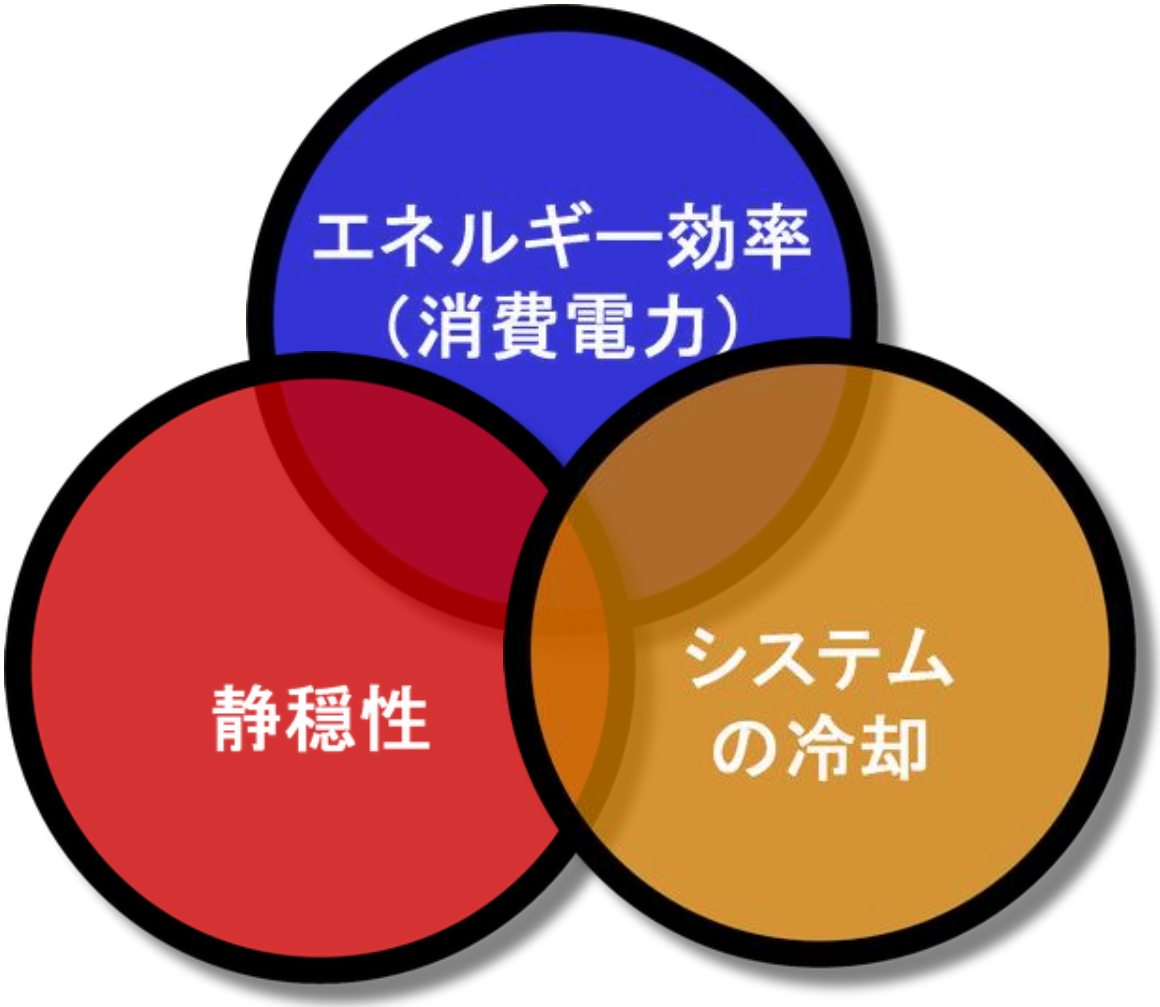
筐体、電源の最適配置

アプライアンスクラスタシステム

アプリケーション認証済み

Windows CCS & Linux

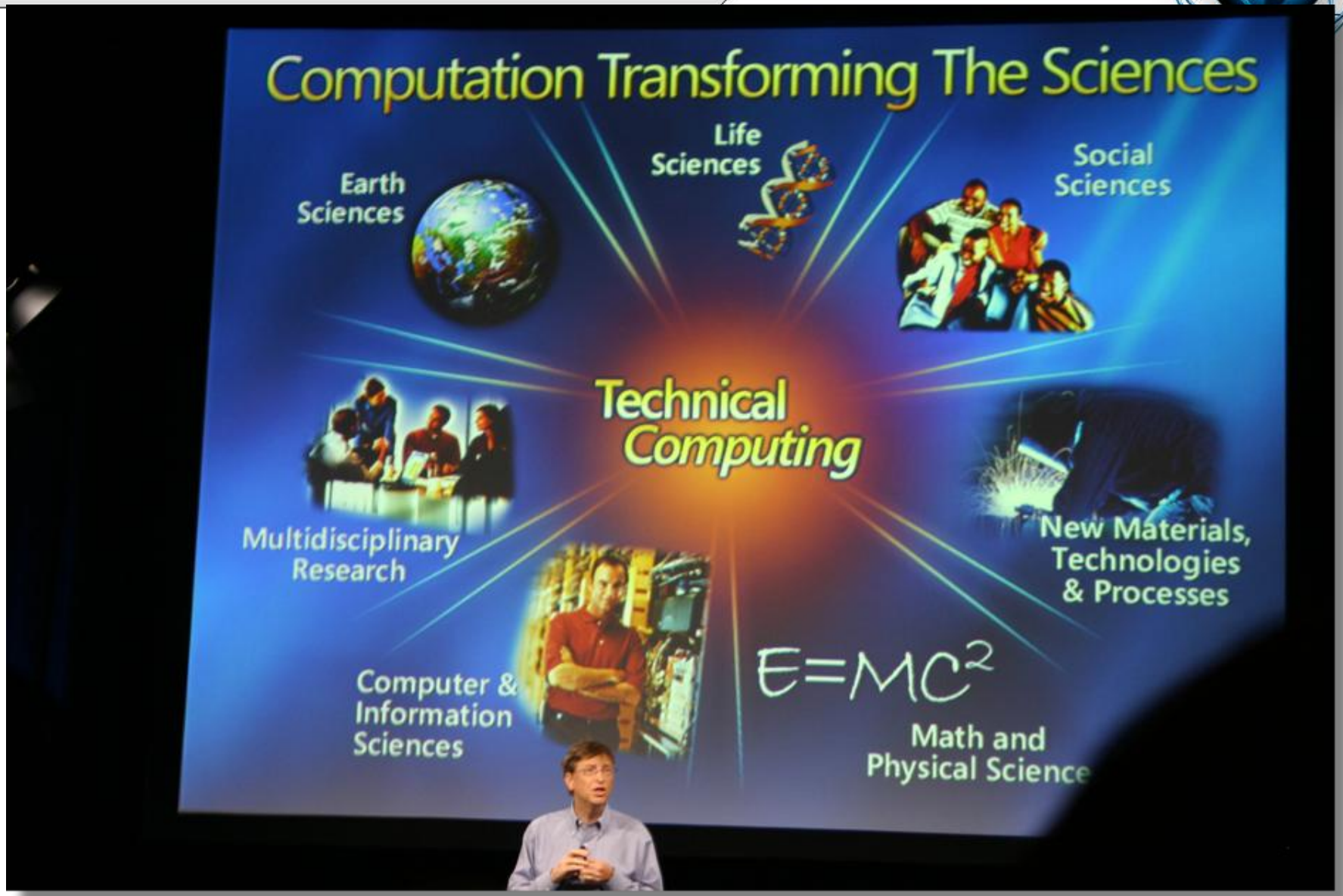
パーソナルクラスタへの要求



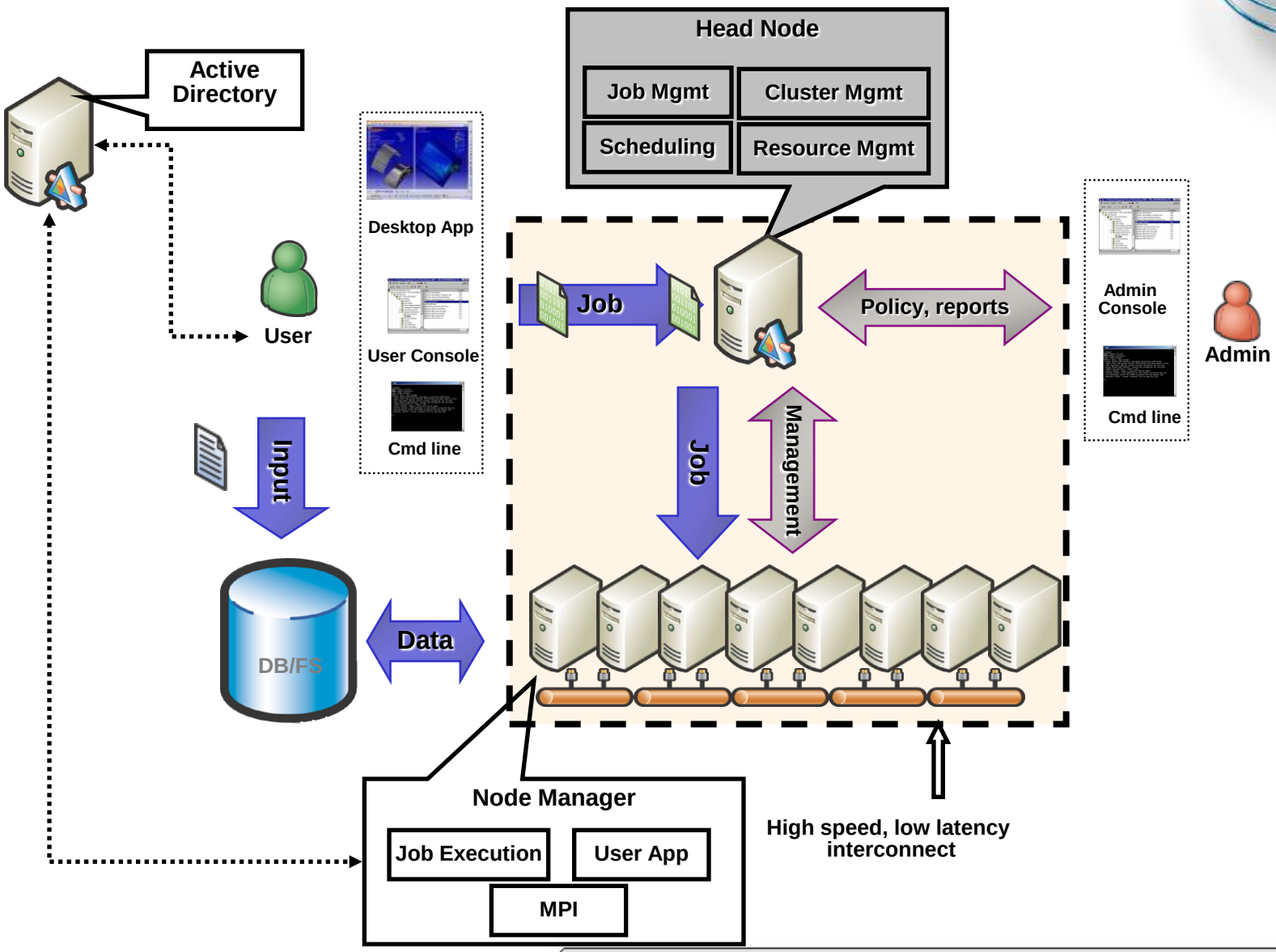
ソフトウエア環境

- OSサポート: 用途と顧客の要望に合わせて、広範囲な選択肢
  - Microsoft Windows Compute Cluster Server（Windows CCS)での動作保証
  - RedHat と SUSEもサポート
  - 仮想化支援
- システムマネージメント
  - インテルの新しいサーバマネージメント機能のサポート
  - リモートからの管理機能の強化
- 開発環境
  - 豊富な開発環境の選択肢

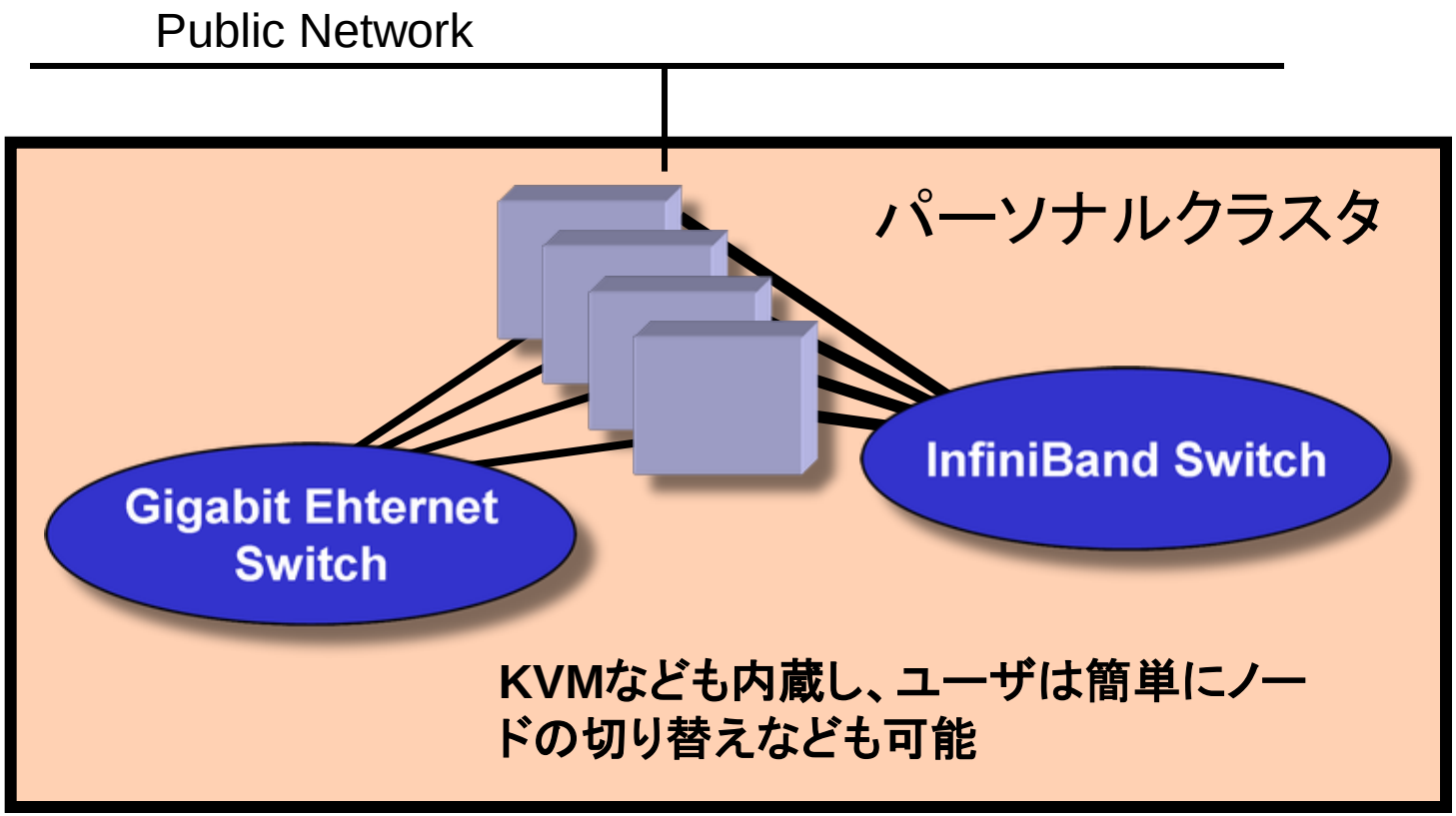
SC|05でのビル･ゲイツ氏の基調講演



Microsoft Compute Cluster Server



パーソナルクラスタ構成図



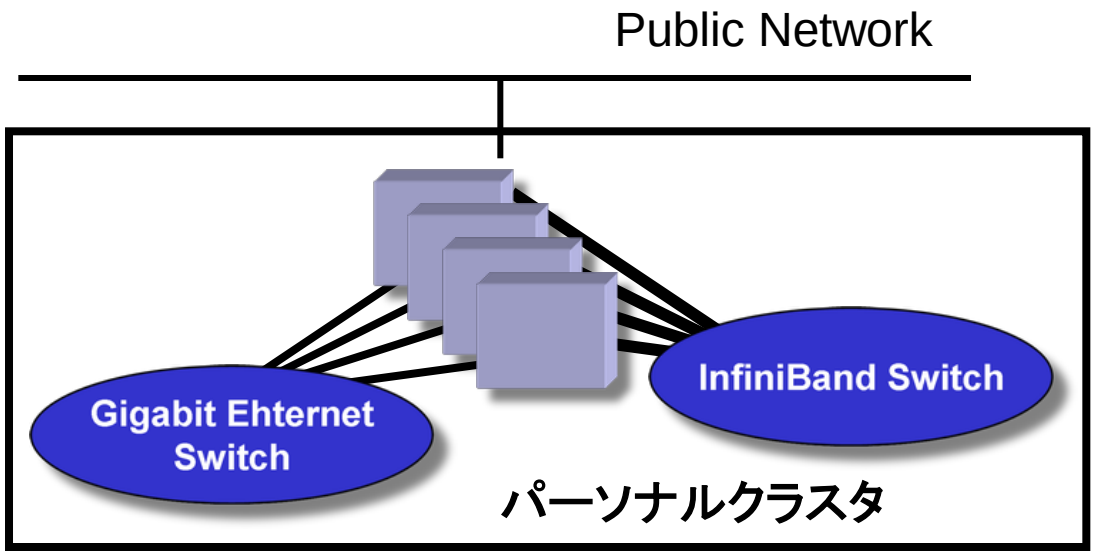
パーソナルクラスタのメリット・デメリット

- パーソナルクラスタは筐体単位でクローズ

（メリット）導入が容易で、管理も簡便

（デメリット）拡張性が限定される

利用できるプロセッサが限定される

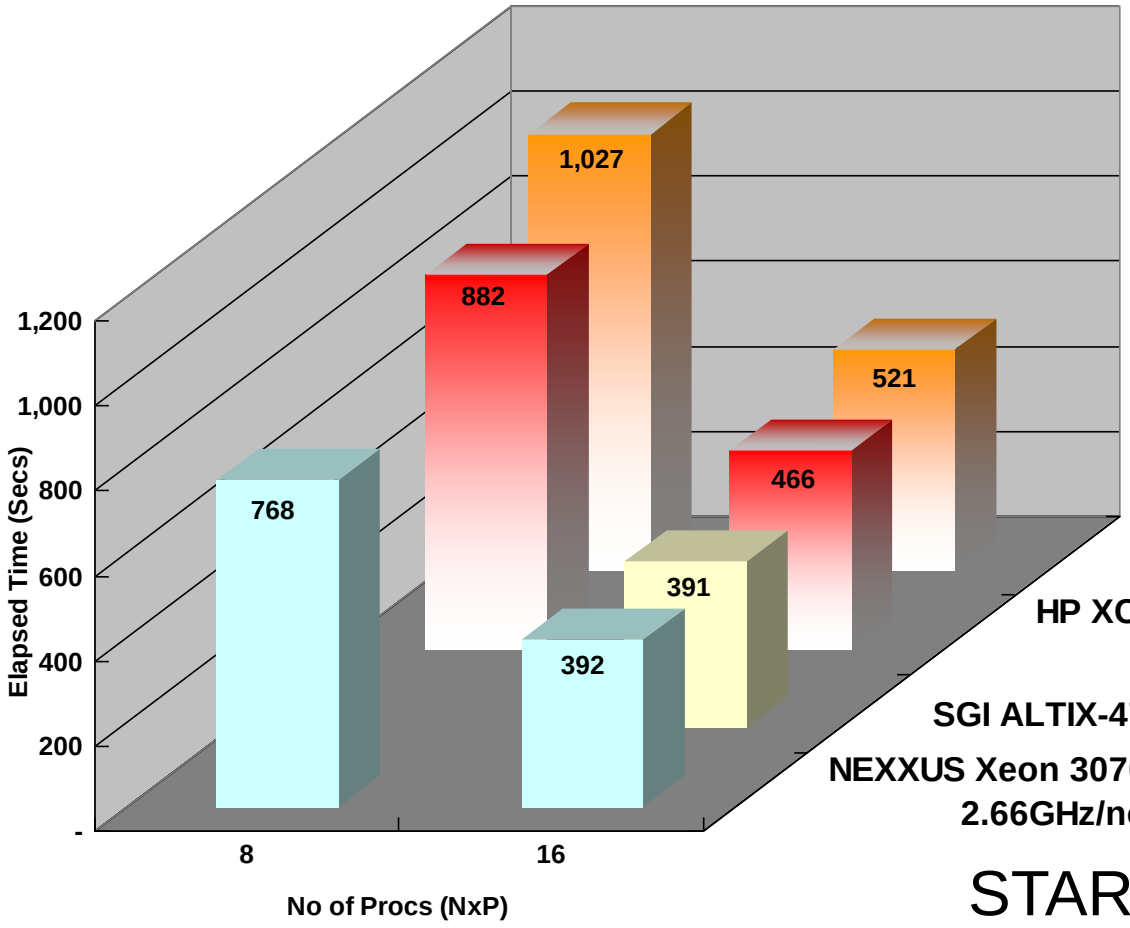


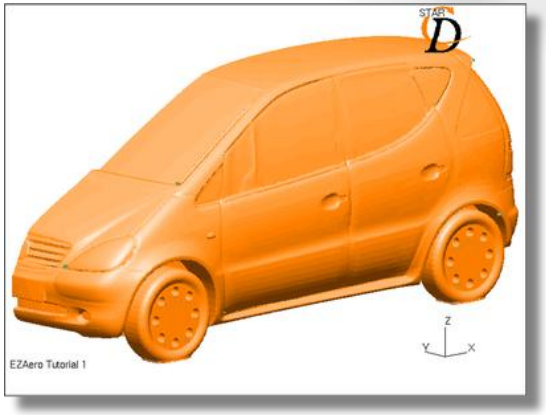
パーソナルクラスタ　事例&デモ

- Tyan Personal Supercomputer
- Intel Quad-Core Xeon Processor
- Mellanox High Performance InfiniBand Interconnect
- Microsoft Windows Compute Cluster Server
- Wolfram grid*Mathematica* Supercomputing Environment

パーソナルクラスタ　事例&デモ

パーソナルクラスタ　事例&性能





HP Woodcrest (2xDual Core 3.0 GHz/Node)
HP XC Opteron D1145XC Dual Core, 2 sockets, 2.2 GHz

SGI ALTIX-4700 1.6 GHz Montecito

NEXXUS Xeon 3070 (Dual Core 2.66GHz/node)

STAR-CDベンチマーク

Benchmarks STAR-CD V3240/V3260
A-Class DATASET

http://www.cd-adapco.com/products/STAR-CD/performance/320/aclass32.html

HPCシステムの新たな可能性
パーソナルクラスタの考察

パーソナルクラスタの背景

- HPCマーケット
- HPCシステムの課題
- マイクロプロセッサの方向性
- TCOの重要性

HPCシステムの二極分化

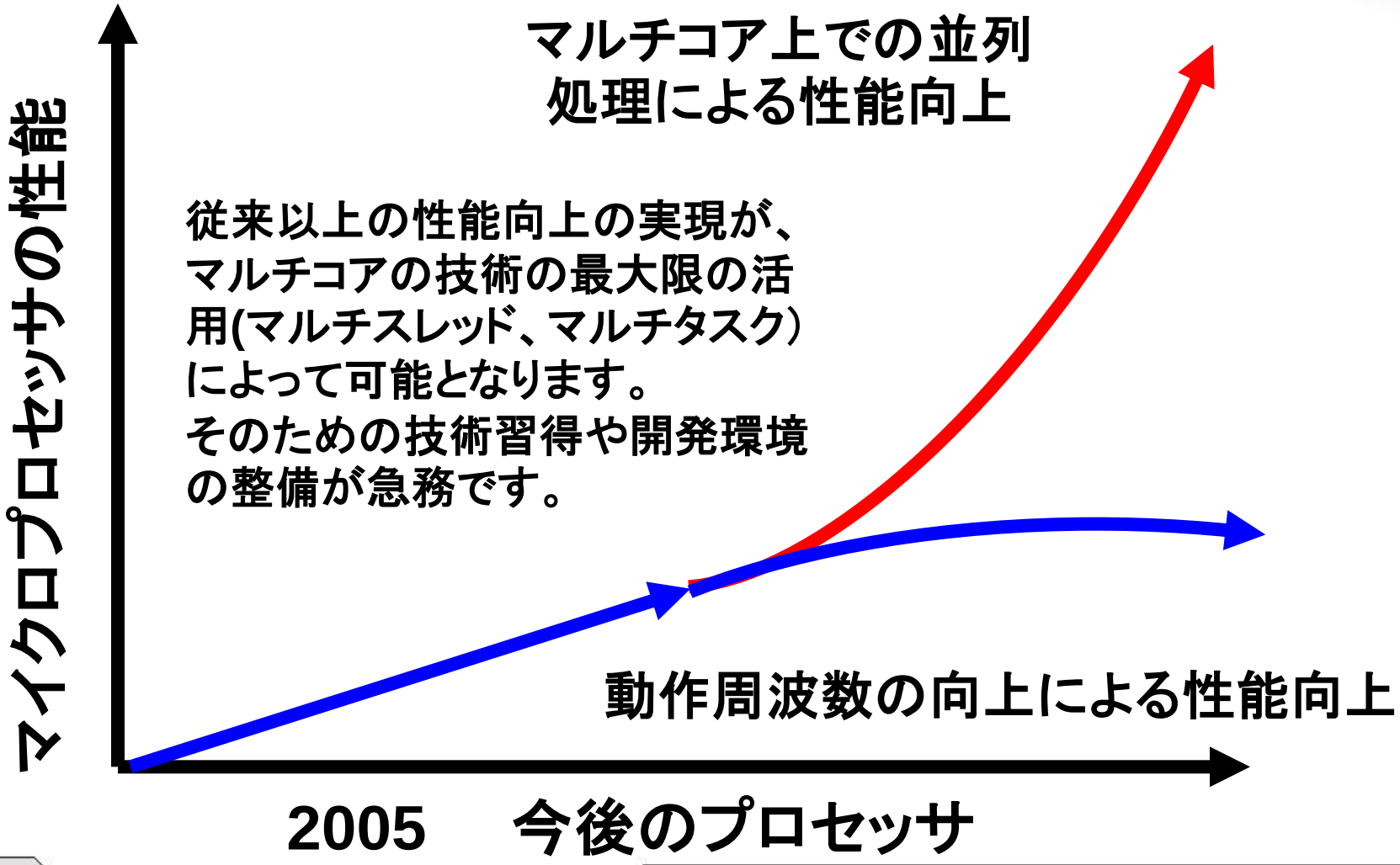
- ペタスケール コンピューティング
- コモディティ コンピューティング

パーソナルクラスタシステム

- システムの特徴
- 並列プログラミング

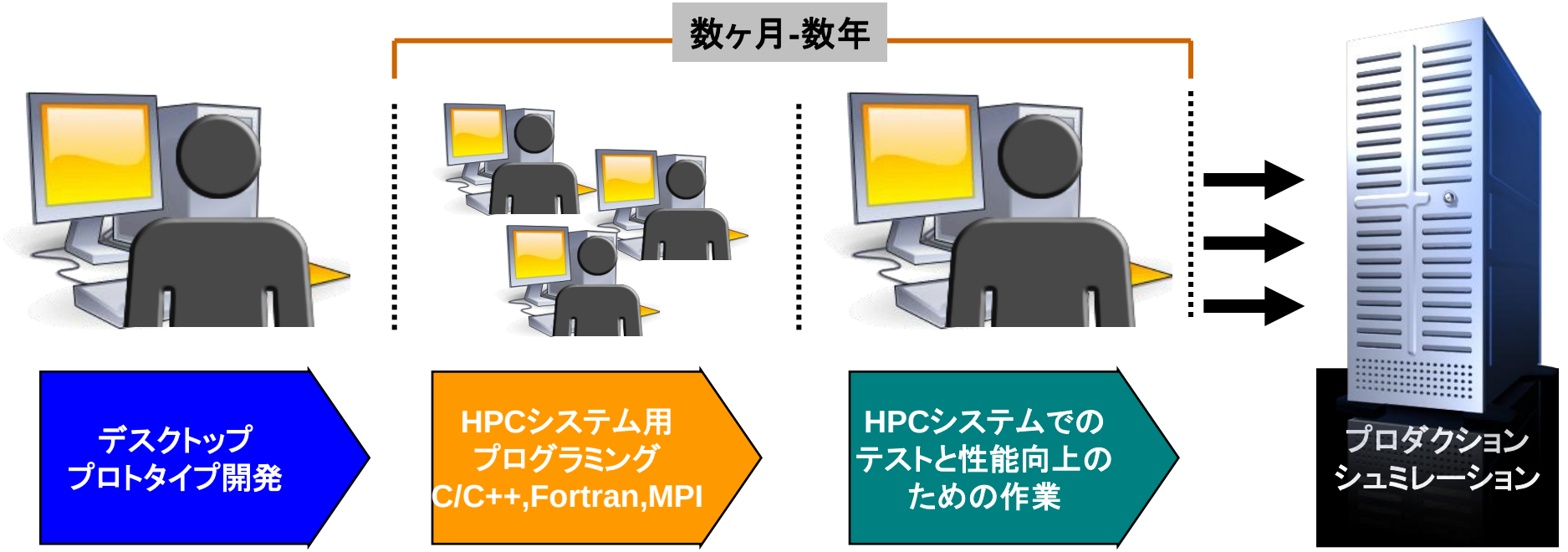
まとめ

マイクロプロセッサの性能向上
動作周波数からマルチコアへ



並列アプリケーション

- 従来型の並列アプリケーションの開発プロセス
  - バッチワークフローでの開発プロセス

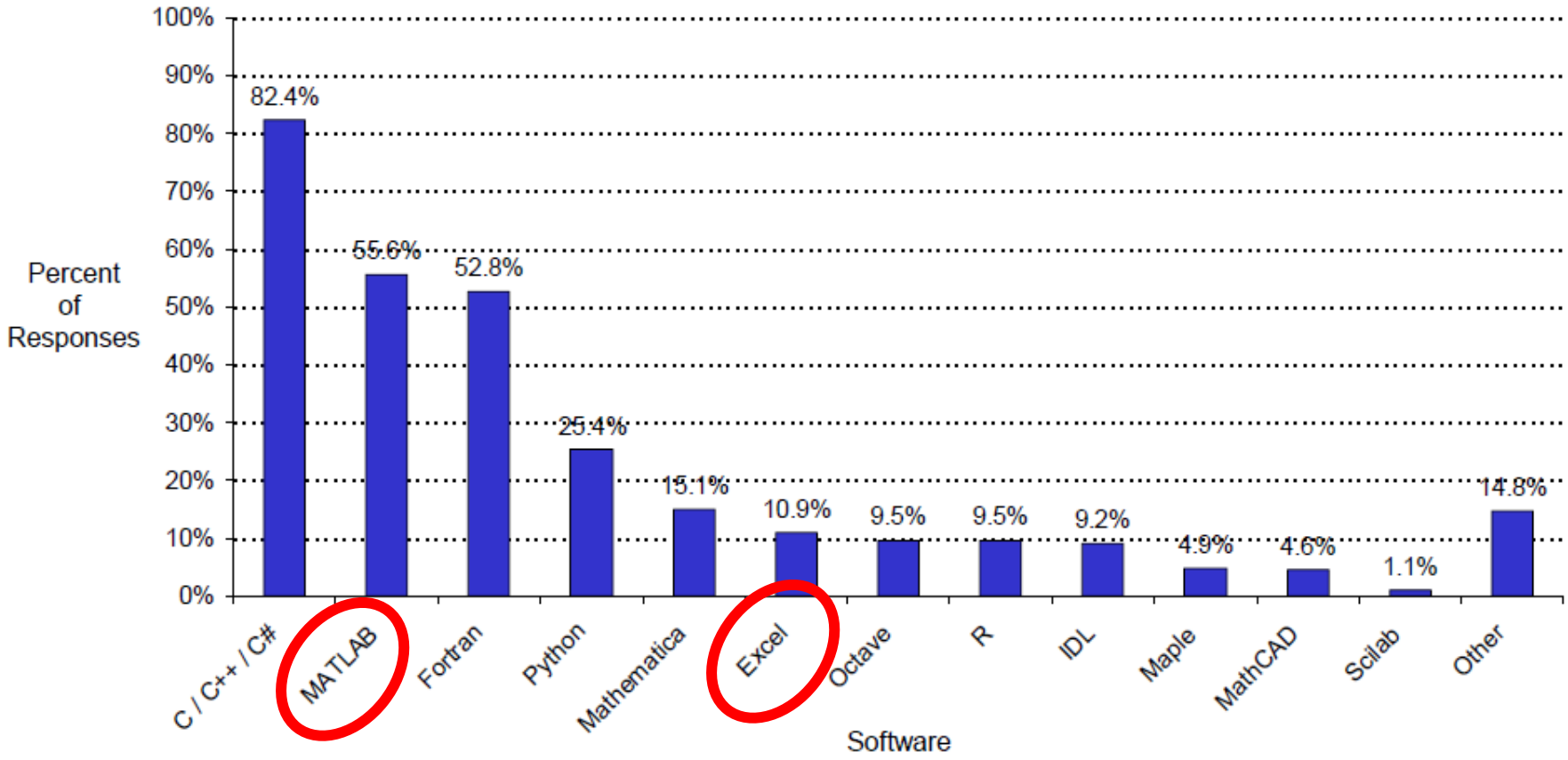


デスクトップ

- デスクトップでのプロトタイプ計算
  - ハイレベル言語（例えば、MATLABなど）
  - マイクロソフト エクセルなどでの処理
- ユーザ
  - システムアーキテクチャやマイクロプロセッサのマイクロアーキテクチャを意識することなくプログラミングを行う
  - GUIを利用した開発環境

並列プログラミングでの課題

- PC上でのプログラミングAPI



The Development of Custom Parallel Computing Applications
Simon Management Group
September 2006

複数選択可

並列アプリケーション

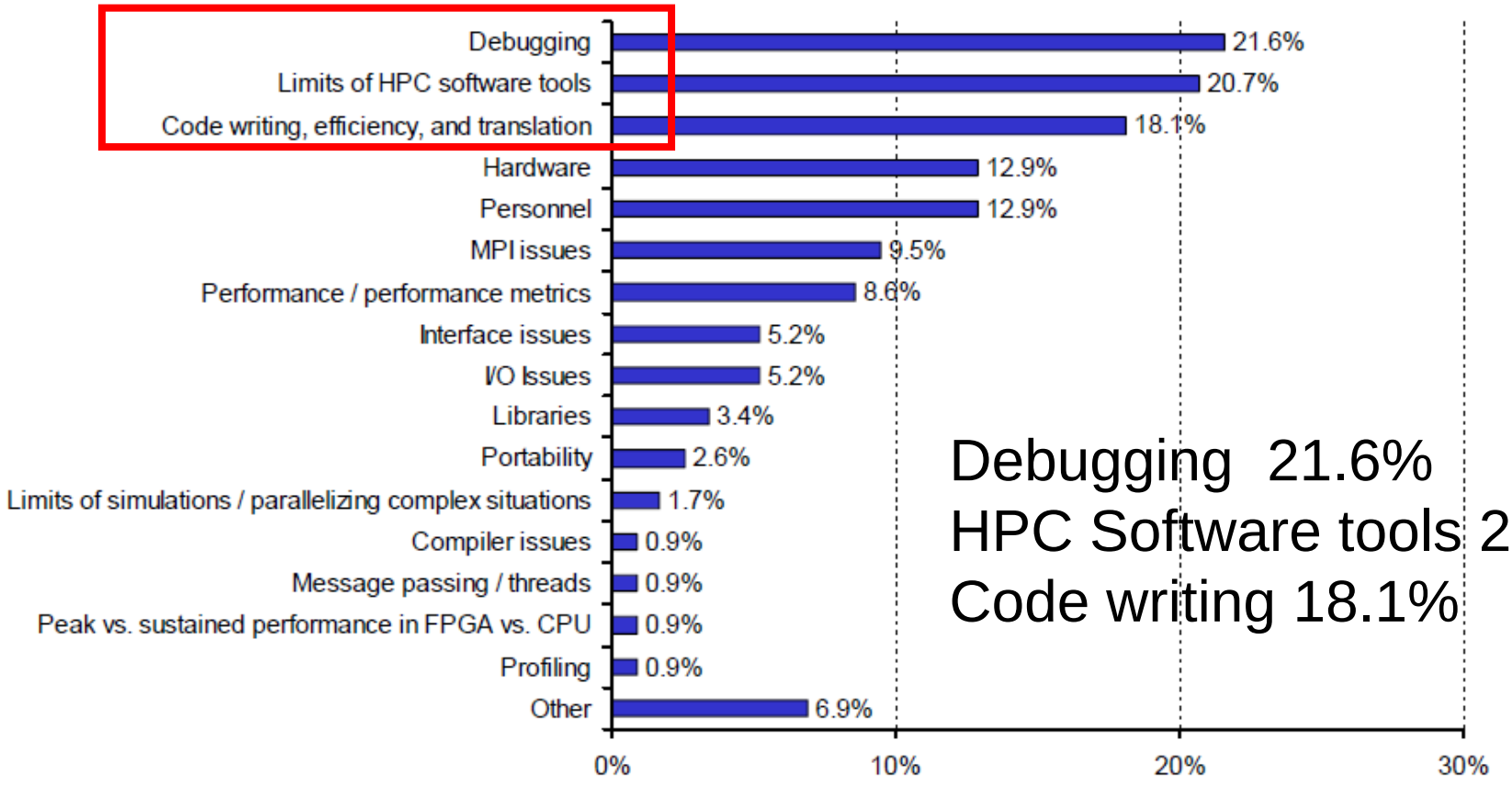
- プログラミング
  - 専門的な知識と並列APIに関する学習
  - C,Fortranなどのコンパイラを利用し、並列処理には、MPI（Message Passing Interface)などのコミュニケーションを明示的に記述
- ユーザ
  - 利用するHPCシステムのアーキテクチャを意識したプログラミング
  - プログラムの開発時にHPCシステムを利用する場合、バッチなどにジョブを投入し、プログラミングの確認を行うことが必要
  - 実際のモデル化やアルゴリズムを実際の解析対象で確認するのはプログラムの完成時まで困難

プログラミングの生産性

- 開発サイクル
  - プログラムのコーディング以外にも様々な作業が必要
- プログラムのデバッグ
  - 実際の解析モデルと入力データが必要
  - 逐次処理や対話処理が必要
- スケーラビリティの実現
  - HPCシステムでの高いスケーラビリティの実現には、高度なプログラミングが必要
  - アルゴリズムの選択やデータのモデル化の検討

並列プログラミングでの課題

- 並列アプリケーション開発でのボトルネック
  - デバッグ環境や開発ツールに関する不満



Debugging 21.6%
HPC Software tools 20.7%
Code writing 18.1%

The Development of Custom Parallel Computing Applications
Simon Management Group
September 2006

複数選択可

並列プログラミング

並列コンパイラ
並列デバッガ

並列数学ライブラリ
並列コード最適化ツール

スレッド解析ツール
最適化ツール

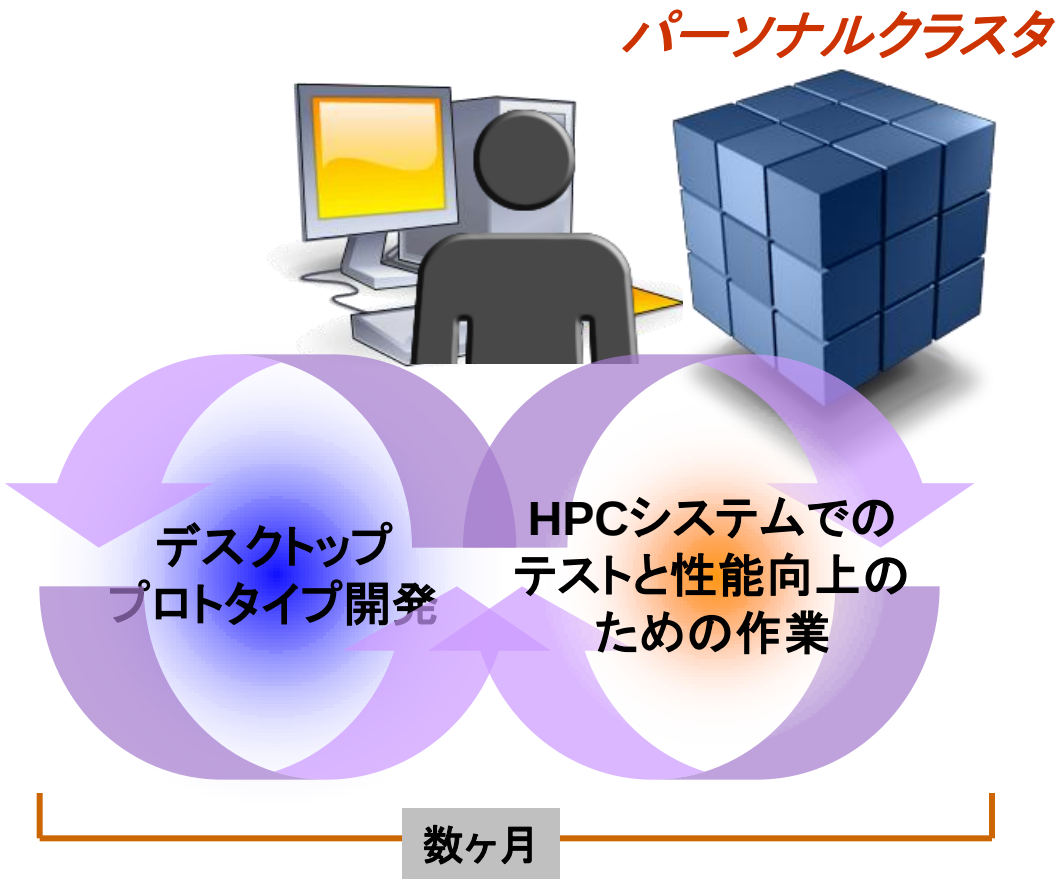
MPIタスク解析ツール
最適化ツール

統合インターフェイス

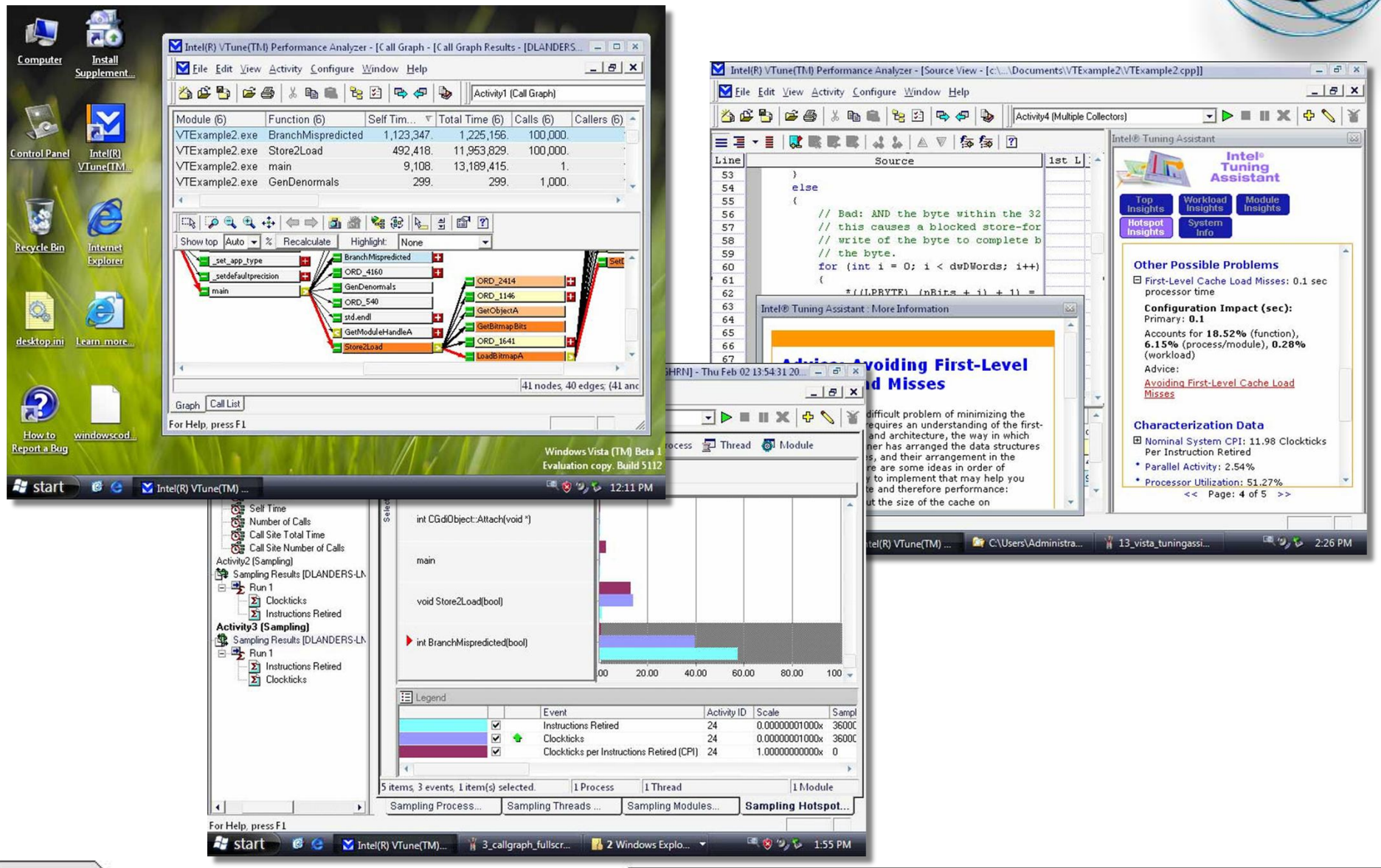
ソフトウエアのギャップの解決

プログラミングの生産性の向上

- 対話処理での開発環境

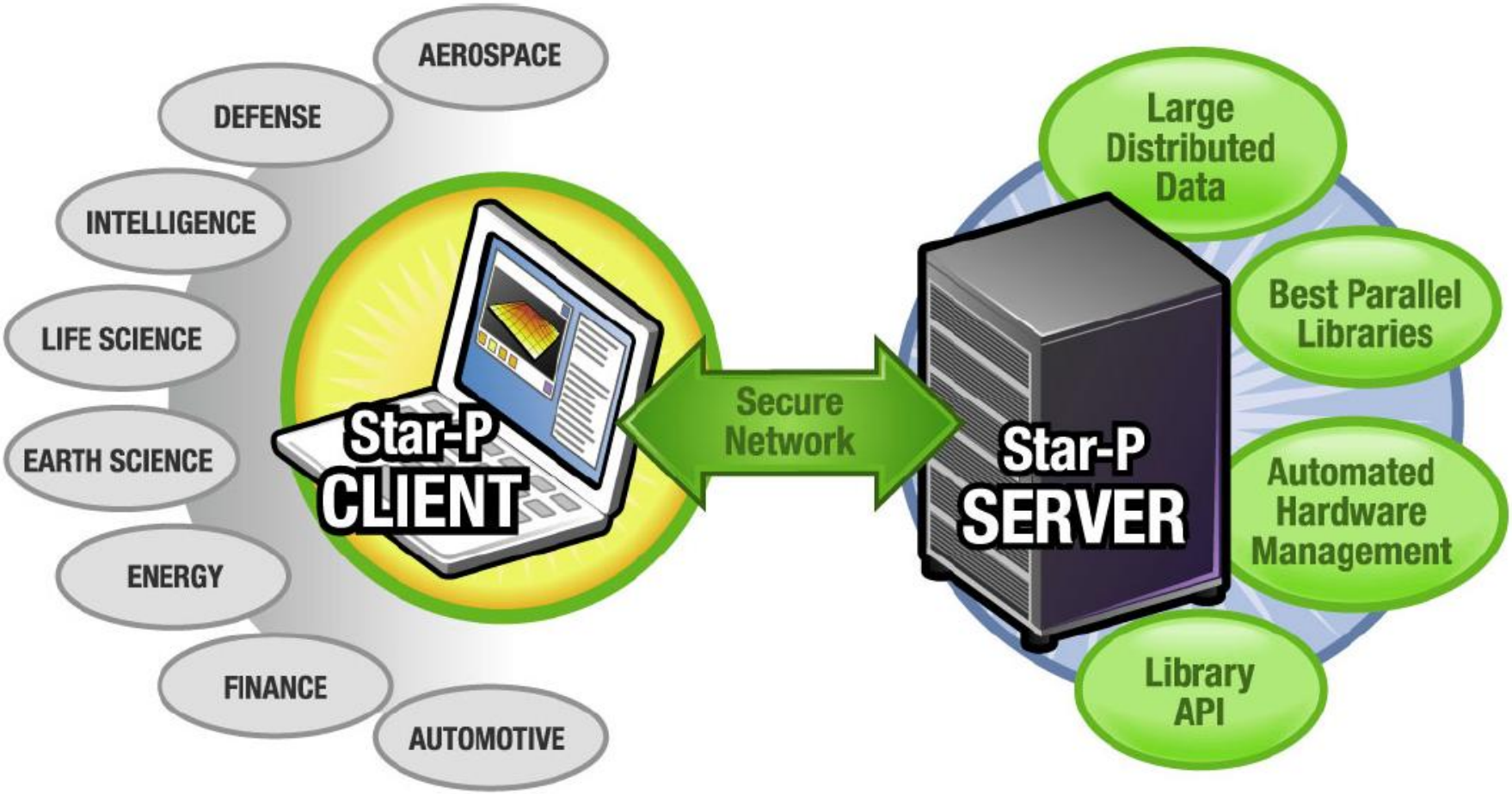


開発ツール

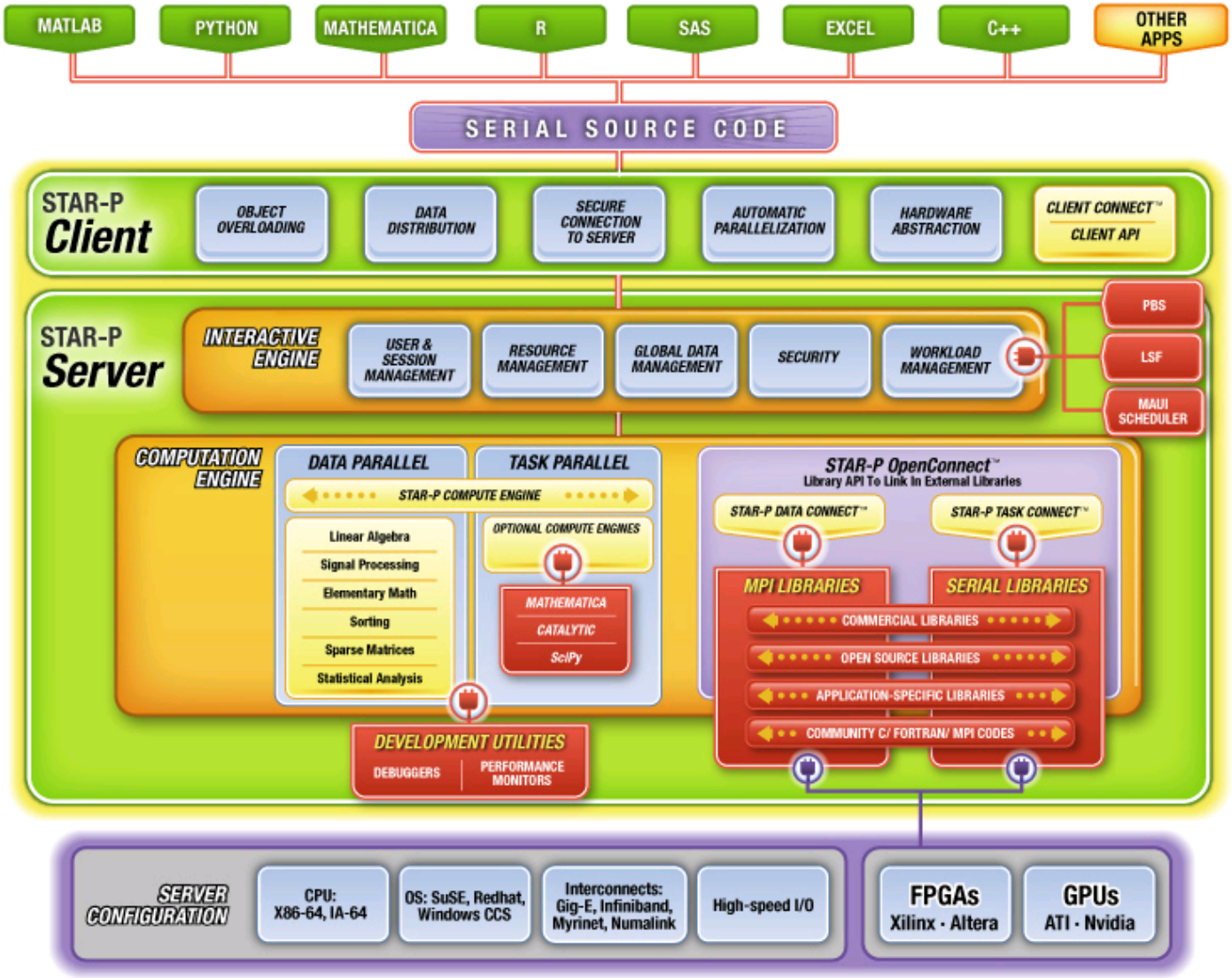

Intel(R) VTune(TM) Performance Analyzer - [Call Graph - [Call Graph Results - [DLANDERS...
Module (6) | Function (6) | Self Tim... | Total Time (6) | Calls (6) | Callers (6)
VTExample2.exe | BranchMispredicted | 1,123,347. | 1,225,156. | 100,000.
VTExample2.exe | Store2Load | 492,418. | 11,953,829. | 100,000.
VTExample2.exe | main | 9,108. | 13,189,415. | 1.
VTExample2.exe | GenDenormals | 299. | 299. | 1,000.
Intel(R) VTune(TM) Performance Analyzer - [Source View - [c:\...\Documents\VTExample2\VTExample2.cpp]]
Intel® Tuning Assistant
Other Possible Problems
First-Level Cache Load Misses: 0.1 sec processor time
Configuration Impact (sec): Primary: 0.1
Accounts for 18.52% (function), 6.15% (process/module), 0.28% (workload)
Advice:
Avoiding First-Level Cache Load Misses
Characterization Data
Nominal System CPI: 11.98 Clockticks Per Instruction Retired
Parallel Activity: 2.54%
Processor Utilization: 51.27%
Intel® Tuning Assistant : More Information
Avoiding First-Level Cache Load Misses
Activity2 (Sampling)
Activity3 (Sampling)
Clockticks
Instructions Retired
Clockticks per Instructions Retired (CPI)
int CGdiObject::Attach(void *)
main
void Store2Load(bool)
int BranchMispredicted(bool)
Sampling Process...
Sampling Threads ...
Sampling Modules...
Sampling Hotspot...

並列計算プラットフォーム事例

- Star-P 並列計算プラットフォーム



Star-P アーキテクチャ



プログラム例と性能データ

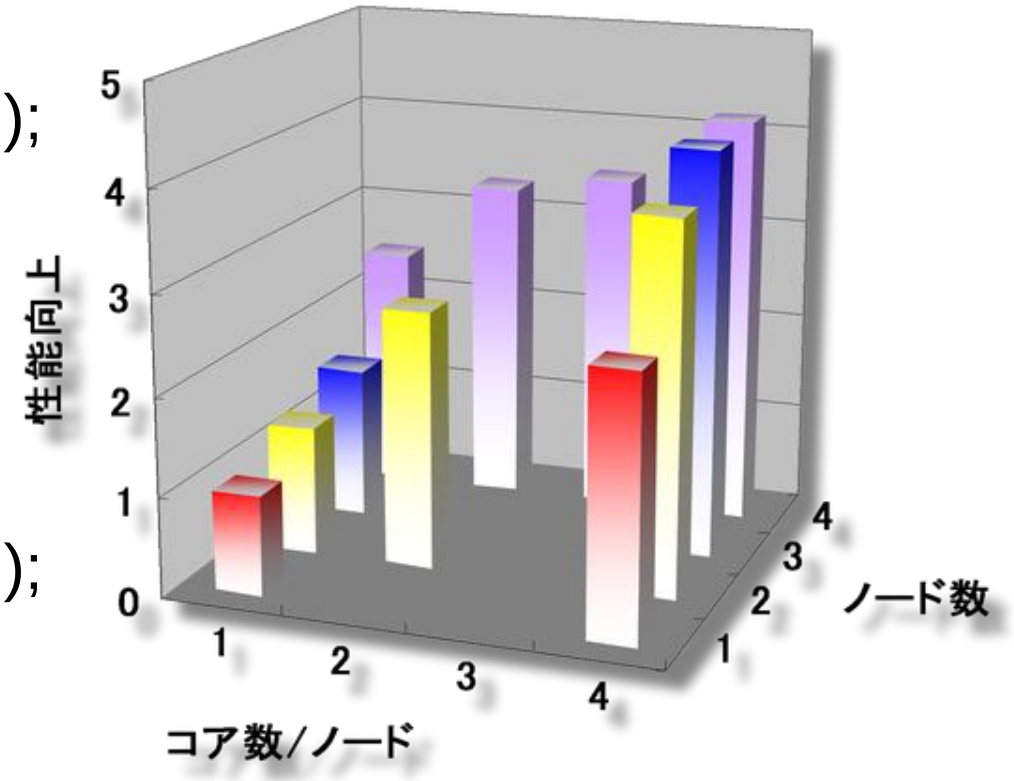
- NxN　行列の逆行列の計算

MATLAB

```
n = 7000;
a = rand(n); b = rand(n);
tic;a*b;inv(a);fft(a);toc
```

Star-P

```
n = 7000*p;
a = rand(n); b = rand(n);
tic;a*b;inv(a);fft(a);toc
```



ベンチマークシステム：NEXXUS 4820-AL

プログラム例と性能データ

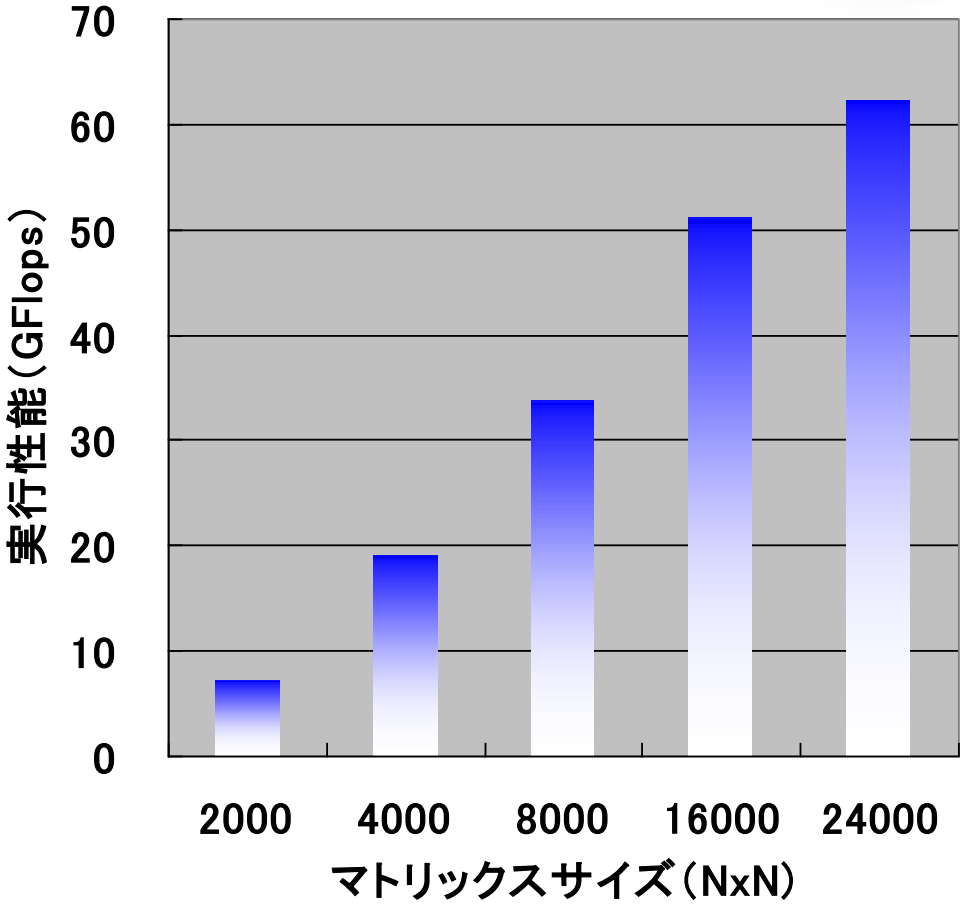
- NxN　行列積の計算

MATLAB

```
clear;
n = 24000;
a = rand(n); b = rand(n);
tic;a*b;toc
ppwhos a
```

Star-P

```
clear;
n = 24000*p;
a = rand(n); b = rand(n);
tic;a*b;toc
ppwhos a
```



ベンチマークシステム：NEXXUS 4820-AL

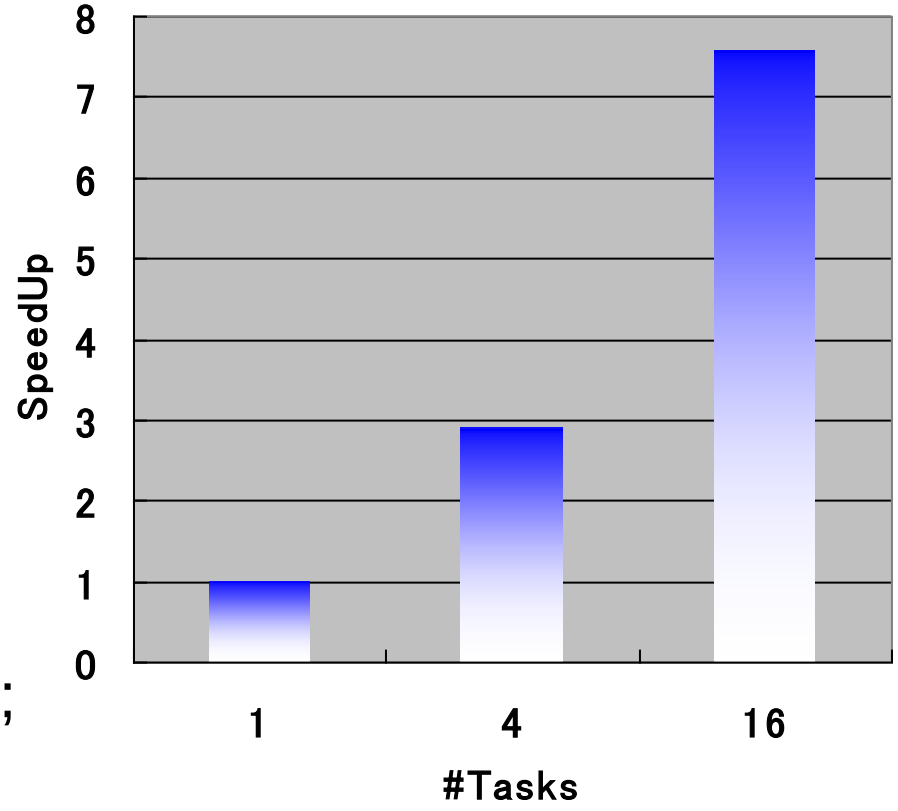
プログラム例と性能データ

- M回のNxN　行列の逆行列の計算

Star-P

```
b=rand(1000,1000,40);
a=rand(1000,1000,40);
tic;
for I=1:40
   b(:,:,I)=inv( a(:,:,I) );
end;
toc
a=rand(1000,1000,40*p);
%  b=rand(1000,1000,40*p);
tic; b=ppeval('inv', a); toc
```

プログラム前半の逐次実行部分
に対する実行時間での評価



ベンチマークシステム：NEXXUS 4820-AL

シングルAPIでの並列処理

MPI
OpenMP

Cluster OpenMPは、ノード内（SMP)とノード間で同一の並列APIでのプログラミングを可能とします。

OpenMP プログラムのコンパイルと実行例

```
$ cat -n pi.c
     1  #include <omp.h>                                  // OpenMP実行時関数呼び出し
     2  #include <stdio.h>                                // のためのヘッダファイルの指定
     3  #include <time.h>
     4  static int num_steps = 1000000000;
     5  double step;
     6  int main ()
     7  {
     8          int i, nthreads;
     9          double start_time, stop_time;
    10          double x, pi, sum = 0.0;
    11          step = 1.0/(double) num_steps;            // OpenMPサンプルプログラム:
    12          #pragma omp parallel private(x)           // 並列実行領域の設定
    13          {       nthreads = omp_get_num_threads(); // 実行時関数によるスレッド数の
取得
    14                  #pragma omp for reduction(+:sum)  // "for" ワークシェア構文
    15                  for (i=0;i< num_steps; i++){      // privateとreduction指示
句
    16                          x = (i+0.5)*step;         // の指定
    17                          sum = sum + 4.0/(1.0+x*x);
    18                  }
    19          }
    20          pi = step * sum;
    21          printf("%5d Threads : The value of PI is %10.7f¥n",nthreads,pi);
    22  }
$ icc -O -openmp pi.c
pi.c(14) : (col. 3) remark: OpenMP DEFINED LOOP WAS PARALLELI
pi.c(12) : (col. 2) remark: OpenMP DEFINED REGION WAS PARALLELIZED.
$ setenv OMP_NUM_THREADS 2
$ a.out
    2 Threads : The value of PI is  3.1415927
```

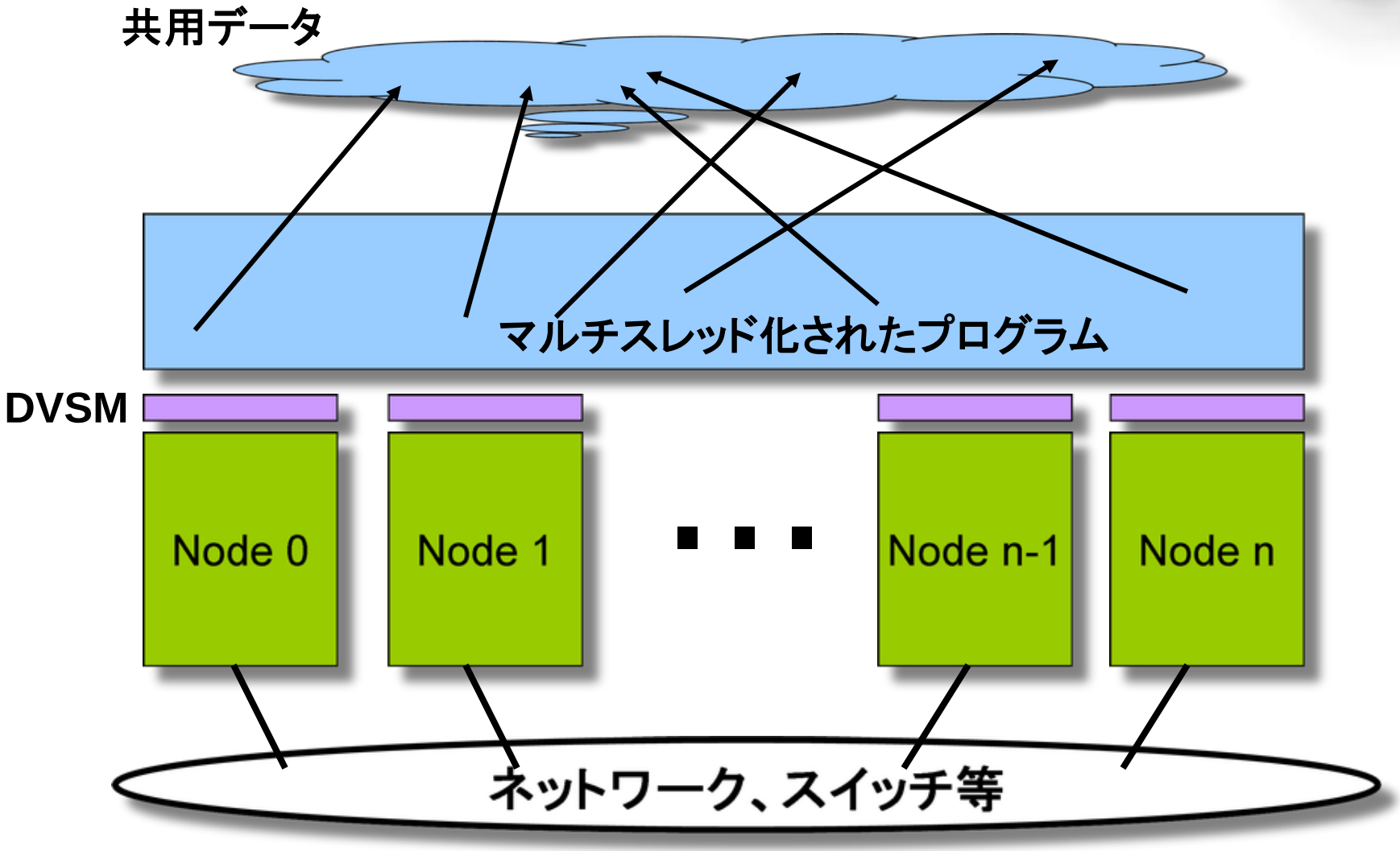
OpenMP指示行

OpenMP実行時関数

コンパイルとメッセージ

環境変数の設定

分散仮想共有メモリ(DVSM)
インテル　クラスタOpenMP



Cluster OpenMP プログラム コンパイルと実行例

クラスタ間共有データの定義

OpenMP実行時関数

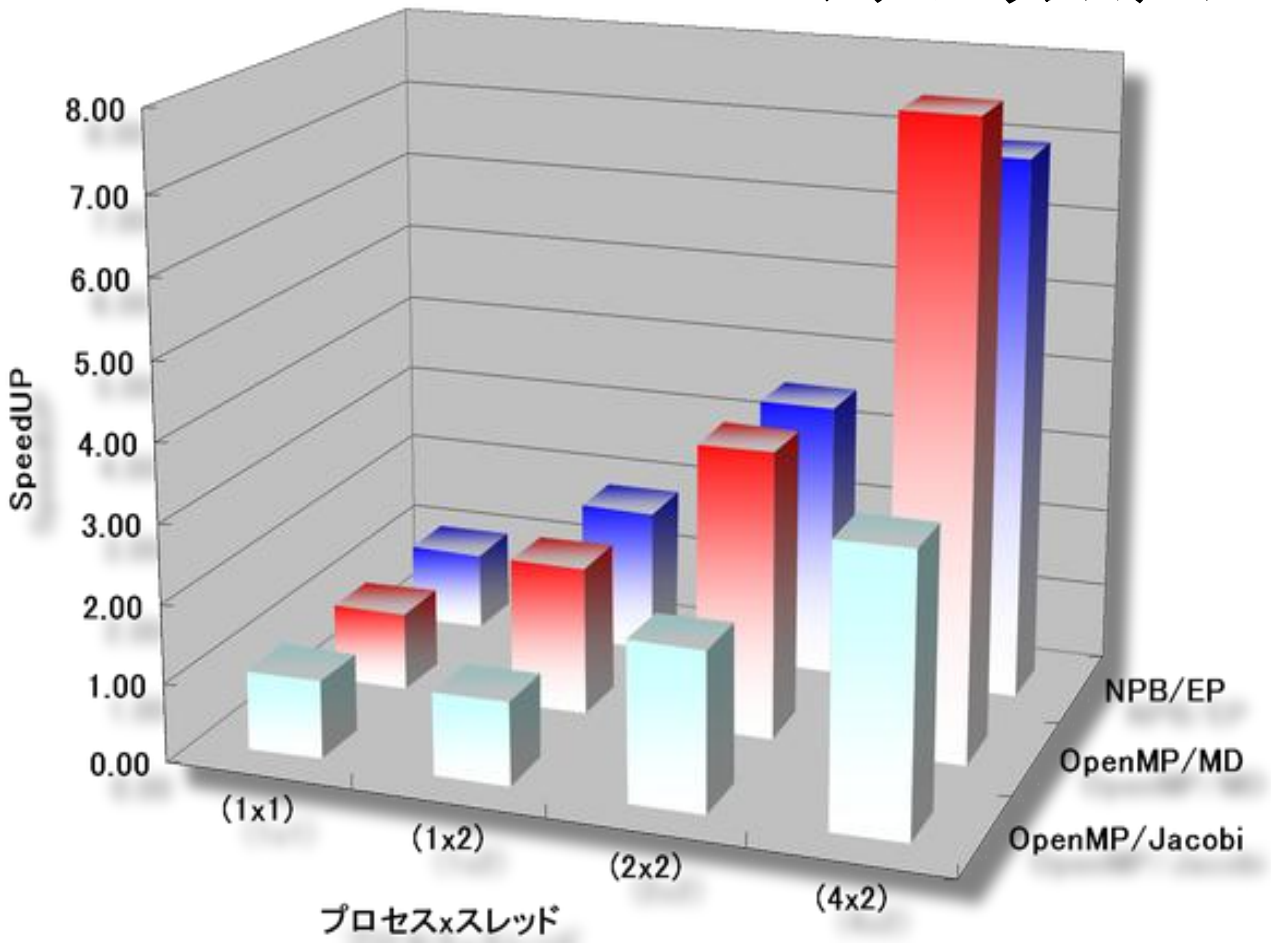
コンパイルとメッセージ

並列実行処理環境の設定

```
$ cat -n cpi.c
     1  #include <omp.h>                          // OpenMP実行時関数呼び出し
     2  #include <stdio.h>                        // のためのヘッダファイルの指定
     3  #include <time.h>
     4  static int num_steps = 1000000;
     5  double step;
     6  #pragma intel omp sharable(num_steps)
     7  #pragma intel omp sharable(step)
     8  int main ()
     9  {
    10  int i, nthreads;
    11  double start_time, stop_time;
    12  double x, pi, sum = 0.0;
    13  #pragma intel omp sharable(sum)
    14  step = 1.0/(double) num_steps;            // OpenMPサンプルプログラム:
    15  #pragma omp parallel private(x)           // 並列実行領域の設定
    16    {
    17      nthreads = omp_get_num_threads();     // 実行時関数によるスレッド数の取得
    18  #pragma omp for reduction(+:sum)          // "for" ワークシェア構文
    19      for (i=0;i< num_steps; i++){          // privateとreduction指示句
    20        x = (i+0.5)*step; // の指定
    21        sum = sum + 4.0/(1.0+x*x);
    22        }
    23    }
    24      pi = step * sum;
    25      printf("%5d Threads : The value of PI is %10.7f¥n",nthreads,pi);
    26  }
    27
$ icc -cluster-openmp -O -xT cpi.c
cpi.c(18) : (col. 1) remark: OpenMP DEFINED LOOP WAS PARALLELIZED.
cpi.c(15) : (col. 1) remark: OpenMP DEFINED REGION WAS PARALLELIZED.
$ cat kmp_cluster.ini
--hostlist=node0,node1 --processes=2 --process_threads=2 --no_heartbeat --startup_timeout=500
$ ./a.out
4 Threads : The value of PI is  3.1415927
```

Cluster OpenMP プログラム

ベンチマークシステム：NEXXUS 4820-PT



- NAS Parallel Benchmark / EP ベンチマーク
- OpenMPサンプルプログラム（分子動力学サンプル、nparts=10000で実行）
- OpenMPサンプル(Jacobi法サンプル、5000x5000）

HPCシステムの新たな可能性
パーソナルクラスタの考察

パーソナルクラスタの背景

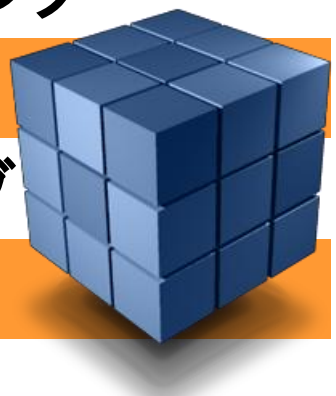
- HPCマーケット
- HPCシステムの課題
- マイクロプロセッサの方向性
- TCOの重要性

HPCシステムの二極分化

- ペタスケール コンピューティング
- コモディティ コンピューティング

パーソナルクラスタシステム

- システムの特徴
- 並列プログラミング



まとめ

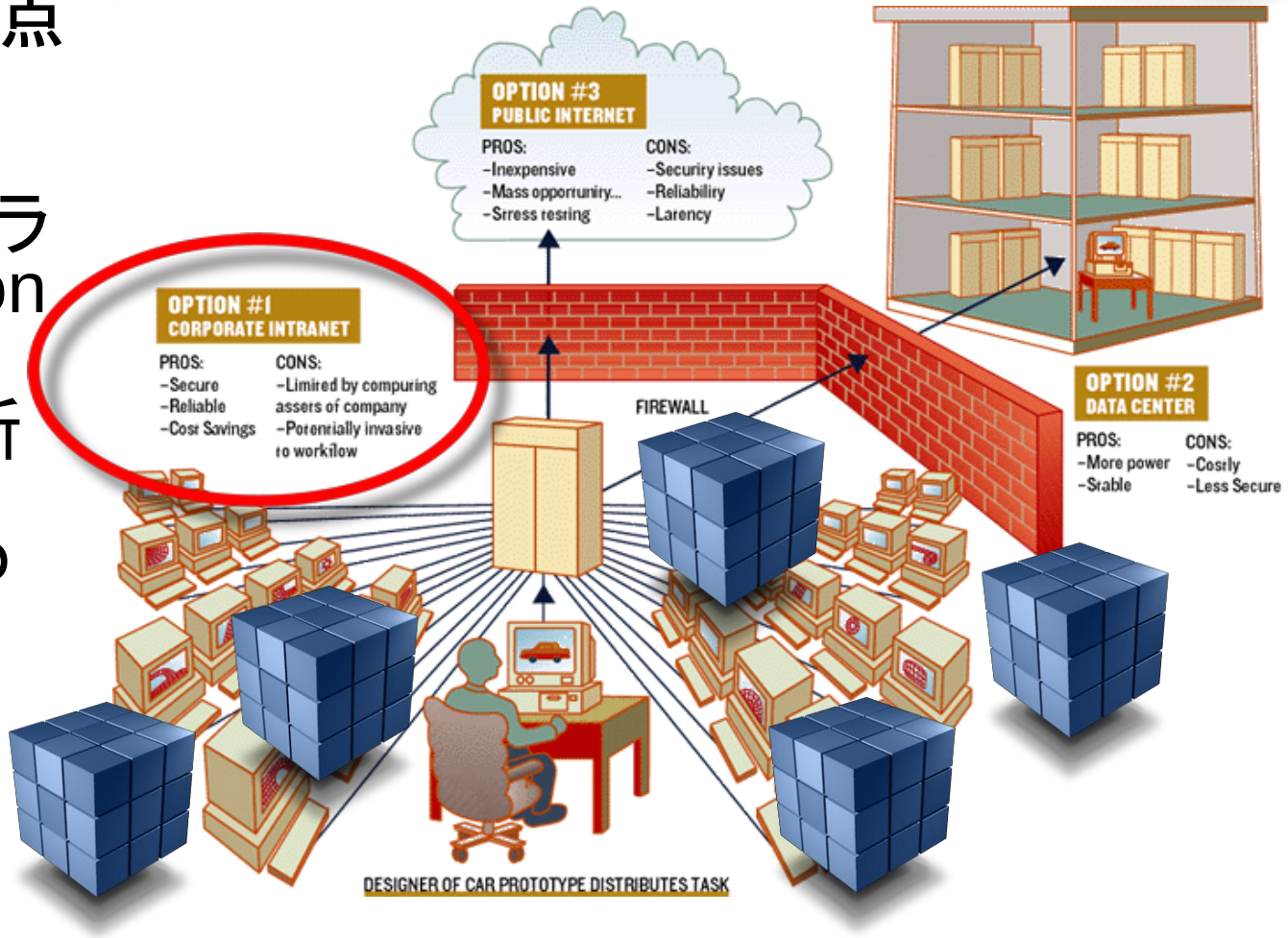
まとめとして・・・

- パーソナルクラスタの可能性
- パーソナルクラスタの将来

コンピュータ利用形態

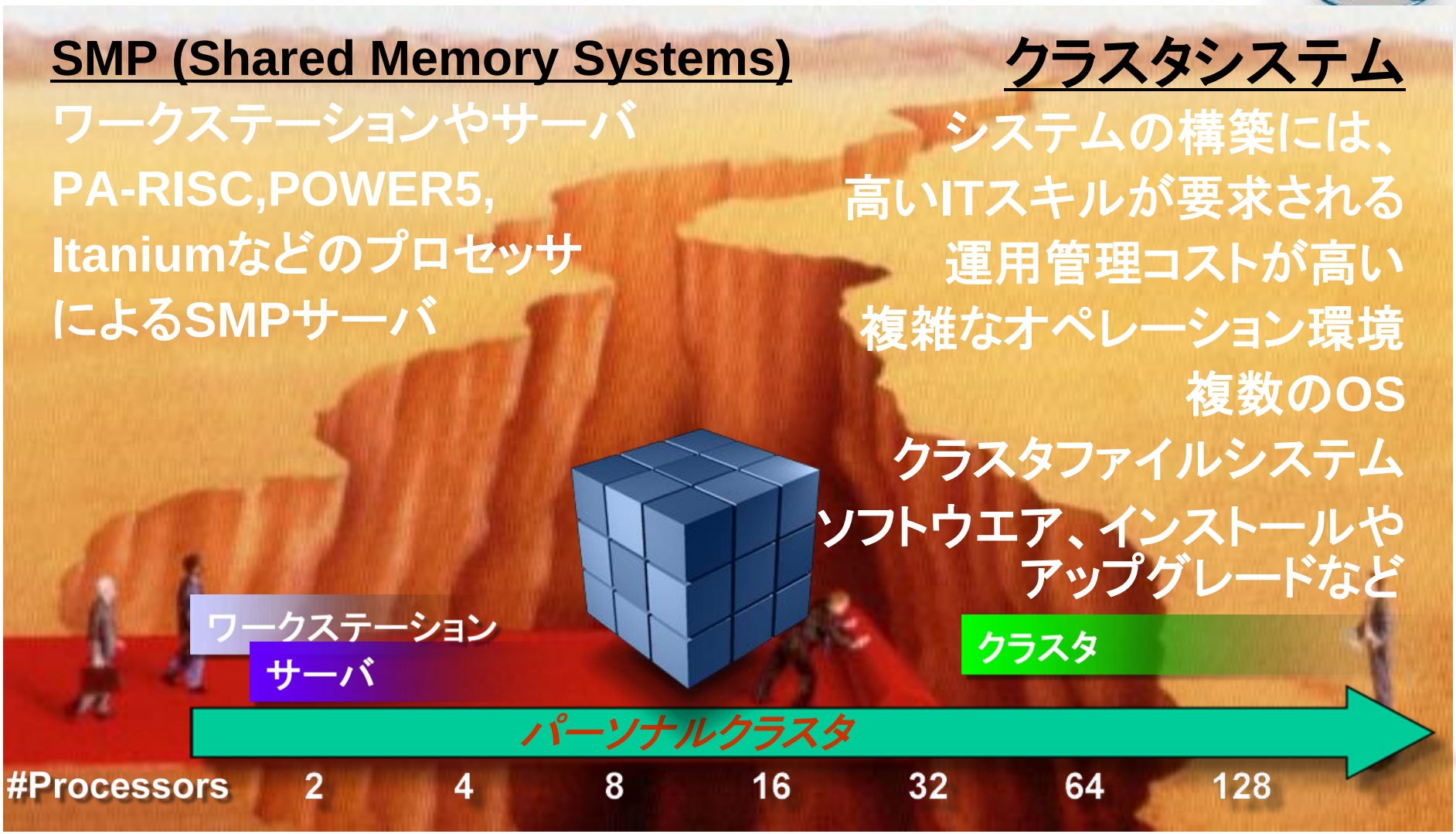
- 2000年11月27日にForbes誌に記載された利用形態での利点と欠点

- パーソナルクラスタは、Option #1としての利用形態に革新を引き起こす可能性を持つ



Forbes 11.27.00

HPCシステムのギャップ



1TFLOPSシステム

1996年
1.06 TFLOPS
7264 x Pentium Pro (57ラック構成)
1.8 TFLOPS
9200 x Pentium Pro (84ラック構成)
設置面積:1500ft^2 (140m^2)
800,000 Watts

2006年
44 x Quad Core プロセッサ
1.8TFLOPS
設置面積:16ft^2 (1.5m^2)
10,000 Watts

2007-2008年
パーソナルクラスタ

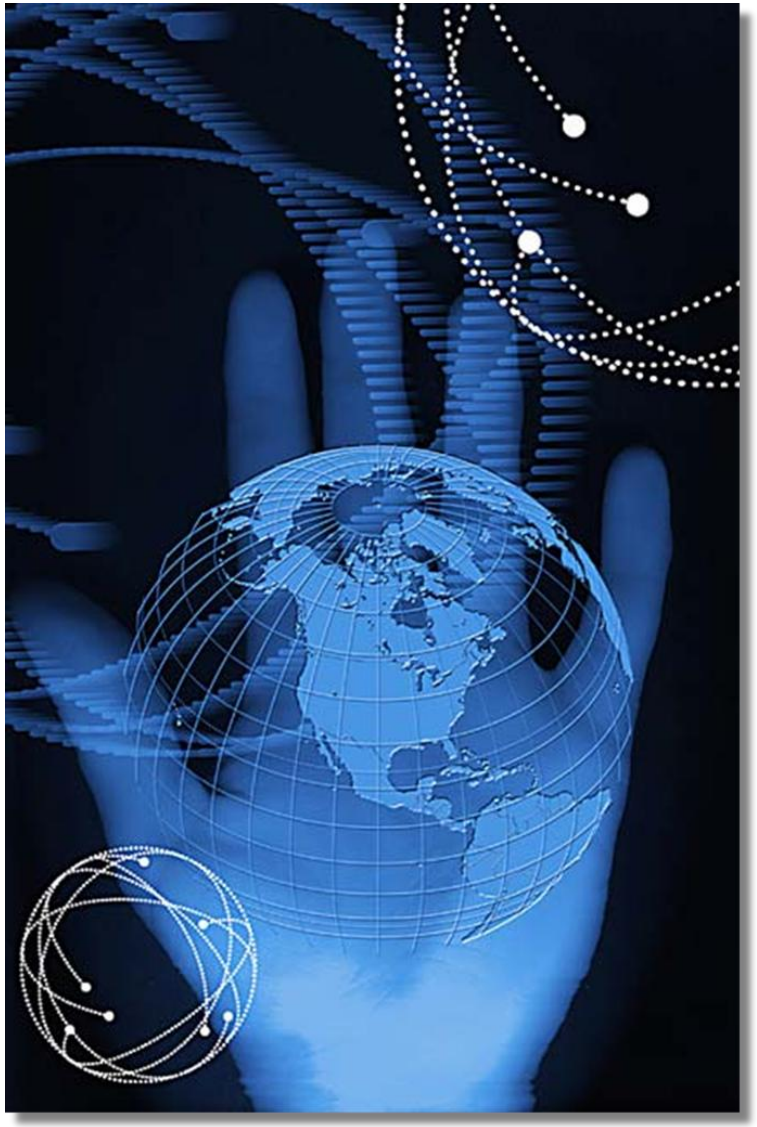
No.1よりOnly One

……
そうさ　僕らも　世界に一つだけの花
　一人一人違う種を持つ
その花を咲かせることだけに
一生懸命になればいい

小さい花や　大きな花　一つとして
　　（小さい花　大きな花）
同じものはないから
No.1にならなくても　いい
もともと特別な　Only one

「世界に一つだけの花 」

この資料について



ここに掲載した資料は、弊社の調査と見解に基くものであり、資料の中で示されている製品やサービスを提供している各社の公式な見解でも、また、マーケティング戦略に基くものではありません。あくまで、弊社としての意見だということにご注意ください。これらの資料の無断での引用、転載を禁じます。

社名、製品名などは、一般に各社の商標または登録商標です。なお、本文中では、特に®、TMマークは明記しておりません。

In general, the name of the company and the product name, etc. are the trademarks or, registered trademarks of each company.

2007年1月

さらに詳しい情報や最新情報は.....

ホームページにて公開しています。
ホームページには、お問い合わせ窓口も開設してありますので、ご利用ください。

コンサルテーション

http://www.sstc.co.jp

製品技術

http://www.hp2c.biz

2007年1月