Empowered by Innovation NEC

CLUSTERPRO

HA/SystemResourceMonitor R4.2 導入ガイド

HP-UX版

第 2 版

2011年 12月

日本電気株式会社

目次

本製品のねらい

本製品のねらい

障害情報や問い合わせの中で、カーネルパラメータの
チューニング不足(ファイルテーブルオーバーフロー等)や、
業務アプリケーションの不具合(メモリリーク、ファイルリークなど)に
よりシステムダウンにつながる事例があり、プロセスの利用する
システムリソースの監視機能への要望がありました。

本製品は、各プロセスが利用するシステムリソースに関する監視を
実現します。
異常を検出した場合には特定のアクション(テキストログ出力、
syslog通知、対象プロセス停止、任意のスクリプト実行)を行います。

導入効果

SystemResourceMonitor 導入前

システムダウン発生!!

・障害発生時に、原因の究明に人手と時間が必要になる。
　→管理コストの増加、不要な管理コストの発生に伴う人的資源の浪費。

・OSや業務APが復旧するまで、業務が停止する。
　→顧客信頼の喪失、ビジネスチャンスの損失が発生。

SystemResourceMonitor 導入後

・障害の発生を予測して対処が可能になる。
　→業務停止を伴うような障害を未然に防止。
　　顧客信頼の喪失、ビジネスチャンスの損失を予防。

・万が一障害が発生しても、原因の早期検出が可能になる。
　→障害発生管理コストの減少に伴う人的資源の有効活用が可能。

システムダウン防止!!

過去の障害事例とSystemResourceMonitorの導入効果

過去の障害事例

【事例1】 nfile(オープンできるファイルの最大数)枯渇によりOSストール

障害事例	ある業務APを導入し、サービスイン当初から順調に運用を続けていたが、1年後OS/APとも動作不能に！！
原因	業務APの不具合で、オープンしたファイルのクローズを行っていない箇所があった。 長期間無停止で運用を続けたことで、ファイルテーブルが枯渇した。
問題点	月次処理であったため、短期スパンではファイルのクローズ漏れを発見することが困難であった。 また、アプリケーションを再起動することで、現象が改善することから、原因の特定が難しかった。

【事例2】 物理メモリ枯渇によりOSスローダウン

障害事例	ある業務サービスを事業拡張のため次々と追加して運用したところOSのスローダウンが発生！ その後、監視系ソフトウェアの内部処理でタイムアウトが発生し、障害を誤検出し系切り替えが頻発！！
原因	物理メモリ確保のためにスワップのIN/OUTが大量発生し、OSのスローダウンを引き起こした。
問題点	APサーバはJavaで構築されており、業務拡張を行うたびにJavaプロセスが増加していった。Javaプロセスは 起動時にヒープ領域を確保するため、起動数の増加と共にメモリ使用量が増加し物理メモリが枯渇した。

(続き)

過去の障害事例

【事例3】 nproc(プロセスの同時起動数)枯渇によりOSストール

障害事例	あるデータベース製品の同時接続数を拡張して運用を開始したところ、データベース起動後しばらくして他のOSプロセス等が軒並み動作不能に！！
原因	接続数が増したことにより起動プロセス数が増加し、nproc(プロセスの同時起動数)が枯渇しOSストールを引き起こした。
問題点	起動プロセス数の増加により、nproc(プロセスの同時起動数)の上限値に達し、それ以上プロセスが起動できなくなった。

【事例4】 nflocks(ロックファイル数上限)枯渇によりアプリケーションが動作不能

障害事例	ある Webサーバの設定を変更して起動したところ、監視系PPが動作不能に！！
原因	Webサーバの使用するロックファイルが増加したことで、nflocks(ロックファイル数上限)が枯渇したため他のロックファイルを使用する監視系PPがロックファイルを獲得できなくなった。
問題点	Webサーバ動作時にロックファイル資源を獲得する仕様であることを確認できていなかった。 また、プロセス数増加とともにロックファイル資源が増えることは検討されていなかった。

(続き)

SystemResourceMonitorの導入効果

リソースの統計情報を継続的に収集、解析

■ プロセス個別の情報を収集

- ファイルリーク／メモリリーク／ゾンビ状態等のプロセスを早期に検出し、障害が発生する前に対処が可能となります。

■ システム全体の情報を収集

- 物理メモリの枯渇やリソースの開放漏れによるカーネルパラメータに関する異常を早期に検出し、
  障害が発生する前に対処が可能となります。

障害発生時にはさまざまなアクションが実行可能

■ syslog通知、テキストログ出力

- 障害の原因となるプロセスの早期検出を行うことで、起こりうる障害を未然に察知し、早急に対処することが
  可能になります。
- 障害の要因を早期に検出することができ、スローダウンなどの業務停止を引き起こすような障害を未然に防止し、
  継続的に安定した業務運用が可能になります。

■ 対象プロセスの停止

- 障害の原因となるプロセスを強制的に停止することで、スローダウンなどの業務停止を引き起こすような障害を未然に
  防止し、継続的に安定した業務運用が可能になります。

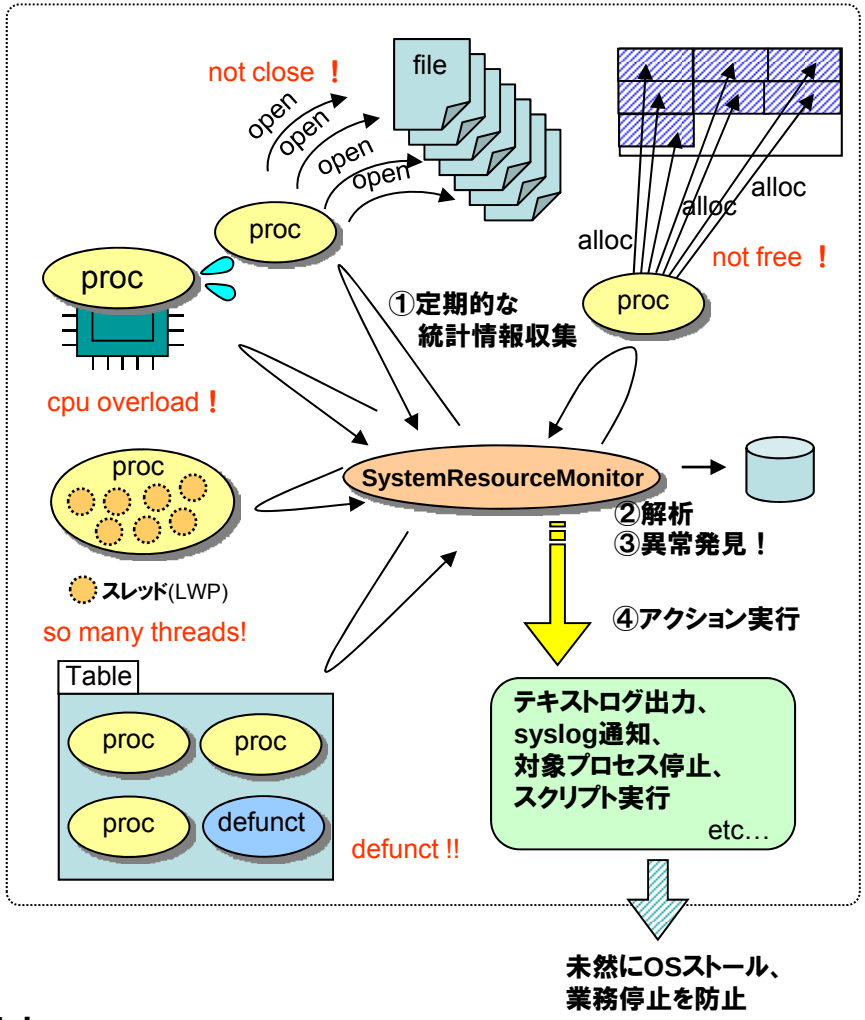
■ スクリプト実行

- ユーザが任意のスクリプトを指定することができ、障害発生時に有効な処理を自動的に実行することが可能です。
  障害発生時のシステムの状態を採取するスクリプトを提供しています。

プロセス単位のリソース監視について

システム上で動作するプロセスの統計情報の収集／解析を行います。
解析結果から以下の項目について、異常と判断すると既定のアクションを実行します。

- **メモリリーク、ファイルリークおよびスレッドリーク監視**
  - → 長期間統計的に情報を収集しなければ発見が困難な微量なメモリリークやファイルリーク、スレッドリークを自動検出し、既定アクション(テキストログ出力、syslog通知、対象プロセス停止等)を行います。
- **高CPU使用率プロセス監視**
  - → CPU使用率の高いプロセスを自動検出し、既定アクション(テキストログ出力やsyslog通知、対象プロセス停止等)を行います。
- **プロセス多重度監視**
  - → 同名のプロセスがどれだけ存在するか(多重度)を監視し、閾値を超えた場合に既定アクション(テキストログ出力やsyslog通知、対象プロセス停止等)を行います。
- **ゾンビプロセス(defunct)監視**
  - → ゾンビプロセス(defunct)の存在を検出すると通報し、既定アクション(テキストログ通知やsyslog通知等)を行います。



システム全体のリソース監視について

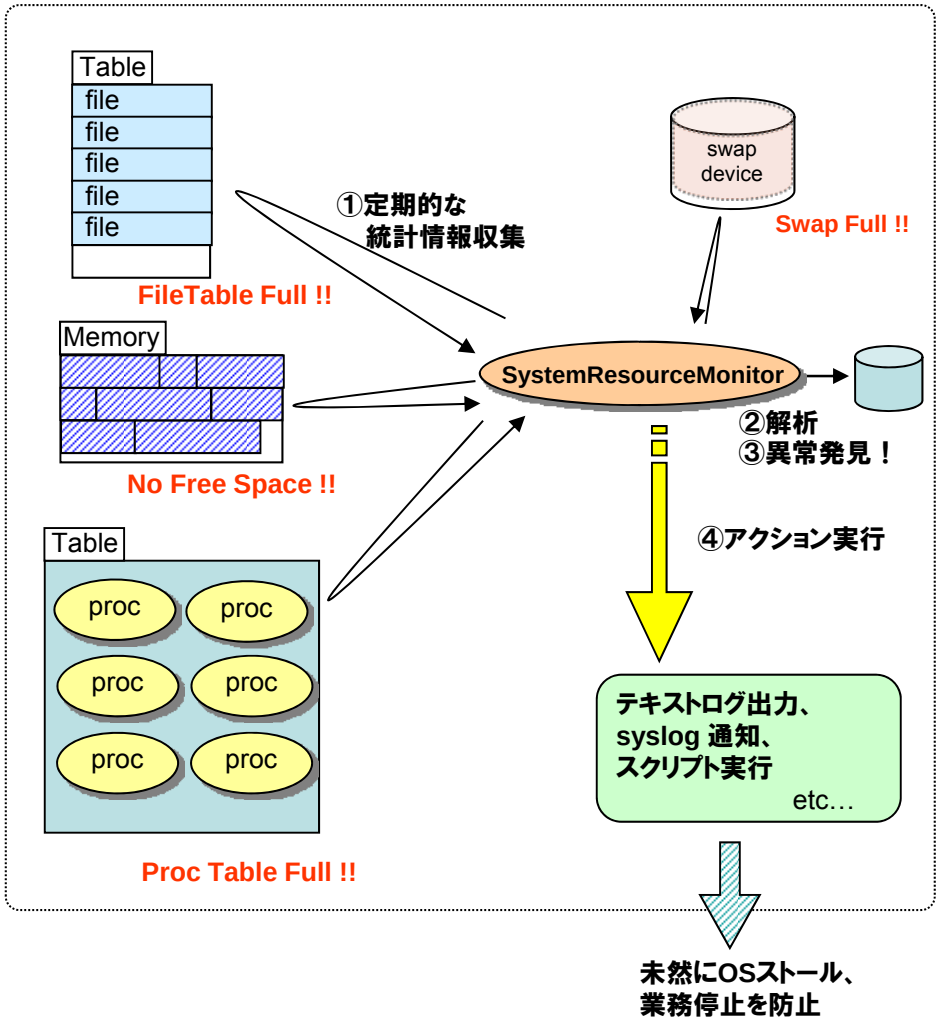
システム上で動作するプロセス全体の利用リソースの統計情報の収集／解析を行います。解析結果から以下の項目について、異常と判断すると既定のアクションを実行します。

システムリソース監視

【監視項目】

総オープンファイル数、総プロセス数、
総ロックファイル数、総メモリ量、CPU使用率、
総スワップメモリ量、ユーザごとの起動プロセス数、
総スレッド数

→ 閾値を超えると通報し、既定アクション（テキストログ通知やsyslog通知等）を行います。

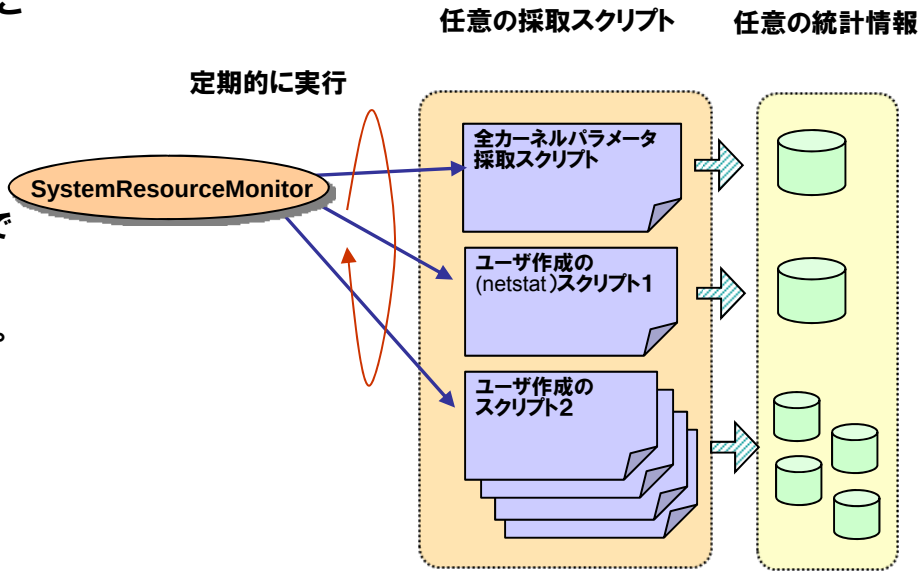


定期的な統計情報の採取

システム上の様々な情報を統計的に採取する機能を提供します。
これらの情報はシステムの稼動状態の把握や将来予想されるシステムリソースに関する
性能問題の早期検出、システム拡張を判断するための材料などさまざまな用途に利用できます。

- ユーザ作成の統計情報採取スクリプトの組み込み

  ユーザが必要とする情報を採取するスクリプトを作成することで、これを定期的に実行し統計情報として保存する仕組みを提供します。
  例えば、ネットワークステータスの表示コマンドnetstat(1)を実行するスクリプトを作成することで、アクティブソケットのステータスやプロトコルの内部状態などの情報を採取することができます。



また、本製品では、以下のような情報採取スクリプトを提供しています。

- 全カーネルパラメータ情報定期採取スクリプト
- システム動作情報(sarコマンド)定期採取スクリプト
- プロセス間通信情報(ipcsコマンド)定期採取スクリプト
- 仮想メモリ統計情報(vmstatコマンド)定期採取スクリプト
- システムパフォーマンス情報(glanceコマンド)定期採取スクリプト

得られた情報からパラメータ値の変化傾向などを分析することで、
性能問題や将来予測されるリソース枯渇問題などを未然に見つけ出すことも可能です。

プロセス単位のリソース監視異常を検出する条件

本製品では、「最大値更新回数」、「増加率」という2つのパラメータを組み合わせて検出を行います。

ファイルリーク／スレッドリーク ・・・最大値更新回数が閾値を超えると検出します。
メモリリーク ・・・・・・・・・・・・・・・・・・・・・最大値更新回数が閾値を超え、増加率も閾値以上の場合に検出します。
ゾンビプロセス（defunct）・・・・・・・・最大値更新回数が連続して閾値を超えると検出します。
高CPU使用率プロセス ・・・・・・・・ CPU使用率上限値を超えた回数が連続して閾値を超えると検出します。

［パラメータのデフォルトの閾値］

初期値からの増加率	10%
最大値を更新した回数（増加回数）/ CPU使用率上限値以上の回数	288回
CPU使用率上限値	90%
モニタ間隔	5分

常にメモリ、オープンファイル数が増加するプロセス、常にゾンビ（defunct）状態のプロセスおよび常にCPU使用率上限値を超えているプロセスの場合、デフォルト値（モニタ間隔5分、閾値288回）で運用すると、約1日でこれらの異常を検出します。計算式は以下のとおりです。

計算式 ： 統計情報を収集する間隔のデフォルト（モニタ間隔）＝5分
24(時間)＊60(分)／5(分) ＝ 288(回)

［デフォルトで運用した場合の検出パターン］

	メモリリーク監視		ファイルリーク / スレッドリーク 監視	ゾンビプロセス（defunct）監視	高CPU使用率プロセス監視
最大値更新回数 ＼ 増加率	10%未満	10%以上	条件なし	条件なし	条件なし
288回未満	×	×	×	×	×
288回以上	×	○	○	○	○

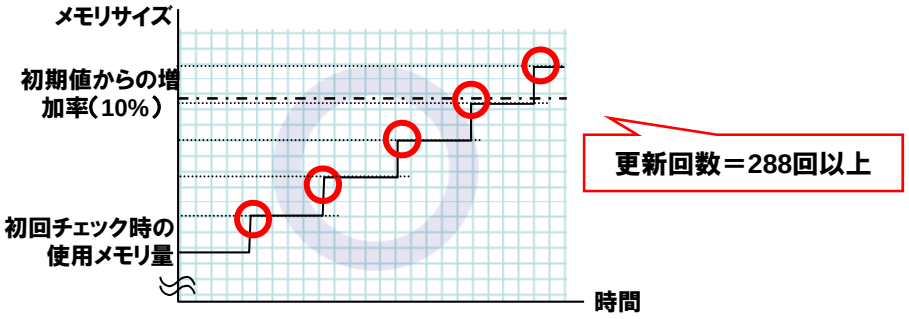
※ ○ ・・・ 異常を検出します。　× ・・・ 異常を検出しません。

プロセス単位のリソース監視異常検出のパターン

メモリリークの検出パターンは以下のようになります。

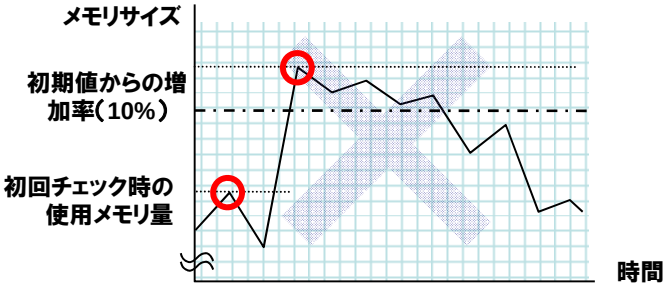
●メモリ使用量があるタイミングごと(ユーザ、サービス等を追加時)更新し、増加率を超えた

ユーザ、サービス等の追加により、一定のメモリ使用量が増加し、
そのつど最大値更新回数はカウントされます。
最大値更新回数及び増加率が閾値を超えるとメモリリークと判定します。



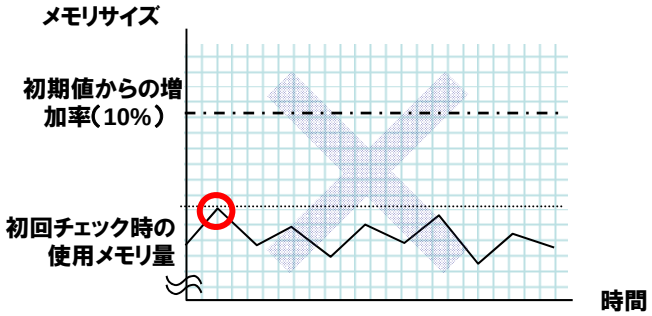
●メモリ使用量が一時的に増加率上限値を超えて増加

一時的に増加率が初期値の10%を超えているが、一定範囲内での一時的なメモリ使用量の増減のため、メモリリークと判定しません。



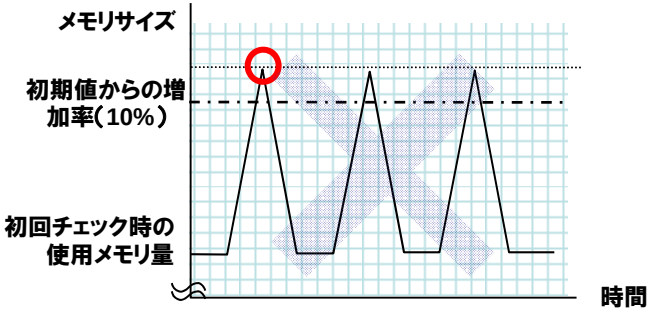
●メモリ使用量が経過時間と共に一定の範囲内で増減

メモリ使用量は一定の値を超えない範囲で増減しているため、更新回数はカウントされません。



●メモリ使用量が一定回数以上更新し、増加率を超えた

メモリ使用量は、ある期間(月次処理等の場合)になると初期値の10%を超えて増加するが、ある範囲内での増加にとどまり、その後は最大値が更新されないため、メモリリークと判定しません。

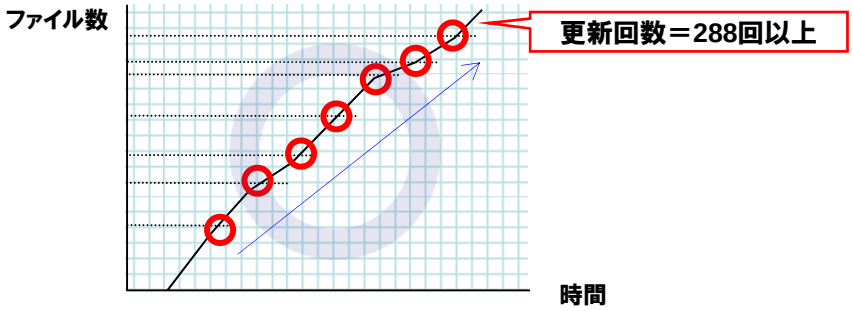


(続き)

ファイルリークの検出パターンは以下のようになります(スレッドリークについても同様です)

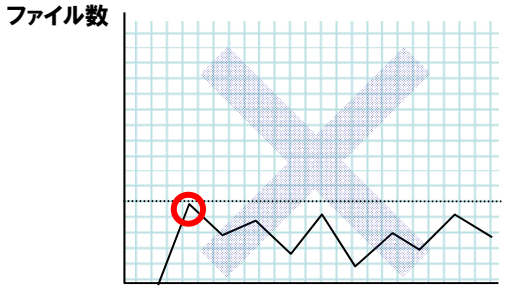
●オープンファイル数が経過時間と共に常に増加しつづけ、更新回数の上限値を超えた

オープンファイル数の最大値は毎回更新され、規定回数(288回)を超えたためファイルリークと判定します。



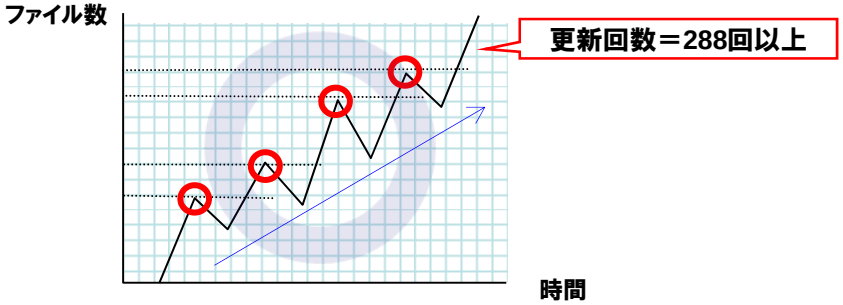
●オープンファイル数は経過時間とともに一定の範囲内で増減

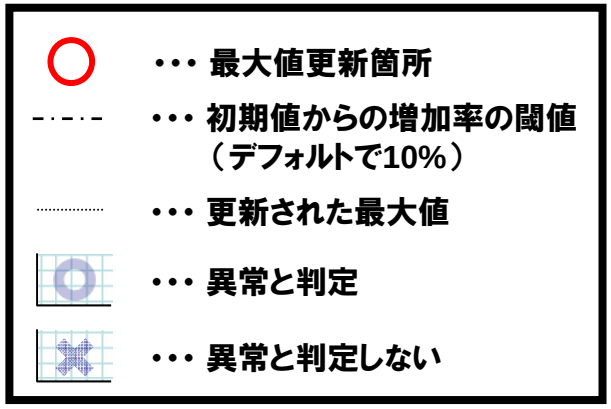
オープンファイル数は一定量を超えない範囲内で増減しているため、更新回数はカウントされません。
そのため、ファイルリークとは判定しません。



●オープンファイル数が経過時間と共に増減を繰り返し、更新回数の上限値を超えた

オープンファイル数の最大値が規定回数を超えて更新されているため、ファイルリークと判定します。



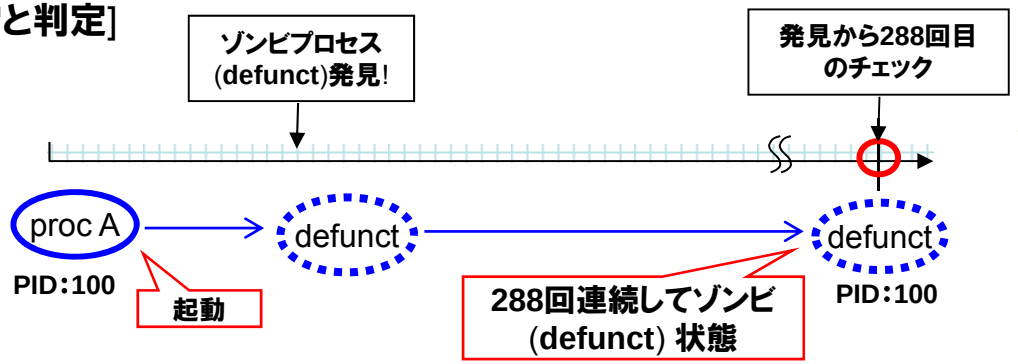


※ 前ページのメモリリークについても、同様です。

(続き)

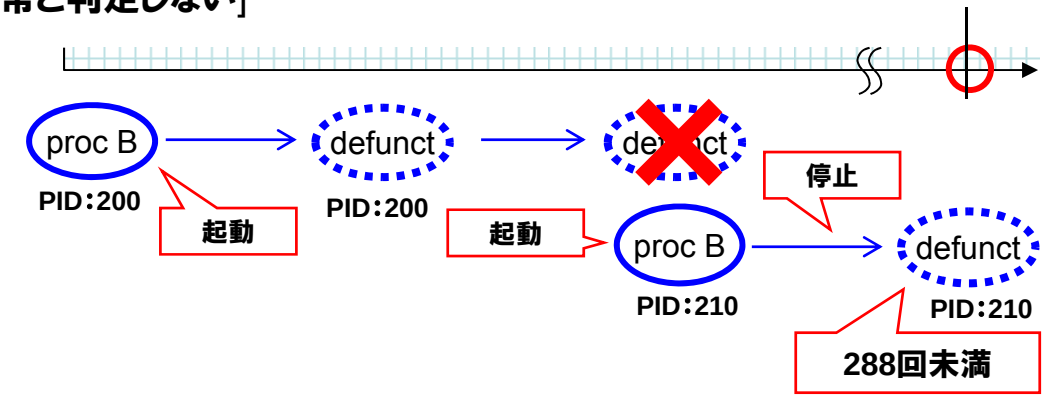
ゾンビプロセス(defunct)の検出パターンは以下のようになります。

[異常と判定]



同じPIDのプロセスが規定回数(デフォルトでは 288 回=1 日)以上、連続してゾンビ状態(defunct)であった場合に、ゾンビプロセス(defunct)と判定します。

[異常と判定しない]



ゾンビ状態(defunct)のプロセスが規定回数(デフォルトでは288 回=1 日)の範囲内で常に PID を変化するような場合は、ゾンビプロセス(defunct)と判定しません。

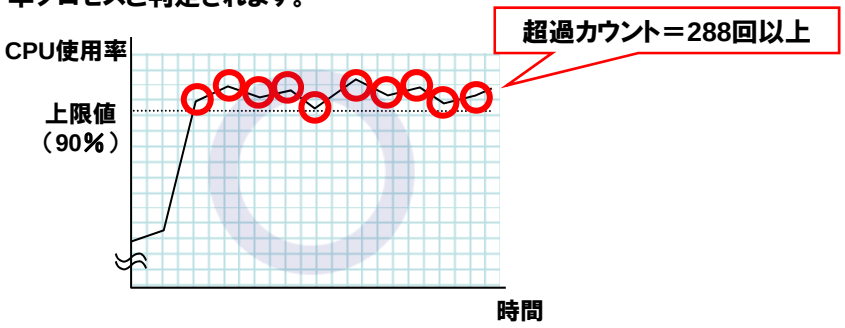
※ 常に起動・停止を繰り返すプロセスの場合、ゾンビプロセス(defunct)と判定しません。
停止時にゾンビプロセス(defunct)となるが、しばらくすると消滅するようなプロセスの場合はゾンビプロセス(defunct)と判定しません。

(続き)

高CPU使用率プロセスの検出パターンは以下のようになります。

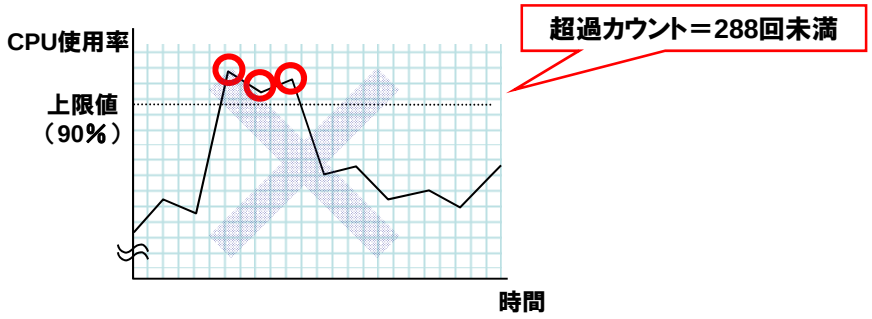
●CPU使用率が経過時間と共にCPU使用率上限値を超え続け、閾値を超えた

CPU使用率上限値を毎回超えているため、上限値超過回数は毎回カウントされ、規定回数(288回)を超えたため高CPU使用率プロセスと判定されます。



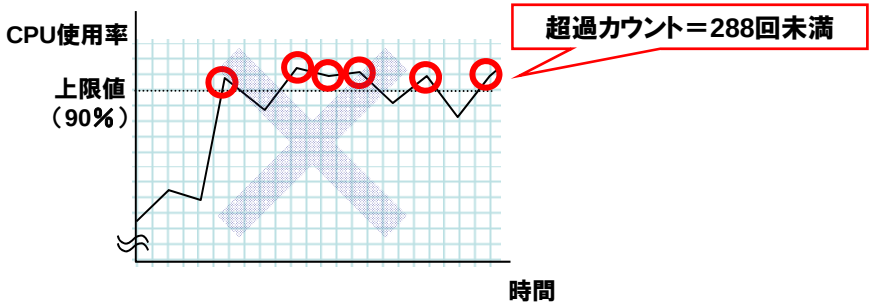
●CPU使用率が一時的にCPU使用率上限値を超え、その後上限値を下回って推移

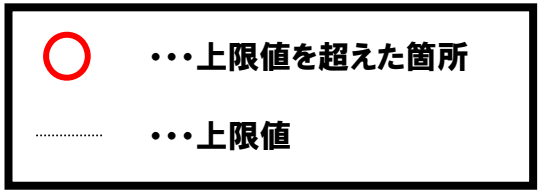
一時的にCPU使用率上限値を超えるが、その後規定回数(288回)を超える前に上限値以下で推移しているため、高CPU使用率プロセスとは判定しません。



●CPU使用率が経過時間と共に、CPU使用率上限値の前後で増減している

CPU使用率が一度でも上限値を下回った場合、上限値超過回数は0にリセットされるため、規定回数(288回)を超えない。
そのため、高CPU使用率プロセスとは判定しません。





システム全体のリソース監視異常を検出する条件

本製品では、「使用率上限値」、「連続超過時間」という2つのパラメータを組み合わせて検出を行います。

総プロセス数／ユーザプロセス数／総スレッド数／総メモリ量／総スワップメモリ量／総オープンファイル数／
総ロックファイル数／CPU使用率　・・・　使用率上限値を連続超過時間、連続して超えた場合に検出します。

［パラメータのデフォルトの閾値］

使用率上限値	90%
連続超過時間	60分

常にシステム全体のリソースが使用率上限値を超えている場合、
約60分後にこれらの異常を検出します。

［デフォルトで運用した場合の検出パターン］

連続超過時間	オープンファイル数上限監視	プロセス数上限監視	ロックファイル数上限監視	ユーザプロセス数上限監視	スレッド数上限監視	メモリ量上限監視	CPU使用率上限監視	スワップメモリ量上限監視
60分未満	×	×	×	×	×	×	×	×
60分以上	○	○	○	○	○	○	○	○

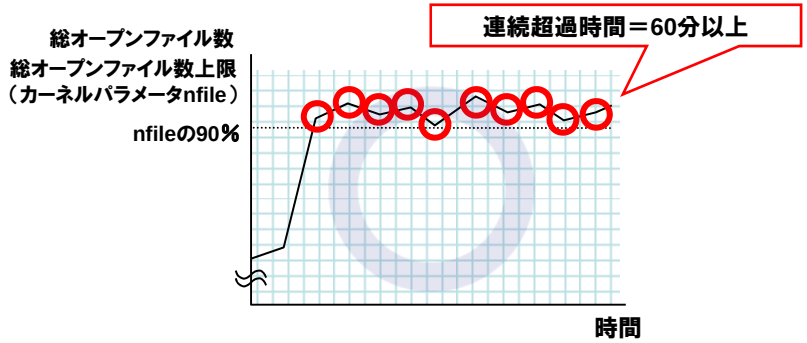
※ ○ ・・・ 異常を検出します。　　× ・・・ 異常を検出しません。

システム全体のリソース監視異常検出のパターン

システム全体のリソース監視の検出パターンは以下のようになります。（以下はオープンファイル数上限監視の例となります）

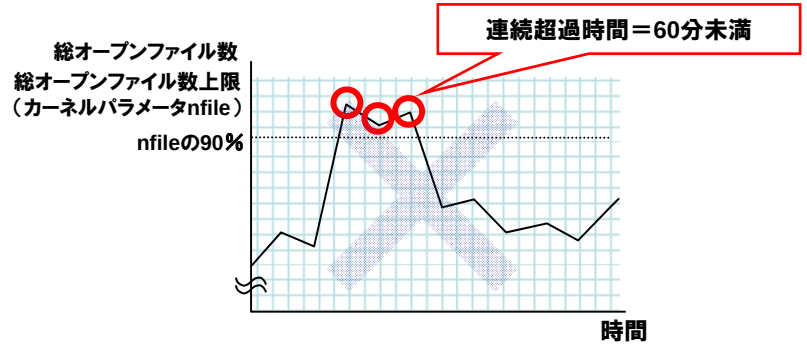
●総オープンファイル数が連続超過時間、連続して使用率上限値以上であった

総オープンファイル数が毎回上限値以上であり、連続超過時間（60分）を超えたため、総オープンファイル数上限異常と判定します。



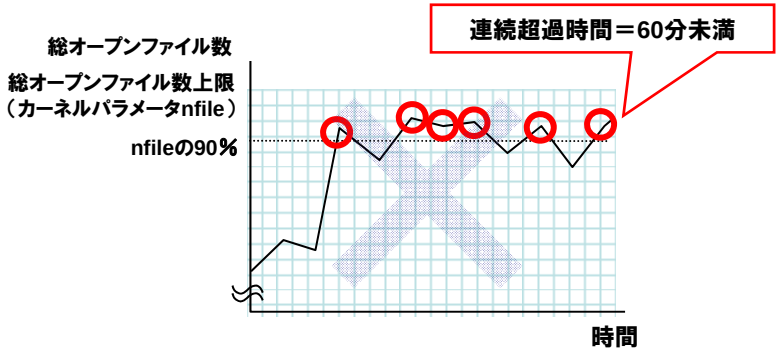
●総オープンファイル数が一時的に上限値を超え、その後上限値を下回って推移

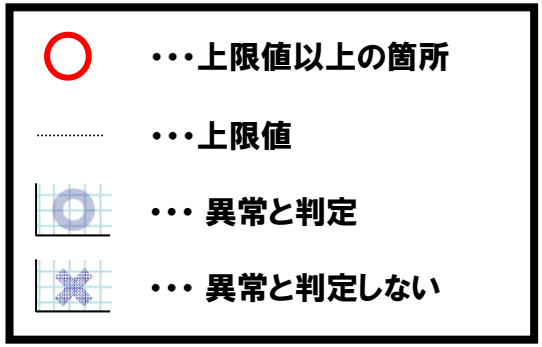
一時的に総オープンファイル数が上限値を超えるが、その後連続超過時間（60分）を超える前に上限値以下で推移しているため、総オープンファイル数上限異常とは判定しません。



●総オープンファイル数が連続超過時間中に、使用率上限値の前後で増減している

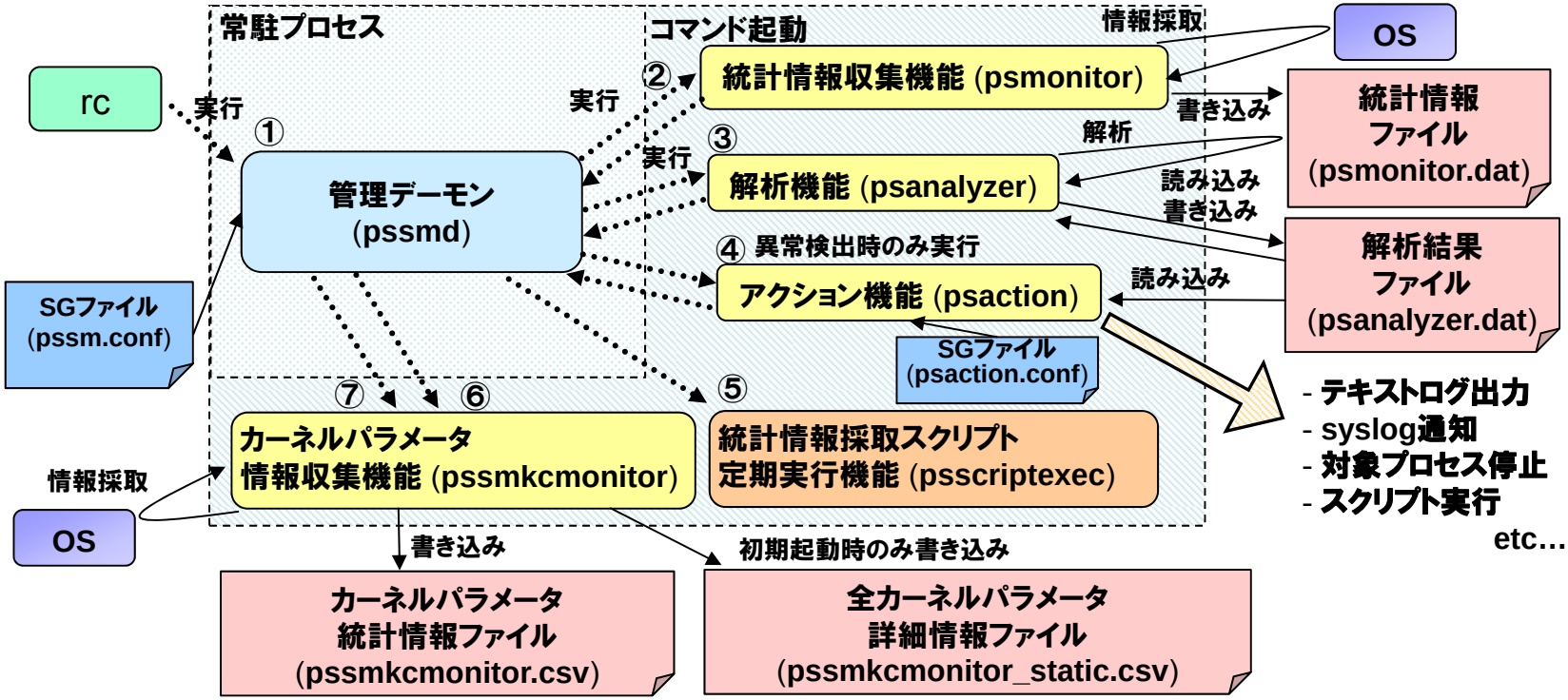
総オープンファイル数が一度でも使用率上限値を下回った場合、リセットされるため、連続超過時間（60分）を超えない。
そのため、総オープンファイル数上限異常とは判定しません。





製品構成の概要

① 管理デーモンプロセス(pssmd)を起動します。
② pssmdデーモンが一定間隔でpsmonitorコマンドを実行し、プロセスの利用するリソースの統計情報を継続的に収集し、統計情報ファイル(psmonitor.dat)に書き込みます。
③ pssmdデーモンがpsanalyzerコマンドを実行し、収集した統計情報を一定間隔で解析し、解析結果を解析結果ファイル(psanalyzer.dat)へ書き込みます。
④ 解析の結果、異常と判断するとpsactionコマンドを実行し、syslog通知および対象プロセス停止等の処理が実行されます。
⑤ 統計情報採取用の任意のスクリプトを定期的に実行することができます。
⑥ pssmdデーモンが起動時にpssmkcmonitorコマンドを実行し、全カーネルパラメータの詳細情報を一度だけ収集し、全カーネルパラメータ詳細情報ファイル(pssmkcmonitor_static.csv)に書き込みます。
⑦ pssmdデーモンが一定間隔でpssmkcmonitorコマンドを実行し、カーネルパラメータの統計情報を継続的に収集し、カーネルパラメータ統計情報ファイル(pssmkcmonitor.csv)に書き込みます。



Serviceguard連携

Serviceguardと連携することで、プロセスリソースの障害時にクラスタシステムでのソフトウェアの可用性を向上させます。

SystemResourceMonitorのServiceguard連携は、ProcessSaverを利用して行うか、
またはダミープロセスを使用して行います。
SystemResourceMonitorのリソース監視機能によって、特定のプロセスの異常を検出するとプロセス停止、
さらには、ProcessSaverと連携してプロセスの再起動を行います。
これにより、無用なパッケージ切り替えやノード切り替えを防止します。
プロセス再開後にも障害が継続的に発生する場合は、Serviceguard連携により、適切なパッケージ
切り替えやノード切り替えを行い、業務を継続することが可能です。

Serviceguard連携が可能なリソース監視

- メモリリーク監視
- ファイルリーク監視
- プロセス毎のオープンファイル数上限監視
- 高CPU使用率プロセス監視
- スレッドリーク監視
- プロセス毎のスレッド数上限監視

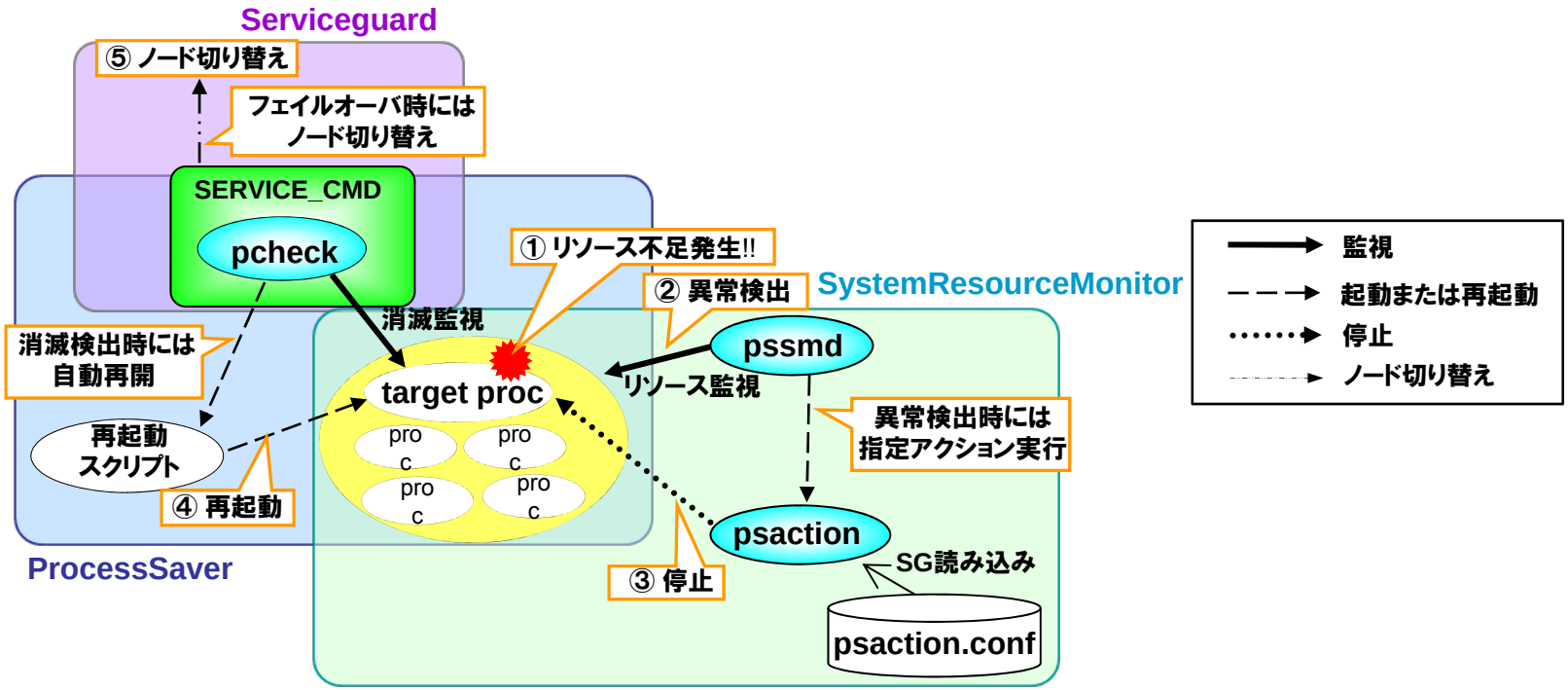
★ SystemResourceMonitorの
プロセスKILLアクションが指定されている
リソース監視について連携が可能です。

※ プロセスKILLアクションは、異常と判定されたプロセスを
kill(2)システムコールによって停止します。
そのため、defunctプロセスや、initプロセス等、
kill(2)システムコールでは停止できないプロセスを
プロセスKILLアクションによって停止することは
できません。

(続き)

ProcessSaverを介したServiceguard連携は以下のような構成で動作します。

SystemResourceMonitorのServiceguard連携は、ProcessSaverを利用して実現します。
① ターゲットプロセスが原因となるリソース不足が発生。
② SystemResourceMonitorの監視により、ターゲットプロセスの異常を検出します。
③ SystemResourceMonitorが検出したターゲットプロセスを停止します。
④ ProcessSaverによってターゲットプロセスの停止を検知し、再起動を行います。
⑤ 自系再開を試みてもシステムが復旧しない場合は、pcheck(1M)が停止します。
⑥ Serviceguardがpcheck(1M)の停止を検知し、パッケージ移動やノード切り替えを行います。



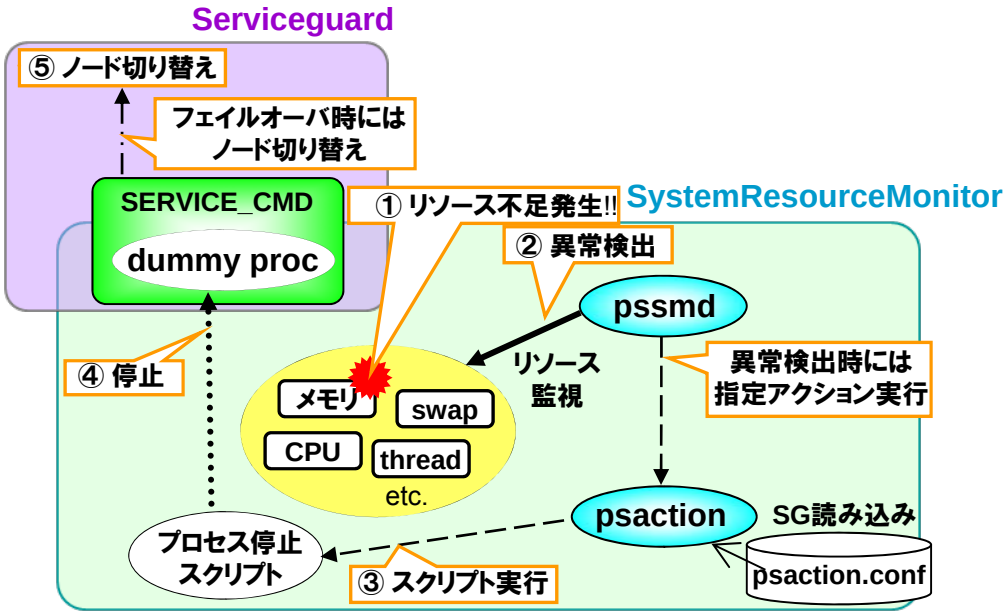
(続き)

ProcessSaverを介さず、直接Serviceguardとの連携を行う場合は以下のような構成となります。

システムリソース監視のServiceguard連携を行う場合に有効です。システムリソース監視はプロセスKILLアクションの
対象外であるため、スクリプト実行アクションを使用してServiceguardとの連携を実現します。

ターゲットプロセスを、Serviceguardのサービスとして指定します。
（システムリソース監視のように、プロセスの特定が困難な場合は、ダミープロセスを用意し、ターゲットプロセスとします。）
SystemResourceMonitorのスクリプト実行アクションに、ダミープロセスを停止するスクリプトを指定します。
SystemResourceMonitorによって異常が検出された場合には、スクリプト実行アクションによって、
ダミープロセスが停止され、Serviceguardがサービスの停止を検知し、パッケージの切り替え等を行います。

① システムリソースの枯渇が発生。
② SystemResourceMonitorの監視により、システムリソースの異常を検知します。
③ SystemResourceMonitorが、システムリソース監視異常検出時のアクションに指定されているスクリプトを実行します。
④ ③で実行されたスクリプトによって、ダミープロセスを停止します。
⑤ Serviceguardがターゲットプロセスの停止を検知し、パッケージ移動やノード切り替えを行います。



統計情報ファイルの利用方法

統計情報ファイル(psmonitor.dat)は、指定時間(デフォルトでは5分)間隔でリソースの統計情報を記録します。
統計情報ファイルから以下のように障害の解析、性能分析を行うことができます。

記録される情報

[プロセス個別情報]
- ユーザ名 （UNAME）
- プロセスID （PID）
- 親プロセスID （PPID）
- プロセスグループID （PGID）
- プロセスステータス （STATUS）
- メモリ使用量 （MEMORY）
- オープンファイル数 （FILE）
- ロックファイル数 （LOCKF）
- CPU使用率 （CPU)
- CPU使用時間 （CPUTIME）
- 同名プロセス数 （PNUM)
- スレッド(LWP)数 （THREAD)
- プロセス名 （COMMAND）

[CPU 個別情報]
- CPU番号
- 収集時点から過去1分間の平均の run queue の長さ
- 収集時点から過去5分間の平均の run queue の長さ
- 収集時点から過去15分間の平均の run queue の長さ
- exec()システムコールの回数

[システム情報]
- ホスト名
- 情報取得日時
- 総情報取得回数
- 現在の総プロセス数 / 総プロセス数上限
- 現在の総メモリ量 / 総メモリ量上限
- 現在の総オープンファイル数 / 総オープンファイル数上限
- 現在の総ロックファイル数 / 総ロックファイル数上限
- 現在の総スレッド(LWP)数 / 総スレッド(LWP)数上限
- 現在のCPU使用率 / CPU使用率上限
- コンテキストスイッチの回数
- 現在の総スワップメモリ使用量 / 現在の総スワップメモリ予約量 / 総スワップメモリ量上限
- 現在のユーザごとの起動プロセス数最大値 / ユーザごとの起動プロセス数上限(ユーザ名)

障害の解析

障害が発生した場合に、各プロセスのステータス、メモリ使用量などのプロセス個別情報から、どのプロセスがいつまで起動していたか、どのような状態であったかなどの解析を容易に行うことができます。

性能分析

プロセス・システムリソース情報の履歴を確認することで障害の原因となったプロセスや、どのリソースが枯渇したのか、どのような形で不具合が起こったのかなどの解析を容易に行うことができます。

(続き)

以下のような事例の場合に、統計情報ファイルを利用して詳細な解析を行うことができます。

事例1: ある監視系PPが突然、動作不能に…

ある監視系PPが動作不能になったのでsyslogを確認したところ、SystemResourceMonitorからメッセージが出力されており、プロセスが消滅していた。メッセージ内容に基づき、調査のために統計情報ファイルを参照した。消滅前後の時間帯を見るとカーネルパラメータの上限に達した状態がしばらく続いていることがわかったため、これが消滅の原因と断定。プロセスの再起動とカーネルパラメータの上限値UP、リソースを無駄に消費している原因への対策を行った。

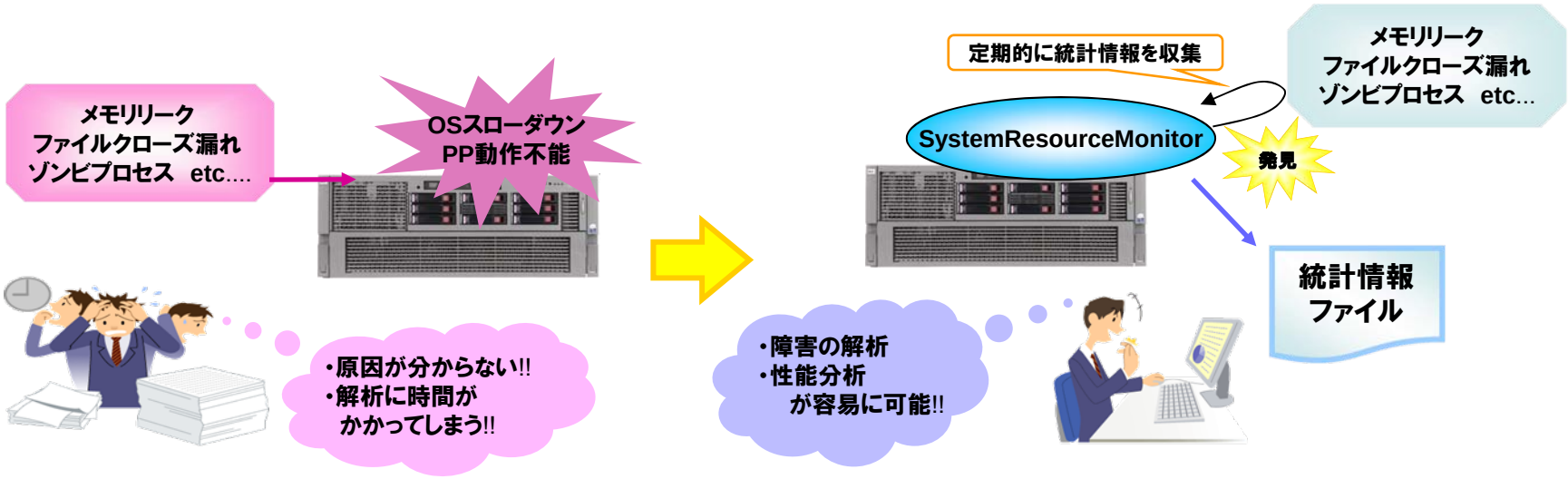
事例2: Webシステムが突然スローダウンし、ブラウザのレスポンスが遅くなった

調査のためにシステム情報を見ると、使用メモリ量が物理メモリ量の上限近くに達していた。
また、プロセス個別情報で各プロセスのメモリ使用量をチェックすると、あるAPサーバによって起動されるjavaプロセスが一定量のメモリ(ヒープ領域)を確保していたことが判明したので、物理メモリの追加などのシステム要件の見直しを行い、慢性的なメモリ不足を解消し、Webシステムのスループットを高めることができた。

(続き)

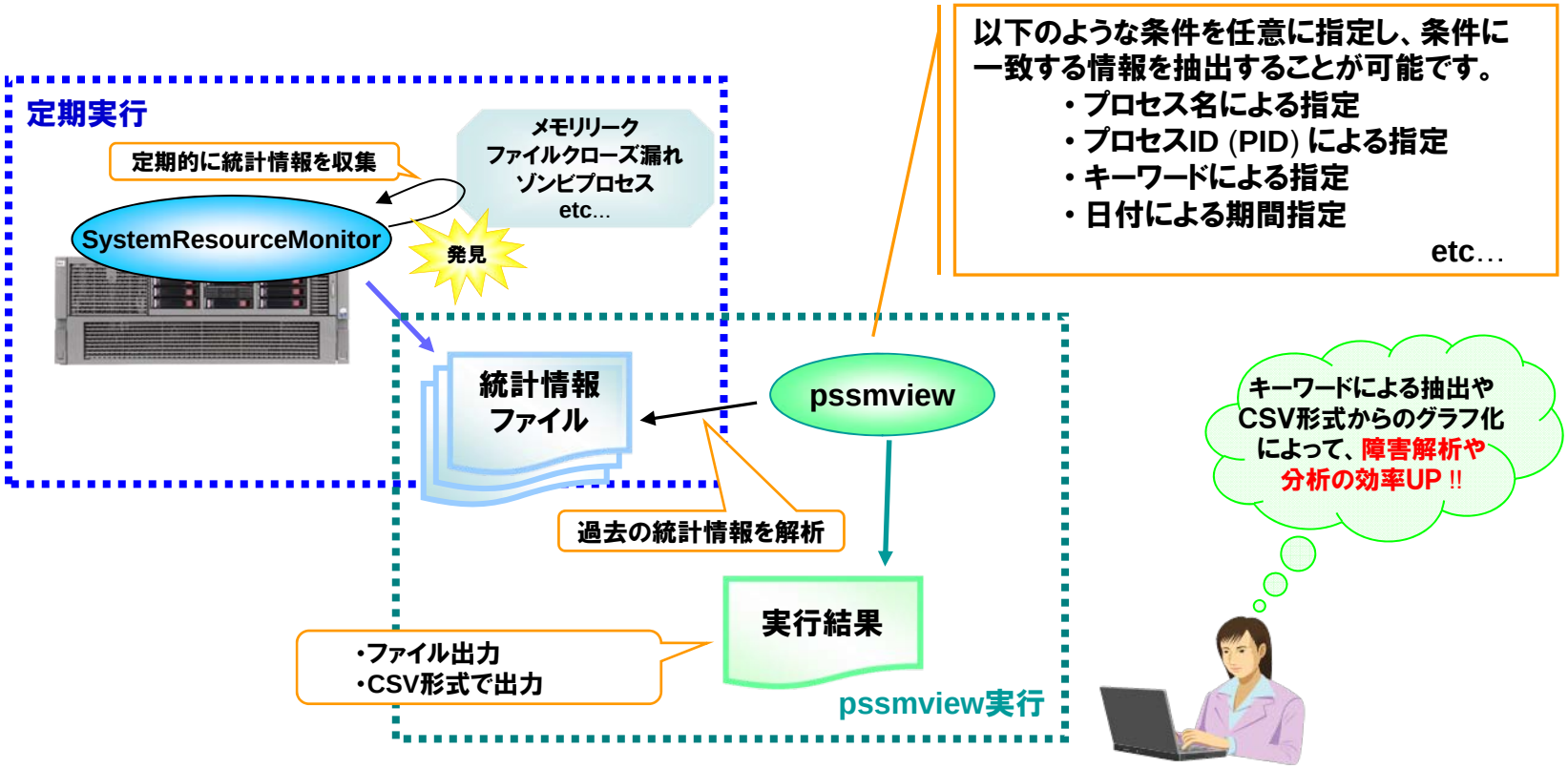
事例3: カーネルパラメータに設定された上限値に近づいた

オープンファイル数が設定されていた閾値を超えたことを示す本製品のメッセージがsyslogに出力されていた。
ただちに統計情報ファイルを確認すると、システム全体のオープンファイル数が閾値を超えていることが確認された。
さらにプロセスの個別情報から、あるプロセスが起動以降オープンファイル数を増やし続けていたことが分かった。
そのプロセスの仕様上、ファイルをオープンし続けることは無いため、プロセスの不具合であることが判明。
動作不能に陥る前にプロセスの再起動など暫定対処を行い、最悪の事態を回避することができた。
後日不具合修正版をリリースし、恒久対処を行った。



統計情報ファイル編集コマンドについて

統計情報ファイルを解析するための支援ユーティリティとして、統計情報ファイル編集コマンド（以降、pssmviewと記す）を提供しております。
pssmviewは、HA/ SystemResourceMonitorで収集した統計情報ファイルから必要な情報を抜き出し、整形・編集することで、問題解決の支援、一般の表計算ソフトによるグラフ化等が可能になります。
日々のリソース使用状況の確認や、障害解析時に有効です。



(続き)

統計情報ファイルのイメージ

/var/opt/HA/PSSM/log/psmonitor.dat

```
================================================================================================================
HOST    : host1
DATE    : 2008/06/01 14:23:53
COUNT : 616
INDEX   :
 PROCESS UNAME:16 PID:6 PPID:6 PGID:6 STATUS:8 MEMORY:10 FILE:6 LOCKF:6 %FILE: 6 %CPU:7 •••
 SYSTEM TOTAL_PROC_NUM:8 PROC_NUM_MAX:8 TOTAL_PROC_MEM:8 TOTAL_MEM_MAX:8 •••
----------------------------------------------------------------------------------------------------------------
TYPE UNAME PID PPID PGID STATUS MEMORY(KB) FILE LOCKF %FILE %CPU %CPU/2 CPUTIME PNUM THREAD %THREAD (COMMAND)
PROCESS  root  0 0 0 SLEEP 64 0 0 0.00 0.02 0.01 0:57 1 1 0.08 (swapper)
PROCESS  root  1 0 0 RUNNING 544 0 0 0.00 0.05 0.03 1:36 1 2 0.17 (init)
PROCESS  root  2 0 0 SLEEP 64 0 0 0.00 0.02 0.01 6:17 1 1 0.08 (vhand)
                    :                                    :                                    :
                    :                                    :                                    :
                    :                                    :                                    :

PROCESS  root   9157   9156   3  SLEEP   368  0   0  0.00   0.02   0.01   4   1    0.08   -sh
PROCESS  root   9172   9157   3  SLEEP  4072  5   0  0.12  0.02   0.01   3    1   0.08   tcsh
PROCESS  root   9742   3881   0  RUNNING    472  1   0  0.02  0.00  0.00    1   1   0.08   /opt/HA/PSSM/bin/psmonitor -x -o
/var/opt/HA/PSSM/log/psmonitor.dat -f 10 -n 6 -c 1126
----------------------------------------------------------------------------------------------------------------
TYPE PROCESS MEMORY(MB) FILE LOCKFILE CPU(%) CSWITCH THREAD SWAP(KB) MAXUPROC
SYSTEM 166 4200 1125 2018 914 NOVALUE 24 4096 8.18 200 55501309 822 8416 103964 726580 4194304 8 256 sfmdb
----------------------------------------------------------------------------------------------------------------
TYPE CPUID RUNQ_1MIN RUNQ_5MIN RUNQ_15MIN SYSCALL SUBINFO1 SUBINFO2
CPU 0 0.0349 0.0238 0.0206 78466 4153 2786
:                :                :
:                :                :
CPU 4 0.0174 0.0067 0.0050 95562 36346 34071
================================================================================================================
```

ヘッダ部

プロセス個別情報部

システム全体情報部

CPU個別情報部

(続き)

pssmviewのファイル出力イメージ

#/opt/HA/PSSM/bin/pssmview -i /tmp/psmonitor.dat -u /tmp/pssmview.dat

```
### ANALYZE INFORMATION ###
HOST    : host1
DATE    : 2008/05/23,08:26:29,2008/05/23,12:10:50
COUNT : 1-38

### ERROR PROCCESS ###
# OPEN FILE LEAK PROCESS #
-------------------------------------------------------------------------------------------------------------------------
    PID   OPEN FILE                                COMMAND
            START       END       MAX
-------------------------------------------------------------------------------------------------------------------------
  25453        10       330       330            procA
-------------------------------------------------------------------------------------------------------------------------

### HISTORY ###
=========================================================================================================================
UNAME  PID  PPID  PGID  STATUS MEMORY(KB)  FILE  LOCKF  %FILE  %CPU %CPU/2 CPUTIME  PNUM THREAD %THREAD COMMAND
-------------------------------------------------------------------------------------------------------------------------
[1] 2008/05/23 08:26:29
   root    0     0     0   SLEEP        64     0      0   0.00  0.02  0.01   90:27     1      1    0.03 swapper
   root    1     0     0 RUNNING       548     0      0   0.00  0.04  0.02    0:10     1      2    0.07 init
   root    2     0     0   SLEEP        64     0      0   0.00  0.02  0.01    1:55     1      1    0.03 vhand
   root    3     0     0   SLEEP        64     0      0   0.00  0.02  0.01    1:17     1      1    0.03 statdaemon
   root    4     0     0   SLEEP        64     0      0   0.00  0.02  0.01    0:04     1      1    0.03 unhashdaemon
   root    8     0     0 RUNNING       384     0      0   0.00  0.00  0.00    0:00     1      2    0.07 ioconfigd
   root    9     0     0 RUNNING       768     0      0   0.00  0.00  0.00    0:00     1     12    0.40 ObjectThreadPool
   root   10     0     0 RUNNING       384     0      0   0.00  0.02  0.01    2:36     1      2    0.07 nfsktcpd
             :                          :                                                              :
```

ヘッダ部

エラープロセス情報部

プロセス個別情報部

(続き)

pssmviewのファイル出力イメージ（プロセス名指定）

#/opt/HA/PSSM/bin/pssmview -n proc1 -i /tmp/psmonitor.dat -u /tmp/pssmview.dat

```
### ANALYZE INFORMATION ###
HOST    : host1
DATE    : 2008/12/03,14:51:58,2008/12/03,17:10:50
COUNT : 1-40

### ERROR PROCCESS ###
# OPEN FILE LEAK PROCESS #
-------------------------------------------------------------------------------------------------------------------
   PID   OPEN FILE                                COMMAND
         START       END       MAX
-------------------------------------------------------------------------------------------------------------------
  25453       10       330       330            proc1 –i 60 10
-------------------------------------------------------------------------------------------------------------------

### HISTORY ###
===================================================================================================================
 UNAME  PID  PPID  PGID  STATUS MEMORY(KB)  FILE  LOCKF  %FILE  %CPU %CPU/2 CPUTIME  PNUM THREAD %THREAD COMMAND
-------------------------------------------------------------------------------------------------------------------
[1] 2008/12/03 14:51:58
-------------------------------------------------------------------------------------------------------------------
[2] 2008/12/03 14:53:02
  root  25453  10264     3   SLEEP          252      10        0     0.24     0.00     0.00        1:54        1          1        0.03  proc1 –i 60 10
-------------------------------------------------------------------------------------------------------------------
[3] 2008/12/03 14:54:04
  root  25453  10264     3   SLEEP          252      20        0     0.49     0.00     0.00        2:36        1          1        0.03  proc1 –i 60 10
                :                              :                                  :
                :                              :                                  :
```

ヘッダ部

エラープロセス情報部

指定プロセス情報部

(続き)

pssmviewのファイル出力イメージ （CSV形式）

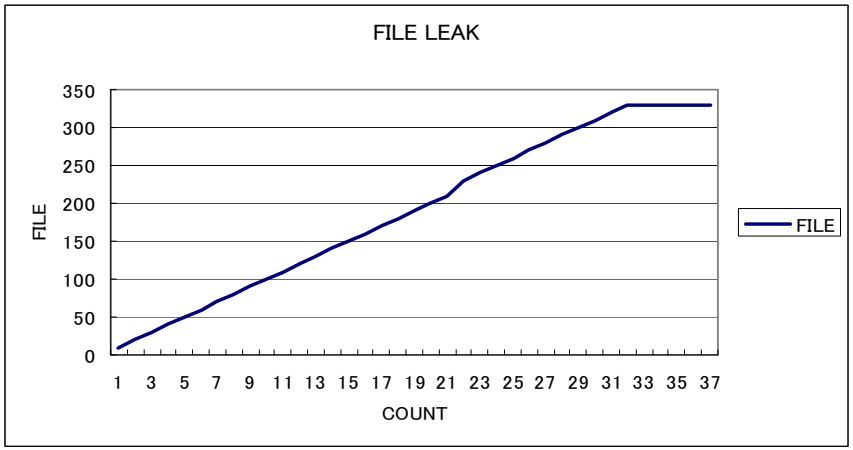
```
# /opt/HA/PSSM/bin/pssmview -o /tmp/psmonitor.csv -n proc2
```

```
host1
2007/12/03,14:53:02,2007/12/03,18:10:50
1,38
COUNT,DATE,TIME,UNAME,PID,PPID,PGID,STATUS,MEMORY,FILE,LOCKF,%FILE,%CPU,%CPU/2,CPUTIME, PNUM,
THREAD,%THREAD,COMMAND
2,2007/12/03,14:53:02,root,25453,10264,3,SLEEP,252,10,0,0.24,0.00,0.00,1:54,1,1,0.03,proc2
3,2007/12/03,14:54:04,root,25453,10264,3,SLEEP,252,20,0,0.49,0.00,0.00,2:36,1,1,0.03,proc2
4,2007/12/03,14:55:05,root,25453,10264,3,SLEEP,252,30,0,0.73,0.00,0.00,2:50,1,1,0.03,proc2
5,2007/12/03,14:56:07,root,25453,10264,3,SLEEP,252,40,0,0.98,0.00,0.00,3:00,1,1,0.03,proc2
6,2007/12/03,14:57:08,root,25453,10264,3,SLEEP,252,50,0,1.22,0.00,0.00,3:05,1,1,0.03,proc2
7,2007/12/03,14:58:10,root,25453,10264,3,SLEEP,252,60,0,1.46,0.00,0.00,3:18,1,1,0.03,proc2
8,2007/12/03,14:59:11,root,25453,10264,3,SLEEP,252,70,0,1.71,0.00,0.00,3:48,1,1,0.03,proc2
9,2007/12/03,15:00:14,root,25453,10264,3,SLEEP,252,80,0,1.95,0.00,0.00,3:55,1,1,0.03,proc2
        :                                   :                                   :
```

ヘッダ部

指定プロセス情報部

表計算ソフトを使用して、グラフ化も可能です。（右図参照）



システム要件

■ HP-UX版

対応機種	NX7700iシリーズ
対応OS	HP-UX 11i v2, HP-UX 11i v3
使用ディスク容量	480.0MB以上（デフォルト）
使用メモリ容量	1.5MB以上

製品価格

■ HP-UX版　NX7700i/5000 シリーズ

HA/SystemResourceMonitor R4.2（2011.10 リリース）					
	型番	製品名	価格（円）	月額保守料金（円）	ライセンス体系
24H対応SWASバンドル（※1）	UQ5218-E00402	HA/SystemResourceMonitor R4.2	236,000	3,000	コア
SWASバンドル（※2）	UQ5218-H00402	HA/SystemResourceMonitor R4.2	230,000	2,600	コア
SWLSバンドル（※3）	UQ5218-G00402	HA/SystemResourceMonitor R4.2	220,000	1,600	コア
メディア	UQ5218-00402M	HA/SystemResourceMonitor メディア	10,000	ー	ー

■ HP-UX版　NX7700i/7000 シリーズ

HA/SystemResourceMonitor (1socket) R4.2（2011.10 リリース）					
	型番	製品名	価格（円）	月額保守料金（円）	ライセンス体系
24H対応SWASバンドル（※1）	UQ5218-E0040T2	HA/SystemResourceMonitor (1socket) R4.2	590,000	7,500	CPU
SWASバンドル（※2）	UQ5218-H0040T2	HA/SystemResourceMonitor (1socket) R4.2	575,000	6,500	CPU
SWLSバンドル（※3）	UQ5218-G0040T2	HA/SystemResourceMonitor (1socket) R4.2	550,000	4,000	CPU
メディア	UQ5218-00402M	HA/SystemResourceMonitor メディア	10,000	ー	ー

（注）SystemResourceMonitor をご利用いただくためには、SystemResourceMonitor のライセンスとメディアが必要です。
SystemResourceMonitor のライセンスとメディアを併せてお求めください。

※1 24H対応SWASバンドル
SWASを、24時間365日ご利用いただけます。
※2 SWAS（ソフトウェアアシストサービス）バンドル
SWLS に加え、弊社技術者がお電話により問題処理を含め各種問い合わせにご回答させていただくサービスがご利用いただけます。
※3 SWLS（ソフトウェアライセンスサービス）バンドル
ご購入いただいたプログラム・プロダクトのバージョンアップ、リビジョンアップ媒体をご利用いただけます。

製品価格は、出荷バージョン等により変更になる可能性があります。
詳細は、弊社営業部門またはサポート部門にお問い合わせください。

商標

- HP-UX、Serviceguardは、米国における米国Hewlett-Packard Companyの登録商標です
- CLUSTERPROは、日本電気株式会社の登録商標です
- 記載の製品名および会社名はすべて各社の商標または登録商標です

Empowered by Innovation