



ジェニファー基本活用法
Version 1.0

2007年08月01日



はじめに

本書は、新しい概念と視点(viewpoint)によるAPM製品として多くの開発者およびシステム運用者から関心をお寄せいただいているジェニファーの基本的な活用方法を記したものです。

ジェニファー(Jennifer)は既存のAPM製品とは異なり、システム障害及び状況に対して直感的で分かりやすい情報提供能力を持っていますが、ツール(Tool)としてはユーザにとって使いにくい部分もあります。

本書は、初めてジェニファーをお使いになる方や、ジェニファーで提供される情報がどのように使えるのかよく分からない方のためのものです。ジェニファーのインストール後、本書をお読み頂いて使うとより分かりやすく効果的にジェニファーを活用していただけるでしょう。

なお、本文書は細かいオプションについての説明や、ジェニファーが持つ全ての機能のマニュアルではありません。

- ▶ リアルタイム モニタリング
 - ・ システムのサービス能力モニタリング
同時端末ユーザ/tps/応答時間
 - ・ システムリソース パラメータ モニタリング
 - システムサービス能力対比cpu使用量分析
 - システムサービス能力対比メモリ使用量分析
 - リアルタイム アプリケーション トランザクション モニタリング
- ▶ ヒストリデータ分析
 - ・ ユーザ/アプリケーション/スロープット/メモリ/CPUデータ
 - ・ アプリケーション性能分析(推移及び詳細分析)
- ▶ 障害要素分析
- ▶ 使用事例整理
(追加予定)

▶ リアルタイム モニタリング
 ・ システムサービス能力モニタリング

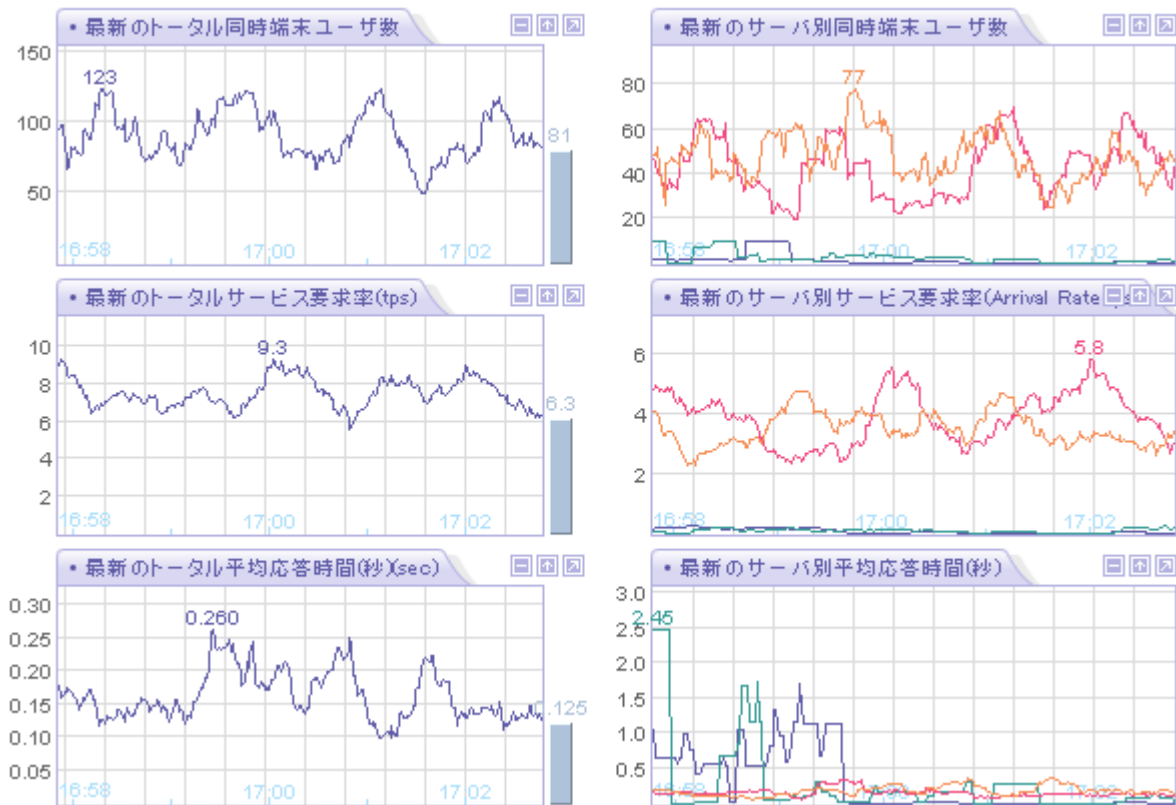
WASシステムを担当する全ての担当者に共通の疑問点の一つは、担当しているシステムがどの程度のサービス能力(service ability)を持っているのかということでしょう。その疑問点を解決するために、一般的ないくつかの方法がありますが、どれも充分なものではありません。

ジェニファーはこのような疑問に対してリアルタイム及び直感的で分かりやすい観点(viewpoint)の情報を提供することによって、今まで疑問であった様々な部分を解決します。

下記の図はジェニファー(jennifer)サーバ及びエージェントを正常に構成後、WASシステムがサービス処理するときサービス処理状況についてジェニファーが提供する基本画面です。この画面はジェニファーで提供する多様なリアルタイム情報を一目で把握できるようになっています。グラフデータに慣れるとジェニファー活用の半分は成功したと言えます。

これから [システムサービス能力]と関連したデータと用語を説明します。



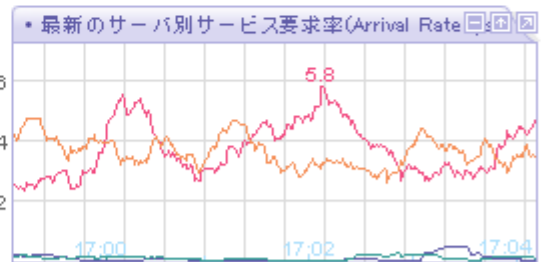
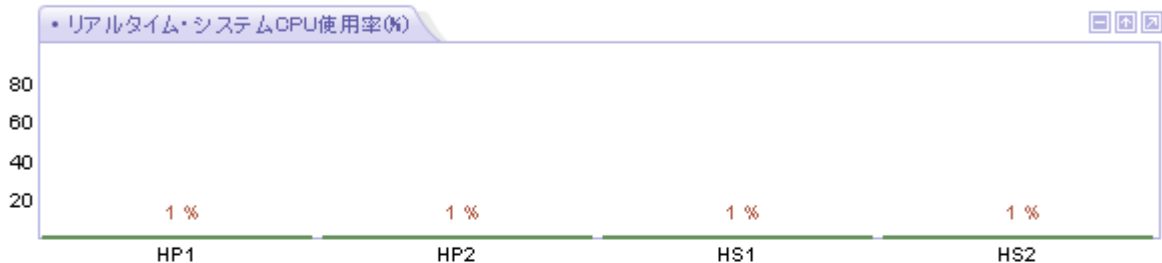
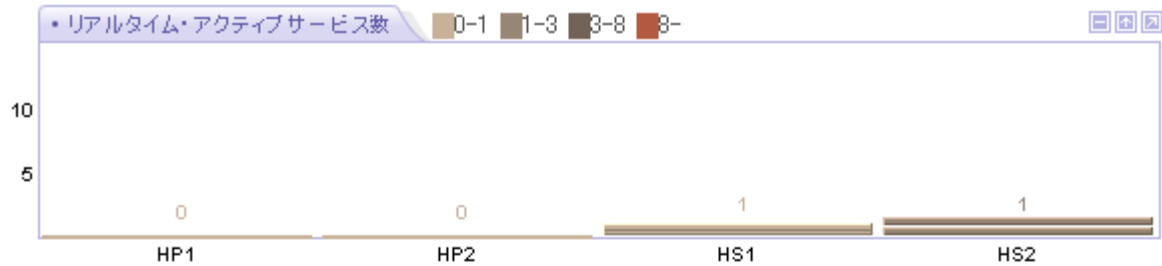


画面の真中にある上図の三つのグラフはシステムがサービスしている能力を測定できる **同時端末ユーザ数/tps/応答時間**をリアルタイムで表示します。

各グラフのX軸は時間の流れを表します。Y軸はサービス能力についての数字を表します。(該当データは1秒毎にアップデートします。)

図の左側は構成されたエージェントデータに対する全体(total)データを表し、右側の図は各エージェントでサービスしているデータを表します。

構成されたシステムが多い場合、各エージェント別データを区分できない場合がありますが、そのときはサーバ別グラフで特定線グラフを選択するか、これが多すぎて複雑な場合は画面上段の[リアルタイムアクティブサービス数], [リアルタイム CPU使用率] グラフで確認したいインスタンスを選択すると、該当インスタンスと関連したデータが反転して表示される為、区分しやすくなります。(下記図参照)

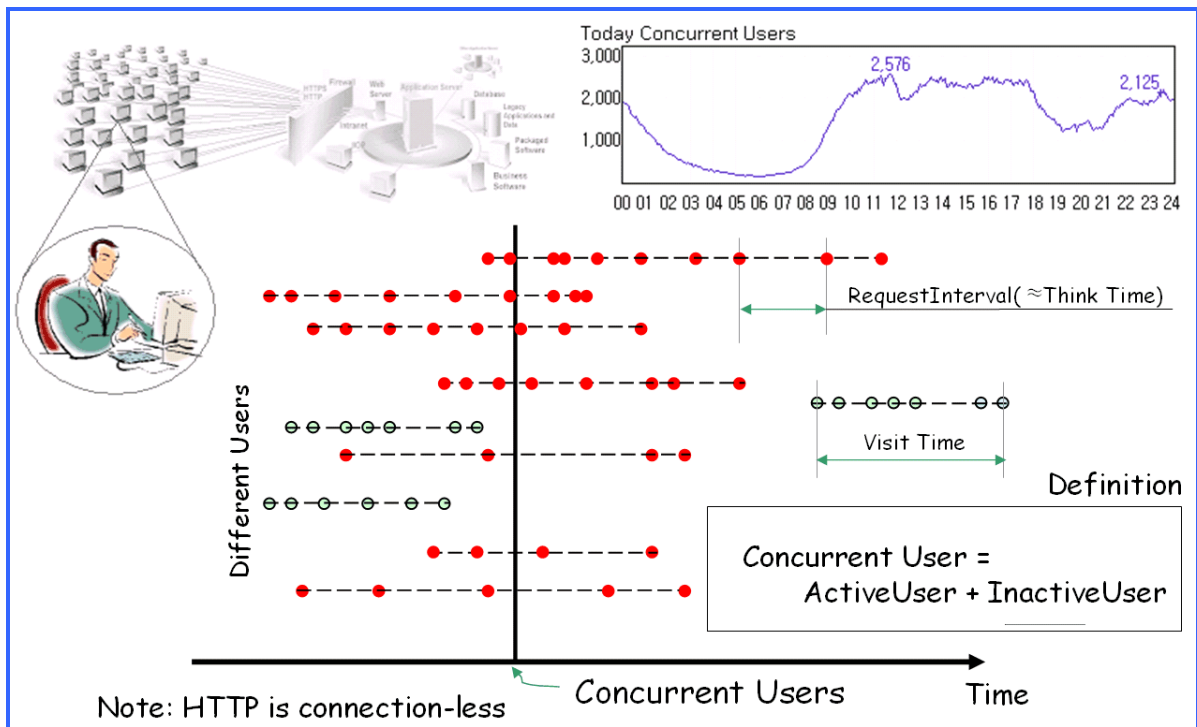


用語説明

同時端末ユーザ : 過去メインフレーム基盤のダミー(Dummy)3270エミュレータまたは、C/Sシステムの“同時接続者”数を求める事はそれほど難しくはありません。ユーザがシステムに接続するとユーザ当り TCP/IPコネクションが作られ、現在TCP/IPコネクションの数又は、IPアドレスを数える事で、“同時接続者”を測定するという方法です。しかし、HTTP(Hyper Text Transfer Protocol)プロトコル(Protocol)基盤のウェブシステムでは、TCP/IPプロトコルを使う点は同じですが、HTTPプロトコルはその属性として、リクエスト毎に新しいTCP/IPコネクションが作られるコネクションレス(Connection Less)という特性¹を持っているので、単純に

¹ 勿論、HTTP Connection Less特性はHTTP 1.1で記述されたKeep-Alive機能と関連がありますが、

コネクションされたTCP/IP数が現在該当システムを使うためにPCの前にいる“同時接続者”を意味するわけではありません。



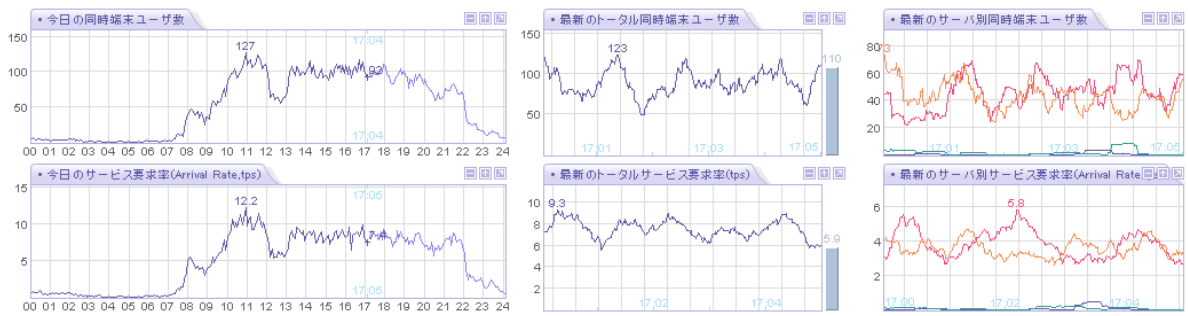
上図のように個別ユーザのクリックを一つの点で表現して見ると、各ユーザは個別リクエスト間隔(Request Interval)でクリックを繰り返します。ある特定の時点で、本システムを使用するためにPCを使っているユーザ数は何名でしょうか？ 上図では、7名が該当時点での“同時端末ユーザ数”です。このように、本システムでは、同時端末ユーザ数(Concurrent User)がPCを使っているユーザ数を定義します。

多くのエンジニアの間で、“同時ユーザ”という単語になじみがある反面、その用語の意味には混同がかなりあります。ある人は“同時ユーザ”という語を、“同時端末ユーザ”という意味に、別の人は“アクティブ(Active)ユーザ”という意味に解釈してしまいます。このため、本書では“同時ユーザ”，“同時接続者”という語を使わず、“同時端末ユーザ(Concurrent User)”という語を用いることにします。

TPS: 単位時間あたりに処理した件数をスロープット(Throughput)と呼びます。Throughputは実際には“処理率”と訳すべきですが、広い意味ですでに“スロープット”として通用しているため、本文書でも“スロープット”と表記します。)

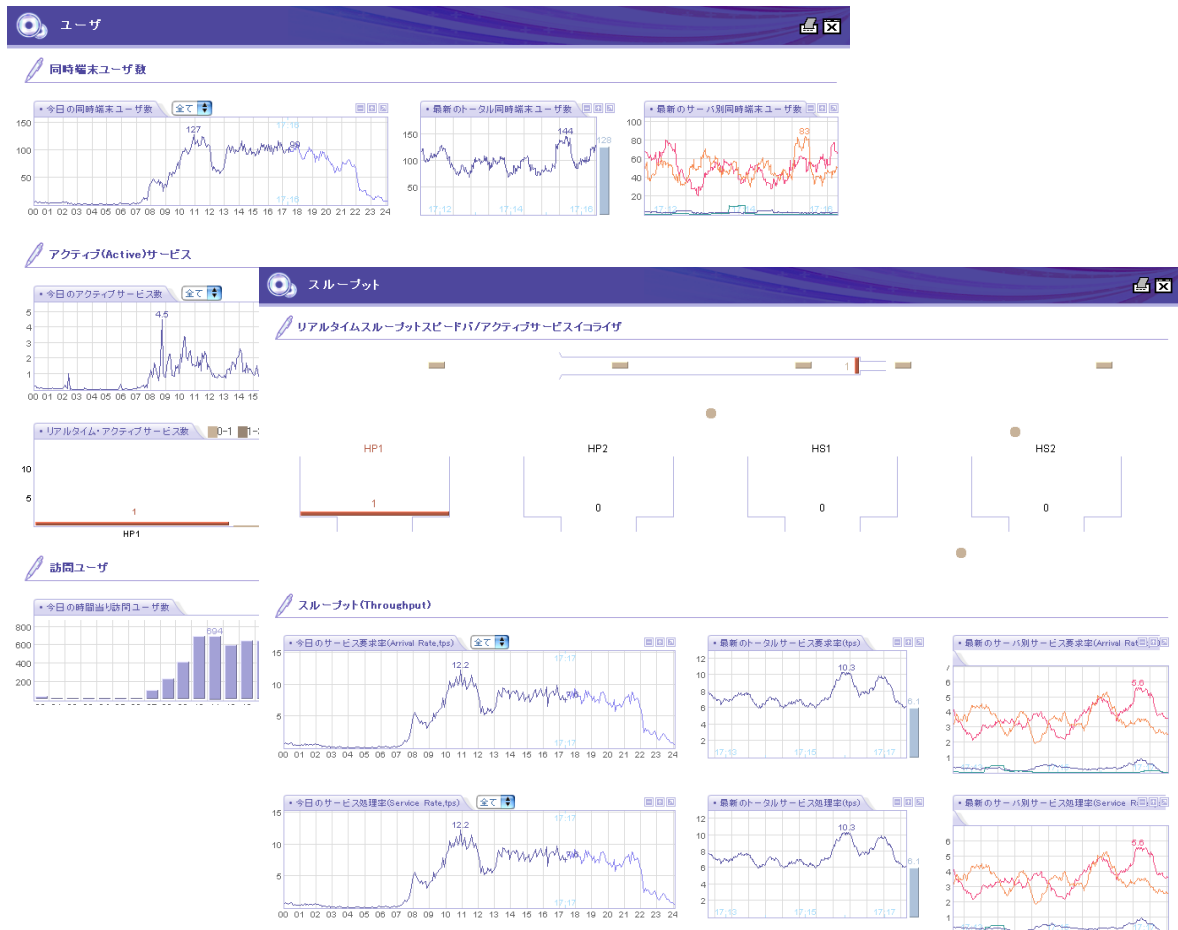
応答時間： 応答時間はリクエストが発生した時点から、リクエストによる最後のデータが全て到着した時点までの時間です。

ウェブサーバでそのKeep-Alive Timeout時間を指定することができます。Keep-Alive



上図左側は[本日同時端末ユーザ数]と[本日サービス依頼率]グラフを通して一日のシステムサービス能力の変化量を確認することができるグラフです。

各グラフのタイトルを選択すると項目別に関連する詳細情報を見ることができる新しいウィンドウが作成されます。(図参照)



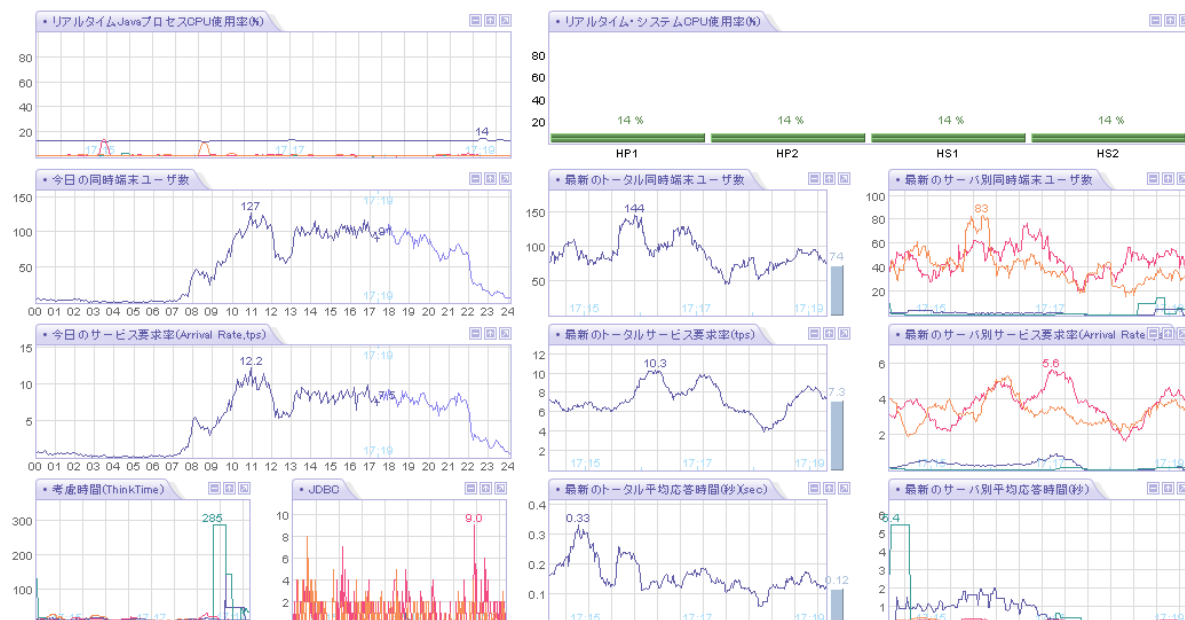
この情報は一日単位に保存されこのような情報をベースに日ごとのpeak load、週ごとのpeak load、月ごとのpeak load、分期peak load、年間peak loadなど一般的なload量分析と共に、サイトのイベントなどの特定ユーザが多数アクセスする時に、次の章で確認する[システム リソース要素モニタリング]分析と共に安定したシステム運用をサポートする重要指標を提供します。

・ システム リソースパラメータモニタリング

先に確認したシステムのサービス能力指数を見ると、CPU使用率とメモリ使用率のようなシステムリソースとの相関関係区分を確認したくなります。

例えば、WASシステムが2000名に提供すべき業務サービスがある場合、1000名程度が使った時点でCPUが100%に達してそれ以上サービスできない状況、又はCPUボトルネックによってtps又は応答時間の低下が発生することは、多くのサイトでおこっており、管理者としてはこのような、ボトルネックパラメータを確認する必要があります。

ジェニファーは(下図参照)[サービス能力]対比システムリソース制限要素確認によってCPU(システム全体、各JVM CPU情報)とメモリ使用量に関する情報も提供することができます。



上の図で最上段の二つのグラフのうち、左側のグラフは各JVM別cpu使用量を線グラフで表示しています。右側グラフはWASが設置されているUNIX(or windows)システムの全体CPU使用量を表示しています。

※ 現グラフには表示されていませんがイコライザー棒グラフの緑色はuserプロセスのCPU使用量、黄色はIO関連CPU使用量、赤色はKernelプロセス使用量を意味します。

また、一日のCPU使用量の変化などについての情報を得るためには、各グラフ上のタイトル(リアルタイム JAVAプロセス CPU使用率/リアルタイムシステム CPU使用率)のリンクをクリックすると、新しいウィンドウが開き、CPUと関連する情報をより具体的に確認することができます。

システムCPU



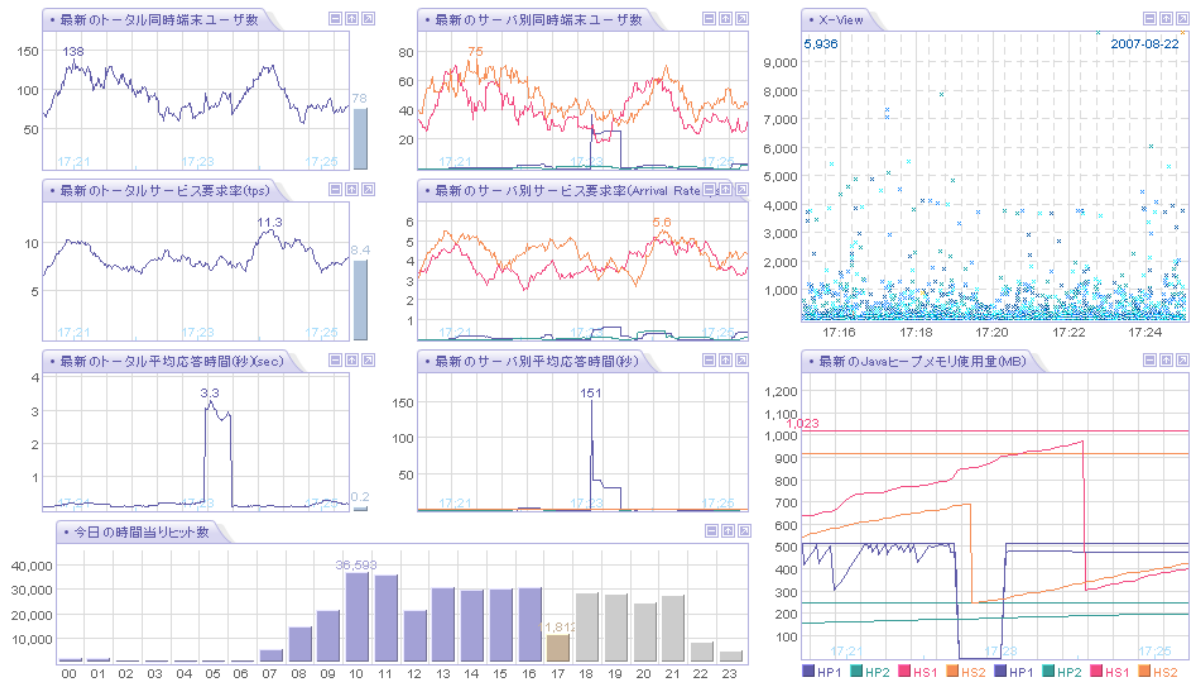
リアルタイムJavaプロセスCPU使用率(%)



Copyright © 2007 JenniferSoft, Inc. All rights reserved.

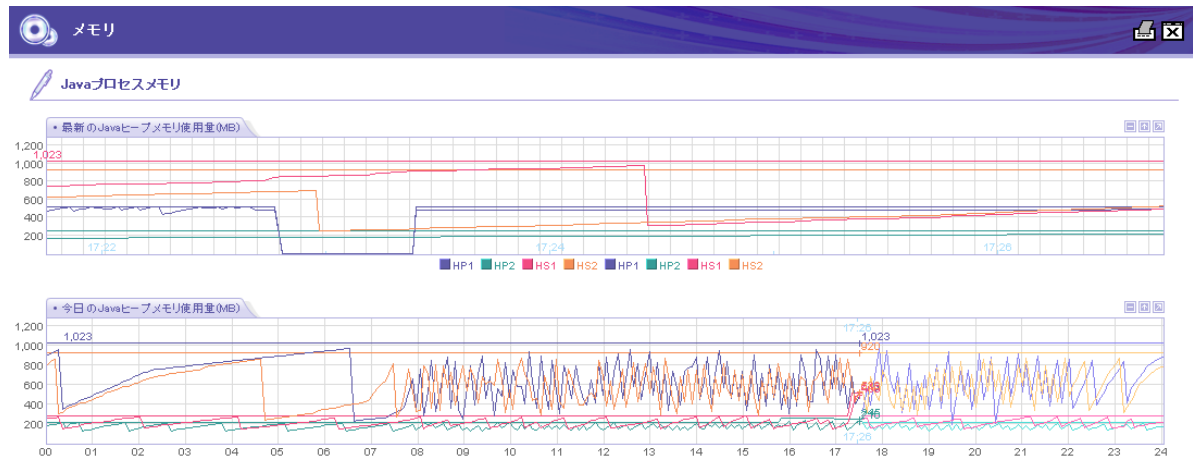
□参考 システムメモリは、UNIXシステムの場合、物理的に割り与えられたメモリを全て使用する形態でグラフ数値が表示されます。このような数値が出る理由はUNIXシステムのメモリ管理特性上、可用なメモリをファイルキャッシュタイプで使うためであり、該当グラフの数値からメモリ不足だと判断することはできません。その判断はページ(一部システムスワップ)などの数値を比較しながら判断することが必要です。

ヒープ (Heap) メモリ使用の変化量に関する情報は、同じ画面で確認できるようにすることで、各種[サービス能力]関連数値と比較分析が可能です。

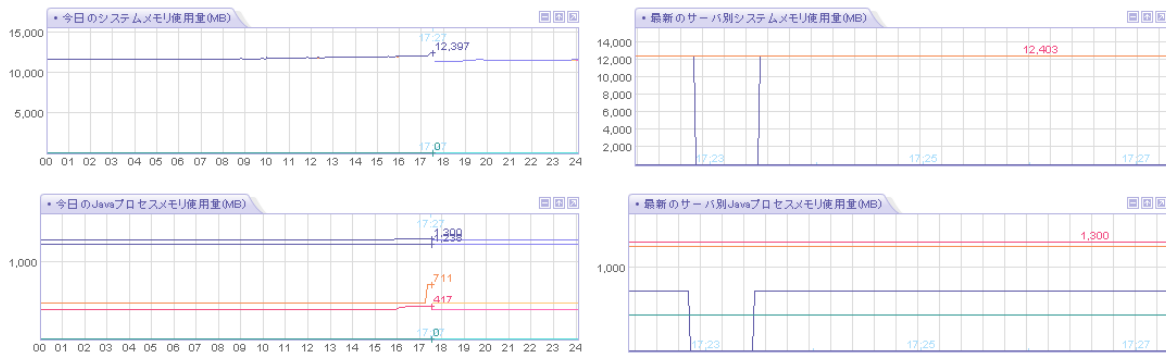


[リアルタイムHeap Memory変化量]

特に、CPUと同じようにシステムメモリ使用量と一日のメモリ使用量などの変化量についてより具体的な情報を得るためには、Heapメモリ使用量グラフのタイトルのリンクをクリックすると下記の新しいウィンドウが開き、希望する情報を見ることができます。



メモリ使用量



Copyright (c) 2007 JenniferSoft, Inc. All rights reserved.

[Heap Memory及びシステムメモリ一日使用変化量]

・リアルタイム アプリケーション トランザクション モニタリング

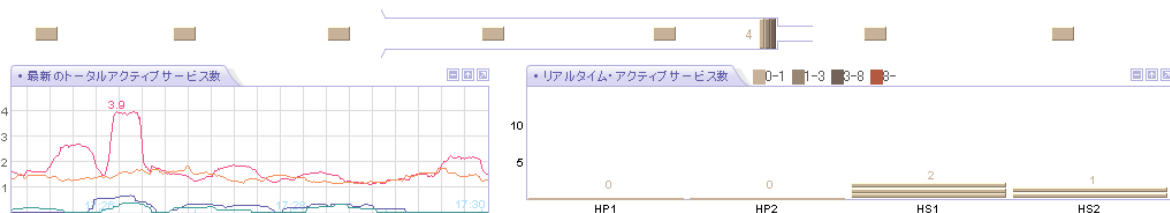
これまでジェニファーの基本ウィンドウで提供する、データを通じてシステムのリアルタイムサービス能力とリソース使用量を確認する方法と詳細情報を見る連結部分について説明しました。

本章ではジェニファーだけが提供しているリアルタイムアプリケーションスレッドモニタリングの方法を説明します。これにより、運用中、又は障害発生時、多角的な観点(viewpoint)でアプリケーションをモニタリングして管理することができます。

初めてジェニファーに接する方にとって新しいアプレット要素は、画面上段にあるイコライザーでしょう。(ジェニファーで提供する各種グラフィック資料はアプレットで提供します。)

(http://www.javaservice.com/~java/bbs/read.cgi?m=&b=jsfaq&c=r_p&n=1119284288)

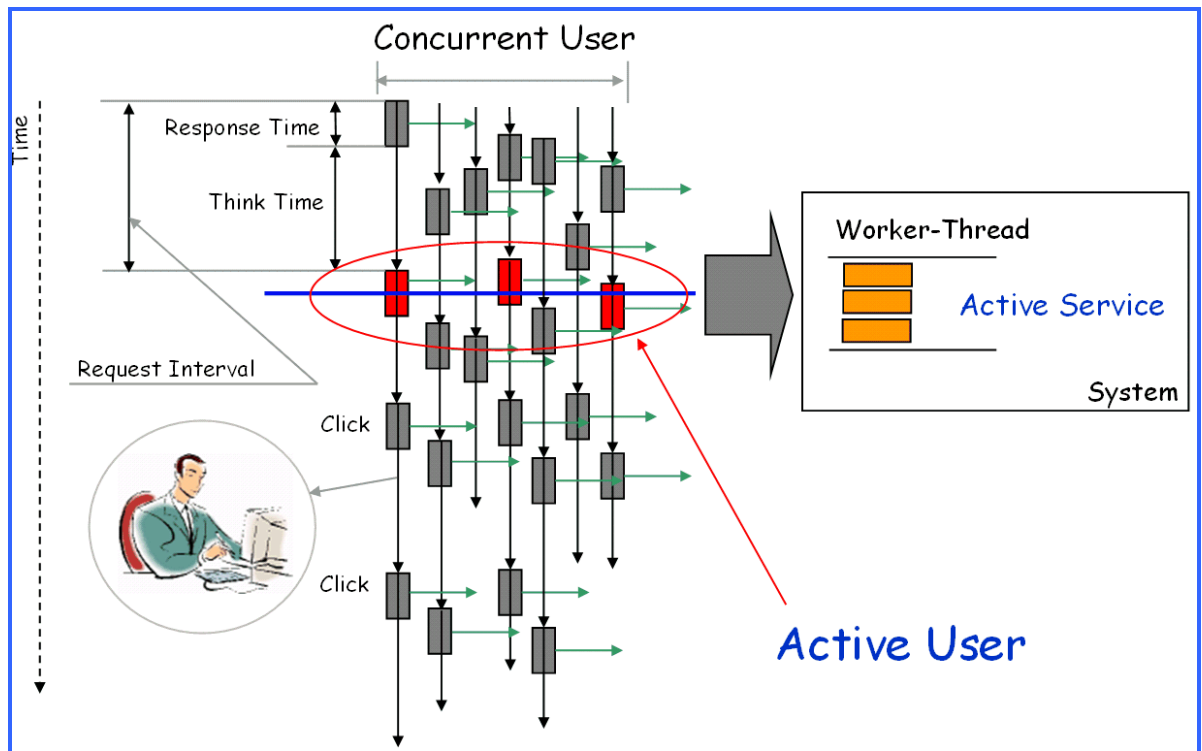
最上段のスピードバーは右下にある[リアルタイム アクティブサービス数]と同じ意味ですが、違いは、スピードバーは全体構成されたエージェント内にある実行中のアクティブサービスの数の合計で、下段にある図は各インスタンス別実行中のアクティブサービスの数です。



アクティブサービス(又はユーザ)に関する用語の定義を、始めに確認しておきます。

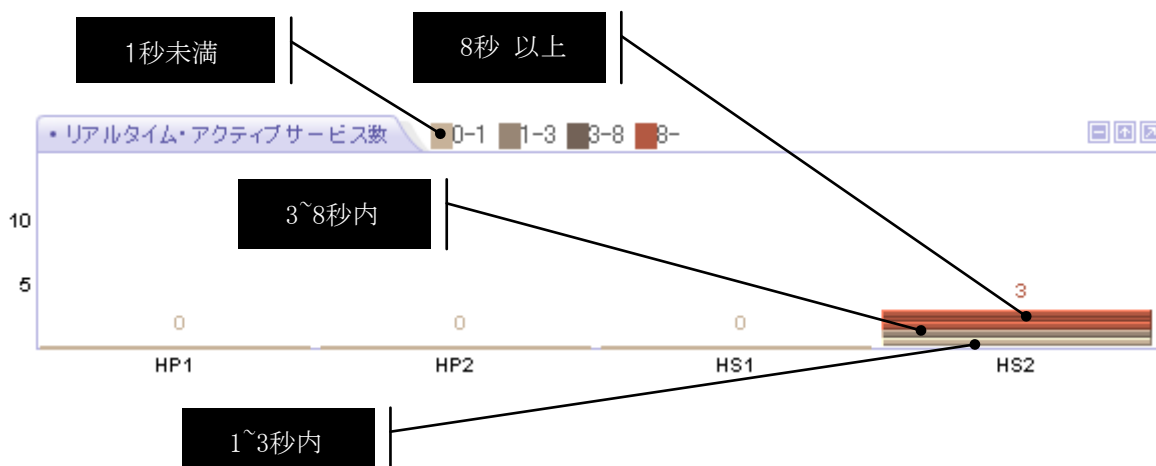
用語説明

アクティブ(Active)ユーザ：下図の様に、各個別同時端末ユーザは各応答時間(Response Time)と待機時間(Think Time)を持って、クリックを繰り返します。特定時刻で時間を切ってみると、該当該時刻にクリック以降まだレスポンスを受けていないユーザ、又は該当該リクエストが現在実行中であるユーザなどが存在します。下図では3名のユーザが該当該時点でリクエスト後、レスポンスを待っているユーザです。このように、“リクエスト後、レスポンスを待っているユーザ”を“アクティブ(Active)ユーザ”として定義します。



ネットワークボトルネックがないと、アクティブ(Active)ユーザのリクエストはサーバ側で該当時点にサービスが実行され(running)ています。このようにサーバ側で該当時点にサービスが実行されている数を“アクティブ(Active)サービス”と定義します。アクティブユーザがクライアント側からの観点であれば、アクティブサービスはサーバ側からの観点から見た数値です。本文書では二つの用語を同義語として使います。²

現在WASインスタンス内でリクエストされたサービス进行处理する為、実行中のアプリケーションスレッドだとお考えください。また、各バーの色が意味するのは実行中のアプリケーションの経過時間(秒)です。(下図参照)



現在WAS内で実行中のアプリケーションスレッドの数は、経過時間ごとに色分けされているので、実行状態を直感的に判断できます。

実行中のアプリケーションについての詳細情報を見るには、上のグラフで確認したいインスタンスを選択しダブルクリックすると、新しいウィンドウが開いて表示されます。



上図のようにリクエストを実行中であるアプリケーションについて様々な情報を得ることができます。下記の表は提供する情報を整理したものです。

区 分	内 容
IP	アプリケーションをリクエストしたクライアントのIP
APPIVAL	アプリケーションがリクエストされた時間
ELASPED	現在までのアプリケーション実行時間
CPU	アプリケーションのCPU使用量
THREAD	JAVAスレッドはOSから割り当てられたpthreadマッピング情報
STATUS	アプリケーションが現在実行中の段階 (下記の参考内容参照)
URL	実行中のアプリケーション名と実行したsql分の数、現在までのfetch件数も同時に表示される。もし、リアルタイム アクティブ サービスリストがリロード(自動10秒、又はF5キー)する時、クエリ数が実行中の場合は一番右側に実行中のクエリまで提供する。

リアルタイムにWASで実行中のアプリケーションについての各要素別が整理されて提示されます。特に STATUS項目は一般的にウェブアプリケーションが実行するロジック、JDBCプログラミング、ホストコネクションなどの内容を段階別に整理し、実行中のアプリケーション状態を簡略なコードで提供することで簡単にアプリケーションの実行把握ができます。



NOINIT

Initializing to get URI, client IP address, Thread ID

INITZG

Registering Active service and Arrival Count

REQST1

Extracting basic information, such as User ID from Cookie

ARRSND

Sending UDP runtime packet to Jennifer server

REQST2

Processing the first HTTP GET/POST operation and character set

REQEND

Processing Visit user and Thinktime calculating after GET/POST operation

INITED

Checking if this request should be rejected by PLC(Peak Load Control)

RJCTNG

Sending PLC message to client

RJCTED

Sent the PLC message to client

APPRUN

After the initializing task, now executing an application logic

[アプリケーションSTATUSコード整理]

ジェニファーに取り入れた新しい技術LWST(Light Weight Stack Trace)は、実行中のアプリケーションのリアルタイム スタックトレース(Stack Trace)情報を提供することで、アプリケーションが現在どのメソッドを実行中であるかの情報を提供します。

以下の例では、[リアルタイムアクティブサービスリスト]で使用する実行中のアプリケーションの名前を選択すると、新しいウィンドウが開き、詳細情報を提供します。



スレッド名	WebContainer : 194 / 0x7925 (isAlive: true, isDaemon: true, isInterrupted: false, isSuspended: false)
URL	/BSIS/BM0000J/BM2220J_R.jsp
HTTPメソッド	POST
到着時間	2007-08-24/12:05:13
経過時間	8.364 seconds
CPU経過時間	0.000 seconds
CPU使用率(%)	0.0 %
状態	JEXENG (Executing JDBC SQL query)
SQL数	4
SQL照会数	2
現在実行中のSQL	SELECT EMP.BIZENO,EMP.BSIZE,SUM(CAST(DTP.BRSAMT AS BIGINT))BRSAMTHUP,SUM(CAST(DTP.BRSQTY AS BIGINT))BRSQTYHUP FROM ((BIS.BRSDTP DTP INNER JOIN BIS.BITMAP MAP ON DTP.BITMCD = MAP.BITMCD AND DTP.BCOLNO = MAP.BCOLNO) INNER JOIN BIS.BITEMP EMP ON DTP.BITMCD = EMP.BITMCD AND DTP.BCOLNO = EMP.BCOLNO AND DTP.BIZENO = EMP.BIZENO) INNER JOIN BIS.BCODEP DEP ON MAP.BITMTP = DEP.BSMLCD WHERE DEP.BLRGCD = '02' AND DTP.BHOPCD = '92158' AND DTP.BBRDGU = '4' AND DTP.BITMCD = '427353' AND DTP.BCOLNO = '1' AND DTP.BRSYMD BETWEEN '20060908' AND '20070824' AND DEP.BSMLNM = '??' AND DTP.BRSCLS = '*' GROUP BY EMP.BIZENO, EMP.BSIZE WITH UR
スタックトレースクライアントIP	public ResultSet java.sql.PreparedStatement.executeQuery() 58.238.15.234
クライアント番号	-6091700106689389883

Copyright(c) 2007 JenniferSoft, Inc. All rights reserved.

上図のように、リアルタイムスタックトレース(Stack Trace)情報を提供することで、より多角的な側面から実行

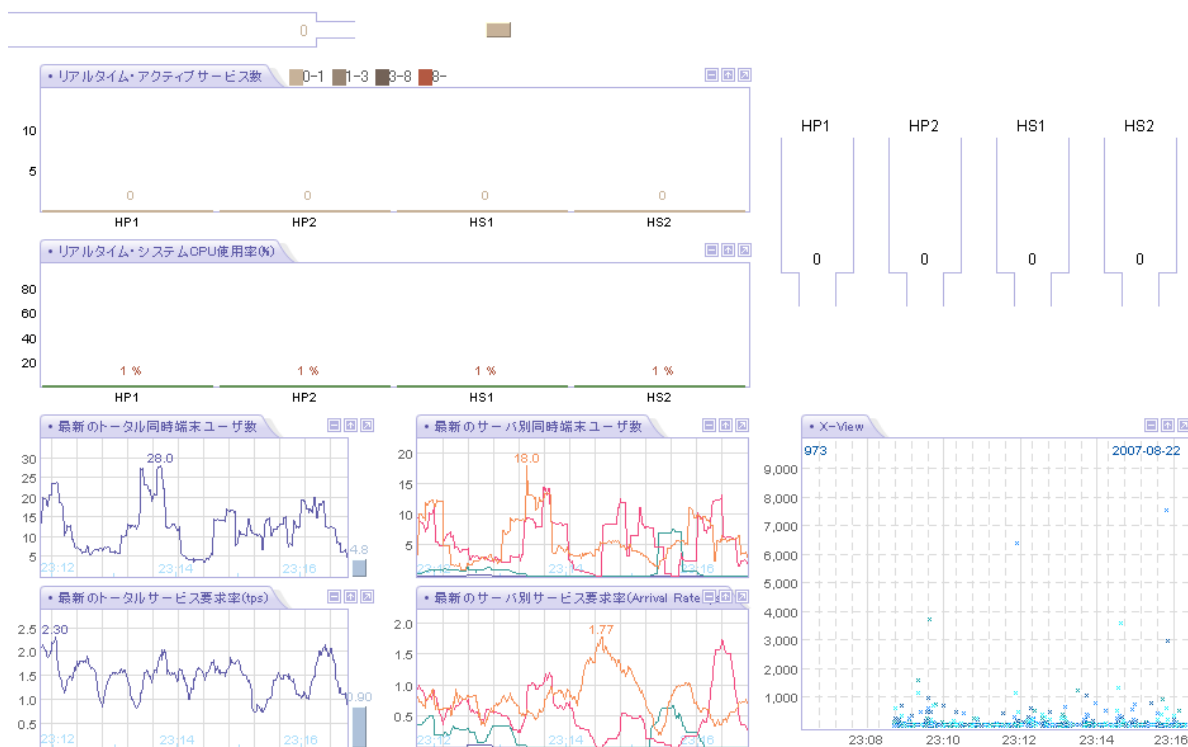
中のアプリケーションについての分析が可能になります。



Copyright(c) 2007 JenniferSoft, Inc. All rights reserved.

アプリケーションを選択した場合、上のようなメッセージが表示されたら、該当アプリケーションスレッドの実行は完了したという意味です。

また、実行を完了したアプリケーションは統計データで各種値(リクエスト件数、平均応答時間、偏差など)に保存されます。同時に、ジェニファーから新しく導入した[XView]という新機能により、各トランザクション情報をレスポンス時間別に点グラフで表しながら詳細な実行の内訳情報を提供します。



上図で、赤いボックスで表示されたグラフがx-viewデータを表示するグラフです。[x]軸は各アプリケーショントランザクションの終了時間を表します。[y]軸は各アプリケーショントランザクションの実行時間を意味します。

グラフ右側の[x-view]タイトルをクリックするか、メインメニューの障害診断を選択すると、下記のような拡大されたx-viewグラフを見ることができます。

また、x-view画面でレスポンス時間台の拡大及び時間台の拡大及び縮小の場合はグラフ上段の Maxと Timeメニューの値を入力すること、グラフのデータ範囲を分析に便利に使うようになっています。

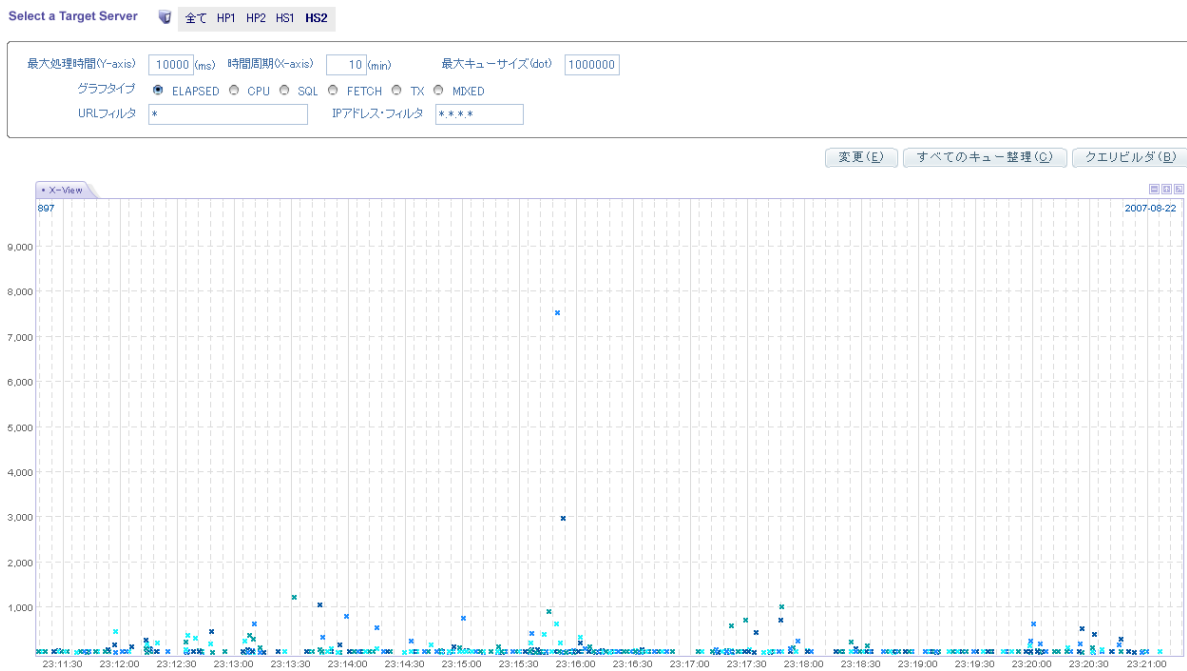
TIP: x-viewグラフのデータ範囲変更の為に提示した、直接入力による方法以外にもHotKeyを利用して簡単に操作することができます。

▶ HotKey整理

HotKey	説明
矢印キー(up/down)	elapsed max 調整
矢印キー(left/right)	右側 base time 移動
+/-	view interval 拡張/縮小

以上全ての場合、shift/ctl keyを同時に押すと、「早く/より早く」という機能として動作します。

注: 点(point)のパターン分析によるシステムの障害及び性能チューニングについては、本書では扱いません。別の文書で説明しています。



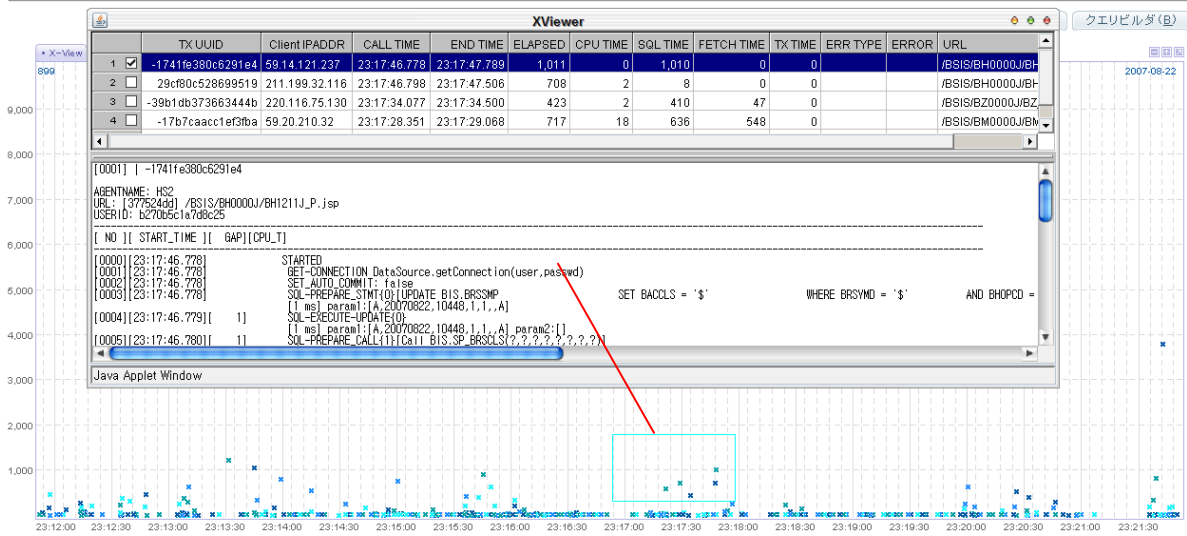
使用例: 上のグラフで8秒台に実行されたトランザクションについて知りたい場合、該当時間台のトランザクションをマウスで選択(ドラック)すると下記の図のように各トランザクションについての実行内訳を提供する新しいウィンドウが開きます。

Select a Target Server 全て HP1 HP2 HS1 HS2

最大処理時間(Y-axis) 10000 (ms) 時間周期(X-axis) 10 (min) 最大キューサイズ(dot) 1000000

グラフタイプ ELAPSED CPU SQL FETCH TX MIXED

URLフィルタ * IPアドレス・フィルタ *.*.*.*



	TX UUID	Client IPADDR	CALL TIME	END TIME	ELAPSED	CPU TIME	SQL TIME	FETCH TIME	TX TIME	ERR TYPE	ERROR	URL
1	-1741fe380c6291e4	59.14.121.237	23:17:46.778	23:17:47.789	1,011	0	1,010	0	0			/BSIS/BH0000J/BH
2	29c890c528699519	211.199.32.116	23:17:46.798	23:17:47.506	708	2	8	0	0			/BSIS/BH0000J/BH
3	-39b1db373663444b	220.116.75.130	23:17:34.077	23:17:34.500	423	2	410	47	0			/BSIS/BZ0000J/BZ
4	-17b7caacc1ef3fba	59.20.210.32	23:17:28.351	23:17:29.068	717	18	636	548	0			/BSIS/BM0000J/BM


```

(0000) [23:17:46.778] STARTED
(0001) [23:17:46.778] GET-CONNECTION DataSource.getConnection(user,passwd)
(0002) [23:17:46.778] SET_AUTO_COMMIT: false
(0003) [23:17:46.778] SQL-PREPARE_STMT(0) [UPDATE BIS_BRSSMP SET BACCLS = '$ WHERE BRSYMD = '$ AND BHOPCD =
[0004] [23:17:46.779] [ 1] SQL-EXECUTE-UPDATE(0)
[0005] [23:17:46.780] [ 1] [1 ms] param1:[A,20070822,10448,1,1,,A] param2:[]
SQL-PREPARE_CALL(1) [Call1 BIS.SP_BRSDCLS(?,?,?,?)]
[0 ms] param1:[]
[0006] [23:17:46.780] SQL-EXECUTE(1)
[717 ms] param1:[] param2:['20070822','10448','1','1','2','S110448']
[0007] [23:17:47.497] [ 717] SQL-PREPARE_STMT(2) [UPDATE BIS_BRSSMP SET BACCLS = '$ WHERE BRSYMD = '$ AND BHOPCD =
[0 ms] param1:[A,20070822,10448,1,2,,A]
[0008] [23:17:47.497] SQL-EXECUTE-UPDATE(2)
[2 ms] param1:[A,20070822,10448,1,2,,A] param2:[]
[0009] [23:17:47.499] [ 2] SQL-PREPARE_CALL(3) [Call1 BIS.SP_BRSDCLS(?,?,?,?)]
[0 ms] param1:[]
[0010] [23:17:47.499] SQL-EXECUTE(3)
[15 ms] param1:[] param2:['20070822','10448','1','2','2','S110448']
[0011] [23:17:47.514] [ 15] SQL-PREPARE_CALL(4) [UPDATE BIS_BRSSMP SET BACCLS = '$ WHERE BRSYMD = '$ AND BHOPCD =
[1 ms] param1:[A,20070822,10448,1,3,,A]
[0012] [23:17:47.515] [ 1] SQL-EXECUTE-UPDATE(4)
[1 ms] param1:[A,20070822,10448,1,3,,A] param2:[]
[0013] [23:17:47.516] [ 1] SQL-PREPARE_CALL(5) [Call1 BIS.SP_BRSDCLS(?,?,?,?)]
[0 ms] param1:[]
[0014] [23:17:47.516] SQL-EXECUTE(5)
[257 ms] param1:[] param2:['20070822','10448','1','3','2','S110448']
[0015] [23:17:47.774] [ 258] COMMIT
[0016] [23:17:47.774] CLOSE-CONNECTION
[0017] [23:17:47.774] GET-CONNECTION DataSource.getConnection(user,passwd)
[0018] [23:17:47.774] SQL-PREPARE_CALL(6) [Call1 BIS.BS4010SBR(?,?,?,?)]
[0 ms] param1:[]
[0019] [23:17:47.774] SQL-EXECUTE(6)
[3 ms] param1:[] param2:['20070822','10448','1','1','BRMAGAN']
[0020] [23:17:47.777] [ 3] SQL-PREPARE_CALL(7) [Call1 BIS.BS4010SBR(?,?,?,?)]
[0 ms] param1:[]
[0021] [23:17:47.777] SQL-EXECUTE(7)
[1 ms] param1:[] param2:['20070822','10448','1','2','BRMAGAN']
[0022] [23:17:47.778] [ 1] SQL-PREPARE_CALL(8) [Call1 BIS.BS4010SBR(?,?,?,?)]
[0 ms] param1:[]
[0023] [23:17:47.778] SQL-EXECUTE(8)
[11 ms] param1:[] param2:['20070822','10448','1','3','BRMAGAN']
[0024] [23:17:47.789] [ 11] CLOSE-CONNECTION
[0025] [23:17:47.789] END
TOTAL 1,011 0

```

新しく生成されたウィンドウは上位と下位の二つのセクションに区分され、上位のウィンドウは各トランザクションの実行アプリケーション統計情報を、下位のウィンドウは該当アプリケーションが実行されたトランザクション情報を表示します。

・ 上位セクションで提供する統計情報整理

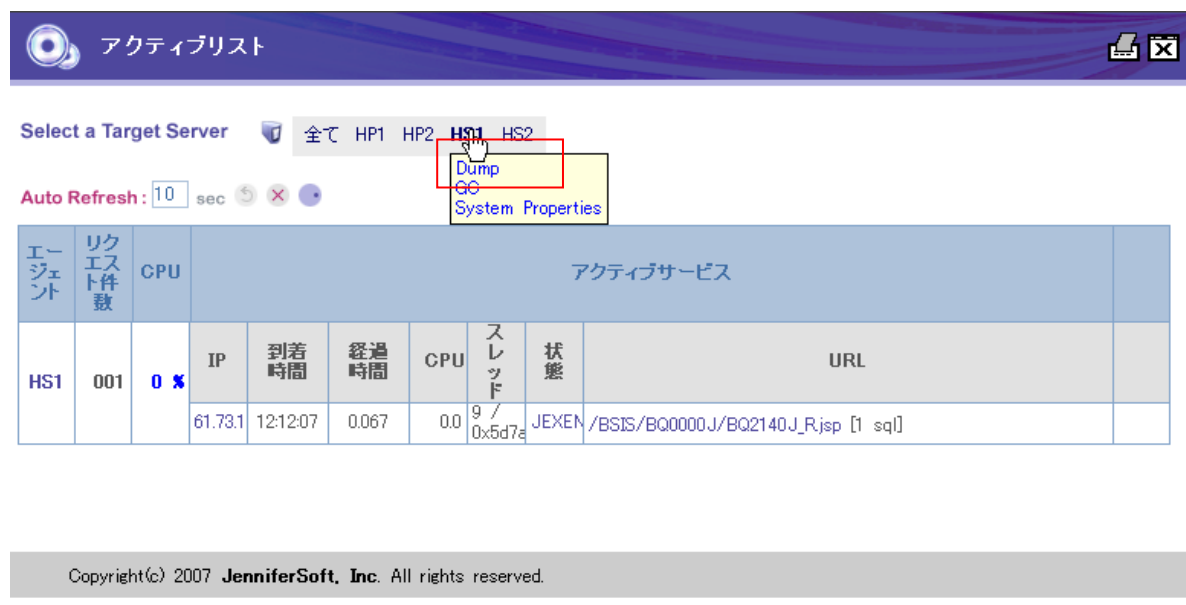
区分	内容
NO	x-viewで点選択時、選択されたトランザクションの数
TX UUID	各リクエスト(Request)毎にユニークに生成されたトランザクションユニーク番号
CLIENT IP	サービスをリクエストしたクライアント側IP番号
CALL TIME	アプリケーションがリクエストした時間
END TIME	アプリケーションが実行完了した時間
ELAPSED	アプリケーショントータル実行時間
SQLTIME	conn.prepareStatement(...), stmt.executeUpdate()/Upsdate() 実行時間の合計
FETCHEDTIME	resultsetが生成された時刻よりrs.next()の結果にfalseが出る時点までの時間の合計
ERROR	該当アプリケーションの、ジェニファーが定義した障害項目要素に含まれた状態
URL	アプリケーション名

・ 下位セクションで提供する統計情報

下位セクションでは上位セクションで選択された個別アプリケーションについての詳細トランザクション情

報を提供します。基本値をそのままインストールした場合は、jspリクエスト内容と実行中使用したファイル、JDBCコネクション及び終了情報、クエリ実行内訳が実行段階別に示されます。必要によってサイトで使用中のフレームワーク(framework)又はビジネスクラス情報を追加で構成すると、ビジネスメソッド実行内訳まで細かくモニタリングできます。このような情報は障害診断及び性能チューニングを簡単にする為の機能として役立ちます。

これまでジェニファーが提供する多様なリアルタイム情報について三つの側面から整理しました。

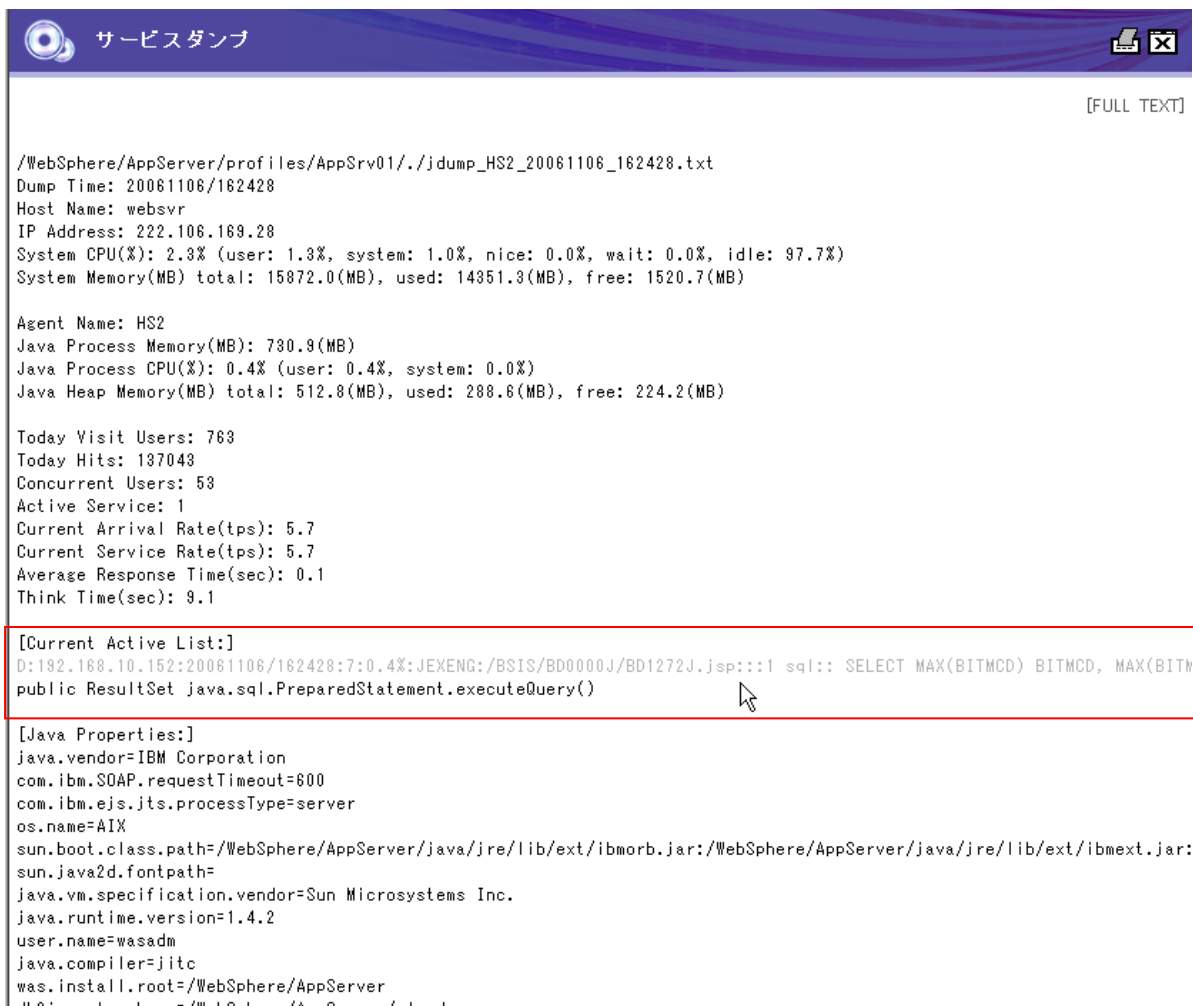


TIP : リアルタイムスタックトレス情報保存方法

リアルタイム アクティブ サービスをモニタリングして、あとでそれを確認、分析するためにその結果を保存するには、上図右上段の[Dump Now]を選択すると、実行中のアプリケーション情報のみならずその関連各種情報(システム使用ダンプ、ダンプ時点のサービス能力関連情報)を全てテキストファイルで保存し、同時に新しいウィンドウで確認できるようにします。



赤色で表示されたファイル内容が生成リクエストによって生成された最新のファイルです。該当ファイルを選択(右側のviewクリック)すると下記の図のように新しいウィンドウが開きダンプ時点のシステム状況情報と実行中のアプリケーションのスタック情報を保存、提供します。



```
サービスダンプ [FULL TEXT]

/WebSphere/AppServer/profiles/AppSrv01/./jdump_HS2_20061106_162428.txt
Dump Time: 20061106/162428
Host Name: websvr
IP Address: 222.106.169.28
System CPU(%): 2.3% (user: 1.3%, system: 1.0%, nice: 0.0%, wait: 0.0%, idle: 97.7%)
System Memory(MB) total: 15872.0(MB), used: 14351.3(MB), free: 1520.7(MB)

Agent Name: HS2
Java Process Memory(MB): 730.9(MB)
Java Process CPU(%): 0.4% (user: 0.4%, system: 0.0%)
Java Heap Memory(MB) total: 512.8(MB), used: 288.6(MB), free: 224.2(MB)

Today Visit Users: 763
Today Hits: 137043
Concurrent Users: 53
Active Service: 1
Current Arrival Rate(tps): 5.7
Current Service Rate(tps): 5.7
Average Response Time(sec): 0.1
Think Time(sec): 9.1

[Current Active List:]
D:192.168.10.152:20061106/162428:7:0.4%:JEXENG:/BSIS/BD0000J/BD1272J.jsp:::1 sql:: SELECT MAX(BITMCD) BITMCD, MAX(BITM
public ResultSet java.sql.PreparedStatement.executeQuery()

[Java Properties:]
java.vendor=IBM Corporation
com.ibm.SOAP.requestTimeout=600
com.ibm.ejs.jts.processType=server
os.name=AIX
sun.boot.class.path=/WebSphere/AppServer/java/jre/lib/ext/ibmorb.jar:/WebSphere/AppServer/java/jre/lib/ext/ibmext.jar:
sun.java2d.fontpath=
java.vm.specification.vendor=Sun Microsystems Inc.
java.runtime.version=1.4.2
user.name=wasadm
java.compiler=jitc
was.install.root=/WebSphere/AppServer
db2inst1.home=/WebSphere/AppServer/aleudscpp
```

※該当機能は、生成リクエストによる生成のみならず、アクティブサービス(active servie)数が指定した値(基本値は90個)以上インスタンス内に累積されると自動的に生成され、モニタリングしなかった時に発生した問題の分析に活用できます。

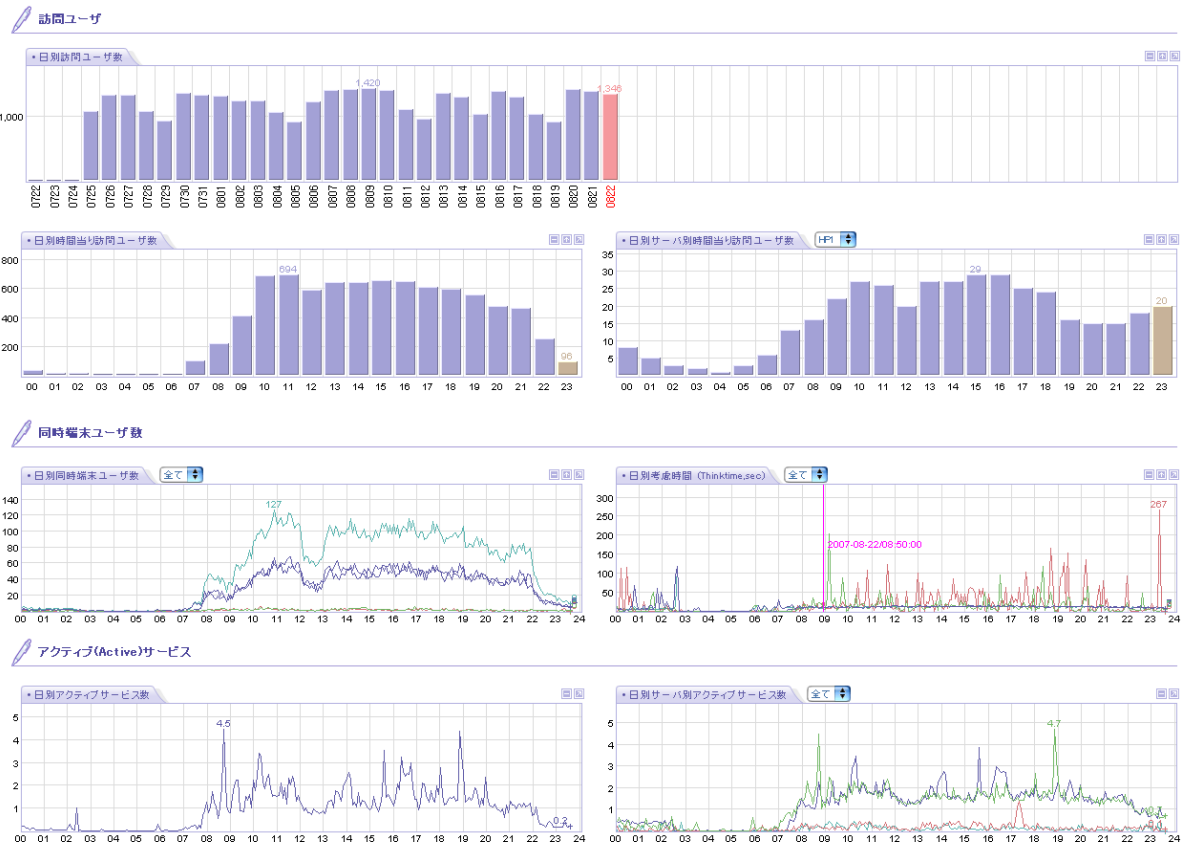
▶ ヒストリデータ分析

ここまでリアルタイムモニタリング要素とアクセス方法を説明しました。この章では前の章で紹介したリアルタイムデータの統計化されたヒストリデータの活用法について説明します。

・ ユーザ / アプリケーション / スロープット / メモリ / CPU データ

ジェニファーの上位 メインメニュー(main menu)中、統計分析を選択すると、前章で確認した全てのリアルタイム情報などが統計化され日時別/時間別に提供されます。

(下図参照)

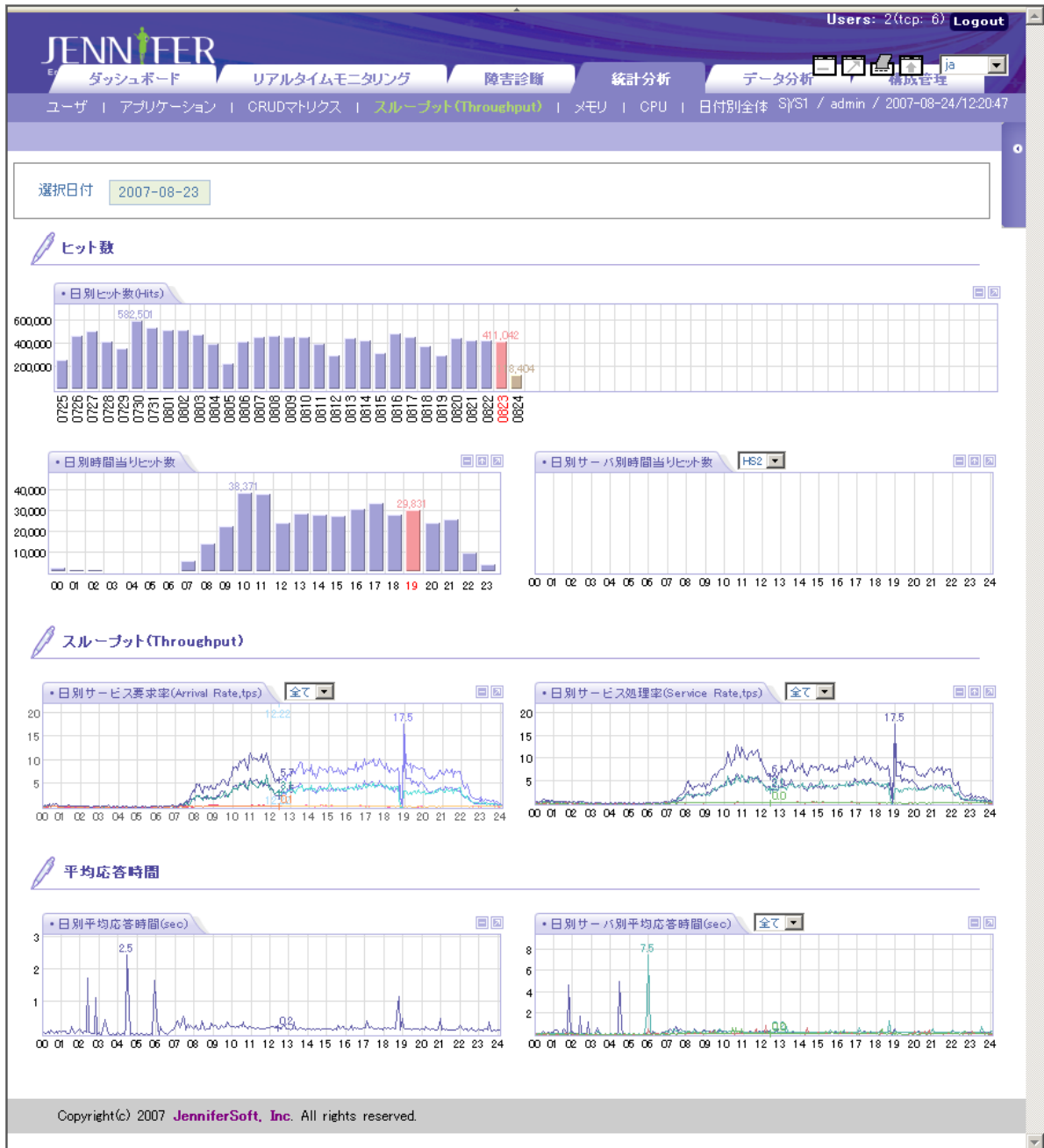


Copyright (c) 2007 JenniferSoft, Inc. All rights reserved.

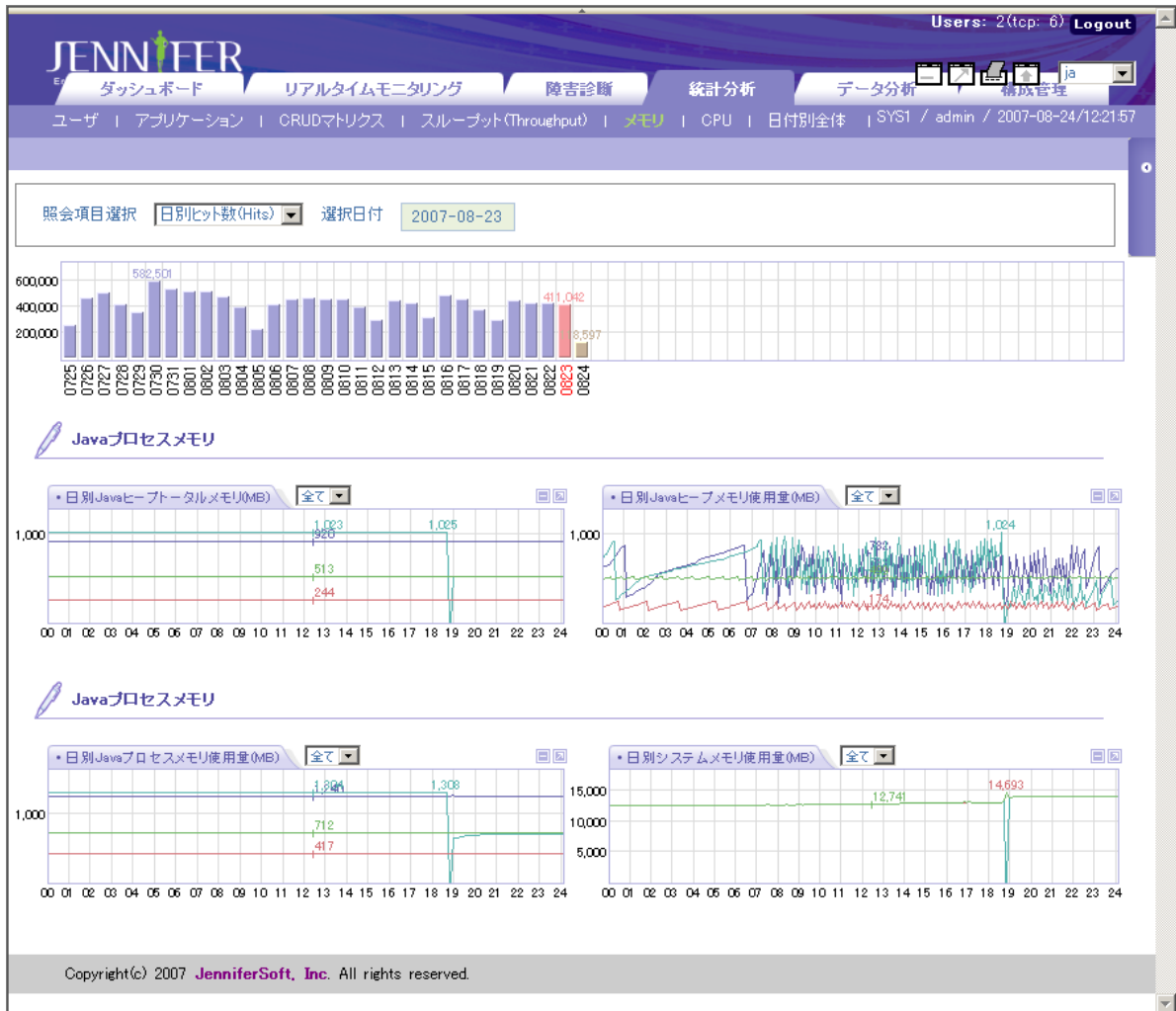
[ユーザ統計データ]



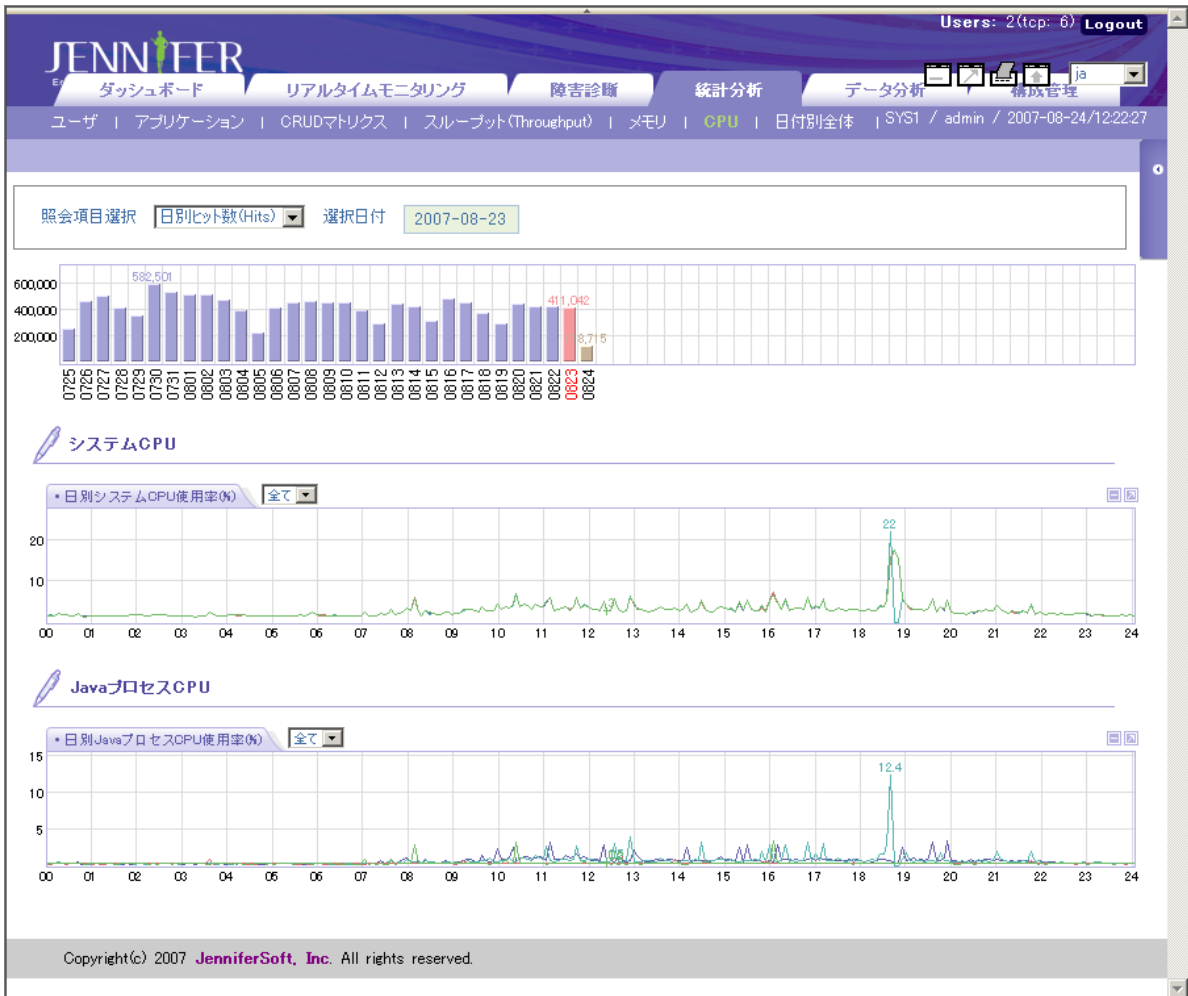
[アプリケーション実行内訳統計データ]



[スロープット[Throughput] 統計データ]



[メモリ使用量統計データ]



[CPU使用量統計データ]

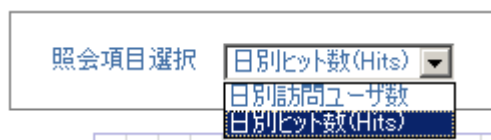
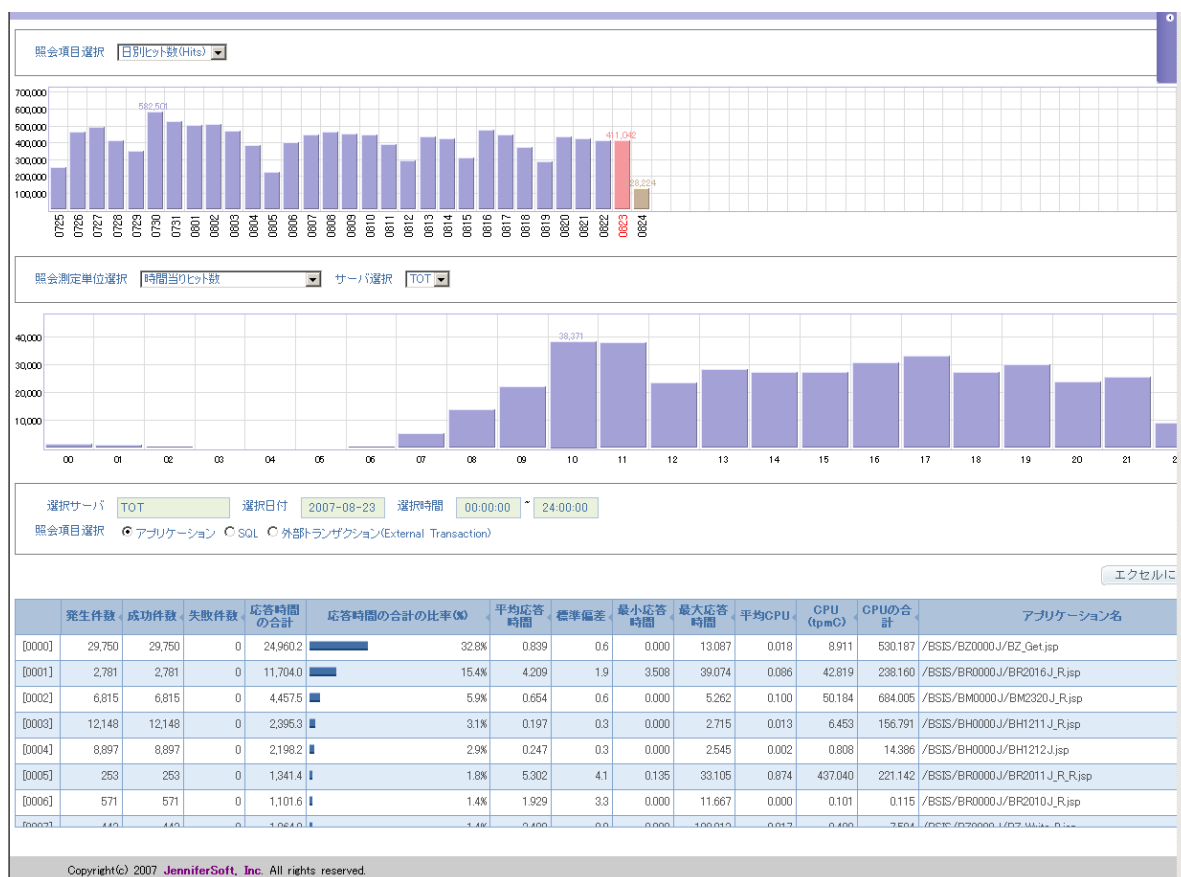
上の部分で提示された統計データは前の部分で説明した[システムサービス能力]についての日時/週/分期/年間データを提供します。同時に[システムリソース]についての情報も同じ基準で提供されるため、現在のシステムリソースの管理や、将来のプロジェクトを検討する際の根拠となるデータとして活用できます。

・ アプリケーション性能分析

- グラフ及び統計情報使用法

WASを利用してアプリケーションを開発/運用する方の主な関心の一つは“性能チューニング”でしょう。本ガイドにはジェニファーのインストール後、ある程度データが収集された時点ジェニファーから提供されるアプリケーション統計データを使って基本的なチューニング要素を推出する方法を説明します。

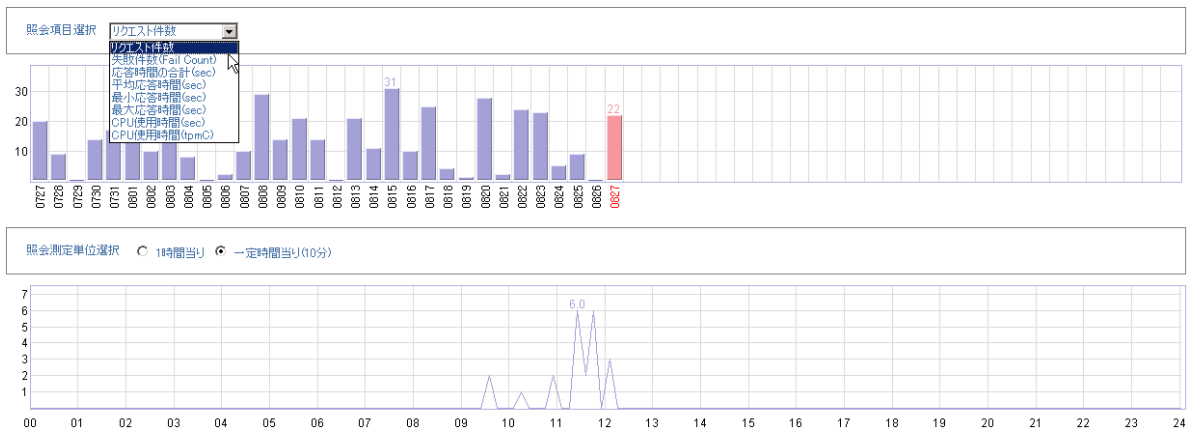
下図はメインメニューの[統計分析]を選択すると基本的に選択されるアプリケーション統計データ 画面です。



一番上のグラフは日にち別訪問者数、日にち別リクエスト件数(基本値)を表示します。

下記のグラフは上位グラフで選択された日の時間別情報を下記のような様々な項目別に検索できるようになっています。(右側選択ウィンドウを利用して各サーバ別選択可能です。(基本は構成された全体サーバのトータル合計です。))

さらに一番下にあるグラフは上から選択したアプリケーション/クエリ/トランザクションについての個別統計情報を多様な側面で時間別に分析できるようにビュー(view)を提供します。

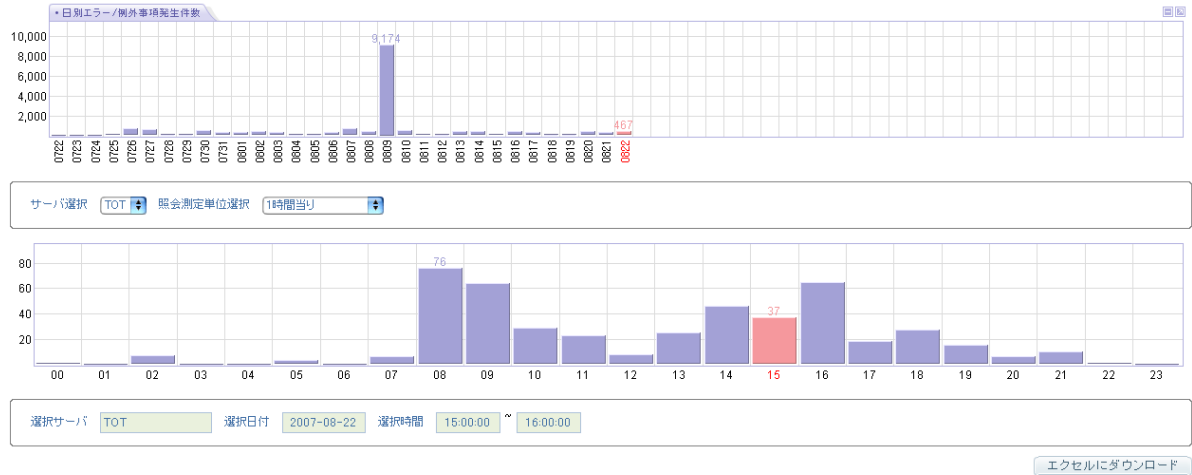


エクセルにダウンロード

統計情報ウィンドウにある[Excel Download]アイコンを選択するとブラウザで提供するテーブルタイムの情報をExcelで管理することができます。

▶ 障害要素(エラー/例外検知) 分析

ジェニファーはアプリケーション実行中発生した、障害要素を検出できる機能があります。これにより、運用中のシステムの問題要因を分析、問題発生の原因を提供し、運用者/開発者が簡単に障害要因を認識しながら修正できるようにします。



選択サーバ: TOT | 選択日付: 2007-08-22 | 選択時間: 15:00:00 ~ 16:00:00

エクセルにダウンロード

発生件数	発生率(%)	エラー/例外項目
[0000]	18	48.6% JDBC BAD RESPONSE TIME
[0001]	6	16.2% JDBC PreparedStatement SQL EXCEPTION
[0002]	6	16.2% JDBC Connection NOT CLOSED
[0003]	3	8.1% APPLICATION BAD RESPONSE TIME
[0004]	2	5.4% TOO MANY ResultSet FETCHED
[0005]	2	5.4% UNCAUGHT APPLICATION EXCEPTION

[エラー/例外メイン画面]

メインメニュー → エラー例外検知 タブを選択すると日別エラー件数と時間別エラー発生件数のグラフを見ることができます。

特定の日付又は該当日付の時間を選択すると発生したエラーの種類と件数をテーブルタイプで確認できます。

選択サーバ: TOT | 選択日付: 2007-08-22 | 選択時間: 15:00:00 ~ 16:00:00

エクセルにダウンロード

発生件数	発生率(%)	エラー/例外項目
[0000]	18	48.6% JDBC BAD RESPONSE TIME
[0001]	6	16.2% JDBC PreparedStatement SQL EXCEPTION
[0002]	6	16.2% JDBC Connection NOT CLOSED
[0003]	3	8.1% APPLICATION BAD RESPONSE TIME
[0004]	2	5.4% TOO MANY ResultSet FETCHED
[0005]	2	5.4% UNCAUGHT APPLICATION EXCEPTION

1. 選択したエラー/例外は該当時間帯で以下のアプリケーションで発生しました。 (20070822 16:00:00 ~ 17:00:00)
 2. 選択したエラー/例外項目は日付別/時間帯別で以下のよう発生しました。
JDBC PreparedStatement NOT CLOSED

エクセルにダウンロード

発生件数	発生率(%)	エラー発生アプリケーション	関連内容
[0000]	3	100.0% /BSIS/BW0000/J/BW1050_JR.jsp	sql[SELECT sum(ABWHTQT) BWHTQT, sum(cast(ABWHTAM as bigint)) BWHTAM FROM BISBWHMSP A WHERE A.BWHTCOD = 'D' AND (ABWHSS BETWEEN 1 AND 999) AND ABWHYMD = '20070822' AND ABWHOLS NOT IN(*)] pstmt created FETCH [0/0] public java.sql.PreparedStatement(Init) public PreparedStatement java.sql.Connection.prepareStatement(String)

該当テーブルでエラー/例外項目で確認したいリストを選択すると、該当エラー/例外状況に登録されたアプリケーションリスト情報を下記のテーブルで確認できます。

各アプリケーションの障害発生に対する詳細内訳を確認する場合は確認したいアプリケーションをダブルクリックすると新しいウィンドウが生成され詳細内訳を確認することができます。

発生件数	発生率(%)	エラー発生アプリケーション	関連内容
[0000]	3 100.0%	/BSIS/EW0000/J/EW1050.J_R.jsp	sql[SELECT sum(ABWHOT) BWHOT, sum(cast(ABWHAM as bigint)) BWHAM FROM BISBWHSMF A WHERE ABWHOD =D' AND (ABWHSS BETWEEN 1 AND 999) AND ABWHYMD = '20070822' AND ABWHCLS NOT IN(*)][] print created FETCH [0/0] public java.sql.PreparedStatement<init> public PreparedStatement java.sql.Connection.prepareStatement(String)

最後に

本書では、ジェニファーをはじめてインストール後、基本的にジェニファーで提示されるデータの活用方法について整理しました。本書で紹介しきれなかった多くの有用な機能に関しては製品マニュアル又はジェニファーソフトを通して各事例文書をお読みいただくと、その活用幅がより広がります。

ジェニファーは最大限現場ベースで作られましたが、ツール(Tools)の特徴を持っている為、少し分かりにくい部分があります。皆様の積極的なご活用とともに、ぜひご意見をお寄せください。