Java障害解決のプロが創った
目からウロコのJavaシステム診断ツール

ENdoSnipe のご紹介

2009/1/29

エスエムジー株式会社
山崎 政憲

1. エスエムジー株式会社のご紹介

- ネットワーク監視制御ソフトウェアの受託開発をメインミッションにしています。
- 10年前よりいち早くLinux／Javaに特化し、ノウハウを蓄積して参りました。

2000	ネットワーク監視制御プラットフォーム「GNMS」をJavaで構築し、パッケージとして販売。
2001	1. Java Trouble Shooting（JTS）サイトを開設。（http://www.smg.co.jp/JavaTroubleshooting/index.html）
2004	Javaトラブルシューティングサービス「JaTS」を提供開始。
2008	1. Javaトラブルシューティングメールマガジンを開始。（https://www.smg.co.jp/seminar/JTSMM/index.html） 2. ENdoSnipe 3.4、3.5を矢継ぎ早にリリース 3. 構成管理SaaS「PROMA-C DevNavi 1.0」をリリース

2. JaTSとは、こんなサービスです。

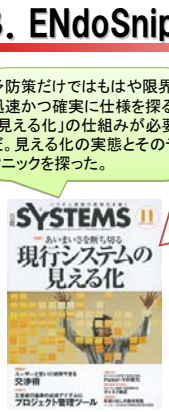
- 世にも珍しい、期間内解決保障型トラブルシュートサービス

期間内解決保障コースで解決できなかった場合は、「半額」をお返しします。

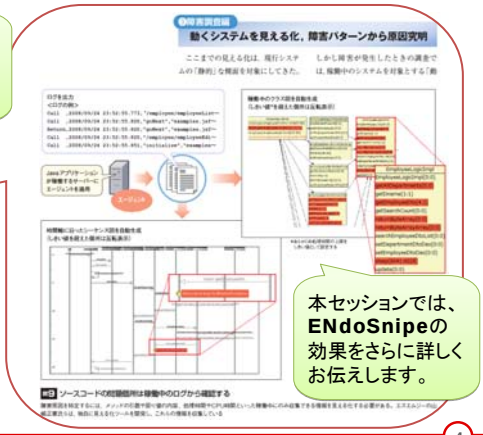
- 年間30件超のトラブルに対応しています。
- 現在のところ、解決率100%
（全く再現せず、調査終了となったものを除く。）
- 「JaTS」で蓄積したJavaトラブルシュートノウハウをパッケージ化したものが、ENdoSnipeです。

3. ENdoSnipeが日経SYSTEMS誌に掲載

予防策だけではもはや限界。迅速かつ確実に仕様を探る「見える化」の仕組みが必要だ。見える化の実態とそのテクニックを探った。



日経SYSTEMS
2008年11月号特集より



本セッションでは、**ENdoSnipe**の効果をさらに詳しくお伝えします。

4. 本セッションの内容

1. 現在のシステム開発の問題点
2. ENdoSnipeの優れた特徴
3. 事例のご紹介
4. 適用範囲
5. 利用シーン
6. 使いやすいライセンス方式と価格
7. 今後の展開

1. 現在のシステム開発の問題点

【1. 現在のシステム開発の問題点】

1. 現在のシステム開発の問題点

短納期化・コスト削減の影響で問題が洗い出されない

- 品質が犠牲になるため、プロジェクト終盤で大量の問題が発生する。
- 特に性能問題やメモリリークなどの長期運用で発生する問題が見過ごされる。

高レベルエンジニアが不足している

- ネットワーク、データベース、OS、Java VM・・・問題解決にマルチスキルが求められているが、そのようなエンジニアは希少。

効率よく問題を解析・解決する手法が不足している

- 大量のログ・データを、長時間かけて解析しなくてはならない（エンジニアが何週間も問題解決にあたるため、多大なコストが必要）。

（さらに）システムの構造・動作を把握する方法が存在しない

- 数年をかけて維持・保守を続けると、システムの全体を把握するためのドキュメントが存在しなくなる。

【1. 現在のシステム開発の問題点】

2. 例えばこんな問題はありませんか？

1. 問題が洗い出されない
   - ① パートナーが開発したシステムの品質が判断できない。（機能要件は満たしているようだが・・・）
   - ② 英語マニュアルを読み込む事なく、サンプルをコピーしたため性能が出ない。
2. 高レベルエンジニアが不足している
   - ① データベースチューニングの基本「実行計画の採取」ができるエンジニアは、全体の10%未満。
3. 障害解析に時間がかかる
   - ① 時間のかかるSQLを発行しているプログラムが特定できない。
   - ② ログをシーケンス図に起こすだけで数日を要する。

【1. 現在のシステム開発の問題点】

3. どうすれば良いのか？

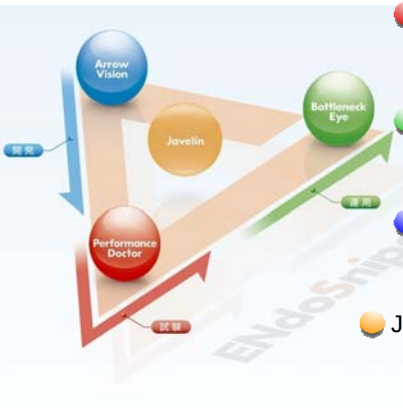
対処策	問題点
専門集団の組織化	• 組織内に力量を備えたエンジニアが見つからない。
教育による技術力の向上	• 芽が出るエンジニアは一部に限られる。（教育を受けるタイミングにもよる） • 効果を発揮するまでは、長い時間と費用がかかる。
外部サービスの利用	• システムの検証を請け負うサービスは多いが、責任範囲は障害の検出までであり、解決には至らない。

「技術者でなくともシステムの品質を確認できる方法」
「効率よく、技術者の力量を底上げする方法」が必要。

性能検証、障害解析から「属人性」を排除する

【1. 現在のシステム開発の問題点】

4. 3つの「見える化」で問題を解決



- PerformanceDoctor
  性能問題の原因を「見える化」
- BottleneckEye
  システムの構造をクラス図で「見える化」
- ArrowVision
  システムの動作をシーケンス図で「見える化」
- Javelin
  プログラムに手を加える事無く、これらの「見える化」を実現

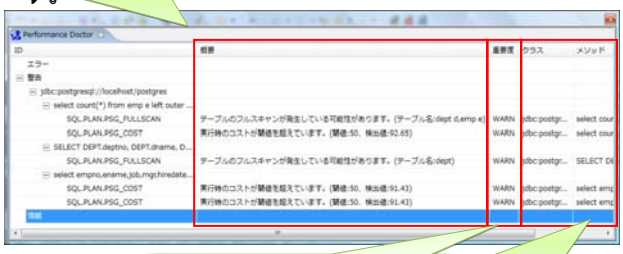
2. ENdoSnipeの優れた特徴

【2. ENdoSnipeの優れた特徴】

1. 効果的な3つの「見える化」で問題を解決

2. PerformanceDoctor

システムのボトルネックとなる危険性が高い処理と、その**理由**を示します。



問題は重要度で分類します。
メソッドまで特定しているので、すぐに原因を究明できます。

2. PerformanceDoctorのルール（1）

No.	カテゴリ	概要
1	Java	メソッドの呼び出し回数
2		メソッドのターンアラウンドタイム
3		スレッドのCPU使用率
4		スレッドの待機およびブロックの回数・時間
5		GCの実行回数・GCによる停止時間
6		メモリリークの発生
7		フレームワークの初期化に代表される処理の頻繁な実行（通常、システム起動時に1回行えば良い。頻繁な実行はCPUやディスクI/O等のリソース浪費に繋がる。）
8		HashMapアクセスでの無限ループ発生のリスク（複数スレッドの同時アクセスによるデータ破壊）

プロファイラやデバッガと異なり、処理に時間がかかった理由を端的に指摘します。

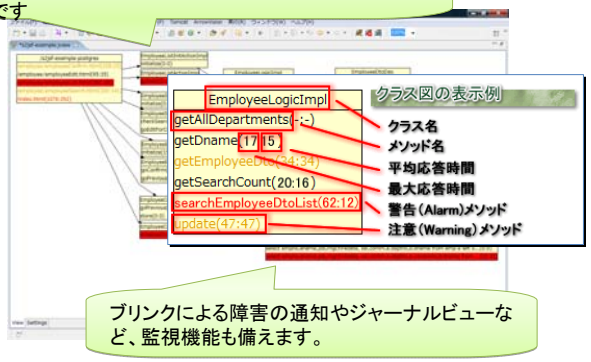
赤字で示したルールは、問題の検出と同時にアラームを発報します。

2. PerformanceDoctorのルール（2）

No.	カテゴリ	概要
9	JDBC	クエリ実行のターンアラウンドタイム
10		クエリの発行回数
11		同一クエリの発行回数
12		SQL大量発行の危険性
13	SQL	フルスキャンの実施
14		実行計画のコスト
15		1クエリ内でのJOINによるテーブルの結合数
16		1クエリ内でのor、unionの個数

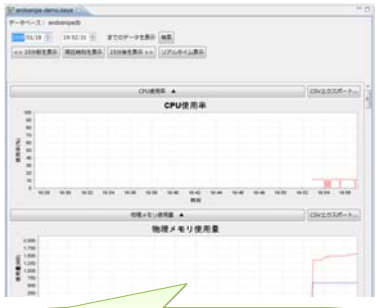
3. BottleneckEye～クラス図の動的生成

クラス図形式でシステムの構造をわかりやすく図示します。実際の動作に基づいた内容なので、正確かつ詳細です



ブリンクによる障害の通知やジャーナルビューなど、監視機能も備えます。

3. BottleneckEye～リソース監視

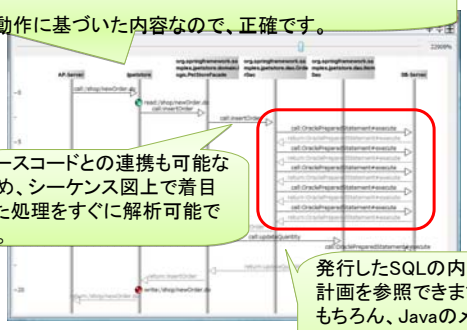


CPU使用率やメモリ使用量などを、リアルタイムにグラフ表示するので、負荷の発生状況が一目でわかります。

データはDBおよびCSVファイルに蓄積するため、レポート作成にも便利です。

4. ArrowVision

シーケンス図形式でシステムの動作をわかりやすく図示します。
実際の動作に基づいた内容なので、正確です。



ソースコードとの連携も可能なため、シーケンス図上で着目した処理をすぐに解析可能です。

発行したSQLの内容や、実行計画を参照できます。
もちろん、Javaのメソッド呼び出しのパラメータや戻り値にも対応しています。

【2. ENdoSnipeの優れた特徴】

5. Javelin

ソースコードに変更を加えることなく、情報の取得を実現

- Bytecode Instrumentation技術を採用(JDK 5.0以降を対象)
  - プログラムのロード時に情報収集用の処理を追加(コンパイル不要)
  - 低オーバヘッドを実現(設定次第だが、CPU使用率が数パーセント上昇するのみ)

トラブルシュータが行う調査手順を自動化(スキル習得が不要になります)

- 時間のかかるSQL→実行計画＋**スタックトレースの取得**
- 排他制御による長時間の停止→スレッドダンプの取得
- メモリリーク→コレクションサイズの追跡/メモリダンプの取得
- CPU使用率やメモリ使用量などの性能情報を収集

3. 事例のご紹介

【3. 事例のご紹介】

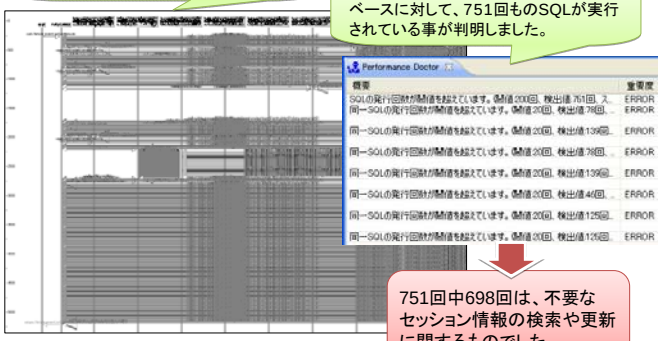
1. 事例一覧

No.	種別	規模	概要
1	Webアプリケーション	600 kL	結合試験完了後のシステムに対して、ENdoSnipeを用いた性能評価を実施。(結合試験では性能評価は未実施) **→データベース設計に問題がある事を発見。**
2	装置制御アプリケーション	200 kL	結合試験として、性能評価を実施。開発側でも性能評価は実施していたが、問題は検出されていなかった。 **→多数の性能問題を発見した。**
3	Webアプリケーション	200 kL	運用環境で発生しているメモリリークの原因究明にENdoSnipeを適用。その後、性能評価を実施。 **→メモリリークの原因を1日で究明。ボトルネックを多数発見した。(適用は「2年目」のSMG社員が担当)**

【3. 事例のご紹介】

2. 事例1:無駄なSQL実行①

ある画面を表示する処理のシーケンス図を、以下に示します。



1度、画面にアクセスしただけで、データベースに対して、751回ものSQLが実行されている事が判明しました。

751回中698回は、不要なセッション情報の検索や更新に関するものでした。

【3. 事例のご紹介】

2. 事例1:無駄なSQL実行②

No.	サイト	画面名	SQL実行回数	備考
1	サイトA(管理者向け)	ログイン画面(ID不正)	35	ログインに失敗しているのだから、IDおよびパスワード確認の1回で十分なはず。
2		画面C(一覧表示画面)	751	698回がセッション情報へのアクセス。
3		画面D(詳細表示画面)	180	詳細情報を1件表示するだけで、180回のSQL実行は多すぎる。
4		画面E	219	セッション情報へのアクセスは210回。
5				
6	サイトB(エンドユーザ向け)	画面G(一覧画面)	167	セッション情報へのアクセスは8回。(データ本体へのSQL回数が**159**回と、非常に多い。)
7		画面H(詳細表示画面)	636	セッション情報へのアクセスは2回。(データ本体へのアクセスが**634**回と、著しく多い。)

データベース設計、クエリ設計に問題のある機能が見つかる。

【3. 事例のご紹介】

2. 事例1:インデクス設計漏れ

No.	サイト	画面名	対象テーブル
1	サイトA(管理者向け)	ログイン画面	なし。
2		画面C(一覧表示画面)	マスタA
3		画面D(詳細表示画面)	なし。
4		画面E(検索画面)	なし。
5		画面F(一覧表示画面)	なし。
6	サイトB(エンドユーザ向け)	画面G(一覧画面)	マスタA, マスタB, マスタC, トランザクションD, トランザクションE

サイトBはインデクスの設計にも漏れが多い事が判明。

サイトBを請け負ったベンダ／エンジニアのデータベース設計の力量に、大きな問題がある事が、PerformanceDoctorの診断結果よりわかりました。

【3. 事例のご紹介】

3. 事例2:ロック取得待ち

排他制御による待ち時間が増加し、システムの応答性能が著しく悪化する事を、PerformanceDoctorが検出しました。

COD.THRD.BLK_TIME	スレッドのsynchronized待ちになった時間が閾値を超えています。(閾値:500msec、	検出値:5,606msec、	スレッドpoo...
COD.THRD.BLK_TIME	スレッドのsynchronized待ちになった時間が閾値を超えています。(閾値:500msec、	検出値:6,861msec、	スレッドpoo...
COD.THRD.BLK_TIME	スレッドのsynchronized待ちになった時間が閾値を超えています。(閾値:500msec、	検出値:7,985msec、	スレッドpoo...
COD.THRD.BLK_TIME	スレッドのsynchronized待ちになった時間が閾値を超えています。(閾値:500msec、	検出値:11,874msec、	スレッドpo...
COD.THRD.BLK_TIME	スレッドのsynchronized待ちになった時間が閾値を超えています。(閾値:500msec、	検出値:18,653msec、	スレッドpo...

```
thread.monitor.thread =
  ThreadId:22,
  ThreadName:http-8080-1,
  ThreadState:BLOCKED,
    at XXXXXXXX.aaaa(XXXXXXXX.java:97)
    at XXXXXXXX.bbbb(XXXXXXXX.java:60)
```

BLOCK!

```
thread.monitor.owner =
  ThreadId:23,
  ThreadName:Thread-9,
  ThreadState:TIMED_WAITING,
    at java.lang.Thread.sleep(Native Method)
    at XXXXXXXX$YYYY.cccc (XXXXXXXX.java:118)
```

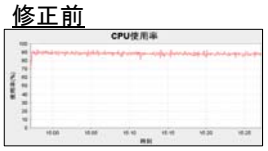
結合試験中は、スループットに注目していたため、このような応答時間の悪化に、試験担当者は気づく事が出来ませんでした。

ENdoSnipeは「停止したスレッド」「停止させたスレッド」双方の、ソースコード上の位置をレポートします。
(解析効率が大幅に向上します。)

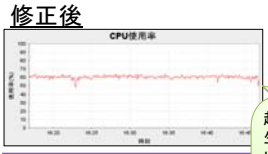
【3. 事例のご紹介】

3. 事例2:フレームワークの初期化でCPU使用率が激増

修正前



修正後




```
public void processEvent(Event e)
{
  GenericApplicationContext context =
    new GenericApplicationContext();
  try
  { ... }
  catch(Exception ex)
  {}
}
```

```
private GenericApplicationContext context =
    new GenericApplicationContext();
    ※起動時にのみ実行する。
public void processEvent(Event e)
{
  try
  { ... }
  catch(Exception ex)
  {
    context = new GenericApplicationContext();
  } ※例外発生時にのみ実行する。
}
```

起動時と例外発生時にのみ、フレームワークの初期化を行うように修正するだけで、CPU使用率が30%も低下しました。

同様な問題が発生するフレームワークを示します。
ENdoSnipeは、問題の発生する処理を自動的に検出します。

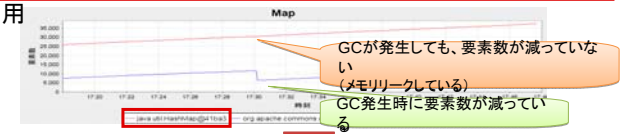
No.	フレームワーク	クラス
1	Spring	GenericApplicationContext
2	Jakarta HttpClient	HttpClient
3	Apache Velocity	Velocity
4	Seasar 2	SingletonS2ContainerFactory S2ContainerFactory

頻出問題。
(過去、4つのJaTS案件で遭遇。)

【3. 事例のご紹介】

4. 事例3:メモリリークの原因を0.5日で解明

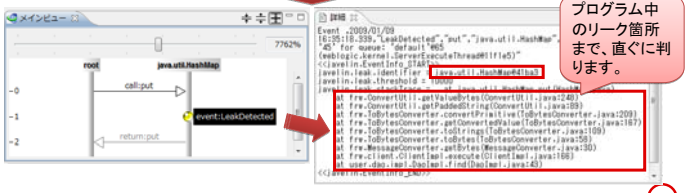
商用環境でメモリリーク発生の報を受け、ENdoSnipeを適用



GCが発生しても、要素数が減っていない
(メモリリークしている)

GC発生時に要素数が減っている

COD.THRD.MEM_LEAK	メモリリークが発生している可能性があります。...	WARN	java.util.HashMap	put



プログラム中のリーク箇所まで、直ぐに判ります。

【3. 事例のご紹介】

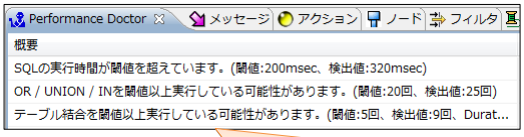
4. 事例3:3日間でボトルネックを多数発見

メモリリークの原因究明後、性能改善に向けて調査を実施

処理	エラー	警告
画面作成処理	5 件	8 件
ビジネスロジック	0 件	0 件
データベースアクセス	2 件	4 件

適用は当社(SMG)の2年目が行いました。

例えば・・・



問題のあるSQLを即座に絞り込む事が可能です。

【3. 事例のご紹介】

5. ENdoSnipeのコスト削減効果

事例2
装置制御アプリケーション

- 12件の障害を検出
  (メモリリーク等の重大な問題を多数含む)
- 顧客受け入れ試験やサービスイン後に発現した場合と比較し、およそ**1200万円の削減効果**を発揮。(顧客試算による)

事例3
Webアプリケーション

- メモリリークを0.5日で究明
- 当社(SMG)の2年目による3日間の適用で、改善ポイントを**20件**摘出。
- **エキスパートによる2カ月の解析**と同等以上の効果を発揮。

4. 適用範囲

【4. 適用範囲】

1. ENdoSnipeの網羅性

■ENdoSnipeの診断機能、監視機能の有効性を示すため、@ITの連載「現場から学ぶWebアプリ開発のトラブルハック」より事例を抜き出し、分析した結果を以下に示します。

ENdoSnipeは全10件（種類）のトラブル事例に対して、以下のような優れた検出・解析支援効果を示します。

① 原因個所を特定するもの ・・・6件
② データ取得が効率化されるもの ・・・4件
③ 効率化できないもの ・・・0件

広範な障害に対し、ENdoSnipeは有効に機能します。

【4. 適用範囲】

1. ENdoSnipeの網羅性～詳細（一部）

掲載回数	事象	原因	ENdoSnipe	
3	数人で利用しているときは、レスポンスが軽快だったシステムで、ユーザー数が増えてくると急激にレスポンスタイムが悪化する現象が発生した。	複数のスレッドから大量のログを単一のファイルに書き込んだため、長時間のロック待ちが発生した。	◎	長時間のロック待ちを検出し、アラームを発報する。 発生個所のスタックトレースを採取するため、原因個所の解析も不要。
4	結合試験工程で実施したパフォーマンステストにおいて、特定機能の応答時間が非常に長い。単体試験工程では、街頭機能の応答時間に問題は無かった。	1:n関連を持つテーブルを、それぞれ別個のSQLで検索していた。 そのため、n個のデータに対しn回のSQLを実行することになり、データの件数が増える度に、SQLの実行回数が著しく増加していた。	◎	データ量とともに実行回数が増加する危険性のある処理を検出し、アラームを発報する。 単体試験工程から実施する事で、問題の早期検出による手戻り工数の削減が可能。
5	システムを起動後、数分経つとOutOfMemoryErrorが発生してアプリケーションが落ちてしまう。	データベースから取得したデータをHttpSessionに登録していたため、解放されなかった。	◎	HttpSessionへの登録オブジェクト数およびメモリサイズの追跡を行う事が可能。 また、HttpSessionへの大量のデータ登録をメモリリークとして検出し、アラームを発報する。 スタックトレースも出力するため、原因個所の解析も不要となる。

@IT「現場から学ぶWebアプリ開発のトラブルハック」より

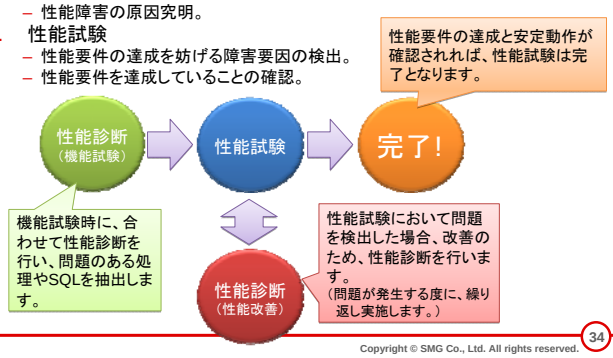
5. 利用シーン

【5. 利用シーン】

1. 性能診断および性能試験

■ 単体試験および結合試験工程における性能確認を、「性能診断」「性能試験」と2つの観点で行います。

1. 性能診断
   - 性能障害の原因となりうる問題の検出。
   - 性能障害の原因究明。
2. 性能試験
   - 性能要件の達成を妨げる障害要因の検出。
   - 性能要件を達成していることの確認。



6. 使いやすいライセンス方式と価格

【6. 使いやすいライセンス方式と価格】

1. ライセンス方式

■ライセンスはサーバ単位となっています。

① CPU数、コア数に関わりません。

② サーバのIP/MACアドレスをご登録頂きます。

– 登録は申請毎に利用期間を設定します。
  ✓期間終了後は、改めて申請する事で他のサーバ用で利用する事が可能になります。
– 申請は全てWebより可能になっています。
  ✓24時間、いつでも申請が可能です。
  ✓その場でライセンスファイルのダウンロードが可能です。

多くの案件に継続してご利用頂けるよう、柔軟に運用できるライセンス管理を行っています。

2. 価格

通常価格

価格	ライセンス形態
690,000円 ／本	サーバーライセンス

※クライアントライセンスは無料でご利用いただけます。
※仮想化環境では仮想化サーバOS毎にサーバライセンスが必要になります。
※初回購入特典として「初級トレーニング」が受講可能です。
※初年度は保守サポート費用は無料になっております。

ボリュームライセンス

▷まとめて20ライセンス以上のご購入の場合、ディスカウントいたします。

購入本数	1ライセンスの料金		初級トレーニング受講可能人数
1～19	690,000円	通常価格	ライセンス数×2 人
20～49	582,000円	15% OFF !	ライセンス数×1 人
50～99	501,000円	27% OFF !	ライセンス数×0.5 人
100～	427,500円	38% OFF !	ライセンス数×0.5 人

3. 関連サービス

No.	名称	価格	概要
1	トレーニング	¥ 60,000-（初級） ¥ 150,000-（上級）	初級：セットアップ・カスタマイズ 上級：障害解析・品質分析
2	導入	¥ 400,000- ～ （ご相談）	ENdoSnipeのインストール・セットアップを行います。
3	診断	¥ 800,000- ～ （ご相談）	お客様のシステムの性能診断を実施いたします。
4	性能改善	¥1,600,000- ～ （ご相談）	お客様のシステムの性能改善を実施いたします。
5	モニタリング	¥ 100,000-/月	3ヵ月以上の中・長期にわたって、運用中のシステムのモニタリングを実施いたします。
6	品質改善 コンサルティング	（ご相談）	システムの診断結果をもとに、お客様の組織の品質改善活動に対して、アドバイス致します。

4. 代理店のご紹介

No.	ロゴ	社名
1	Component Square, Inc.	株式会社コンポーネントスクエア
2	NEC Empowered by Innovation	日本電気株式会社
3	Human Crest	株式会社ヒューマンクレスト

7. 今後の展開

1. ロードマップ～より強力なソフトウェアを目指して

広範な問題に対応すべく、診断項目と監視項目を大幅に追加します。データをDBに蓄積するため、過去のデータをトレースする事が可能になります。

Ver. 4.0
- 診断項目追加
- 監視項目追加
- 履歴DBの導入

システム全体の品質レベルを判断し易くなります。（診断項目、監視項目の追加も、引き続き実施します。）

Ver. 4.5
- レポート機能
- ダッシュボード

Ver. 5.0
- セルフチューニング
- Webサービス化
- .NET対応

ユーザはエージェントをインストールするだけで、高度な診断を受ける事が可能になります。.NETに対応する事で、対象システムを広げます。

2009 → 2010

2. さいごに

■ ENdoSnipeは、日本のシステム開発の現場から生まれた国産のJavaシステム診断ツールです。

1. 現場の技術力向上の必要性、高まるオフショア活用圧力に対応すべく、開発しました。
2. 品質の確認に膨大な工数が費やされる現代のシステム開発方法に、大きな変化をもたらすものと確信しております。
3. ENdoSnipeを活用する事で、エンジニアのシステム理解が進み、技術力を高めるきっかけとなる事も願っております。

ぜひご利用を検討下さい。
（評価版もご利用頂けます）

ご清聴、有難うございました

ENdoSnipe

アンケートにご協力をよろしくお願いします。
（合わせて、本日の発表資料をお受け取り下さい。
第二会場前のブースでも資料の配布・説明を行っています。）

お問い合わせについては、下記までご連絡ください

連絡先
エスエムジー株式会社　小幡 有美
TEL　：045-476-3171
E-Mail　：endosnipe@smg.co.jp