Javaアプリケーションの開発から運用までの頼れる助っ人HeapStats

NTT OSSセンタが開発したHeapStatsは，Java Virtual Machine（JVM）が管理するヒープメモリ（メモリ領域）の詳細な状況を，Javaアプリケーションの性能に極力影響を与えずに取得できるOSS（Open Source Software）の監視・解析ツールです．HeapStatsは開発時のデバッグ・試験から運用時にいたるまで，迅速な問題解決を支援します．本稿では，HeapStatsの特長や適用シーン，解析の事例などを紹介します．

たかお　しんじ　すえなが　やすまさ
髙雄　慎二　/末永　恭正
くぼた　ゆうじ　わき　ひろあき
久保田　祐史　/和氣　弘明
ながふさ　まさひろ
長房　真広

NTT OSSセンタ

HeapStats開発のねらい

NTT OSSセンタでは，Java[*1]に関する技術支援の一環として，お客さまから寄せられる障害解析依頼に対応していますが，これらの依頼の中には，事象発生時に取得できた情報では不十分で，すぐに調査や解析を行えない場合があります．そのような場合は，お客さまに情報の再取得を依頼しますが，事象の発生頻度が低い場合などでは，再現ならびに情報取得に非常に多くの時間を要します．また，ヒープダンプ[*2]など従来の解析で必要とした情報は，取得時にシステムにかける負荷が大きく，かつ出力するファイルサイズも巨大なため，運用中のシステムでは取得が難しい場合も多くあります．ヒープダンプが取得できないときは，テキストファイルであるクラスヒストグラムで代替せざるを得ず，その場合はさらに手作業による解析に時間が必要なうえ，得られる情報も限られてしまいます（**図1** ㈎）．

そこでNTT OSSセンタでは，Javaアプリケーションのヒープメモリ（メモリ領域）の不足などの異常に起因する問題発生時に迅速な解析を可能とすることを主なねらいとし，Java Virtual Machine（JVM）監視・解析ツール「HeapStats」を開発しました．

HeapStatsは，Javaアプリケーションの性能に極力影響を与えずに低オーバヘッドで動作し，クラスごとのヒープメモリ使用量やクラス間の参照関係など，ヒープメモリの内部の状況に踏み込んだ詳細な情報を継続的に取得し，結果をグラフィカルに表示して分析することが可能です．このためシステム運用時であっても，障害解析に十分な情報を常に取得し続け，突然のトラブル発生にも即座に対応することができます（図1 ㈏）．

私たちはHeapStatsを多くのプロジェクトで使ってもらい，ユーザからのフィードバックを得て改善していくことを目的に，IcedTeaコミュニティ上にOSS（Open Source Software）として公開しました(1)．IcedTeaは数多くのLinuxディストリビュータに採用されているOpenJDKパッケージを開発しているコミュニティであり，多数の開発者と利用者が参加しています．現在，IcedTeaのHeapStatsプロジェクトは，NTT OSSセンタメンバ3名のコミッタ[*3]を中心に運営されています．

HeapStatsの概要

HeapStatsは，JVM監視エージェント（エージェント）とアナライザという2つのプログラムで構成されています（**図2**）．エージェントは障害解析に必要な情報を取得するプログラムです．エージェントは，Javaプロセス起動時に起動オプションを追加するだけで簡単に起動することができ，ヒープメモリ使用量やデッドロックの発生などを常時監視します．また，SNMP（Simple Network Management Protocol）トラップによってほかの運用監視ツール等にアラートを通知することができ，簡易な監視ツールとしても使用可能です．アナライザは，エージェントが取得したJVMの各種情報を表示し，問題の解析を支援するGUI（Graphical User Interface）アプリケーションです．

■エージェント

エージェントはヒープメモリに関する情報を取得します．その際，監視対

＊1　Javaは，Oracle Corporationおよびその子会社，関連会社の米国およびその他の国における登録商標です．
＊2　ヒープダンプ：JVMが管理するヒープメモリをファイルに出力させたもの．
＊3　コミッタ：ソースコードリポジトリを変更する権限を持つ，OSSプロジェクトの管理者．

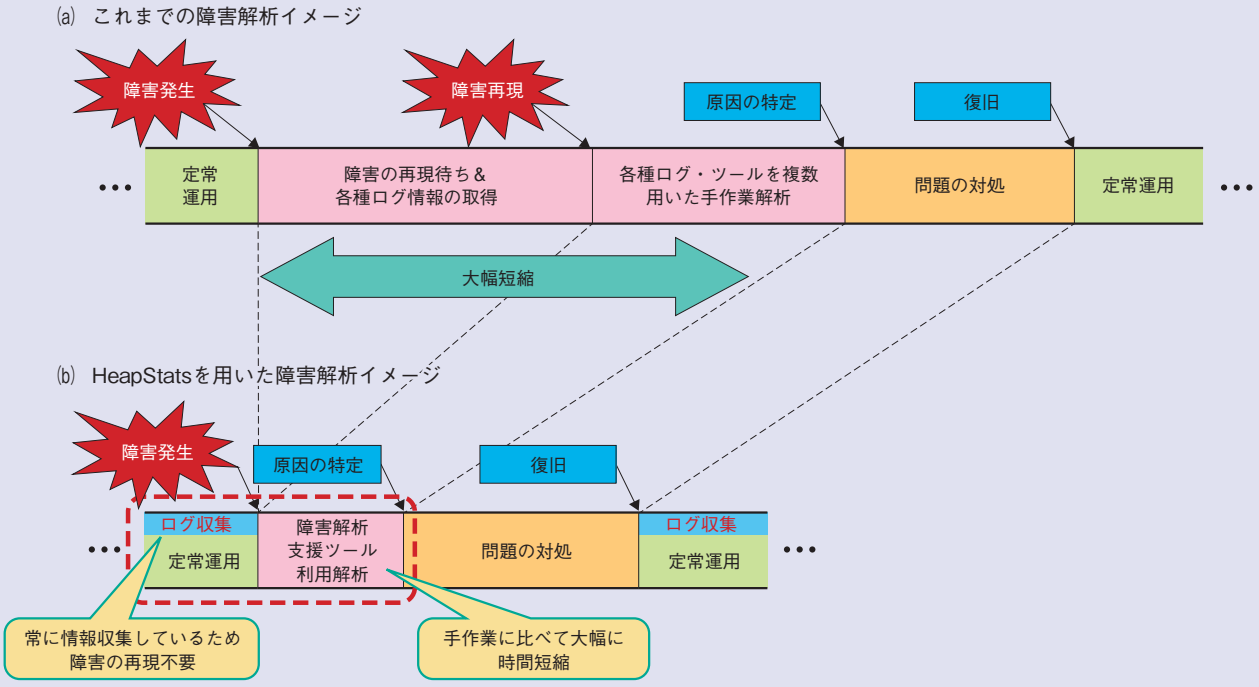


図1　HeapStatsを用いた障害解析イメージ

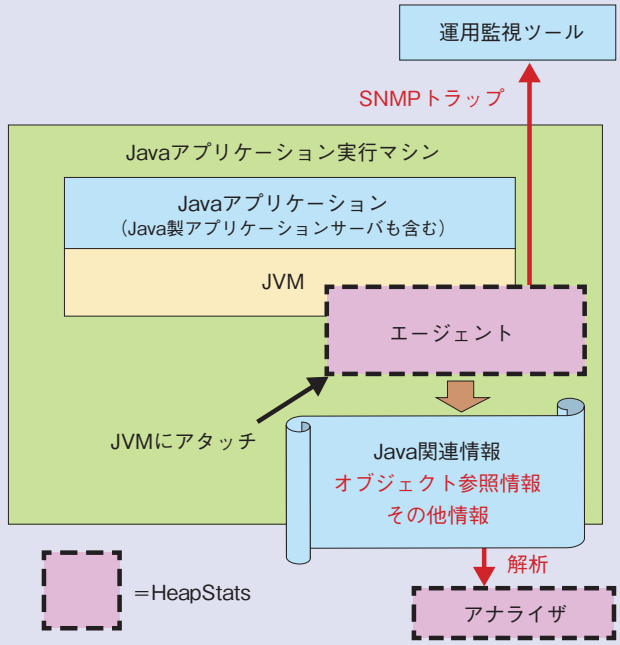


図2　HeapStatsシステム構成

象アプリケーションのオーバヘッドを非常に小さく抑えるために，以下に述べるJVMのガベージコレクション（GC: Garbage Collection）の実装に即してつくられています．

GCは，使われなくなったJavaオブジェクトが位置するヒープメモリを回収する際に，使用中のメモリを誤って回収しないように，使用中のオブジェクトを探索してマーク付けを行います（**図3**）．このマーク付け処理を行う関数内に，各種情報を取得する処理を割り込ませることで，エージェントとGCのマーク付けを一体で動作させています．その際，API（Application Programming Interface）関数コールのオーバヘッドをなくすため，直接ヒープメモリ上の必要な情報のアドレ

スを参照して情報を取得します．

エージェントが取得する情報は，解析に必要な情報に絞り込むことによって，ヒープダンプと比べはるかに小さいサイズに抑えています．GCごとに取得される情報はその都度ログ（スナップショットファイル）へ出力するので，メモリを圧迫することはありません．

これらの工夫によって，エージェントは低オーバヘッドでの情報取得を実現しています．Java実行環境のパフォーマンスを測定する標準的なベンチマークSPECjvm2008[(2)]を用いてHeapStatsの使用有無でスコアを比較したところ，オーバヘッドは4.51％にまで抑えられていました*4（**図4**）．これにより，ヒープダンプと異なり，システム運用中でも常に情報を取得し続けることが可能となっています．

エージェントは，ヒープメモリ以外にもサーバリソース関連情報など多様な情報を収集します．定常的な情報収集（**図5** ㈀）のほか，JVMでメモリ不足例外（OOME: Out Of Memory Error）やデッドロックが発生すると，さらに多くの情報を収集します（図5 ㈁）．

エージェントはx86およびx86_64系アーキテクチャ上のLinuxとJava SE 6以降で動作し，Red Hat Enterprise Linuxなどに対応したインストールパッケージが用意されています．

■アナライザ

アナライザは，エージェントが収集したスナップショットファイルやサーバリソース関連情報などを表示します．特にスナップショットファイルに記録された情報は，解析をしやすくするためグラフ化しています．**図6** ㈀は，ヒープメモリ利用状況の情報を時系列で表示し，クラスごとにオブジェクト数・ヒープメモリ使用量を表示した画面です．図6 ㈁は，クラスオブジェクトの参照関係（あるクラスのオブジェクトがどのクラスから参照されているか）を図示した画面です*5．

これらの情報はしばしば大量となり，従来のヒープダンプ解析ツール等では解析が難しい場合がありました．アナライザは，解析をしやすくするため，

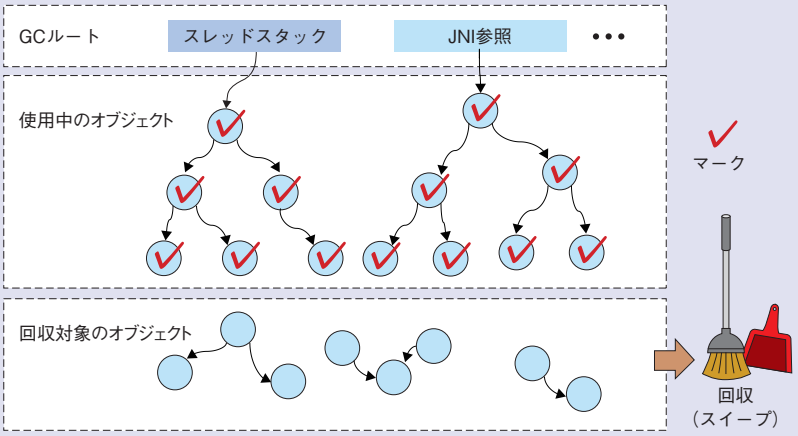
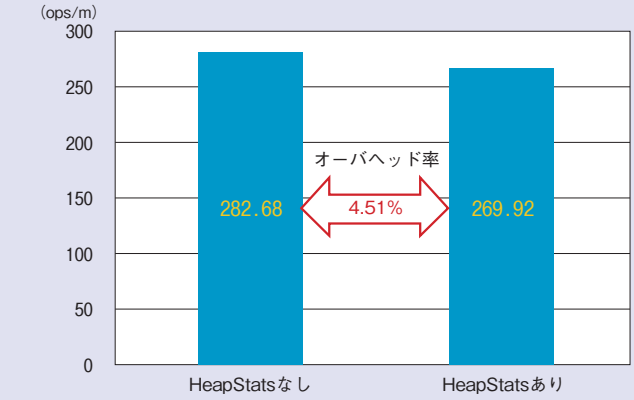


図3　GCのマーク付け



※測定環境
- ベンチマークツール：SPECjvm2008 1.01
- マシン：DELL PowerEdge R810（Xeon X7542，メモリ32 GB）
- OS：Red Hat Enterprise Linux Server release 6.3 x86_64
- Java：java-1.7.0-openjdk-1.7.0.25-2.3.10.4.el6_4.x86_64
- Java起動オプション：-Xms4500m -Xmx4500m -XX:+UseG1GC -agentpath：〈エージェントライブラリ〉

図4　HeapStatsオーバヘッド率（SPECjvm2008 Composite Result）

*4　アプリケーションとマシン環境により異なります．
*5　クラスの参照関係表示はHeapStats 1.1.0以降で利用可能です．

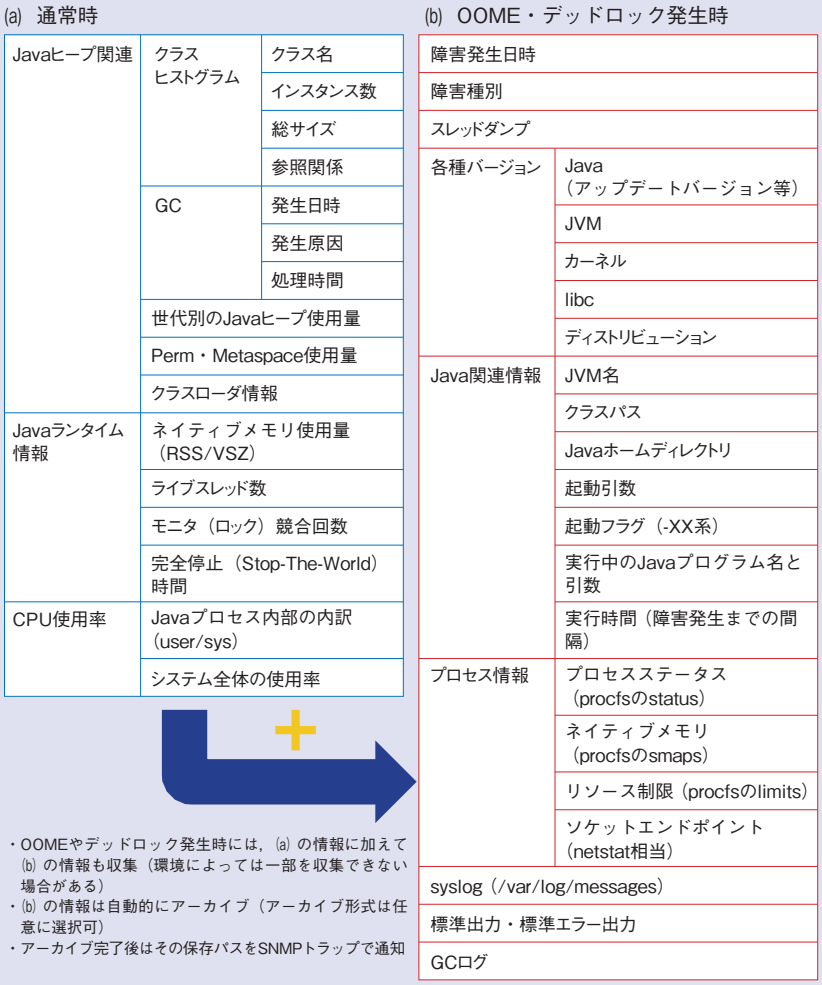


図 5　HeapStatsの情報収集項目

特定のクラス名称による絞込みや，クラスごとのオブジェクト数・ヒープメモリ使用量によるソート機能を備えています．

アナライザを使った解析の具体例は後述します．

HeapStatsの適用シーン

HeapStatsを開発段階から使用することで，ヒープメモリ関連の試験やデバッグを効率化できます．Javaプログラムはヒープメモリ管理をJVMにゆだね，プログラムがメモリの明示的な解放を行わなくてもよい仕組みを持っています．しかし，プログラムが意図せずオブジェクトを参照し続けると，JVMがメモリを解放せずヒープメモリ使用量が増加し，ヒープメモリの空き領域が不足する不具合（メモリリーク）が起こり得ます．HeapStatsのヒープメモリ解析機能は，メモリリークを早期に検出するうえで役立ちます．

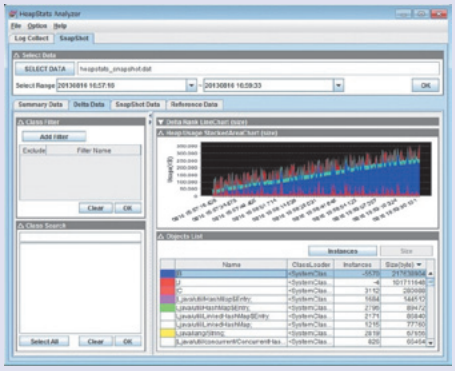
通常，メモリリークは開発の試験段階で発見されますが，長い時間をかけて少しずつ進行する場合や，何らかの操作がトリガとなって発生する場合など，運用段階で顕在化することもあります．また，メモリリーク以外にも，設計時の想定を超えた多数の処理や大きなヒープメモリを要する処理が発生した場合にもヒープメモリが不足します．これらは，ユーザ数や蓄積データの増加等が関係するため，一定の運用期間後に顕在化する可能性があります．したがって，十分なデバッグと試験を経てリリースされたプログラムであっても，運用段階での問題発生を視野に入れた監視が必要なので，HeapStatsは運用段階でも効果を発揮します．

SNMPを通してHeapStatsと運用監視ツールを連携させた利用イメージを**図 7** に示します．特定クラスによるヒープメモリ大量使用など，ヒープメモリの詳細な状況に踏み込んだ障害の予兆検知と，発生時の迅速な対応が可能になります．

解析事例

次にNTT OSSセンタ内の検証において発生した不具合事象の原因を，早期に突き止めた事例を紹介します．HeapStatsを導入したあるシステムの検証中に，Major GC[*6]が頻繁に発生するようになり，パフォーマンスが低下する事象を確認しました．ヒープメモリ利用率積み上げグラフ（**図 8**）から，プログラムがヒープサイズに対し

*6　Major GC：JVMがヒープメモリ全体を対象として回収を行う処理．



(a) オブジェクト数・ヒープメモリ使用量表示

(b) クラスオブジェクトの参照関係表示

図 6　HeapStats分析画面

てかなり多くのメモリを恒常的に使用しており，そのうち多くの割合をbyte型の配列が占めていることが分かりました．次にクラス間の参照関係図（**図9**）を表示させたところ，アプリケーションサーバの冗長構成（クラスタリング）のためにアプリケーションサーバどうしが交換するメッセージのクラスが大量にbyte型の配列を使用していることを突き止めました．それにより，クラスタリングに関連する設定のチューニングに集中した調査を行い，早期に原因を特定することができました．もしもHeapStatsを使用しなければ，クラス間の参照関係を知ることが難しいため，byte型配列が使用されている原因が分からず，原因特定までにより長い時間を要したものと思われます．

今後の展開

今後は，より多くの開発・運用プロジェクトにHeapStatsを理解してもらえるよう努め，導入いただいたプロジェクトの課題解決やTCO削減に貢

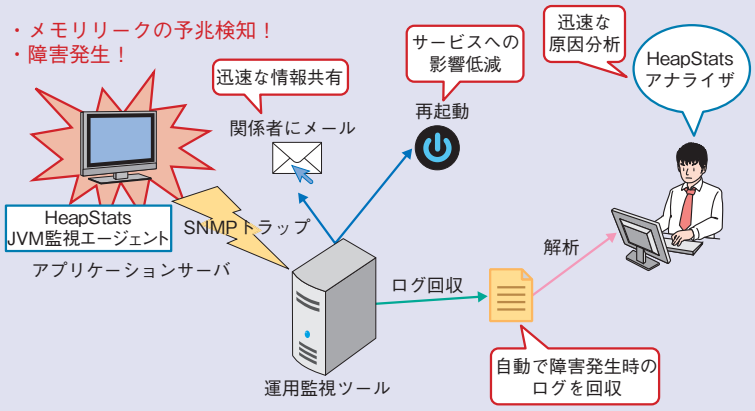


図 7　運用監視ツールとの連携イメージ

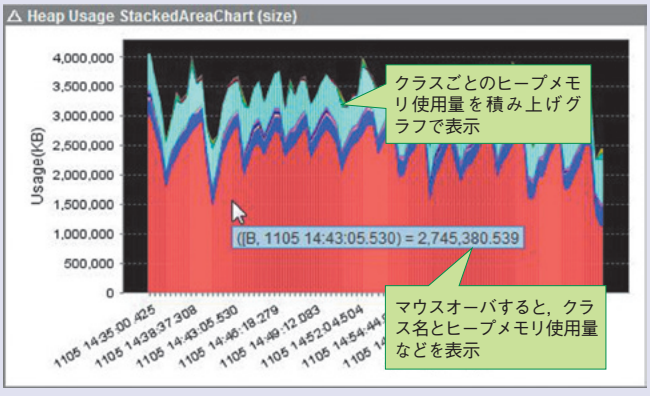


図 8　積み上げグラフ

コラム

HeapStats 導入に向けて

NTTコムウェア　エンタープライズビジネス事業本部　担当課長　大江　孝

NTTグループ会社がサービス提供しているシステムの更改にあたり，業務アプリケーションを再構築し，Java製の全文検索エンジンやJBossなどのOSS製品を導入して更改することとなったため，Javaヒープメモリを監視項目の1つとして重要視していました．

今回，OOMEやデッドロックが発生した際のJVM情報が即時に取得でき，かつ低オーバヘッドで動作するということに期待して，HeapStatsを採用しました．

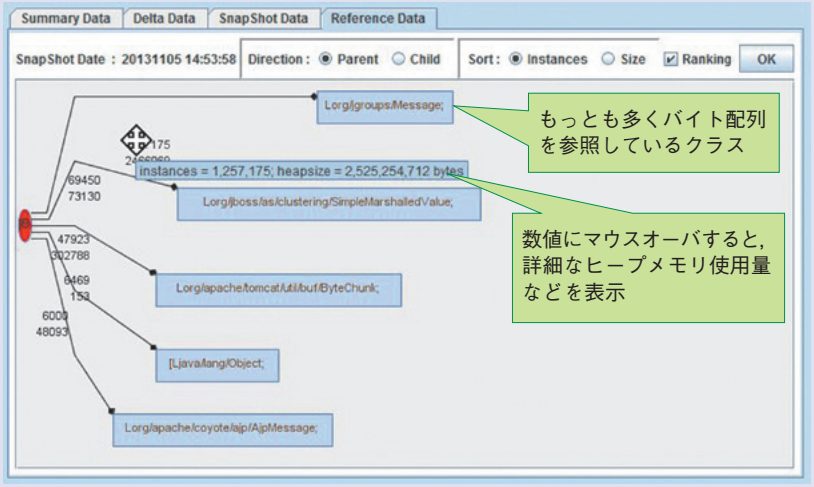


図 9　クラス参照関係グラフ

HeapStats は，IcedTea にある HeapStats プロジェクトのサイトから，ツールをダウンロードしてお試しいただけます．また OSSVERT（OSs Suites VERified Technically）に同梱されており，NTT OSS センタで問合せサポートを行っていますので，ぜひ活用をご検討ください．

献していきたいと思います．また，ユーザからのフィードバックを基にIcedTeaコミュニティ上で品質改善や機能追加，さらにはほかのOSSの解析ツールとの連携や，性能面のさらなる改善などを検討していきたいと考えています．

■参考文献

⑴ http://icedtea.classpath.org/wiki/HeapStats/jp
⑵ http://www.spec.org/jvm2008/

（左から）長房　真広／末永　恭正／
高雄　慎二／和氣　弘明／
久保田　祐史

HeapStatsは，突然訪れるトラブルに迅速に対応できる手段を提供し，Javaシステムの開発から運用まで「もしも」のときの助っ人として，システムの安定稼動に貢献します．HeapStatsの強力な情報収集・解析機能をぜひご活用ください．

◆問い合わせ先

NTT OSSセンタ
応用技術ユニット
TEL　03-5860-5055
FAX　03-5463-5491
E-mail　oss-contact@lab.ntt.co.jp