JENNIFER
JENNIFERSOFT, Inc.

www.jennifersoft.com

Case Study:

Out of Memory (OOM) Detection and Analysis

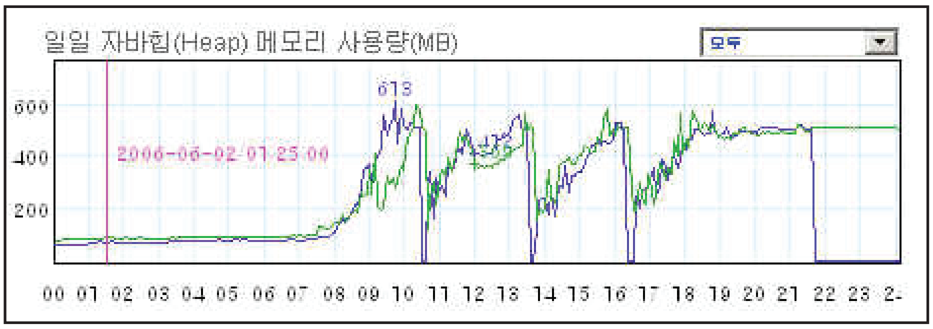
Background

"A" System was developed as portal system for "M" City Hall employee for use of daily operations, and is accessed by more than 4,500 users. "A" System is a large scale system with max concurrent users count of approximately 5,000 users and hourly visiting users count is approximately 10,000 users per hour.

"A" System is experiencing performance problem where Out-of-Memory error is occurring randomly and Full Garbage-Collection (GC) is executed frequently, leading to frequent system downtime.

System administrator has narrow down the cause of the performance problem to one of the newly added Groupware servers and attempted to identify the cause of performance problem but neither the administrator nor Groupware vendor could not find the cause of performance problem.

Analysis with JENNIFER



[Pic 1 : Java Heap Memory Monitoring]

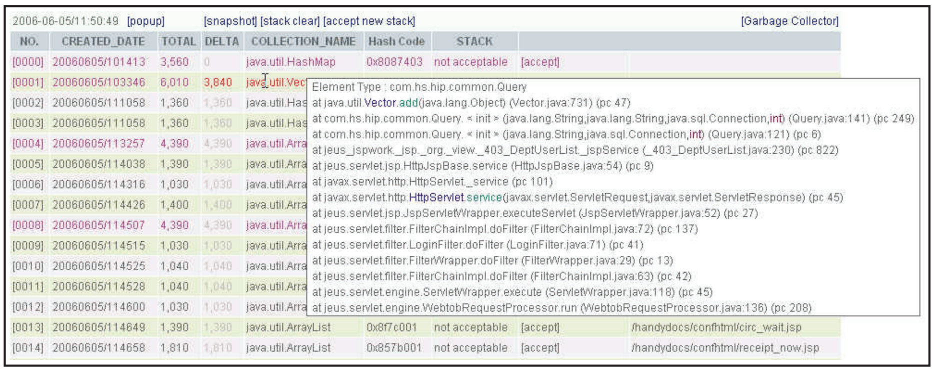
After JENNIFER was installed, system administrator observed that Java heap memory usage increased suddenly approximately every 3 hours, leading to OOM errors and system downtime (See above graph).

To investigate the problem further, system administrator enabled JENNIFER's Collection Monitoring capability to gather more in-depth information.

Collection Monitoring

Many web applications are using java collection (vector, hashtable etc…) to manage large amount of data. When Java collection fails to properly create and delete objects or cannot properly manage element insert logic or delete logic, it may lead to Out of Memory errors.

JENNIFER's Collection Monitoring traces all Java collection object's creation and deletion activity within heap memory, allowing system administrator to easily trace the cause of the OOM errors.



```
2006-06-05/11:50:49  [popup]          [snapshot] [stack clear] [accept new stack]                                  [Garbage Collector]
NO.     CREATED_DATE     TOTAL  DELTA  COLLECTION_NAME    Hash Code   STACK
[0000]  20060605/101413  3,560  0      java.util.HashMap  0x8087403   not acceptable  [accept]
[0001]  20060605/103346  6,010  3,840  java.util.Vec   Element Type : com.hs.hip.common.Query
[0002]  20060605/111058  1,360  1,360  java.util.Has   at java.util.Vector.add(java.lang.Object) (Vector.java:731) (pc 47)
[0003]  20060605/111058  1,360  1,360  java.util.Has   at com.hs.hip.common.Query. < init > (java.lang.String,java.lang.String,java.sql.Connection,int) (Query.java:141) (pc 249)
[0004]  20060605/113257  4,390  4,390  java.util.Arra  at com.hs.hip.common.Query. < init > (java.lang.String,java.sql.Connection,int) (Query.java:121) (pc 6)
[0005]  20060605/114038  1,390  1,390  java.util.Arra  at jeus_jspwork._jsp._org._view._403_DeptUserList._jspService (_403_DeptUserList.java:230) (pc 822)
[0006]  20060605/114316  1,030  1,030  java.util.Arra  at jeus.servlet.jsp.HttpJspBase.service (HttpJspBase.java:54) (pc 9)
[0007]  20060605/114426  1,400  1,400  java.util.Arra  at javax.servlet.http.HttpServlet._service (pc 101)
[0008]  20060605/114507  4,390  4,390  java.util.Arra  at javax.servlet.http.HttpServlet.service(javax.servlet.ServletRequest,javax.servlet.ServletResponse) (pc 45)
[0009]  20060605/114515  1,030  1,030  java.util.Arra  at jeus.servlet.jsp.JspServletWrapper.executeServlet (JspServletWrapper.java:52) (pc 27)
[0010]  20060605/114525  1,040  1,040  java.util.Arra  at jeus.servlet.filter.FilterChainImpl.doFilter (FilterChainImpl.java:72) (pc 137)
[0011]  20060605/114528  1,040  1,040  java.util.Arra  at jeus.servlet.filter.LoginFilter.doFilter (LoginFilter.java:71) (pc 41)
[0012]  20060605/114600  1,030  1,030  java.util.Arra  at jeus.servlet.filter.FilterWrapper.doFilter (FilterWrapper.java:29) (pc 13)
                                                       at jeus.servlet.filter.FilterChainImpl.doFilter (FilterChainImpl.java:63) (pc 42)
                                                       at jeus.servlet.engine.ServletWrapper.execute (ServletWrapper.java:118) (pc 45)
                                                       at jeus.servlet.engine.WebtobRequestProcessor.run (WebtobRequestProcessor.java:136) (pc 208)
[0013]  20060605/114649  1,390  1,390  java.util.ArrayList  0x8f7c001  not acceptable  [accept]  /handydocs/confhtml/circ_wait.jsp
[0014]  20060605/114658  1,810  1,810  java.util.ArrayList  0x857b001  not acceptable  [accept]  /handydocs/confhtml/receipt_now.jsp
```

[Pic 2 Collection Monitoring]

After Collection Monitoring feature was been enabled, objects were carefully examine and system administrator observed that Vector Object increased steadily over time. To identify the applications which are using Vector Objects, stacktrace information was gathered from application and administrator quickly found that Query.java program was using the problematic vector objects.

Administrator took the stack trace information provided by JENNIFER and consulted with Groupware's engineer. After source-code analysis concerning vector objects, vendor's engineer discovered that vector object used by Query.java which is responsible for data control was set in "static mode" and "delete-element" logic was commented out. This meant that after each time GC is executed, vector was not deleted and it remained in memory. Also, element-delete logic was not enabled so number of element object continued to increase, resulting in elements piling up in memory, eventually leading to Our of Memory (OOM) errors.

Conclusion

It turned out that Groupware's developer had changed vector object in Query.java for debugging purposes then forgot to change the source-code back. In this case, the mistake wouldn't have been discovered in a test environment; this type of problem would only surface during production environment when there is multitude of requests being processed. To prevent and quickly resolve such case of performance problem, a real-time collection monitoring tool is necessary.

Key Message :

1. System "A" is experiencing performance problem where Out of Memory (OOM) error is occurring randomly and GC is executed frequently, leading to system downtime, unable to maintain normal operation.

2. Out of Memory error is frequently resolved with Real-time monitoring of collection object with tracing capability in production environment.

Note

JENNIFER Review Downloads:
http://www.jennifersoft.com/docs/apm-jennifer-installation--file-download.html

JENNIFER Introduction Download:
http://www.jennifersoft.com/docs/apm-jennifer-documents.html