CUHK ALE Middleware - Test Cases

Prepared By   :   Daiming Qu

Report No   :   CUHK-Middleware -TC

Version   :   1.0

Issue Date   :   31 August 2007

Table of Content

Revision History

Date	Ver./Rev.	Description	Author
31 Aug 2007	1.0	First Release	Daiming Qu

1 Introduction

This document details the test cases prepared for CUHK ALE Middleware, which are designed based on the state diagram of The Application Level Events.   The tests were conducted as listed in this document, and the corresponding results are listed for reference.

The objective of these test cases is to validate the workflow of the middleware conforms to EPCGlobal's The Application Level Events (ALE) Specification, Version 1.0.

2 Test Environment

Software	• CUHK ALE middleware • CUHK Tag Viewer • CUHK Reader Emulator • CUHK ECSpec Explorer (a simple software which implements all ALE logics) • System Internals' Process Explorer • Health Monitor from http://healthmonitor.zucchetti.com [GPL] • Apache Jmeter • MySQL Administrator
Operating System	Windows XP
Database	MySQL
Hardware Configuration	CPU: Pentium D 2.8GHz /w PAE (Physical Address Extension) Memory: 2GB Disk: 200GB

3 Functional Test Cases

The test cases are designed based on EPCGLOBAL ALE Specification Version 1.0, Section 8.1 ALE Main API Class, ALE state diagram. The tests are performed via CUHK ECSpec Explorer. Middleware state can be obtained from application server logs.

Case No.	Scenario	Expected Result	Actual Result	Remarks
1.	Define an ECSpec	ECSpec is defined, and middleware enters UNREQUESTED state for that ECSpec	Passed	NIL
2.	After test case 1, subscribe an defined ECSpec	ECSpec is subscribed, and middleware enters REQUESTED state for that ECSpec	Passed	NIL
3.	After test case 2, start CUHK reader tag emulation	Middleware enters ACTIVE state.	Passed	NIL
4.	After test case 3, wait until the ecspec reader cycle expire	middleware enters REQUESTED state for that ECSpec, and ECReport is received	Passed	NIL
5.	After test case 4, unsubscribe the ECSpec for all subscribers	middleware enters UNREQUESTED state for that ECSpec	Passed	NIL
6.	After test case 5, undefine ECSpec	middleware undefines the ECSpec	Passed	NIL
7.	After test case 1, start Reader Emulation and poll the ECSpec	middleware enters ACTIVE state	Passed	NIL
8.	After test case 7, send a STOP trigger	middleware enters UNREQUEST state	Passed	NIL

4 Performance Test Cases

| Case No.	Scenario	Expected Result	Actual Result	Remarks
1.	Test Duration: 24 hours Number of ECSpecs: 2 Number of Physical Readers: 4 Number of Logical Readers: 2 Tag Submission Rate: 100 tag/s, randomly to 1 physical reader every 100 ms	Resource consumption: CPU < 50% MEMORY < 512 MB	Passed	See Section 4.1.1 for details
2.	Test Duration: 24 hours Number of ECSpecs: 20 Number of Physical Readers: 4 Number of Logical Readers: 2 Tag Submission Rate: 100 tag/s, randomly to 1 physical reader every 100 ms	Resource consumption: CPU < 50% MEMORY < 512 MB	Passed	See Section 4.1.2 for details

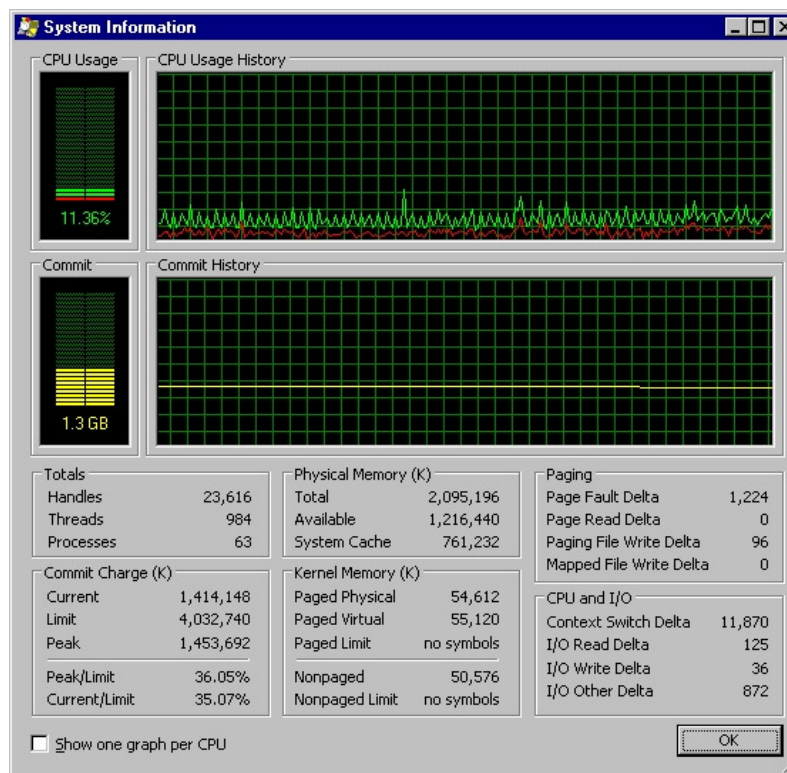### 4.1 Performance Test Case Detail

4.1.1 Test Case 1

4.1.1.1 Resource Consumption
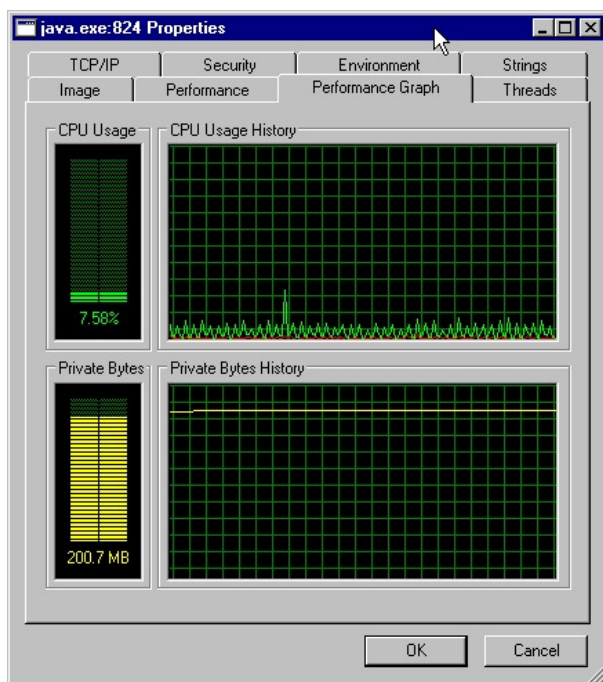
Resource Consumption of JBoss
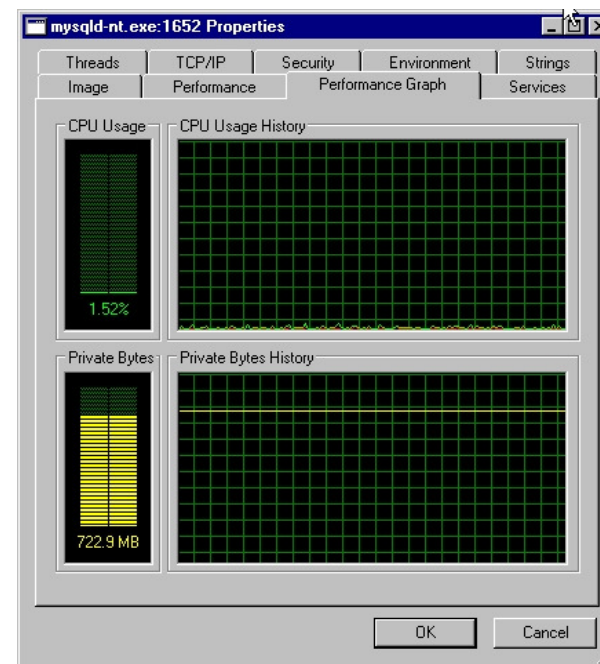
Resource consumption of MySQL

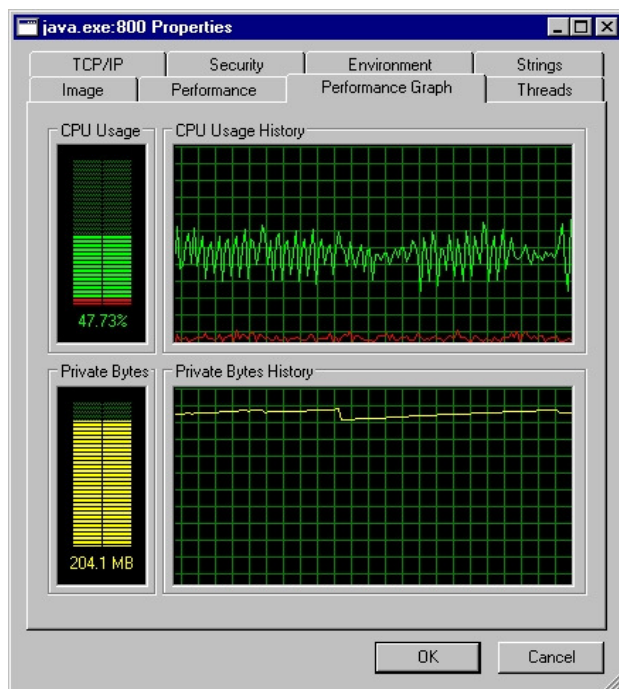Overall System Resource Consumption

4.1.1.2 Findings

- JBoss occupies less than 7 % (peak around 8%)
- MySQL is using negligible amount of CPU resources.
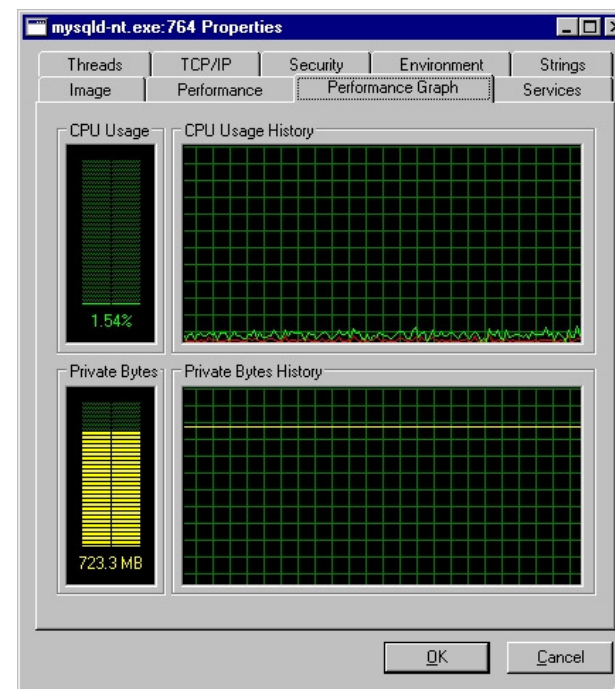- The system is stable over 24 hours.
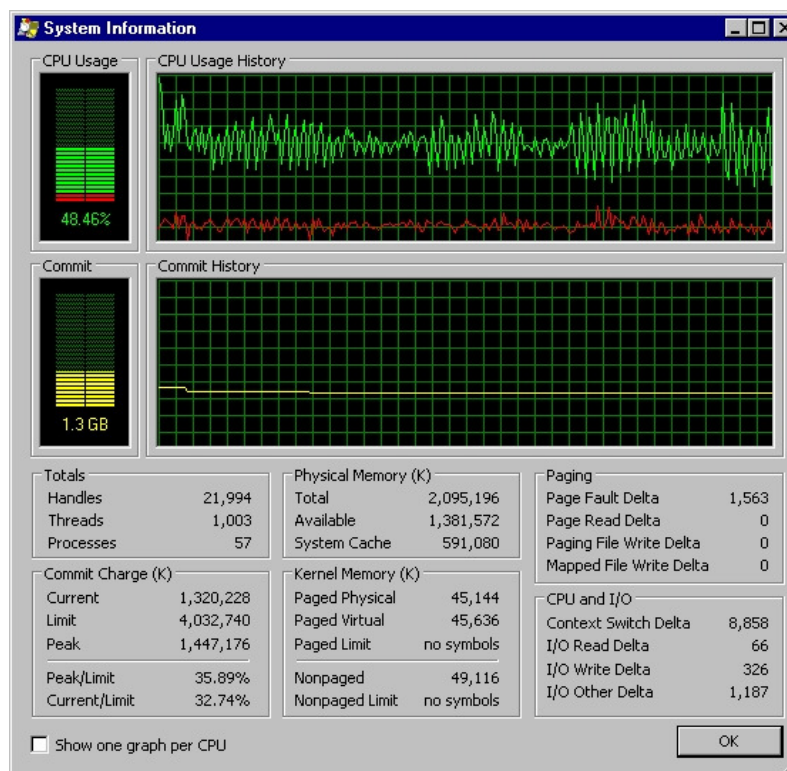
4.1.2   Test Case 2

4.1.2.1 Resource Consumption

Resource Consumption of JBoss

Resource consumption of MySQL

Overall System Resource Consumption

4.1.2.2 Findings
- CPU usage is proportional to the number of ECSpecs - 4% CPU is allocated for 1 ECSpec
- Memory consumption is constant

4.1.3 Overall Findings
- More tunings/disabling of the debugging logics are needed in order to have full performance of JBoss.
- JBoss and MySQL are very stable – there are cases that the Windows hangs, i.e. no console input is accepted, but JBoss and MySQL are still running.