eZ-Audit Performance Test Executive summary

The purpose of the Performance Test was to identify and resolve bottlenecks in the application, architecture, and infrastructure. The Performance Test effort also helped determine the optimal configuration of production environment. The Integration Partner Integrated Technical Architecture (ITA) team run four performance test cycles for eZ-Audit. Six scripts were tested for eZ-Audit: Case-team log-in, Search, Create Users, Annual submissions, Institution log-in, and Audit Clearance Document (ACD). There were several key bottlenecks that were identified and resolved. The resolutions improved the performance of the eZ-Audit application significantly. All issues identified during the eZ-Audit Performance Test have been resolved. The following actions were taken to resolve the issues:

Session Failover

The eZ-Audit sessions were not persistent on the database. If a clone (server) went down, users would lose their data, as the session did not persist in the database. The eZ-Audit team modified the code to make sessions persist in the database.

WebSphere Configuration Changes

Java Virtual Machine (JVM) was changed from 64 MB to 128 MB, the Java Database Connectivity (JDBC) connections were increased from 10 to 15, and Servlet threads were changed from 25 to 50 to improve the EZ-Audit performance. WebSphere e-fix to improve Struts performance and JDBC performance were applied.

Response Time Issue - Long Database Service Time

The page "co-team_login_home.jsp" displays all the submissions and users (in a drop down list). Since there can be 70 submissions and 300 users this page, it takes a long time to display. There are three business processes that will use this page. Code change was done to modify the rendering of Java Server Page (JSP). JavaScript was used instead of Java Code to generate assignee list. There were 300 annual submissions in the co-team lead's work queue which was causing a long response time. The data was removed from the database so only 78 annual submissions were on the case team lead queue.

Scalability

Two application clones were brought up for the scalability test. The users in to the application, the users were getting "session expired" message and failing. The application was using an API which only checked to see if there was an existing session for a particular user on one clone. It did not check to see if this session information was persisted in the session database. Thus, when switching over to the other clone, existing session information was lost and caused application to fail. A different Application Programming Interface (API) was recommended after seeing this issue. The application team used the new API which resolved this issue. A test was run after the code fix and it showed sessions were properly managed.

Closing the JDBC Connection properly

It was observed that the JDBC connections were not being dropped. Two steps were taken to correct this issue. First, the code that closes the connection was moved from the 'catch' block to the 'finally' block of the code. Thus, even if there is an exception in the application the connection will be release back to the WebSphere pool. Second, the ITA team applied a JDBC driver patch that included recent fixes on the driver.