



Case Study

Performance Reengineering for financial business application that monitors real-time ATM transactions for ATMs across the Globe.

1. Capacity of application increased from 100 to 700 virtual users.
2. Improving the transaction response time by up to 50% through suggested changes in the Application Architecture.
3. Saving the hardware up gradation expenses by effective tuning of web and database server.
4. Technology: Windows 2003 server, Database: SQL Server 2005, Web Server: MS IIS, Dot net 2.0 based application.

Abstract

CresTech helped a leading financial solution provider meet its performance objective of supporting 700 concurrent users on its ATM transaction monitoring application.

The recommendations provided by crestech team helped client save the cost of costly hardware up gradation through architectural changes and server tunings

The Company

The client is a leading MNC and business innovator in Financial Services Domain. Primarily the products of the company are being used for monitoring and controlling ATM transactions of different banks across the world. The products are also bundled by a leading hardware vendor selling server solutions across the globe.

The Business Problem

The client aimed to provide cutting edge financial solutions based on dot net technology. Client wanted their application to support 700 concurrent users and having a response time of less than 20 seconds for key transactions like Show Monitor View.

CresTech team was contacted to conduct a baseline test on the existing application and recommend changes\up gradations that can help them achieve their performance objectives.

AUT details

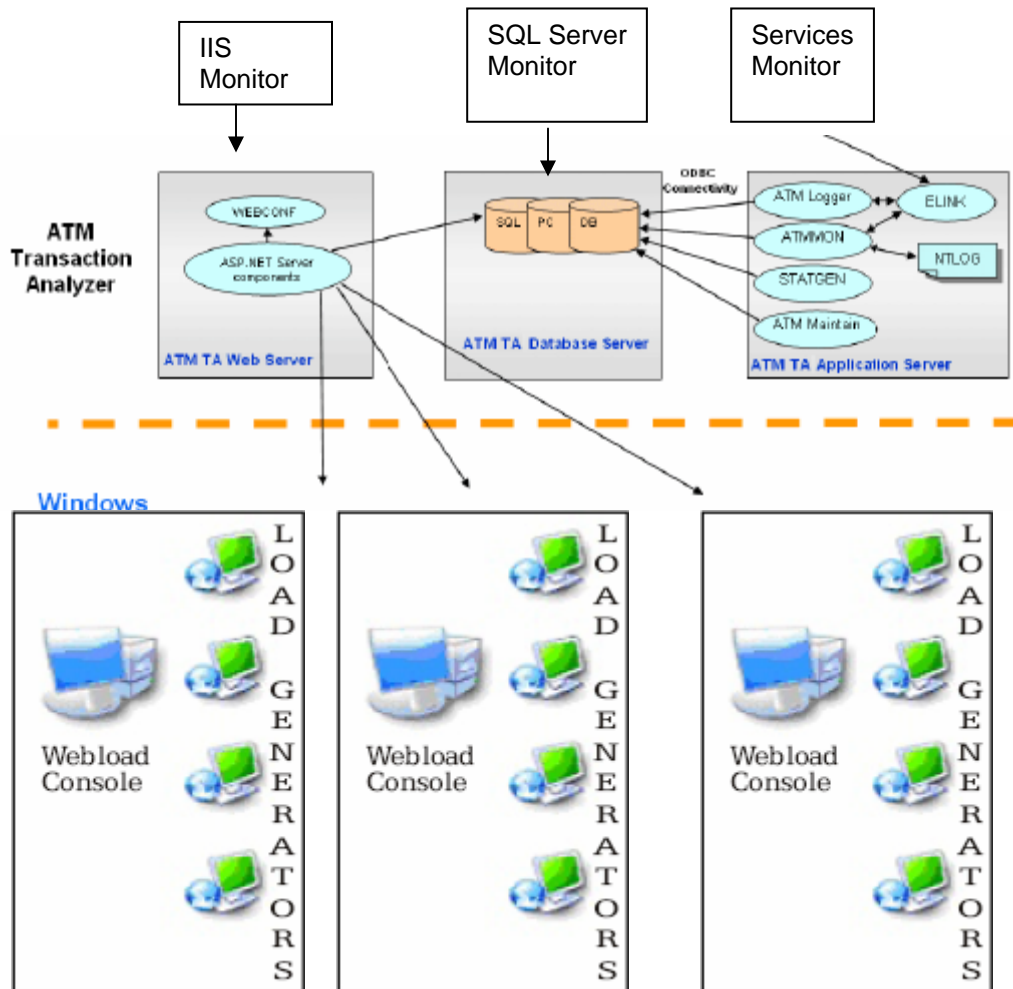
The application had database, Data retrieval Services and ELink communication process as its main components. ELink communication process extracts data from backend mainframe system and inserts in a table in a SQL server database. Data retrieval services then work on the data (in raw format) and after processing inserts in other tables defined by complex business logic. The presentation logic then work on it and present data to the user based on different views defined.



- **Database** The AUT Database is installed on a system which has **SQL Server** installed on it. It can reside on an independent system or coexists with other AUT components.
- **AUT Application Server**
The various AUT services, for example: ATMLOGGER-Service, ATMSTATGEN Service, ATMSMON Service and ELink. These services can reside on the AUT Database or on an independent system.
- **AUT Web-Server**
The AUT Web Server is a web based application used to monitor ATM transactions through web-browser like Microsoft Internet Explorer. The ATM Web Server should be installed on a system which has **Microsoft IIS 5.0** or above installed on it. The **ASP.NET** server component accesses the information from the AUT database and acts as a gateway to construct information that end-users view over the web.
- **ELink Communication Process**
ELink is an independent communications process residing on both HP Nonstop servers and the front-end AUT Application Server. AUT uses ELink as the communication channel between the AUT Application Server and the HP Nonstop host(s). Acting as a gateway for the nonstop environment, the ELink host running on the nonstop system passes structured information to the AUT Application Server through the ELink service. It is the middleware between the backend and front-end components of AUT. ELink configuration is handled through the **ELKCONF** file on the HP Nonstop server and a configuration file on the AUT Application Server.

Test Environment

The exercise was conducted using Mercury Interactive LoadRunner 8.0. The LoadRunner (LR) Probes were deployed on the Servers and the data collected through LR Controller Console.



Approach

Analyzing the situation, team decided to conduct the tests in a phase wise manner

Phase 1— Deriving Test Scenarios based on the requirements and scripting

In this phase, Crestech team derived test scenarios and user workflows by interacting with different stakeholders in the performance exercise and gleaning server logs to understand the existing usage patterns.

The derived user models acted a starting point for scripting and determined the scripts structure for maximum reusability.

Phase 2—Benchmarking tests conducted

Test Scenarios were prepared in LoadRunner Controller by combining the user models derived in first phase with the scripts

The scenarios were then executed to collect the data points depending on the performance testing objectives. Also the scripts were executed a pre configured number of times to average out any inconsistencies because of occasional spikes in network/Server performance

The performance data for benchmarking tests were presented to the client and determined

Phase 3— Data Collection, Analysis and Bottleneck Identification

The data from different probes on the server was collected and analyzed by our Technology experts at the backend and possible bottlenecks identified.

Key Findings

First Test Run

The first test run indicated that the application could handle only 100 concurrent users. By analyzing the Web server logs, it was identified that the inability to scale was due to insufficient pool queued length of MSIS 6.0 web server and worker processes.

Second Test Run

Increasing the pool queued length to 10000 and worker processes to 100 on the web server increased the number of concurrent users to 250. However going above the user-load of 250 users started throwing server timeout error.

On further analyzing the database monitors logs; the problem was traced to generation of dead lock conditions on database. The services running at the backend and the user load executed queries without ATMTA when load of 500 concurrent users is applied.

Third Test Run

Implementing the database changes suggested by the Crestech Team increased the performance of application by up to 700 virtual users at the same time reducing the response time of key transactions up to 4 times of its previous value

Key Suggestions

Web server Configuration changes:

Suggested configuration changes on web server e.g. increasing the pool queued length to 10000 and worker processes to 100 to name a few.

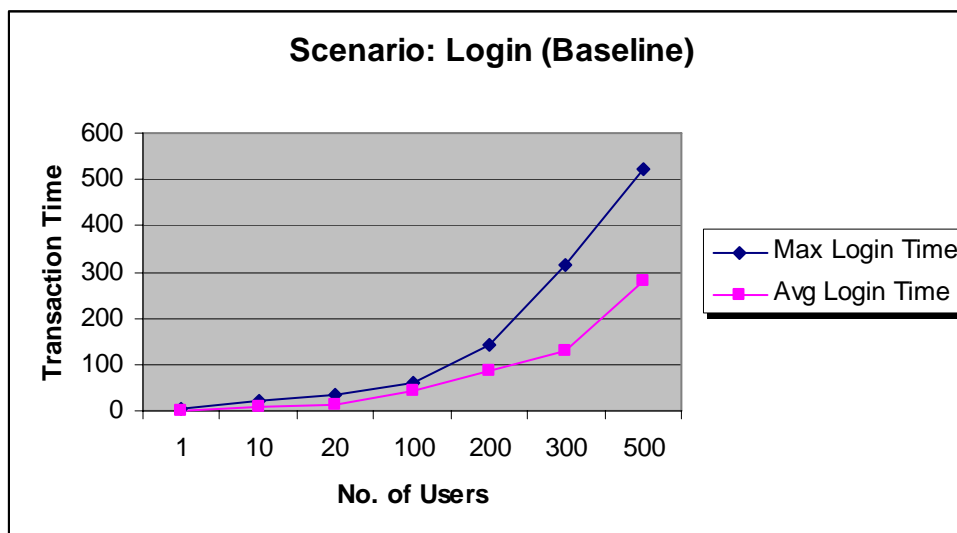
Application changes

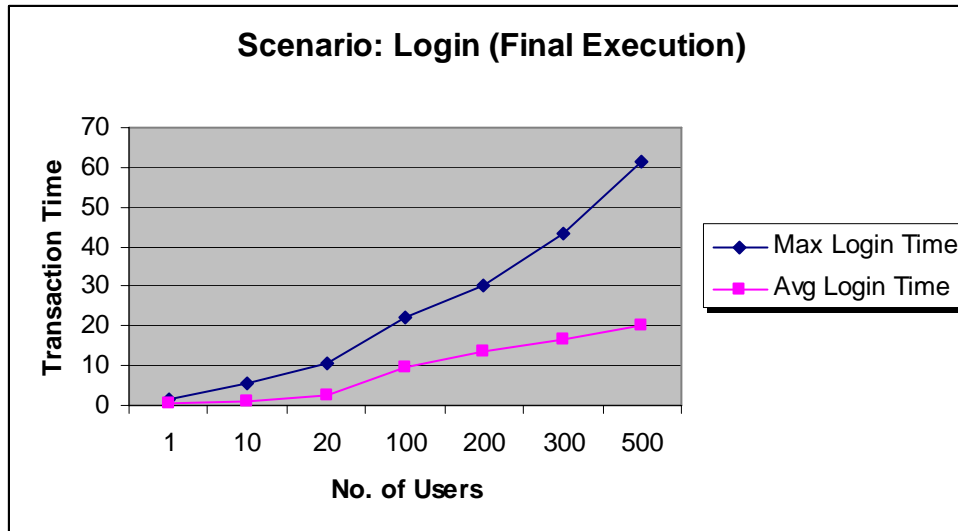
Suggested changes at the Architectural and component level of the application to enhance the performance. E.g. optimizing the data layer of the application to reduce the data I/O operations. Segregating the OLAP and OLTP operations to boost the transaction performance with the cost of a slight window time to show results.

Database design changes

These included suggestions to execute queries without Locks where the data conflict had the least possibility. Using stored procedures instead of firing direct queries to database. Changing the indexing structure for the optimization of search queries.

Test Execution and Result





Business Value

- Meeting Performance objectives for the application helped our client project their solution to HP (the hardware vendor) and ultimately edge out other competitors based on performance
- Scalability projections given by the crestech team helped client gear up for expected user load before hand preventing them last minute headaches and saving customer from all important data loss during peak load on the application