WebCAT: The Development, Performance Analysis, And Deployment of a Web-based Crime Analysis Toolkit


Abstract—The Systems and Information Engineering Department at the University of Virginia has teamed with the Virginia Department of Criminal Justice Services since 2000 to develop WebCAT, the Web-based Crime Analysis Toolkit. WebCAT provides crime analysts with a tool to access Virginia’s statewide crime data and effectively analyze it. The authors have prepared WebCAT for deployment by finishing its development, conducting an integrated system performance evaluation, and performing a life cycle analysis. Regarding deployment, the data acquisition, query time efficiency, and spatial mapping capabilities have been improved. A performance evaluation assessed the effectiveness of WebCAT’s developments to ensure that it is now a more efficient tool. Finally, the life cycle analysis provided a methodology for maintaining WebCAT after deployment. The work done this year guarantees that WebCAT is not only ready for deployment, but also that it will be a sustainable toolkit that will aid in crime prevention in years to come.

I. INTRODUCTION

Crime prevention is vital to providing a safe environment for citizens of Virginia. Information sharing and effective crime analysis tools are necessary for crime prevention. Crime analysts in Virginia and across the country, however, lack statewide data sharing and the necessary tools to easily analyze the statewide data. Currently, crime analysts in Virginia cannot easily obtain cross-jurisdictional crime data, resulting in a lack of communication between local law enforcement agencies [3]. The lack of crime analysis tools can be seen in the results of the 2004 Virginia Crime Analysis Network (VCAN) conference survey, which stated that 40% of crime analysts use no software programs to analyze data, and 47% of crime analysts use limited programs such as Microsoft Excel [2].

This project’s goal was to improve crime prevention in Virginia by improving the information sharing and analysis capabilities of WebCAT. To do this, the project’s objectives were the (1) development, (2) performance evaluation, and (3) deployment of WebCAT. Regarding development, the following improvements were implemented to increase the information sharing and analysis capabilities of WebCAT:

1. Data Acquisition Capabilities
2. Query Time Optimization
3. Spatial Analysis Capabilities

With these new developments, we conducted a performance evaluation of WebCAT to measure the changes in information sharing and analysis capabilities. Lastly, we performed a life cycle analysis to ensure that the improvements would be maintained as WebCAT is used.

The remaining sections discuss the data acquisition capabilities, query time optimization, spatial analysis capabilities, integrated system performance evaluation, life cycle analysis, and conclusion.

II. DATA ACQUISITION CAPABILITIES

The crime data acquired by WebCAT is a crucial component for its information sharing and analysis capabilities. If WebCAT can obtain thorough crime data in an efficient manner, it will increase cross-jurisdictional information sharing and enable more effective and precise analysis. To maximize WebCAT’s data acquisition capabilities, three alternatives were developed and assessed through a system evaluation to determine the most efficient strategy for acquiring detailed statewide crime incident data, specifically crime address information. The results were then implemented.

A. Alternatives

The following alternatives were considered: No Action, Direct Connection, and XML File Upload.

1. No Action: WebCAT obtains Virginia statewide crime data from the Virginia Department of Criminal Justice Services (VDCJS) in a monthly crime report following the National Incident Based Reporting System (NIBRS) format.
2. Direct Connection: WebCAT obtains Virginia statewide crime data by acquiring accounts of all Virginia jurisdictions’ crime database computers. WebCAT
administrators would remotely access each jurisdiction’s crime data and upload it to WebCAT.

3. XML File Upload: WebCAT acquires Virginia statewide crime data using an upload page on the WebCAT interface that would allow jurisdictions to upload XML crime data files to WebCAT’s database using the Global Justice XML (GJXML) schema. This schema is part of the Global Justice XML Data Model (GJXDM), which is a set of standards designed to allow justice agencies across the country share information effectively [6].

B. Analysis

To choose the optimal solution, the three alternatives were compared using data completeness, cost, data latency, and organizational compliance as metrics.

Regarding data completeness, taking no action yields incomplete data. The NIBRS monthly report contains limited crime incident details, specifically crime address data, which is crucial to spatial analysis. This would limit WebCAT’s future information sharing and analysis capabilities. Direct Connection and XML File Upload allow WebCAT to obtain all relevant crime data needed for cross-jurisdictional analysis.

Direct Connection’s initial costs are small, but the long term maintenance costs are large. Since Direct Connection uploads crime data from each jurisdiction’s computer, the process will be different for each jurisdiction because data is stored differently within each computer. The maintenance of WebCAT’s compatibility with each jurisdiction’s database computer as they change will be costly. XML File Upload has a greater initial cost because the GJXML schema is extremely complex, increasing its implementation costs. However, the XML File Upload, once implemented, is self-sufficient and requires limited maintenance because it places the burden of compliance on the jurisdictions. By taking no action, there would be no initial cost and WebCAT would continue to incorporate data within its database on a monthly basis which does not require much maintenance.

Taking no action results in high data latency because the crime data is obtained and incorporated within WebCAT’s database only monthly. XML File Upload is more efficient than taking no action because jurisdictions can upload their information at any time. This means that WebCAT’s data has the potential to nearly be in real-time. However, the data latency of XML File Upload is indirectly proportional to the frequency at which upload are made. Direct Connection results in the most efficient data acquisition because WebCAT has control over the frequency of obtaining. WebCAT data can be as close to real-time as needed by controlling the frequency of the Direct Connection uploads.

Regarding organizational compliance, Direct Connection is the most problematic. It is unlikely that all jurisdictions will provide WebCAT with user accounts to their databases for security reasons. Jurisdictions do not want other organizations to have the ability to manipulate and possibly corrupt their data. XML File Upload is more organizationally compliant than Direct Connection because the Department of Justice (DOJ) and the VDCJS are encouraging jurisdictions to switch to the GJXML schema as a standard for crime data storage [1]. In fact, projects done by certain states in the area of information exchange are now required to be GJXDM compatible in order to eligible for federal grants [6]. Nearly 2/3 of jurisdictions in Virginia have the GJXML schema implemented [3]. Taking no action is the most organizationally compliant alternative because the method has already existed for over four years with little complaints.

C. Results

The best solution is XML File Upload, based on a comparison of data completeness, cost, data latency, and organizational compliance with the other alternatives. Regarding data completeness, the XML File Upload provides WebCAT with the necessary crime data, including address data, needed to generate more precise queries and conduct spatial analysis. The minimal maintenance costs in the long term compensate for the larger initial costs. Although XML File Upload cannot guarantee near real-time data like Direct Connection can, it does provide this potential. In addition, it provides WebCAT with more timely data acquisition than taking no action. Lastly, XML File Upload is organizationally compliant because many of the jurisdictions in Virginia are already using the GJXML schema. The DOJ and VDCJS are encouraging the remaining organizations to do the same. For these reasons, the XML File Upload was implemented, providing WebCAT with the ability to efficiently obtain the detailed crime data needed in a cost-effective manner.

III. QUERY TIME OPTIMIZATION

Crime analysts must be able to obtain data in a timely manner in order to effectively analyze it. Query performance can deteriorate with the amount of crime data stored in the database and the amount of WebCAT users. Thus, WebCAT’s database and queries must be optimized. The Query optimization will contribute to the overall goal of improving crime analysis by ensuring that analysts can access the data in a timely manner.

A. Systematic Approach to Query Optimization

Database systems are difficult to model because they are all unique. According to Manegold, the necessary functions for modeling a database “highly depend on parameters that are specific to the very DBMS’s software and … to the hardware the database system is running on” [5]. Not only are the software and hardware different among the DBMSs, the database structure, data and workload for each system are different as well. With virtually every database system being different, no global approaches exist for the optimization of all databases.

In order to find the best optimization method for the WebCAT database system, a systematic approach was created. Using query duration run-time as a metric, the approach utilized a structured Design of Experiments (DOE) tree to test the appropriate indexing method as shown in Figure 1. Within the tree, the best alternatives were selected at each decision node, based on their query run-times. This approach systematically traversed the decision tree, yielding an optimal indexing solution.
Clustered and unclustered are two approaches for storing the data for retrieval in a database. The five data fields (ori, incdate, loc, weapon and offcode) were the different data fields by which the database can index the data for retrieval. Single and separate refers to having one index only or having multiple indexes. Indexes were created using multiple combinations of the above listed alternatives. The following section describes the steps taken at each decision node.

B. Decision Node Data Collection and Analysis

For the experiments and tests, queries that searched on the five commonly used data fields were created. These data fields are ori, weapon, offcode, location and incdate. Each data field had multiple options. There were 72 query types generated from the combinations of the five data fields and their options. The 72 query types were evaluated on each of the alternatives present at each decision node. For each indexing method, 100 iterations were generated for each of the 72 query types, resulting in 7200 query run-times. The best indexing method for each decision node was chosen based on an analysis of its 7200 query run-times.

The data collected for each alternative indexing method had Gaussian mixture distributions with large separations between them. 95% confidence intervals for the means and medians of each data set were used for analysis. The confidence intervals were compared to find the alternative path in the tree with the least time and greatest statistical separation.

![Fig. 1. Systematic approach to query optimization](image)

C. Results

The indexing method that was chosen to optimize WebCAT’s database system was the single index clustered by offcode and ori. This indexing method also proved significant compared to WebCAT’s original indexing schema as seen in Figure 2. The implemented indexing schema significantly reduces WebCAT’s query duration run-time, providing a much more efficient crime analysis tool.

![Fig. 2. Comparison of chosen index vs. Old Indexes and Null with the x axis being time in milliseconds](image)

IV. SPATIAL ANALYSIS CAPABILITIES

To improve crime analysis in Virginia, crime analysts need to have a tool that allows them to detect geographic crime patterns and hotspots. This ability will enable them to more effectively analyze crime data which will in turn aid in crime prevention. To improve WebCAT’s spatial analysis capabilities, two alternatives were developed and assessed through a system evaluation to determine the most efficient method for conducting geographic analysis. The results were then implemented.

A. Alternatives

The following alternatives were considered: No Action and Choropleth Mapping.

1. No Action: Spatial analysis is done by tabulating Virginia region crime totals using reports for each region. An individual crime query and report is needed for each region. The crime rates are then calculated and the information is manually plotted on a map to visualize the results.

2. Choropleth Mapping: Spatial analysis is performed using choropleth mapping. Choropleth mapping is a process by which sections of a map are thematically color coded based on attributes of each section. In WebCAT’s case, regions of a digital map of Virginia are color coded based on crime rates for a particular query.

B. Analysis

To choose the optimal solution, the two alternatives were compared using time, cost, and quality as metrics. Taking no action means that the process to obtain the relevant spatial analysis information is slow because it is manual and requires individual queries and reports for each region. Choropleth Mapping, however, allows the user to obtain the same results using a single query, which is more efficient.

Taking no action also proves to be more costly. Because crime analysts will have to spend more time generating the spatial results, they will be able to provide less help on other projects, which will lower productivity. Choropleth Mapping is very cost-effective because it limits the amount of work needed to obtain the necessary results. This means that crime analysts will have more time to work on other projects, which will help increase productivity.

Taking no action results in a final product that is inferior to that of Choropleth Mapping. Because the crime analysts must obtain the spatial analysis information manually
through calculations, there is much more room for error. Also, the final map could prove insufficient if the data is plotted by hand. The quality of the Choropleth Mapping final product is far superior. Choropleth Mapping calculates the results for the crime analyst. In addition, the final choropleth map is digital and provides much more analysis functionality such as a dynamic legend, labeling, and the ability to zoom in and out.

C. Results

The best solution is Choropleth Mapping, based on a comparison of time, cost, and quality with the other alternative. Choropleth Mapping allows the user to much more quickly obtain the geographic analysis results. The cost of Choropleth Mapping is also much lower than it would be if no action was taken because the time taken is reduced. Crime analysts will have more time to work on other projects, maximizing productivity. Lastly, the quality of the results from the Choropleth Mapping is superior because the results are computer-generated, meaning less room for error, and the final product has more functionality. For these reasons, Choropleth Mapping was implemented, providing users with the ability to cost-effectively conduct quality geospatial analysis in a timely fashion. Figure 3 below displays the final product of the Choropleth Mapping implementation.

![Fig. 3. Choropleth Map Page](image)

V. INTEGRATED SYSTEM PERFORMANCE EVALUATION

This evaluation was conducted to assess whether or not the developments made to WebCAT have improved the website’s overall performance. This integrated evaluation was conducted focusing on three types of users: local, regional, and state. Three types of evaluations were performed: XML File Upload Evaluation, Query Time Evaluation, and Spatial Mapping Evaluation.

A. XML File Upload Evaluation

GJXML’s flexibility facilitates WebCAT’s interoperability with approximately 2/3 of Virginia’s jurisdiction’s Record Management Systems (RMS) and has improved WebCAT’s file sharing capabilities [3]. Furthermore, ingesting additional data types (e.g. Arrest Date or Calls for Service) is simplified. With NIBRS data, analysts were limited to only policy level analyses. Using RMS data, analysts can now query on crime address data and detect geographic crime patterns at a much higher resolution. This means the integrated WebCAT system provides better crime analysis for all three types of users.

B. Query Time Evaluation

A query time analysis performed on the database showed that, by changing the indexes, there was significant change to the database query time. The purpose of this experiment was to test the speed of the website as a whole.

The experiment involved small, medium, and large queries representing the three user types. It was assumed that the state agency would have the largest query size because it was searching through the most data and the local agency would have the smallest size because the query would be the most specific. Large queries ranged from 17,000 to 29,000 records. Medium queries ranged from 8,000 to 11,000 records. Small queries ranged from 200 to 2,000 queries. Ten queries were created for each query size based on the assumption of what information an analyst would want to collect.

There were two websites that were tested: Old WebCAT (the original version) and New WebCAT (the updated version). There were also two sets of indexes tested: the original set of indexes and the updated set. Each query was performed on three different website/index combinations: Old WebCAT/Old Indexes (OW/OI), New WebCAT/New Indexes (NW/NI), and New WebCAT/Old Indexes (NW/OI). NW/NI was compared to OW/OI to determine which website ran faster. NW/NI was compared to NW/OI to determine if the indexes caused significant change.

The results varied for each query size. For small queries, the query time differences for each website combination were very small and not significantly different.

For medium queries, OW/OI experienced a problem. It did not have the capability of querying on multiple ORI numbers at once. The individual query times were fastest for OW/OI, but overall it took the longest time because each ORI number had to be run as a separate query. Therefore, NW/NI was the fastest time; it was about one to two seconds faster than NW/OI.

For large queries, the three queries were not significantly different from each other. However, OW/OI was the fastest and NW/NI was the second fastest. The difference in query times could be explained by the addition of code to New WebCAT, particularly the addition of choropleth mapping. This is shown by NW/OI being about four to five seconds slower than OW/OI. However, NW/NI did show a one to two second improvement over NW/OI which shows that the new indexes are helpful.

While the integrated WebCAT system had no significant change to the query times, the functionality was improved. This means that more effective crime analysis can be performed without the additional cost of time.

C. Spatial Mapping Evaluation

Choropleth mapping has made local, regional, and statewide spatial analysis easier and faster. Analysts can now visually represent data to see crime trends instead of comparing data reports and time charts. On a local level, analysts can visually track individual crime trends in their jurisdiction. On a regional level, analysts can compare the
crime distributions of neighboring counties and cities. From the statewide perspective, analysts can easily represent trends and hot spots over time.

VI. LIFE CYCLE ANALYSIS

In the near future, WebCAT will be deployed to local jurisdictions for use. As with any software system, maintenance issues must be considered. Without a thorough strategy, maintenance costs can be unnecessarily high and the system can fail. A life cycle analysis of the WebCAT system can help create a maintenance decision model for future maintenance scenarios. This analysis was performed using Markov decision processes to create the hardware and software maintenance decision models described in the following sections. These models will help ensure that WebCAT is maintained in the future.

A. Hardware Maintenance

WebCAT’s hardware system is observed at the beginning of each time period and classified into one of five possible states, ranging from excellent (x = 0) to inoperable (xi = 4). After each observation, one of three possible actions is taken: (1) do nothing, (2) upgrade by adding more memory, or (3) replace the system.

A policy, denoted R, is a rule for making decisions at each point in time. R is completely characterized by the decisions {d(R), d1(R), d2(R), d3(R), d4(R)}.

The model observes the following constraints:

1. If the system is currently in state i, choosing to upgrade by adding more memory (action 2) will result in the system moving to state (i-1).
2. If the system is in state 0 (excellent), no action is taken.
3. If the system is in state 4 (inoperable), the system is replaced.

When the system is in state i and decision di(R) = k is made following policy R, the system moves to a new state j, with known transition probability pij(k), for all i, j = 0, 1, 2, 3, 4 and k = 1, 2, 3. Thus, each of the nine policies shown in Table 1 below has its own state transition matrix.

TABLE 1

<table>
<thead>
<tr>
<th>HARDWARE MAINTENANCE POLICY OPTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>R0</td>
</tr>
<tr>
<td>R1</td>
</tr>
<tr>
<td>R2</td>
</tr>
<tr>
<td>R3</td>
</tr>
<tr>
<td>R4</td>
</tr>
<tr>
<td>R5</td>
</tr>
<tr>
<td>R6</td>
</tr>
<tr>
<td>R7</td>
</tr>
<tr>
<td>R8</td>
</tr>
</tbody>
</table>

When the system is in state i and decision di(R) = k is made following policy R, a known cost ci(k) is incurred. Upgrade costs were estimated to be $100, and replacement costs were estimated to be $3,000. The cost measure used to compare policies is the long-run expected average cost per unit time. For any policy, the long-run expected average cost per unit time, E(C), can be calculated from the expression

\[ E(C) = \sum_{i=1}^{n} C_i \pi_i, \]

where k = di(R) for each i, and (π0, π1, π2, π3, π4) represents the steady-state distribution of the state of the system under the policy R being evaluated. The policy that minimizes E(C) is sought.

Table 2 below shows the results of the analysis. Policy R0 is the optimal policy with a minimum expected cost of $48.50 per unit time. The hardware system should be replaced when in states 3 and 4 and upgraded when in good or fair condition states 1 and 2.

TABLE 2

<table>
<thead>
<tr>
<th>LONG-RUN EXPECTED AVERAGE COST PER UNIT TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>R0</td>
</tr>
<tr>
<td>R1</td>
</tr>
<tr>
<td>R2</td>
</tr>
<tr>
<td>R3</td>
</tr>
<tr>
<td>R4</td>
</tr>
<tr>
<td>R5</td>
</tr>
<tr>
<td>R6</td>
</tr>
<tr>
<td>R7</td>
</tr>
<tr>
<td>R8</td>
</tr>
</tbody>
</table>

B. Software Maintenance

The software maintenance decision model described herein is based on prior work done by Krishnan et al [4]. Many of the model assumptions still hold, though some modifications have been made.

The state of WebCAT’s software system is based on the sum of new and modified code in the system. Let X = \{x_i | i = 0, 1, 2, 3, ..., 13\} denote the set of 14 possible states of the system. The integer value assigned to the state of the system represents the degree of maintenance work performed in the system since WebCAT’s release. The state of the system x = k indicates that the sum of new and modified code in the system at time t is (k*10)% of the initial system size.

At the time of publication, WebCAT’s system size was approximately 35,000 lines of code, or 35 KLOC. WebCAT’s initial system size when it is released in summer 2006 is estimated to be 40 KLOC. Since WebCAT is a smaller software system than the system Krishnan uses for his case study, WebCAT’s distribution parameters were scaled linearly while its cost parameters were scaled by using the COCOMO software cost estimation model to calculate the proportion of reduced effort with a smaller system.
The review period $t$ for maintenance decisions is specified to be six months. At the beginning of each review period, the programmer reviews the state of the system and makes one of three possible decisions, denoted $d_i$, $i = 1, 2, 3$. The first alternative $d_1$ is to do nothing and just keep the system operational. The second alternative $d_2$ is to order a minor upgrade to the system by fulfilling the users’ maintenance requests in that review period. The third alternative is to replace the system, thus returning the system to State 0.

The distribution of maintenance requests for every six months was estimated as a normal distribution with a mean of 4 KLOC and a standard deviation of 1.2 KLOC. Since the volume of maintenance requests is likely to change with the state of the system, a state transition matrix can be constructed in the following way: For a given state $i$, let $D_i$ be a normal random variable with mean $\mu = 4 + 2i$ and standard deviation $\sigma = 1.2$. The transition probabilities are then calculated according to this shifting random variable.

A cost function is associated with each maintenance decision. These functions are dependent on the current state of the system, $i$, and the next state of the system, $j$. The cost functions for each maintenance decision are as follows:

The cost of doing nothing is:

$$OC(i, j) = 14 + 4.2i^2 + 1.75(j - i)^*10)^{0.75}.$$  

The cost incurred by performing a minor upgrade is:

$$MC(i, j) = 3.5 + 3.5i^2 + 1.75((j - i)^*10)^{0.95}.$$  

The replacement cost is:

$$RC(i) = 70 + 0.75(i^*10)^{0.75}.$$  

The optimal policy in each state of the system is the maintenance decision with the lowest single-period cost. These costs are shown in Table 3 below and given by the following expressions:

$$SC(i, u_i) = \sum_{j=0}^{13} p_j OC(i, j)$$  

$$SC(i, u_j) = \sum_{j=0}^{13} p_j MC(i, j)$$  

$$SC(i, u_k) = RC(i).$$

### TABLE 3

<table>
<thead>
<tr>
<th>STATE</th>
<th>SC(1,0)</th>
<th>SC(1,1)</th>
<th>SC(1,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>15</td>
<td>6</td>
<td>70</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>13</td>
<td>70</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>22</td>
<td>71</td>
</tr>
<tr>
<td>3</td>
<td>82</td>
<td>34</td>
<td>71</td>
</tr>
<tr>
<td>4</td>
<td>151</td>
<td>47</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>252</td>
<td>61</td>
<td>72</td>
</tr>
<tr>
<td>6</td>
<td>387</td>
<td>76</td>
<td>72</td>
</tr>
<tr>
<td>7</td>
<td>561</td>
<td>92</td>
<td>73</td>
</tr>
<tr>
<td>8</td>
<td>777</td>
<td>109</td>
<td>73</td>
</tr>
<tr>
<td>9</td>
<td>1038</td>
<td>122</td>
<td>73</td>
</tr>
<tr>
<td>10</td>
<td>1345</td>
<td>132</td>
<td>74</td>
</tr>
<tr>
<td>11</td>
<td>1702</td>
<td>143</td>
<td>74</td>
</tr>
<tr>
<td>12</td>
<td>2111</td>
<td>155</td>
<td>74</td>
</tr>
<tr>
<td>13</td>
<td>2573</td>
<td>167</td>
<td>75</td>
</tr>
</tbody>
</table>

According to Table 3, it is optimal to replace the software system after state 6. In other words, decision makers should consider rewriting WebCAT’s software after 60% of additional code has accumulated.

### VII. CONCLUSION

This paper has described WebCAT, a system designed to improve criminal incident data sharing and data analysis in Virginia. Major components of this project were: data acquisition capabilities, query time optimization, spatial analysis capabilities, integrated system performance evaluation, and life cycle analysis. Formal analysis has determined alternatives that significantly enhance the functionality of WebCAT and ensure its long term maintenance. Additional analysis of the integrated system shows that it will provide improved data sharing and crime analysis for more than 2/3 of Virginia’s law enforcement jurisdictions.

### ACKNOWLEDGMENT

The WebCAT team wishes to acknowledge the work done by the four previous WebCAT teams in the Systems and Information Engineering Department, which provided a basis for our continued efforts. We would also like to thank Butch Johnstone of the VDCJS for their support and suggestions.

### REFERENCES


