Analysis of Computer Science Curriculum through Development of an Online Crime Reporting System

Linda Kress

Thesis submitted to the College of Engineering and Mineral Resources at West Virginia University in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science

Roy S. Nutter, Ph.D., Chair
John M. Atkins, Ph.D.
Cynthia D. Tanner, M.S.C.S.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2006

Keywords: computer science curriculum, software engineering, senior design, online crime reporting

ABSTRACT

Analysis of Computer Science Curriculum through Development of an Online Crime Reporting System

Linda Kress

There are inherent differences in appropriate methodologies for developing hardware-oriented and software-oriented projects. Due to a recent change in the curriculum requirements for computer science students, the need to revise the senior design curriculum for students working on software-oriented projects has arisen. With graduate student supervision, a senior design team developed an online crime reporting system. The need for revision was apparent throughout the development process. This paper discusses difficulties encountered by the team and offers solutions to those obstacles, particularly in the areas of scheduling, documentation, development environment, and the security of software applications.

For Lenore Kress, who taught me to be thirsty,
and for Tammy McDonald, who helped me to drink.

Acknowledgements

I would like to thank the following people for their help:

Dr. Roy Nutter, for tirelessly offering advice when it was needed,

Dr. John Atkins, for the elegant database solution and for talking me into going to graduate school in the first place,

Cindy Tanner, for being a great sounding board when I needed an experienced ear,

Sergeant Chris Casto of the West Virginia State Police, for being enthusiastic and insightful,

the **West Virginia State Police**, for giving us this opportunity and for their appreciation of our efforts,

and the folks at the **National White Collar Crime Center**, for giving us valuable time and insight into the Internet Crime Complaint Center.

I would also like to extend very special thanks to the members of the senior design team; without their efforts and unwavering dedication, none of this would have happened:

Daniel Baker
Nathan Gunn
Michael Izzi
Omid Jalali
Nathan Moore

Finally, I would like to thank Steven Miller and Larissa Shelton for signing on to continue development and expansion of the WVCRIME.

Table of Contents

List of Figures and Tables

Introduction

A change in the curriculum for undergraduate computer science students at West Virginia University, combined with the desire of the West Virginia State Police to experiment with an online crime reporting system, provided a unique opportunity to assess the feasibility and efficacy of a new paradigm for software development in an academic, collegiate setting. In this model, the computer science undergraduates, (hereafter referred to as the senior design team), were led by a graduate student to define, specify, design, implement and test a software product for an actual client. This provided the students with invaluable experience in developing software in a real-world environment, as opposed to an academic one.

Statement of the Problem

This paper presents a discussion of graduate leadership of a team of senior undergraduate students throughout the development of an online crime reporting system for the West Virginia State Police. This paper suggests changes to the undergraduate senior design curriculum regarding development projects that primarily focus on software development.

Chapter 1 – Background

Senior Design

In 2002, West Virginia University began requiring computer science students to complete a senior design project (computer engineering students were previously required to complete the senior design curriculum). Due to the differences in the disciplines, the senior design project requirements were structured to evaluate projects that heavily involved hardware components. Many deliverables the senior design team completed in the first semester corresponded to hardware-oriented systems. This effectively doubled the documentation effort required by the team. The WVCRIME team attempted to reconcile deliverables midway through the development lifecycle; the results and implications of this reconciliation are discussed in Chapter 7.

Software Lifecycle Methodology

A software process model is the process by which a software development team defines, specifies, analyzes, designs, implements, tests, deploys, maintains and retires a software product. A process model may manipulate various phases as appropriate to fulfill its goals. There are many methodologies from which a development team can choose. Different process models vary radically in their plans of attack, and thus are often suited for different development environments. A brief survey of different process models is presented in this section.

Build-and-fix

This methodology is the crudest form of a process model. It consists of only two phases: implementation and testing. One begins coding immediately with no consideration given to the requirements or design phases. When the program does not perform properly, or features need to be added, the code is enhanced or corrected in an ad-hoc fashion. This method is unsuitable for all but the smallest applications. Use in development projects of any substantial size result in programs that are difficult if not impossible to maintain, poorly satisfy client's needs, and expensive to repair due to the little forethought given to testing [3].

The Classic Waterfall Model

Winston Royce proposed the classic waterfall process model in 1970. In this process model, each phase of the software lifecycle must be completed before the next phase can begin. The original process model provided for feedback loops between phases, but the process is generally regarded as a linear one [1]. The phases in the waterfall model are: gathering system requirements, gathering software requirements, preliminary program design, analysis, program design, coding, testing, and operations [2]. The waterfall method was the first refinement of the Stagewise process model. Due to its linear nature, the waterfall process model is not considered suitable for development that incorporates many changes to the requirements or design after these phases have been completed; another limitation is that incorrect requirements that are not discovered until late in the development process can be catastrophic to a project [1]. It is also worth

noting that this model does not consider the retirement phase at all. The retirement phase typically concerns what to do with outdated mediums on which data are stored, and its absence in this model is reflective of changes that have occurred within the technology industry since the model was developed.

The Incremental Model

The incremental model is an iterative version of the waterfall process model [1]. This model progresses through the same lifecycle phases as the waterfall model, but the product is divided into several smaller pieces, or builds, that are developed individually in sequence via the waterfall methodology. When a build is complete, it is delivered to the client, and work begins on the next build. In this way, the client receives the product in small increments, one build at a time, until the entire product has been deployed. This process model is suitable for products that can be easily divided into separate modules.

The Rapid Application Development (RAD) Model

The RAD Model is a version of the incremental model that centers on minimal development time for each build [1]. RAD development teams are small and intensely focused on the task at hand. RAD requires more flexibility than its parent model, and the team must be selected carefully. A poorly chosen team can easily fail in the high-tension environment that this process model fosters [4]. Each team member must be highly experienced and the team itself must be free to make design decisions on its own, without input from stakeholders; the stakeholders in turn must trust the expertise of the RAD team [4].

The Rapid Prototyping Model

The rapid prototyping process model is characterized by an iterative progression of rapidly-developed prototypes. After the initial requirements phase, a prototype is quickly designed and built. The client then uses the prototype for a trial period and provides feedback to the development team. The team then improves the prototype and returns it to the client, and the process repeats itself until the client is satisfied.

While this process model is often used in conjunction with other process models, it is not often employed exclusively. Developers may make less-than-ideal choices concerning implementation in the interest of expediently getting the next prototype to the customer, which results in a lower-quality system [1]. When using prototyping as a requirements-gathering tool, developers must take care to ensure that the client understands that the prototype is not an ideal solution, but a tool to foster good communication about requirements. Otherwise, the client may believe that the product is much closer to completion than it actually is, which can result in unreasonable demands [1].

The Spiral Model

The spiral model was proposed to mitigate some of the shortcomings of previously proposed software process models. It is a risk-based model, rather than a code-based model (build-and-fix) or a document-driven model (waterfall) [3]. The spiral process model has a similar process for every lifecycle phase. For example, the requirements phase begins by identifying the goals of the system. Once this has been achieved, any constraints on the development process are delineated. Alternatives are discussed and chosen. Then risks are identified. If a risk cannot be solved or mitigated, then the development team has two options: they may accept the risk, or the project may be terminated. If all risks are accepted, mitigated, or resolved, then the team commits to obtaining funding for the next development phase.

The spiral model is highly flexible in its application to large software projects. Maintenance and additional features added after deployment simply become additional iterations around the spiral, and these phases progress through the same risk assessment as all the others. It mandates review of progress and budgeting constraints, which means that project development is continually changing. At its inception, the spiral process model was not well-tailored to contract-driven development; the flexibility proved an ill match for the control and accountability needs of these clients [3]. Its suitability has not progressed in this arena of development [1]. It also relies heavily on the experience of risk-assessment personnel.

Agile Process Models

All the previously discussed process models share a common factor: they are prescriptive process models. They were designed "to bring order to the chaos of software development" [1]. Agile process models are concerned with rapid adaptation to change. They tend to differ drastically from prescriptive models in many ways, one of which is the active participation of the client in all aspects of the lifecycle. This means that the client must be nearly as available and committed as the development team. The primary representative for the client was Sergeant Chris Casto, a senior forensic investigator of the West Virginia State Police. Sergeant Casto was not available to participate in the development at the level required by agile process models. For this reason, agile process models were not considered when a lifecycle model was selected for this project.

WVCRIME Process Model

The classic waterfall model, in combination with the prototyping model, was selected for the development of the West Virginia Crime Reporting and Information Management Effort, (WVCRIME). The constraints of the waterfall process model, combined with a prototype, fit best with the requirements of the system. Specifically, the system was to be delivered to the client in May 2006; requirements elicitation began in September 2005. In order for the team to deliver the system in May, the client could not make any changes to the requirements after they were agreed upon in late December 2005. This also required that the team spend extra time refining requirements before beginning implementation, as incorrect requirements would have been fatal to the product.

The Alternatives

The build-and-fix model was not selected because it was not appropriate for the size of the project and represents poor software engineering methodology in all but the smallest of undertakings. The spiral model was rejected for two reasons: it depends heavily on experienced risk assessors, and the nature of the team made terminating the project an impossibility. The RAD model was unsuitable for this team because of the nature of the product; it was not possible to deliver the system one piece at a time. None of the modules could function as a separate entity save the database, and it would have been useless without a mechanism for inserting complaints and tips.

The future development of WVCRIME may follow the incremental process model; in selecting a process model, it was beyond the scope of the development team to impose a process model on future development teams.

WVSP Crime Reporting

The West Virginia State Police currently have a free text area that the general public utilizes to file anonymous tips. These tips are delivered to the assigned officer of the West Virginia State Police, who then must decide who should ultimately receive the tip.

The West Virginia State Police also receive complaints from the Federal Bureau of Investigation's Internet Crime Complaint Center, which is hosted at the National White Collar Crime Center (NW3C) in Fairmont, West Virginia. The assigned officer must read these complaints and make decisions about to whom they should be delivered.

The West Virginia State Police officers assigned to these tasks are spending an increasing portion of their time attending to these tips and complaints. The tips are not currently stored for later perusal.

Other relevant systems in place or being researched

Many law enforcement jurisdictions have implemented an online crime reporting system. The author conducted online research to determine what online crime reporting capabilities were currently available. One hundred thirty-nine web sites were examined. These web sites were located through Google searches. The author located sixty-two college or university web sites, thirty-five city or town web sites, 35 county or regional web sites, and ten national or international web sites that provided some type of online crime reporting capability. For the purpose of this survey, those web sites that directed users to email the site were not included; only those sites that provided some type of form for the user to complete were considered. These results reflected a survey of online crime reporting systems; it was not meant to be an exhaustive study.

The sites were examined according to the following criteria: the site allowed anonymous reporting, the site allowed reporting of drug-related crime, the site allowed reporting of sexual assault, the site allowed reporting of violent crime, the site allowed users to file complaints against law enforcement, the site used https, the site displayed a warning to users informing them it was a crime to file a false police report, the site was primarily a general tip line, the site was used to gather information from users on cold cases, and the site was primarily used for gathering information on fugitives. If a site discussed how the information was delivered, it was, without exception, delivered via email.

The author was unable to find a college or university with a tip line for their students in two states, Alaska and Delaware. The author was also unable to find a county

or regional (but within the state) crime reporting system in twenty states: Alaska, Arkansas, Connecticut, Hawaii, Idaho, Iowa, Kentucky, Maine, Massachusetts, Minnesota, Mississippi, Montana, New Hampshire, New Mexico, North Dakota, Oklahoma, Rhode Island, South Dakota, Tennessee, and West Virginia. The author found it surprising that so many sheriffs' departments across the country were not taking advantage of anonymous tip lines.

As represented in Figures 1.9 and 1.10, the majority of sites were not using any form of encryption. This was cause for concern, especially in regard to college or university tip lines. The majority of these tip lines allowed users to report any type of crime, and not all of them allowed the reporter to remain anonymous. The lack of security posed a risk to the reporter, particularly if he or she was not permitted to remain anonymous.

Those sites that did not allow users to report any type of crime generally restricted users to reporting misdemeanors. Restrictions were also placed on the information available about the crime (e.g. the reporter did not have any evidence, the reporter did not have any knowledge concerning a suspect, etc.). The following charts, graphs, and tables detail the results of this survey. The data was drawn from the following sources: [14] through [157].

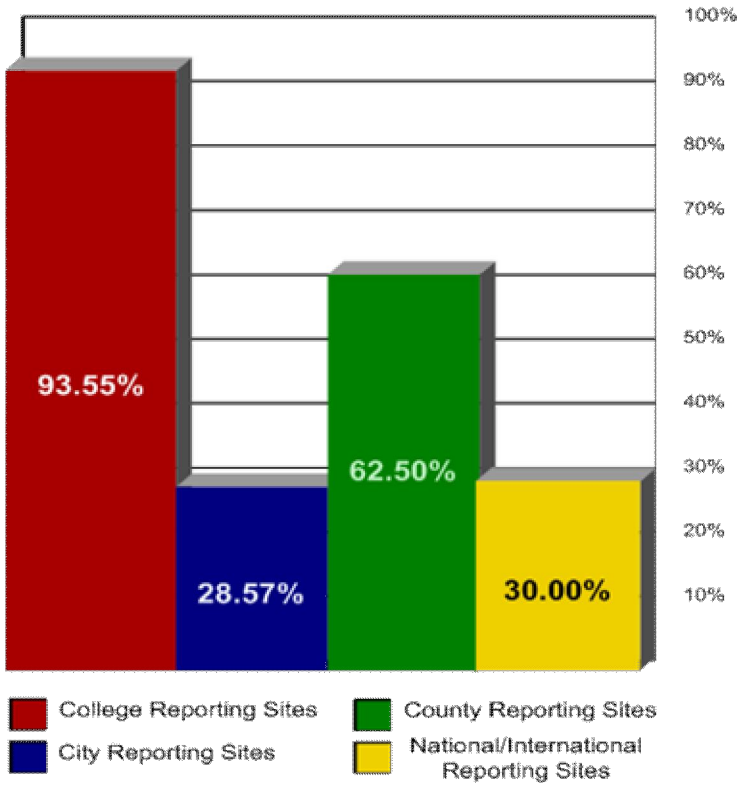


Figure 1.1 Percentage of sites that allow reporters to report crime involving drugs

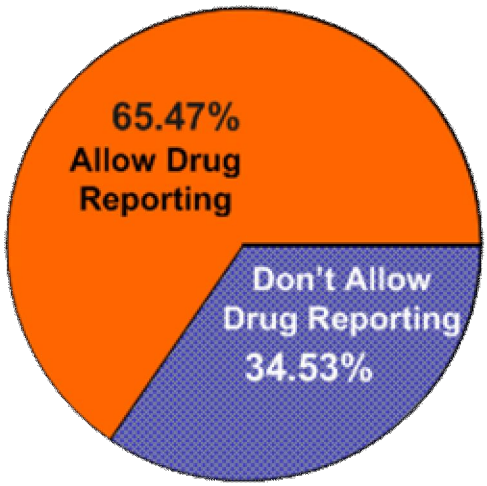


Figure 1.2 Overall percentage allowing drug reporting

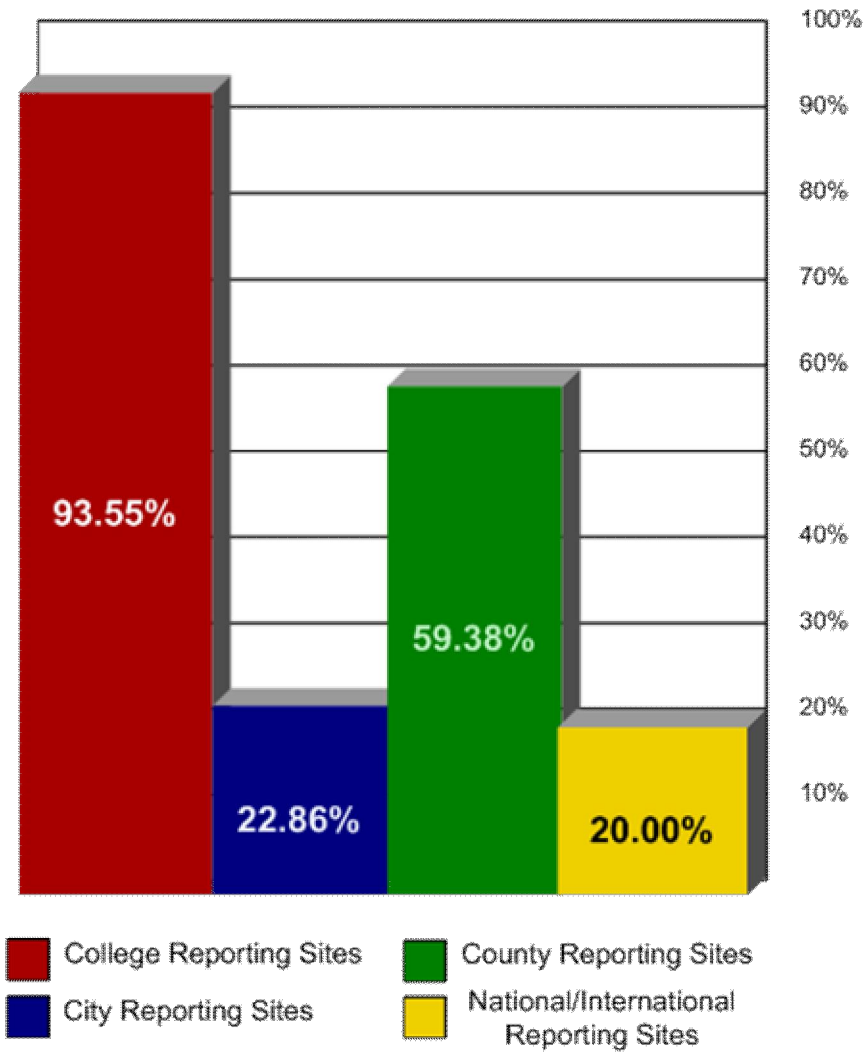


Figure 1.3 Percentage of sites that allow reporting of crime involving sexual assault

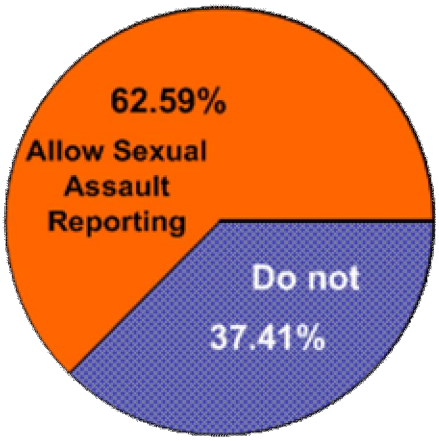


Figure 1.4 Overall percentage allowing sexual assault reporting

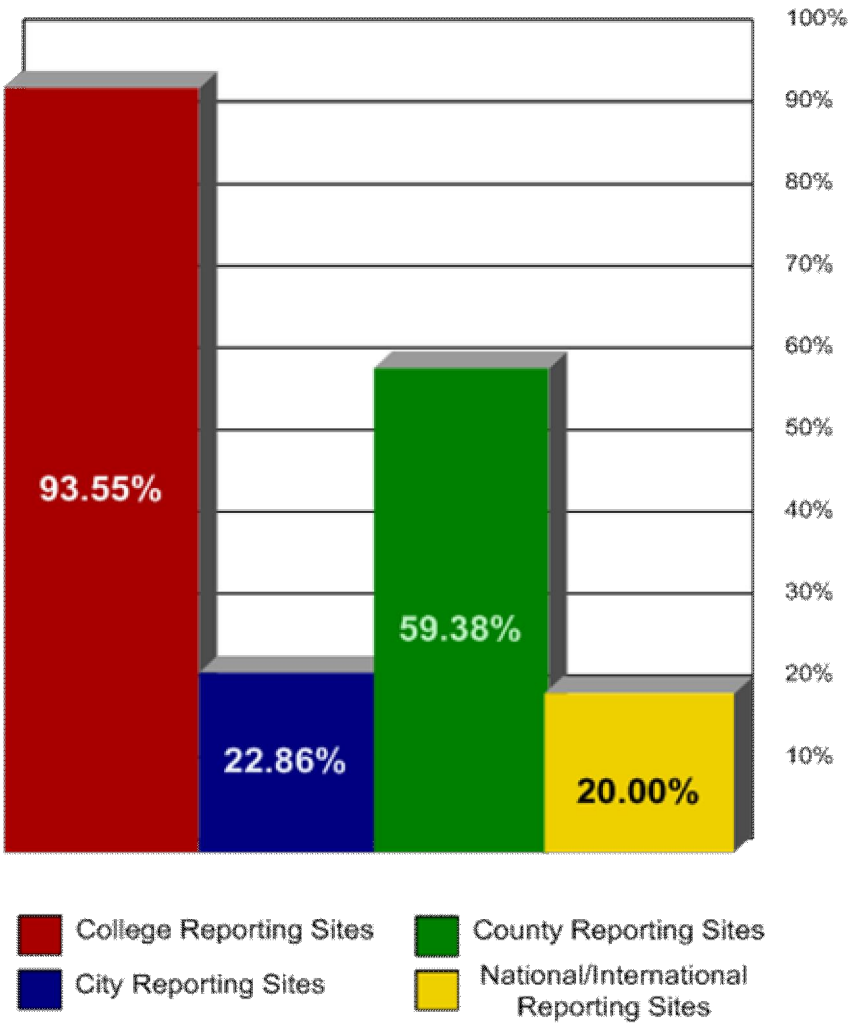


Figure 1.5 Percentage of sites that allow reporters to report violent crime

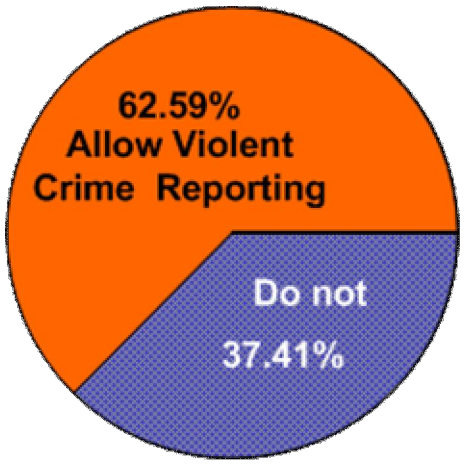


Figure 1.6 Overall percentage allowing violent crime reporting

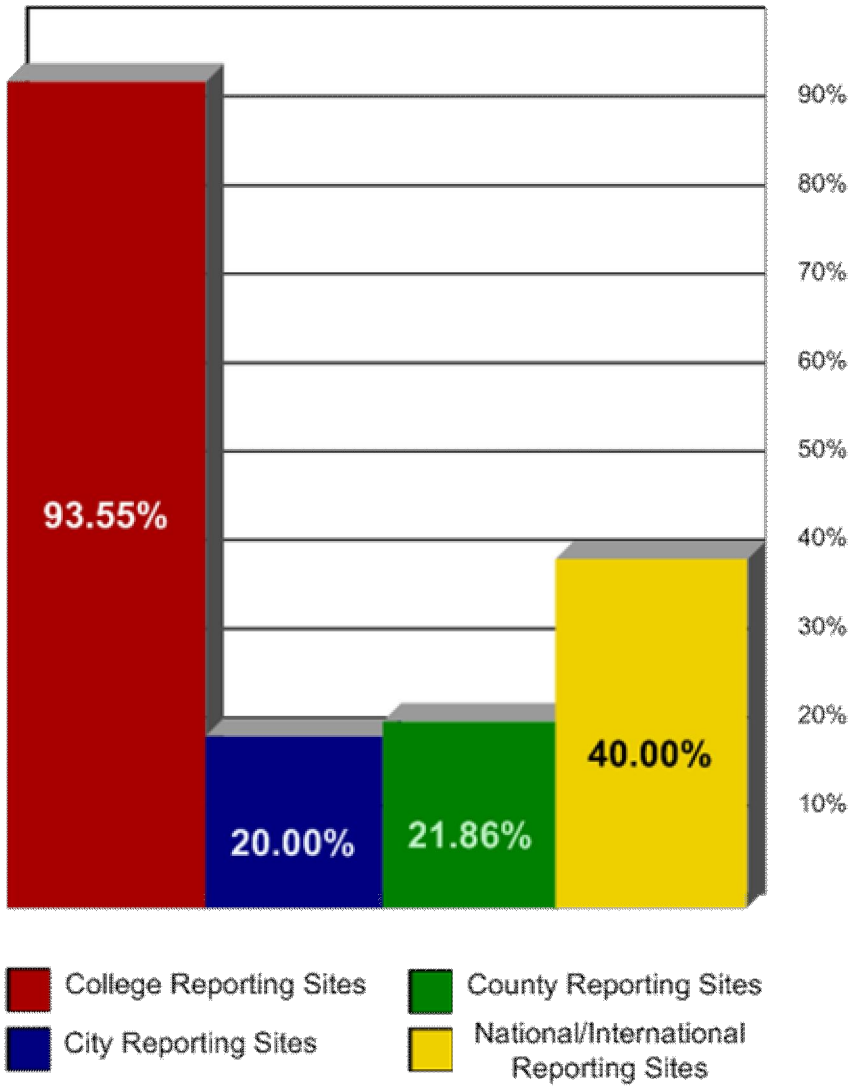


Figure 1.7 Percentage of sites that allow reporters to file complaints against police

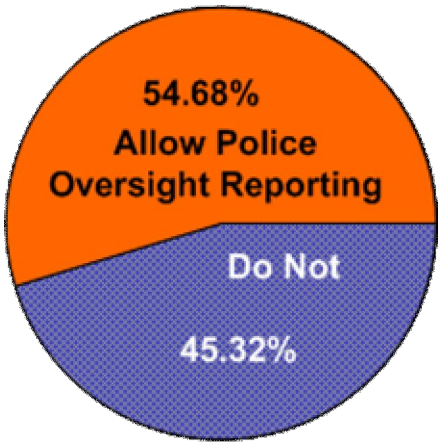


Figure 1.8 Overall percentage allowing reporting against police

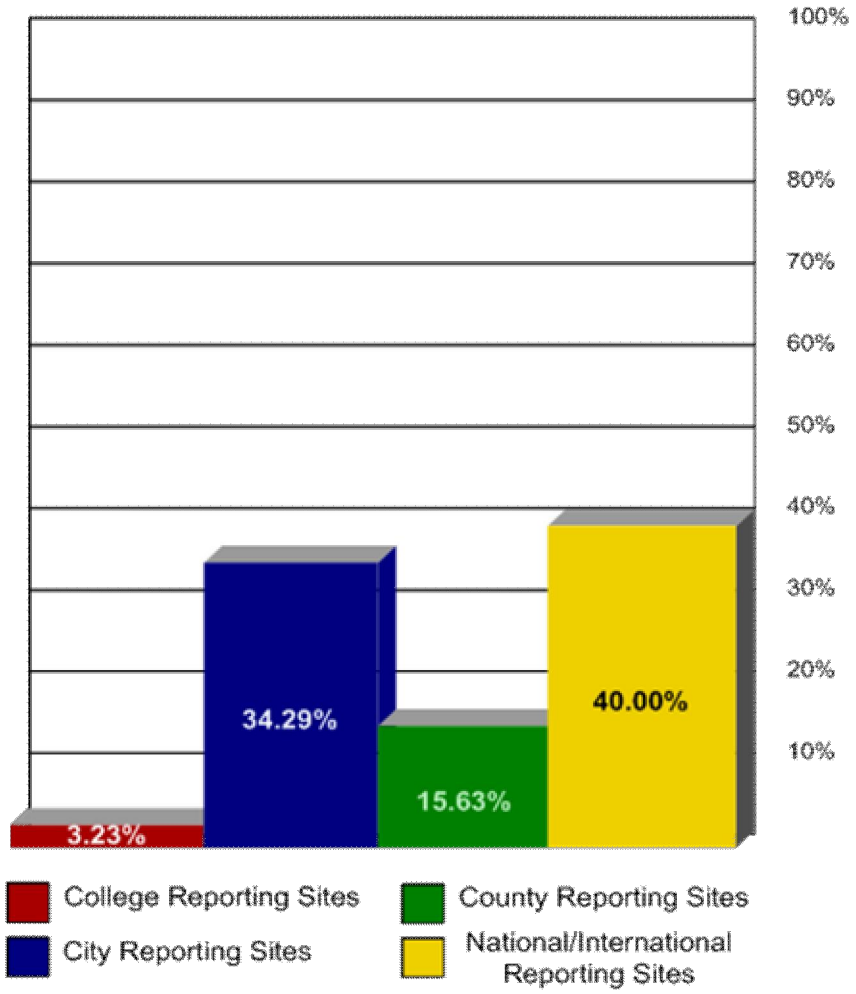


Figure 1.9 Percentage of site using https

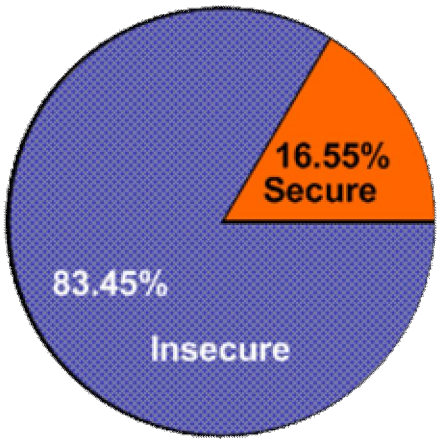


Figure 1.10 Overall percentage of sites using https

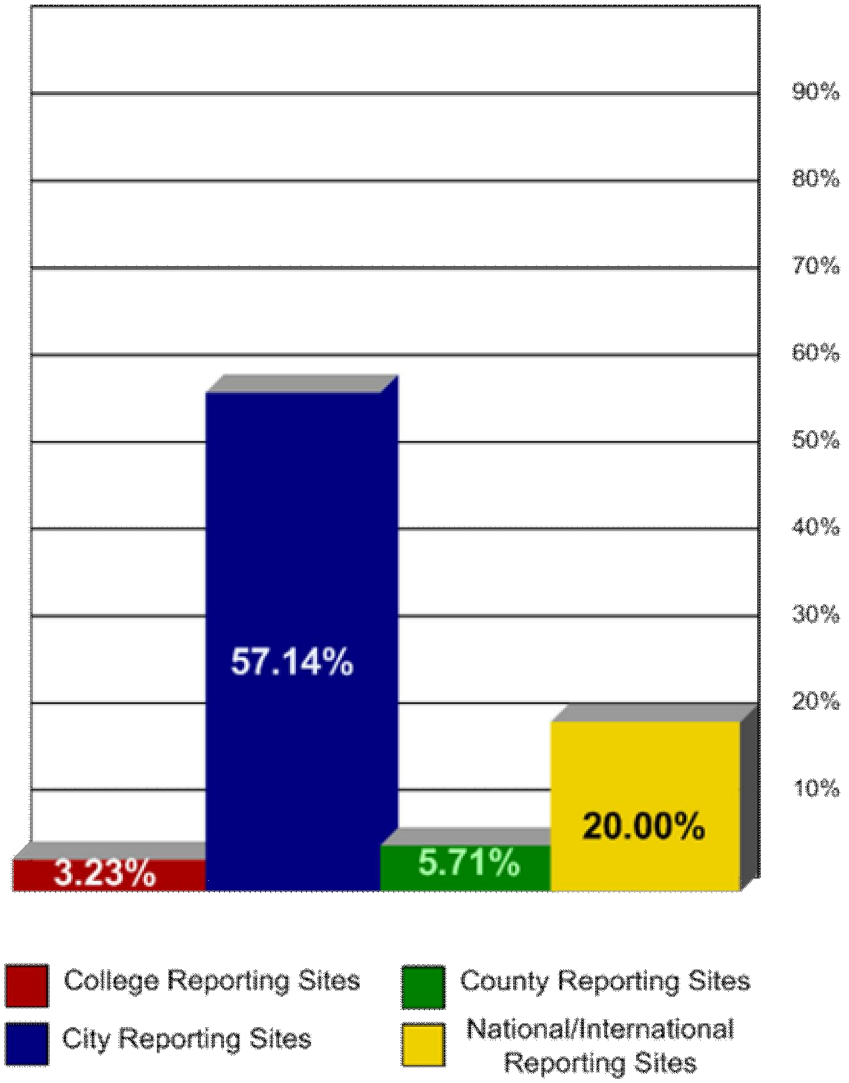


Figure 1.11 Percentage of sites warning reporters about false crime reporting

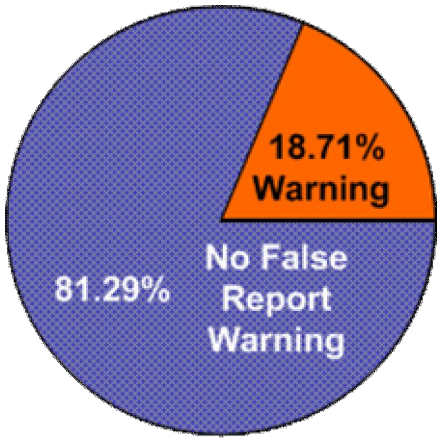


Figure 1.12 Overall percentage of sites warning reporters about false crime reporting

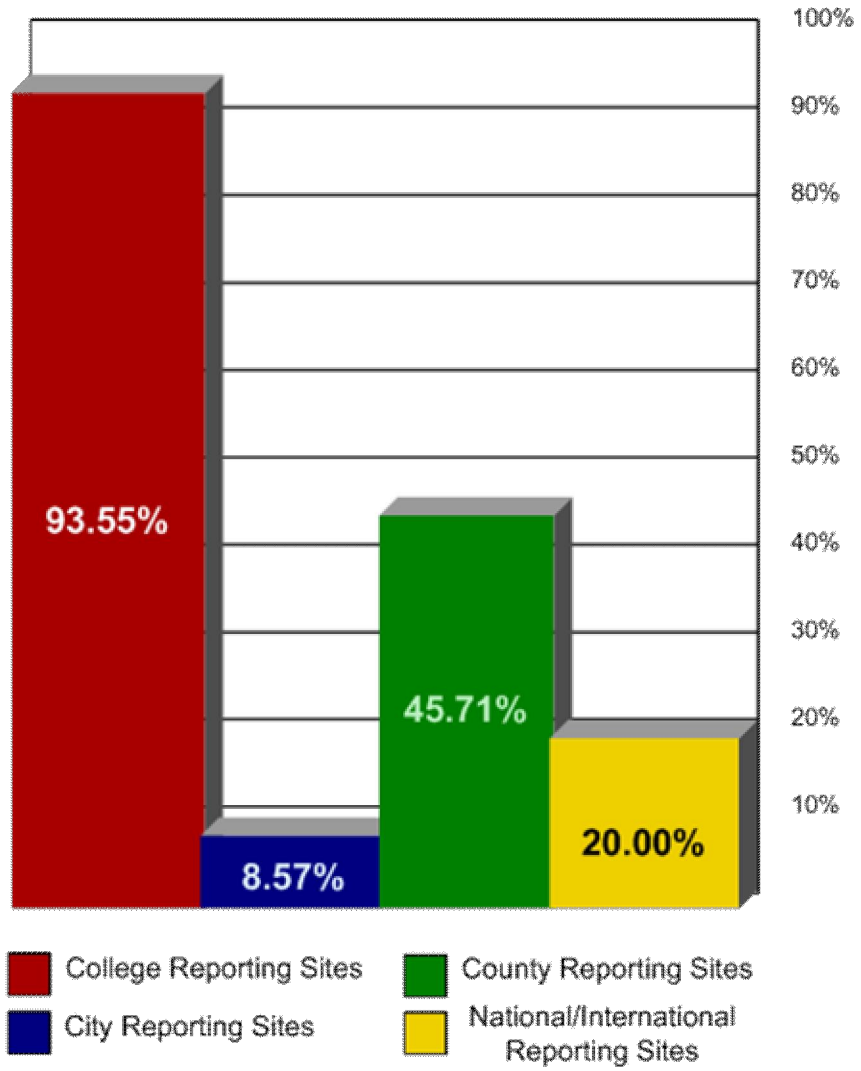


Figure 1.13 Percentage of sites using a general tip line

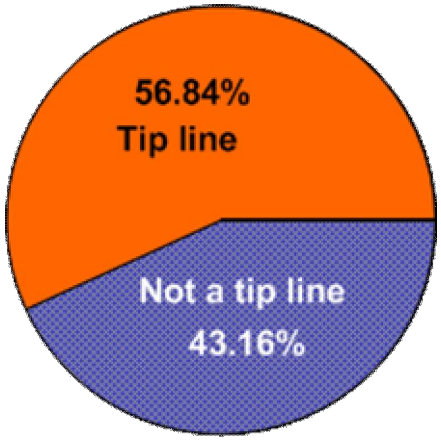


Figure 1.14 Overall percentage of sites using a tip line

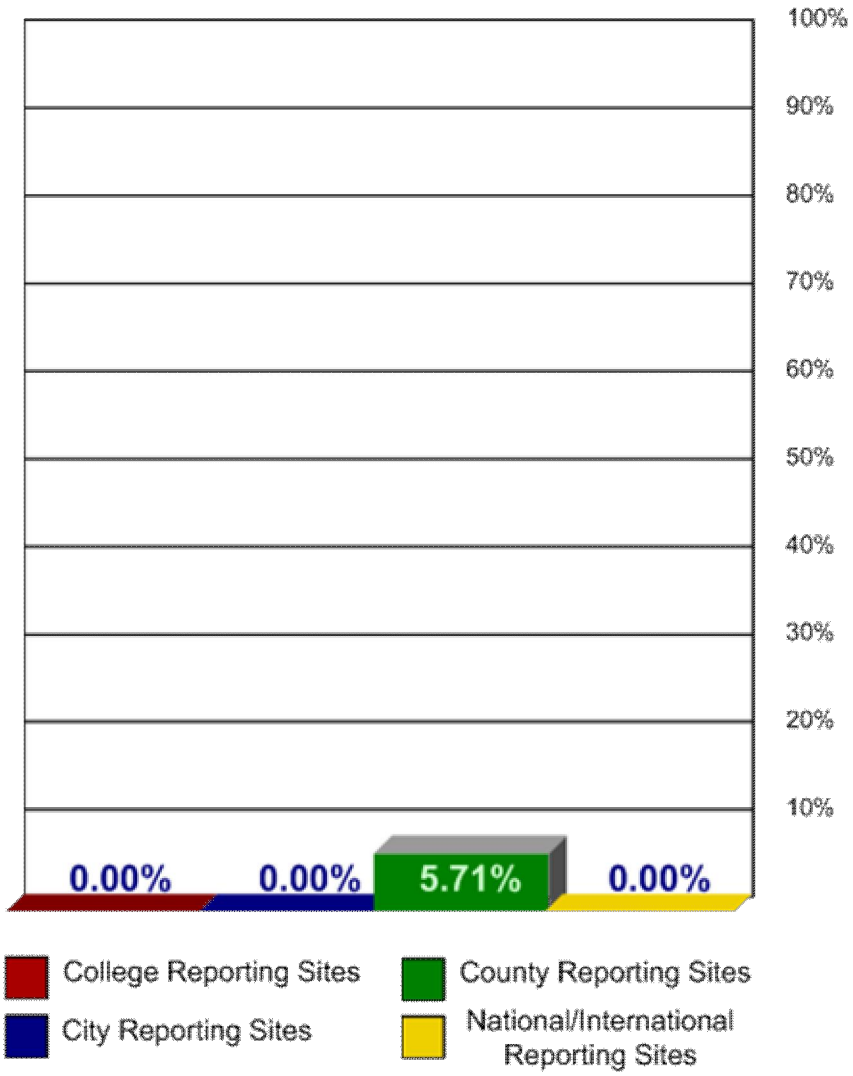


Figure 1.15 Percentage of sites used primarily for cold case information

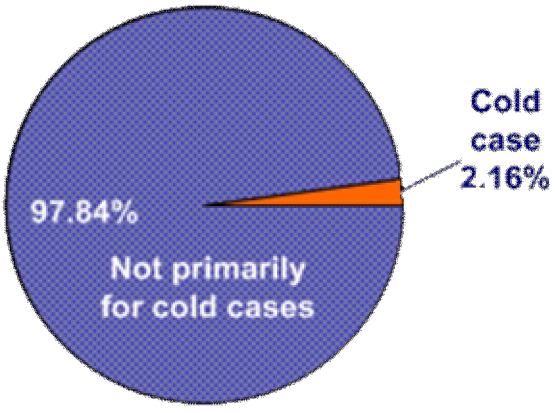


Figure 1.16 Overall percentage of cold case sites

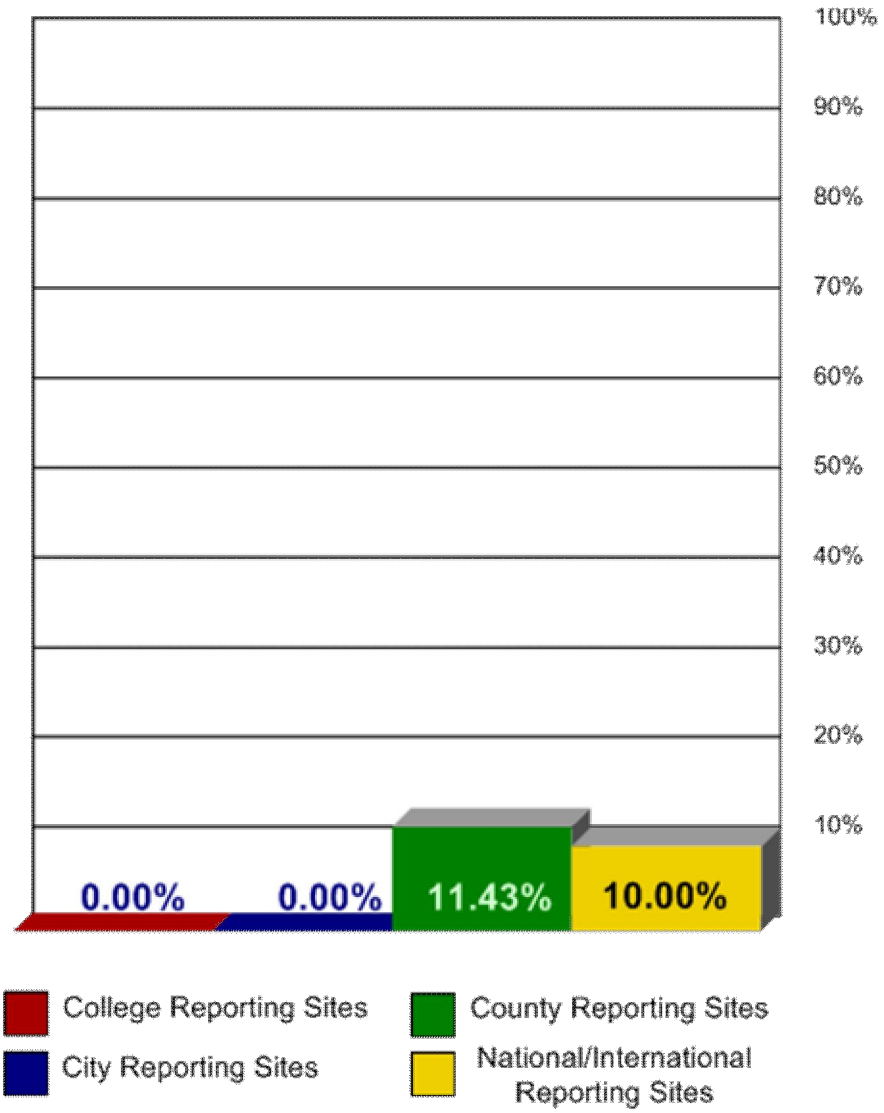


Figure 1.17 Percentage of sites used primarily for fugitive information

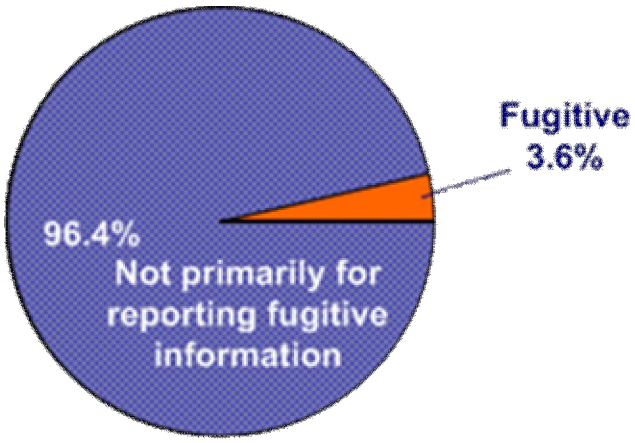


Figure 1.18 Overall fugitive information sites

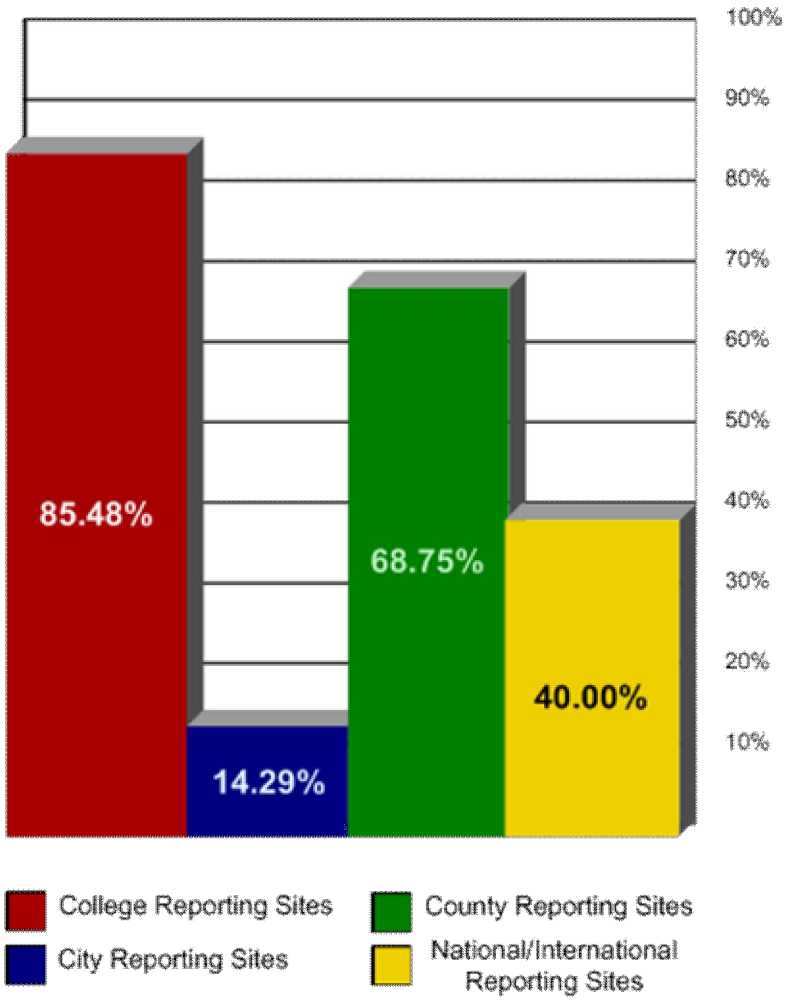


Figure 1.19 Percentage of sites that allow some type of anonymous reporting

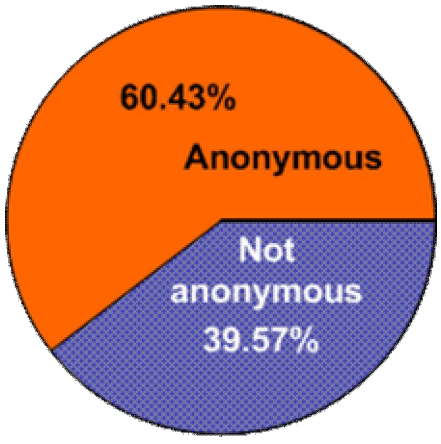


Figure 1.20 Overall percentage allowing anonymous reporting

Many cities and towns restricted which types of crimes users were permitted to report. The following table details the frequencies of the crimes that users could report on restricted web sites.

Crime	Number of sites allowing crime to be reported
Vandalism	**22**
Lost Property	**18**
Theft	**18**
Vehicle Burglary	**16**
Harassing Telephone Calls	**9**
Petty Theft	**6**
Credit/Check/Debit Card Fraud	**5**
Vehicle tampering	**5**
Grand Theft	**4**
Hit and Run	**4**
Other misdemeanors*	**4**
Residential Burglary	**4**
Commercial Burglary	**3**
Suspicious Circumstances	**2**
Trespassing	**2**
Abandoned Vehicle	**1**
Accident (no DUI or Hit and Run)	**1**
Custodial Interference	**1**
Defrauding an Innkeeper	**1**
Fraud	**1**
Graffiti	**1**
Harassment	**1**
ID Theft	**1**
Lost/Stolen License Plate	**1**
Noise	**1**
Public Drunkenness	**1**
Threats	**1**
Traffic Complaint	**1**
Vehicle Prowling	**1**

Table 1.21 Utility of crime reporting by restricted city/town web sites

* These sites were not considered by the author to allow users to report all types of crime because "other *misdemeanors*" was specified.

Some counties and regions restricted which types of crimes users were permitted to report. The following table details the frequencies of the crimes that users could report on restricted web sites.

Crime	Number of sites allowing crime to be reported
Suspicious Circumstances	**5**
Lost Property	**3**
Theft	**3**
Vandalism	**3**
Traffic	**2**
Accident on Private Property with No Injuries	**1**
Aggressive Driving	**1**
Alcohol Violations	**1**
Civil Disputes	**1**
Credit/Check/Debit Card Fraud	**1**
Disorderly Conduct	**1**
Fire Hazard	**1**
Fraud and Abuse	**1**
Forgery	**1**
ID Theft	**1**
Loitering	**1**
Lost/Stolen Property	**1**
Mail Theft	**1**
Noise Violations	**1**
Prescription Fraud	**1**
Quality of Life Complaints (noise, excessive dust from construction, etc.)	**1**
Soliciting	**1**
Speeding	**1**
Telephone Harassment	**1**
Trespassing	**1**
Vehicle Tampering	**1**

Table 1.22 Utility of crime reporting by restricted county/regional web sites

Some national and international systems restricted which types of crimes users were permitted to report. The following table details the frequencies of the crimes that users could report on restricted web sites.

Crime	Number of sites allowing crime to be reported
Child Abuse Online	2
Computer Intrusions	2
Missing Persons	2
Advanced Fee	1
Age Discrimination	1
Bankruptcy Fraud	1
Botnets	1
Corruption	1
Counterfeiting	1
Criminal Racial Content	1
Denial of Service Attacks	1
Desktop Forgery	1
Disability Discrimination	1
Drug Dealing	1
Electronic Stalking	1
Hate Crimes	1
Human Rights Violations	1
Identity Fraud	1
Internet Banking Fraud	1
Internet Service Theft	1
Investment Fraud	1
Malware	1
Online Auction Fraud	1
Other Fraud	1
Password Theft	1
Pirating	1
Property Fraud	1
Prostitution (online)	1
Sex Discrimination	1
Spam	1
Terrorist Threat	1
Theft	1
Trojans	1
Vandalism (Web Site)	1
Violent Web Site	1
Virus Writing	1
Wanted Fugitives	1

Table 1.22 Utility of crime reporting by restricted national/international web sites

The WVCRIME system was initially modeled after the FBI's Internet Crime Complaint Center. Like the IC3, the WVCRIME checked for "hot words," words that were examined by the system and flagged for special response (e.g. murder, death, kill, etc.). Complaints filed via the IC3 that contained hot words were given the highest priority and were handled before other complaints.

When the IC3 received a complaint, it was logged in a database by a proprietary system, checked for hot words, and sent to an employee of the NW3C. The employee decided to whom to send the complaint via email. Recipients of these emails included various state and federal law enforcement agencies throughout the company.

The WVCRIME automated the process of deciding the destination of the complaint or tip and also the delivery of that complaint or tip to the appropriate jurisdiction. The author's research indicated that these capabilities of the WVCRIME make it unique in these respects.

It was possible to receive complaints from the IC3 that did not have a clear destination. In this sense, delivering the complaint to the appropriate jurisdiction was not completely automated. It was necessary to have a default jurisdiction that handled these complaints manually. However, these instances were expected to be occasional, and the WVCRIME handled the delivery of most tips and complaints accurately.

Team Structure

There are many options available for the organization of team structure. Team structure heavily impacts the development process; it affects how decisions are made throughout the software development lifecycle. A brief sampling of team structures and their implications is presented in this section.

Chief Programmer Model

The Chief Programmer team model was proposed in 1972 by Harlan Mills. This team structure is modeled after the interactions between the team of people in an operating room [5]. One person is ultimately responsible for the success (or failure) of the product being developed, the chief programmer. The chief programmer serves as the team leader and single point of accountability for the team. This person should be technically capable of performing any task that any developer in the team is asked to do and serves as the liaison between the team and upper-level management. The chief programmer has the final say on all major decisions that impact the development process.

The back-up chief programmer is required to possess all the qualities of the chief programmer and is prepared to assume leadership of the project should the chief programmer be unable to fulfill his or her duties. This position in the team can be difficult to fill; a person capable of leading the team is asked to "play second fiddle," a position that generally pays less and requires the same level of expertise as the chief programmer position.

The third clearly defined role in this team structural model is the project secretary, or librarian. The team librarian is responsible for caring for all the artifacts (code, documentation, etc.) for the project. He or she controls access to the artifacts and oversees configuration management (control of change). This position is crucial to the team [6].

Additional team members include developers and specialized technical experts. The team is supported by these professionals, who perform tasks assigned by the chief programmer.

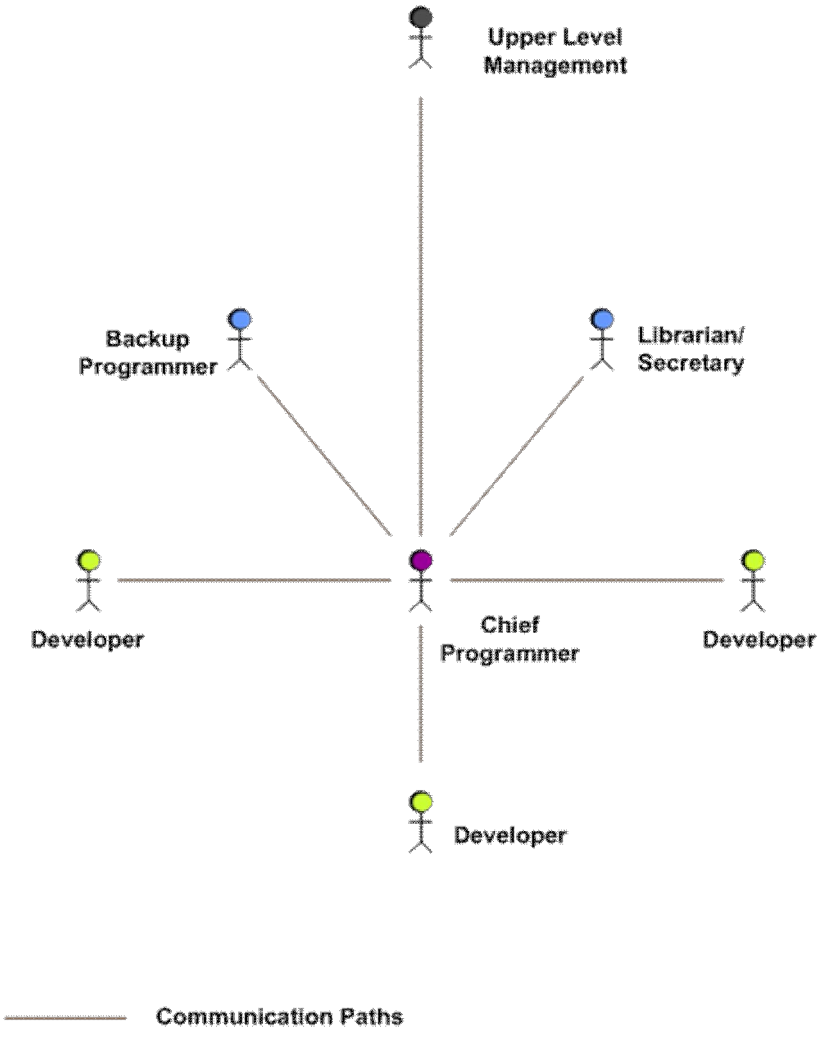


Figure 1.23 Chief Programmer Team Structure

Modified Chief Programmer Model

This team structure is similar to the Chief Programmer team structure. However, the duties of the chief programmer are divided into technical and managerial roles. Team leadership consists of a team leader, who is responsible for all technical aspects of the software development, and a team manager, who serves as the managerial team leader and is the liaison to upper-level management. This structure alleviates some of the difficulties in finding a chief programmer; "competent practitioners often make poor team leaders" [6] because they lack managerial skills.

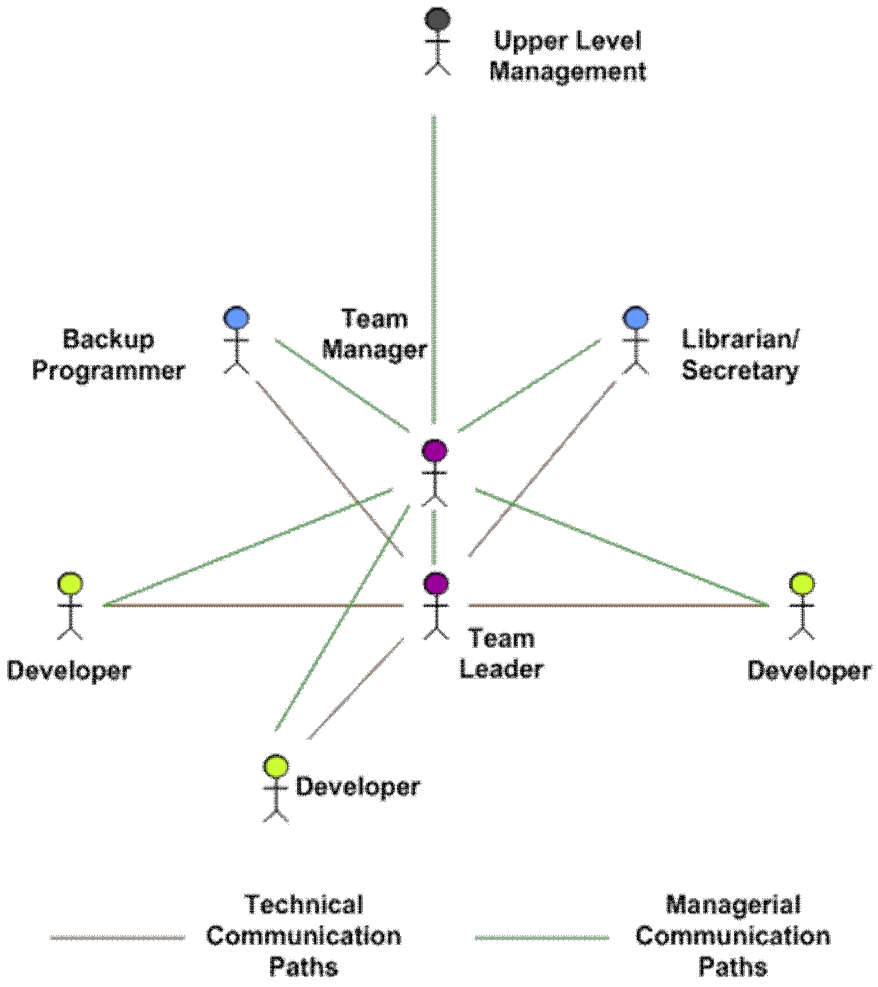


Figure 1.24 Modified Chief Programmer Team Structure

Democratic Model

The Democratic team structure allows all team members to be equal in stature. All team members have input in the decision-making process. While there is no team leader, team members will be selected to be in charge of specific development tasks. This team structure promotes "egoless programming," which means that developers tend to think of all artifacts as belonging to the team rather than an individual developer.

This team structure is appropriate in research environments. It fosters creativity and encourages a sense of team. However, there are some drawbacks to this structure that make it inappropriate for use in large-scale development. Upper-level management of democratic teams can be difficult because there is no single point of accountability. There is no team leader, which means some tasks may be overlooked, as no one is in charge of looking at the "big picture." Also, experienced developers may not be amenable to reviews of their work by less experienced team members.

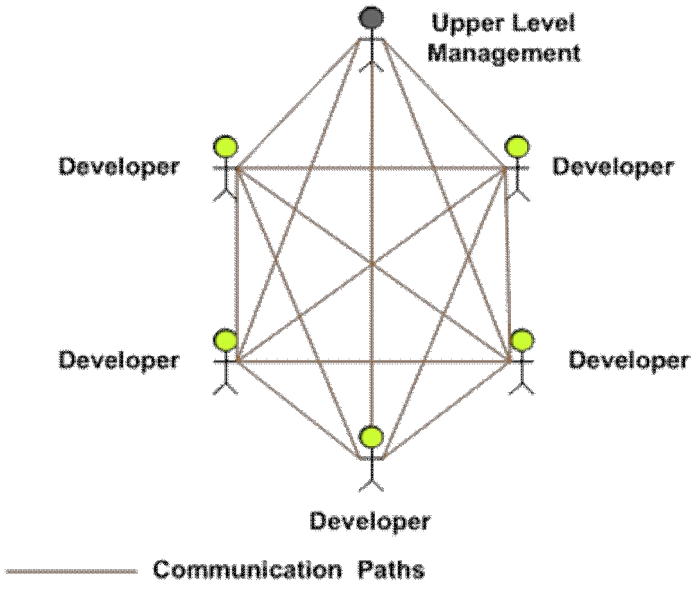


Figure 1.25 Democratic Team Structure

Hierarchical Model

The Hierarchical model is easily scalable, and as such is appropriate for products that may grow or shrink as they are developed. The process model consists at the lowest level of small teams of developers that are responsible for implementing the system. Each small group has a team leader. Communication is generally encouraged among all team members at this level. However, the team leader serves as the liaison between upper-level management, the project leader, and the team. The project leader coordinates several teams. In a pure hierarchical model, the team leader must communicate with other team leaders via the project leader, though this practice is not always enforced in industry. The person responsible for the overall success of the project is the project manager, who supervises several project leaders. Team leaders do not directly communicate with the project leader or project manager; communication must take place through the clearly defined channels prescribed by the hierarchical process model. As the product expands or becomes smaller, adding new teams or downsizing is achieved with relative ease, as the modularization of knowledge is accomplished by the restricted communication paths.

One difficulty that this process model presents is that all of the technical details of the product are hidden from those who make administrative decisions. The further one proceeds up the hierarchy, the fewer details one possesses. This process model also requires more management, which translates into more overhead for the product.

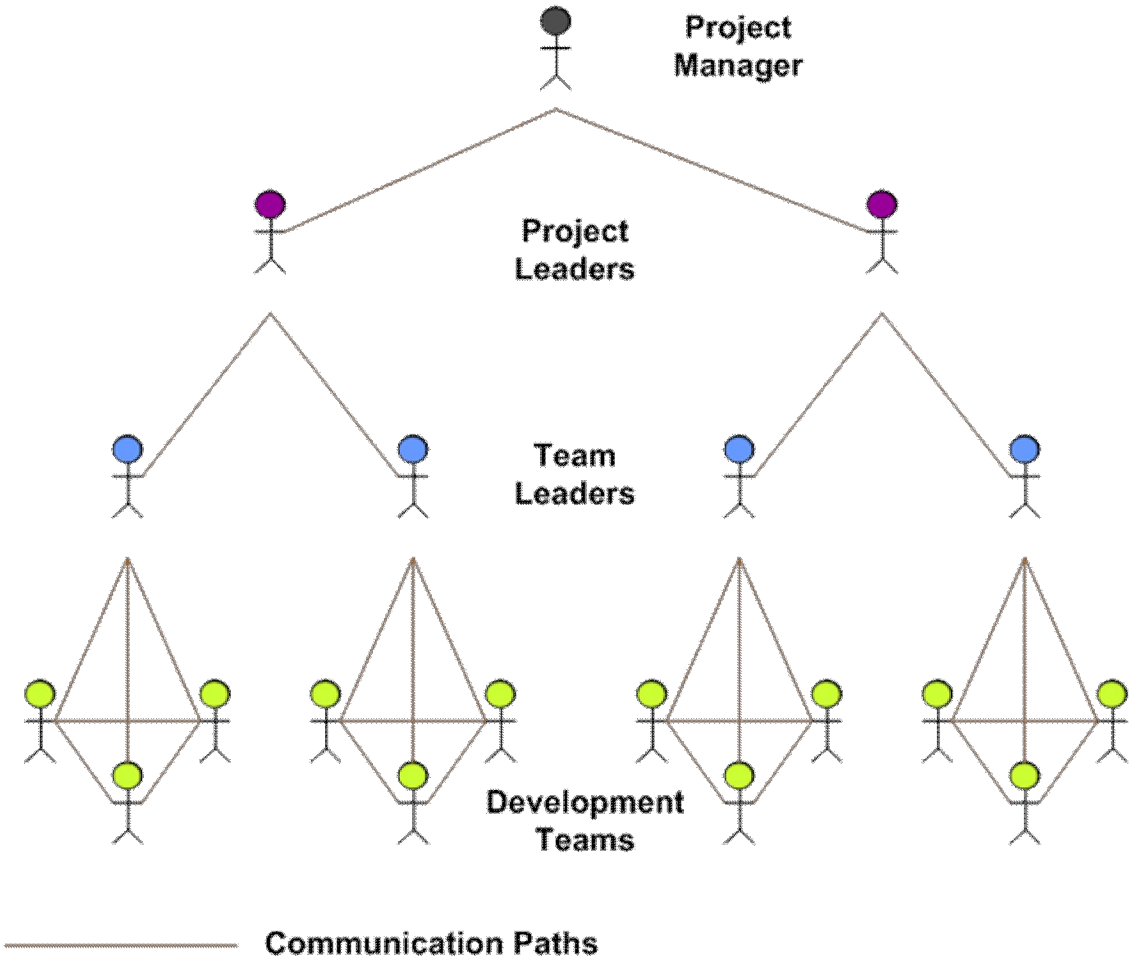


Figure 1.26 Hierarchical Team Structure

The Alternatives

Each of the previously discussed team structures had benefits and drawbacks, but none were appropriate for the WVCRIME development process. The Chief Programmer team structure would have placed too much decision-making responsibility in the hands of the author; the senior design students would not be able to fulfill the requirements of their curriculum, which includes making at least some design decisions. This model places the entire design process squarely in the hands of the team member that fills the

chief programmer role and relegates the other team members to fulfilling assignments doled out by the team leader.

The Modified Chief Programmer presented many of the same problems, especially since the team preferred to work as a democratic entity. Serving only as the liaison for the team to the client would not have allowed the author to have any input into the design process, which was as equally unsatisfactory as the difficulties presented by the Chief Programmer model.

The Democratic Model was found to be inappropriate on all fronts. This model is only appropriate in research environments, and while this project involved research components (this type of automated crime reporting system had not been heretofore developed), the project had a deployment deadline. This made the Democratic team structure inappropriate according to current software engineering theory. This structure also did not provide enough leadership possibilities for a graduate student to make this an appropriate thesis topic.

The Hierarchical model was rejected for two reasons. It posed the same difficulties as the Chief Programmer model, as the teams at the lowest level correspond to smaller versions of that model. The second reason is that such a structure requires at least two teams that are supervised by team leaders, who in turn answer to the project leader. The WVCRIME project did not require such a large number of developers; the number of developers required to follow this structure would have complicated the development process in terms of overhead, and the project would not have been completed on time.

WVCRIME Team Structure

As is often necessary in industry, it was necessary to adapt an existing methodology to suit the needs of the project. These needs extended beyond the software itself; the need for the senior design team to fulfill its curriculum requirements and the need for a suitable thesis topic for the author were given considerable weight. This resulted in a loosely defined team structure for the initial phases of the software development lifecycle. This loose definition of roles led to friction within the team midway through development. This friction is discussed in following chapters, in addition to strategies for avoiding it in future teams of this nature.

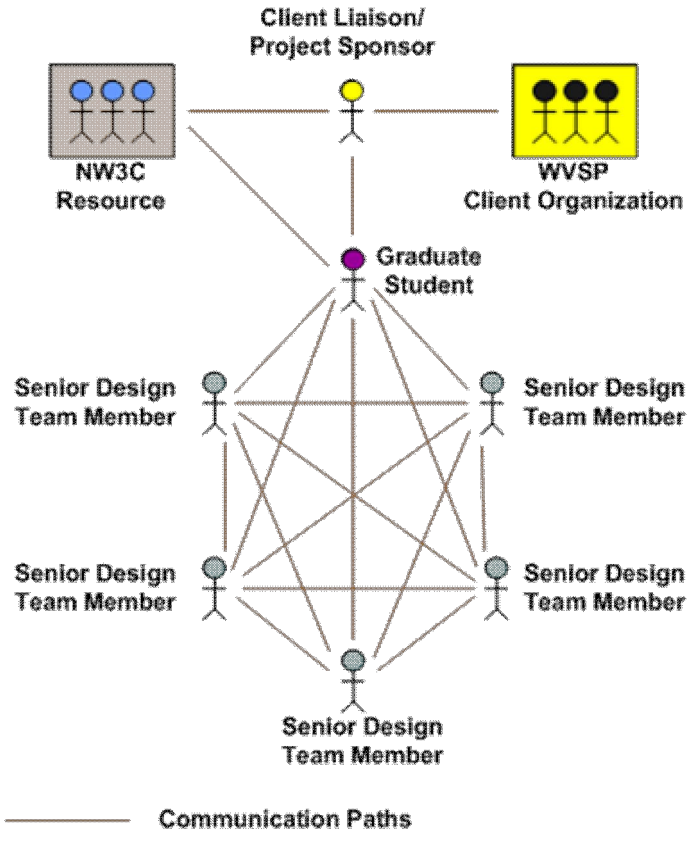


Figure 1.27 WVCRIME Team Structure

Chapter 2 – Requirements

A Note on Organization

From this point forward, the activities of the author are presented in chronological order. This is to give structure to the document and to better describe the development of the software and the author's contributions to that process.

Original proposal

The original proposal for the project provided a broad overview of the system the West Virginia State Police envisioned to replace the current online tip reporting system. The WVCRIME (West Virginia Crime Reduction and Information Management Effort) system was to include an interactive, user-friendly web interface that allowed the general public to file complaints and anonymous tips. The tip or complaint would be forwarded to the appropriate authorities for resolution, and it would be archived in a centralized database. The West Virginia State Police would then be able to search these tips and complaints on various criteria [7]. The proposal in its entirety is in Appendix A.

Team Assembly

Faculty members at West Virginia University announced the WVCRIME project in their classes and asked for those who were interested to contact the faculty advisor for the project, Dr. Roy Nutter. Due to the nature of the project and the skills required, the author felt that this project presented a project tailor-made to her abilities. After discussing it with the faculty advisor, it was decided that the author would be involved with the project. Her involvement with the senior design team was not well defined at the time, as such an experiment had never been undertaken in this department.

The author and the faculty advisor held a meeting with all senior undergraduate students who were interested in working on the project. At this time, a preliminary attempt was made to define the role of the author as a type of "coach." The meaning of "coach" was ambiguous and not well understood by the students.

Student interest in the project was larger than expected, making it necessary to select a team of five students from the pool of applicants. The students were asked to submit resumes, and the faculty advisor and the author chose the best candidates for the project. The team members were selected on the basis of grade point average and appropriate skills. An unintended but beneficial result of this process was that all the team members were computer science majors, as opposed to computer engineering students (although one student was a dual CS and CPE major). This result served to highlight the disparity between the requirements for senior design projects and expected deliverables for a software-oriented project.

In the interest of maintaining good relations throughout the academic department, the author notified those students that were not selected via email. Lack of software development skills, a necessary requirement for successful candidacy, was the primary reason for rejection (many CPE students had not taken advanced CS courses).

Brainstorming

Once the team was in place, the author facilitated a brainstorming session in which questions for the client were proposed, organized, and detailed. The team classified the questions according to subject and the questions were given to the client in advance of the initial interview. The questions attempted to better define the following: who in the client's organization had the final say in requirements, the scope of definitions of particular types of crime, budget issues, systems with which the WVCRIME was to interact, maintenance of the system, training issues, application of the system to actual reports filled out by the West Virginia State Police, the technical expertise of different classes of users, and how to address unintended use of the system (if a member of the general public reported an emergency).

Interviewing

Additional meetings with the client were required in order to allow the development team to produce a requirements definition. Further meetings helped to further clarify the system the West Virginia State Police desired. One meeting in particular helped to clarify the user interface, and a breakdown in communication led to the student who was primarily responsible for the web interface not attending the meeting. This misunderstanding, when coupled with the ambiguous role of the author, would set the stage for an incident later in development that would threaten the productivity and composition of the entire team.

It was agreed with the client at this meeting that a nonfunctional prototype would be delivered in November. The student developing the web interface was upset at having missed the meeting, and also at not being consulted about the delivery date of the prototype. After the author explained to the student the reason for the delivery date, and the consensus that existed among the other team members concerning the date, the student's concerns were mitigated, but not resolved. The failure of the author to realize the stressful environment that her ambiguous role was creating led to the incident midway through development. This incident will be discussed at length at the end of this chapter.

The other interview of note took place at the National White Collar Crime Center (NW3C) in Fairmont, WV. As discussed in chapter one, the NW3C operates the Internet Crime Complaint Center (IC3) for the FBI. The meeting was attended by all team members, the faculty advisor, the client's representative, and various employees of the

NW3C. As with the client interview, a list of questions was provided to the NW3C before the interview.

The interview was educational from the standpoint of group dynamics. While the team was able to obtain answers to many of the questions on the agenda, the meeting was adjourned after two hours without all of the questions answered. This was due to the discussion between the client and the employees of the NW3C about various aspects of the system, such as where the system should be hosted and who should maintain it. The discussion became bogged down in the details of these issues, and the author failed to take control of the meeting and direct it toward a more productive path. Toward the end of the meeting, the faculty advisor was able to channel the discussion back to the topics at hand, but too much time had elapsed for the team to gather all the information they wanted.

Reflection upon these events allowed the author to circumvent this problem at the next meeting with the NW3C. All the developers that maintained the IC3 attended the initial meeting; this plethora of people was not conducive to obtaining specific information in a short amount of time, especially when some of the issues discussed had little effect on the implementation and design of the WVCRIME. Another issue that contributed to the less-than-optimal results of the initial meeting was the degree to which the author was intimidated. The author was concerned about trying to discern the group dynamics at the NW3C before endeavoring to direct the conversation (i.e. she was attempting to discover who she must not interrupt so as to not convey a poor impression of the team or of the establishment she was representing). While this strategy might serve one well when one has the time to take advantage of the information one can gain,

it did not work well when the time allotted to the meeting was short. As a result of these reflections, the author put effort into developing the skill to be direct and focused without appearing aggressive. She was able to employ this new skill at the next meeting with the NW3C with the desired results.

Requirements Analysis

Requirements analysis involved deciphering precisely what the client wants. This section highlights two major contributions the author made to the requirements analysis process.

While the requirements phase is restricted to discovering and documenting what the system should do, it is not inappropriate to extend the discussion at times to methods of accomplishing these goals. Indeed, it is a useful exercise to consider the feasibility of a requirement before committing to developing it. It was in this vein that the author considered how to best identify where a complaint or tip should go after it had been filed. The team discussed using zip codes, but issues of cross-jurisdictional zip codes made this unattractive. The author suggested an interactive map from which users could be forced to select an area before filing a tip or complaint. This solution was later implemented.

Another contribution the author made to this process was the ability to consider abnormal situations that might follow from the use of a system. For example, the West Virginia State Police desired that the system provide a mechanism by which reporters or tipsters can upload files that pertain to the complaint or tip. The author considered this and looked at it from a different angle: what would the consequences be if a criminal uploaded images of a crime *while it was being committed* in order to taunt the police, such as a photograph of a kidnapping victim? What would the consequences be if no one saw that photograph for several days? The system could present a liability to the West Virginia State Police if such an event were to transpire, particularly if the media was informed that the West Virginia State Police had evidence of a kidnapping and no one looked at it for *four days*. Clearly there was a need for someone to cursorily review

uploaded files every day, since they could not be scanned for hot words the way a tip or complaint could. Further consideration on this topic later led the author to consider the question of ascertaining the validity of the uploaded file.

The author also contributed to this phase through leading group analysis and discussion of requirements.

Requirements Definition

The intended audience of the requirements definition is the client. Its purpose is to communicate to the client the development team's understanding of the high-level functionality of the system. It includes all of the functions invoked directly by users while hiding the particulars of how this will be accomplished. In short, it describes, at the highest level of abstraction, what the system is to do, not how it is to be done.

The author oversaw the *content* of the requirements definition as it was submitted to the faculty that evaluated the senior design team's performance. She contributed to some of the diagrams and the introductory section. She provided guidance for the proper development of level one data flow diagrams, specifically that the flow of data must be preserved from one level of diagram to the next. However, she was instructed not to edit or revise the document in any detail before its submission to faculty. Only after the senior design team had fulfilled its requirement to the senior design class was she permitted to edit the document.

Upon editing the document, it was apparent that the document was unacceptable for delivery to the client. Each team member was using his or her own naming, grammar, and capitalization conventions. The definition was inconsistent and ambiguous in sections, and had been poorly proofread in general. These problems led to late delivery of the requirements definition to the client. The author was concerned about the failure to meet the deadline, and proposed two solutions to the faculty advisor. The solutions were to correct the document herself and miss the deadline by days, or to have the team correct the document and miss the deadline by a week or more due to the overhead required in having five people revise a document. The faculty advisor instructed the author to return

the document to the team for correction. The senior design team independently came to the same conclusion, which was indicative of two concepts.

The first concept is that the senior design team desired to improve the work it had done instead of having it done for them, which meant that it was becoming invested in the project as a team. This meant that the team was beginning to "gel," a highly positive factor in the potential of a team's productivity [6]. The second concept is that the requirements definition was the responsibility of the senior design team; it was becoming increasingly clear that the senior design team did not consider the author a team member, which fed into the incident at the end of the requirements phase.

The senior design team committed itself to correcting and polishing the document. The author approved the revisions and the client received the definition a week behind schedule.

To avoid the consistency problems regarding conventions, the team devised a series of baselines to combat this problem. This solution had mixed results. Anyone proofreading the document simply had to consult a list of conventions to determine the appropriateness of capitalization, notation and the like. This made proofreading for uniformity much easier. However, not all team members used the conventions when writing their portions of documents, instead depending upon the document compiler to fix these errors. This led to considerable frustration on the part of the team member who was responsible for this task. Due to time constraints, it was not feasible to point out consistency problems of this nature to the team members responsible for them and have them correct the errors. This is not a trivial discussion; it can take several hours to correct errors of this sort in *one* large document (this project required several documents

that were over two hundred pages). In future endeavors that necessitate heavy documentation requirements, the author recommends that teams decide upon conventions at the outset of the project. In this way, team members will be accustomed to following the agreed-upon conventions from the beginning, and will be able to recall them easily later in the development process.

During the initial interviewing portion of the project, the group discovered the needs for a better communication mechanism and a central repository for all artifacts relating to the project. The former was resolved by creating a Google group, WVCRS. This solution served the purposes of the team well for the most part; however, space constraints on the group account made it infeasible for the WVCRS group to function as a repository for artifacts.

At this point, the team did not have a development laboratory. The senior design team was given space on the University's server for a web page concerning its project. Several team members voiced opposition to keeping artifacts on the school's server because it had had reliability issues in the past. The author proposed opening a Yahoo account for the group in order to use the briefcase functionality it provides. The senior design team did not want to pursue this suggestion. Instead, they opted to have one member open a File Transfer Protocol (FTP) server on his home machine. The author agreed to this on the condition that the server would be constantly available.

The decision to place the artifacts on the team member's home machine was a poor one. It was often necessary to locate the team member in question to obtain the ever-changing address of the server. The address of the server changed every time the machine had to be restarted, every time the electric or cable service went down, and

every time the team member realized that the server had come under attack. The team member in question left for a scheduled school vacation but neglected to make sure the server was available because no one had accessed it for "a while." The choice to place the server on a team member's home machine caused more frustration than any of the alternatives would have caused, and the author recommends any solution over this one.

Client Response to the Requirements Definition

Other than a few minor details dealing mostly with domain knowledge, the client was satisfied that the definition accurately reflected the system desired by the West Virginia State Police. The client emphasized that the diagrams were particularly helpful in clarifying the team's understanding of the system to be developed.

Prototype

The initial development schedule provided that the prototype and the requirements specification be delivered on the same date. At this point in the development lifecycle, it became apparent that this senior design team was performing at least twice the work of any computer engineering senior design team. The author was aware that the senior design team had external requirements to fulfill for the senior design course, but was not aware of the extent of these requirements. As a result of this dual set of deliverables, the team did not meet the requirements specification deadline.

They did meet the prototype deadline, partly due to the willingness of the senior design team to accept the author's help with developing the prototype. The prototype was instrumental in solidifying the requirements concerning the web interface, particularly in clarifying how the system was to determine what type of crime a user was reporting.

As is true of all software engineering endeavors, the client had several changes he wanted the team to incorporate into the system, which required that the requirements definition be updated. The senior design team members began to voice concerns that the system was growing without bound. This concern was addressed with the delivery of the requirements specification.

Requirements Specification & Acceptance Test Plan

The requirements specification is a refinement of the requirements definition. It specifies details of functionality, such as what data items a function receives and outputs. It indicates which type of analysis and design the team is favoring. The audience of this document is the client and the developers, which makes the writing of this document particularly difficult. It must be intelligible to both parties; the challenge comes in writing it in such a way that is clear to both without appearing condescending in tone to either. The author made the decision to include the acceptance test plan as part of the requirements specification to aid in configuration management. The acceptance test plan is used to test the system against the requirements after implementation.

When the author realized that the team would not meet the requirements specification deadline, a meeting of all University-affiliated interested parties was called to resolve the double workload of the senior design team. In attendance were the author's entire thesis committee, all the faculty that evaluated the senior design team, the author, and the senior design team. A rough schedule was fleshed out to eliminate the double workload. The schedule corresponded to deliverables that were more in line with expected deliverables for software development. The delivery date for the requirements specification to the client was pushed back to accommodate the new schedule.

The author's contribution to the requirements specification before the editing process began consisted of developing the data dictionary and specifying the database.

As with the requirements definition, the author was instructed not to edit the requirements specification before submission to faculty. Due to the delay of the

requirements specification, the author was not able to edit the requirements specification until after the University returned from Thanksgiving break. This left a very short amount of time in which to make corrections. When the author received the requirements specification, it was not in acceptable condition to give to the client. Aside from consistency and correctness problems, there was no diagram accompanying each function. The author was adamant that a data flow diagram be included with each function because of the emphasis the client had placed on their capacity to help him visualize what was occurring in any given part of the system.

The week before final examinations is a stressful time for most college students. The senior design team members balked at the idea of creating the requested diagrams, and the author responded by reminding the senior design team that her requests were in the best interest of the client and the project as a whole (they increased the solidity of the understanding of the client, which resulted in more correct requirements). The stress that the ill-defined role of the author was imposing had reached critical mass. The senior design team once again refused, at which point the author stated that the team was not in fact a democracy. The senior design team stated that the senior design team was a democracy, and they a meeting requested with all faculty that were evaluating their performance, one of whom is the faculty advisor for the WVCRIME project and is also the chairperson of the author's thesis committee. They stated that they would not meet with the faculty advisor without the other evaluating faculty present.

The author realized that the development group had reached an impasse and appealed to the faculty advisor of the WVCRIME project. She was advised to drop all communication relating to the project until all parties were able to meet and discuss the

issue. The faculty advisor reaffirmed the author's role in the development process. The author was informed that if the senior design team could not accept the author's role as such, then they would be asked to find a new senior design project.

All requested parties attended the meeting at which this issue was resolved. The author's position as team leader was clearly delineated and was accepted by the senior design team. Members of the senior design team voiced opinions that this should have been made clear from the outset, and with this the author agreed.

If having a graduate student involved with senior design projects is to be a success, then his or her role in the team structure must be clearly defined. This conclusion is not a new idea; poorly defined roles is one of five factors that can lead to team toxicity, a state in which the productivity of a team and the team leader's ability to manage the team is compromised [8]. It was possible that given the nature of this particular team's formation and its experimental nature, this issue could not have been avoided. However, it was the fault of the author that the situation was permitted to persist to the point where the composition of the team was put at risk. The author learned from this nearly fatal mistake, and it is unlikely that she will repeat it.

After the team management issue was resolved, the meeting continued. It was decided that the author would oversee the design of the database. The senior design team member that had previously been assigned to that task voiced immediate frustration about his part in the project. The author informed him that she did not want to usurp his place on the team; she suggested that once the database design was finalized, it would remain under his domain completely. He would be responsible for implementation of the database, and the author would test it when implementation was complete. This was

amenable to the team member. Once that issue was resolved, acquisition of a development space for the team was discussed, as were plans for activities over the approaching holiday break. A final date was also selected for the delivery of the requirements specification to the client.

In order for the system to be delivered by the due date, it was necessary to limit the changes the client was able to make to the requirements. It was determined that in addition to adding this constraint to the requirements specification, the author would personally explain to the client that due to time constraints, he would be unable to make further changes once he approved the requirements specification. Furthermore, it was the author's responsibility to make sure that the client clearly understood the implications of the necessity field that accompanied each function (mandatory, desirable, or optional), the data dictionary, and the entity relationship diagram.

Many of the senior design team members left the area for the break. The author, who remained in town, and the remaining members of the senior design team began setting up the development lab. The author and one of the members of the senior design team were to prepare the framework of the design document over the break. This was not done. The senior design team member was unavailable, and the author was uncertain as to which method of analysis and design was being pursued by the team. The author determined that it was prudent for the entire team to take a break from the project. Though the author was chastised by some of the senior design team members for not completing the framework, the author felt she made the correct decision, especially in light of the recent turmoil the team had undergone.

Review of Requirements Specification with Client

The client approved the requirements specification with a few minor changes. It is noted that the only omission error that the client found in the requirements specification was the omission of one of the fifty-five counties of West Virginia in the data dictionary (the author inadvertently left out Mingo County). He indicated that he understood that any further changes he wanted to make would be in the domain of the next development group.

Chapter 3 – Design

The WVCRIME was designed with structural analysis and design techniques in mind. This choice of design methodology was a natural extension of the data dictionary, data flow diagrams, and the entity-relationship diagram already in place. The primary audience of the design document is the software development team, and so may not be easily understood by non-technical individuals.

Database

The client specified that we implement the database with Microsoft SQL Server 2005, and that we design a relational database (as opposed to an object-relational or object-oriented database). It should be noted that the team members assigned to the database module, the author and a senior design team member, struggled with a solution to modifying columns in the complaint table without wasting an extraordinary amount of space when the types of data collected in a complaint form changed. The elegant solution implemented by the team was suggested by Dr. John Atkins during the requirements phase.

Loss of a Module

At this point in the development process, an additional meeting with the NW3C was needed to further clarify some design issues. The author was concerned about how to verify that an uploaded file was indeed an electronic photograph and not a malicious file. Keeping in mind the previous meeting's outcome, the author was able to keep the discussion focused on the information that needed to be obtained.

Specifically, the development team was at a loss as to how they should secure the files that reporters and tipsters were permitted to upload. Unfortunately, although the IC3 allows the general public to upload files via its system, the team was informed that the FBI handled the process of ascertaining that the uploaded file was what it purported to be; the NW3C was not able to help the development team with this concern. In light of the security threat that this capability posed, the author informed the client of the possible security concerns and advised the client that the upload of evidentiary files should be postponed for the next development group. The client accepted this development.

Design Activities

One requirement of the senior design course was that each student be responsible for a distinct part of the project. Each senior design team member designed his own module and was responsible for implementation of that module. During the design phase, the author focused on team administrative matters.

The author was approached about accepting another graduate student on the team. The author considered the previous difficulties that the team had experienced with poorly-defined roles, and the progress toward cohesiveness the team had made since then. The author determined that given the nature of the request (the graduate student in question needed a topic for a problem report), the potential cost to the team was too great. In order for the graduate student to obtain the requisite information to complete a problem report, the team would have had to spend precious time educating the student about the system, time the team did not have to spare. The author determined that it was not possible to add a new member to the team for that reason without falling victim to Brook's Law, which states that adding new people to a project makes it later [9]. The author also felt that adding a new graduate student to the project might necessitate redefinition of roles. Considering the recent calamities the team had experienced, the author judged that the addition of this student to the team would in no way help the team achieve its goal, and might possibly prevent it from reaching it.

During the design phase, the author engaged in activities that supported the rest of the team. The author dealt with network issues between the University's network and the development lab, software issues, and resource issues.

Chapter 4 – Selection and Integration of New Team Members

The Lane Department of Computer Science and Electrical Engineering at West Virginia University committed to continuing to support the development of the WVCRIME after the current team graduates at the end of the second development semester. One of the responsibilities of the team leader of this team was to select and integrate new team members into the team without falling victim to the aforementioned Brook's Law.

Selection

The first step taken by the author was to email all the people that are part of the department's listserv requesting resumes from interested students. This yielded five resumes; the author judged the team could comfortably accommodate the addition of two members to the team without unduly burdening the current team members, if the integration was done in a controlled fashion.

One factor that differed in the original team selection process concerned the undergraduate level of the applicants. The faculty requested that the new members be from different years in their undergraduate education. The intent was to create a team with members that would graduate in a staggered fashion, thereby never losing all of the development team (and the knowledge of the system they possessed) at one time.

Four of five candidates were interviewed. At the end of the interviews, the author decided that the product would benefit from the addition of three of the candidates.

However, one candidate asked to be dropped from consideration before he was informed that he had been accepted.

Integration

The two remaining candidates were invited to come and meet those team members that had not been present for their interviews. Each team member gave the new members a brief overview about his respective module. The overviews were intended to let the new members get a sense of where they might like to focus their attention when the current team graduates. One new member expressed an interest in the database; he was assigned to help the old team member revise documentation for the database to keep it current. The author believed that in this way, he would learn the details of the database without unduly hampering the old team member's development schedule.

Academic Dishonesty

It came to the author's attention that one of the newly approved team members had engaged in academic dishonesty. On the advice of the faculty advisor, the author waited until the matter has been resolved and then explained to the newly approved member that his or her offer to participate in the WVCRIME had been rescinded.

The team needed another new member. The author asked faculty if she might speak to some of their undergraduate classes about the project. She also emailed some of her former students to inquire if they might be interested in the project. She was able to find another team member to work on the project through her endeavors. Because time grew short at the end of the semester, the author largely bypassed the interview process. The new team member was a former student of the author, which meant that the author had insight into the skills and work ethic of the new member. This team member was assigned to work with the author on documentation so as to not slow the production of the system.

Chapter 5 – Implementation

During this phase, the author continued to provide auxiliary support for the development team and attend to administrative matters. She also began writing code to test the database against the constraints specified in the design document. This code can be found in Appendix B.

Presentation

One of the responsibilities of the team leader during this time was to give a presentation at the West Virginia State Police Academy for high ranking officers in the organization. The author was extremely busy at the time she was notified of the upcoming presentation. The author decided to delegate the responsibility of creating the presentation to one of the senior design students. She approved the presentation after making some changes.

The original presentation had been given by the senior design team as one of its requirements for the senior design course. The changes the author made concerned the intended audience of the presentation. Most technical references were removed. The benefits of the system to the West Virginia State Police were emphasized rather than what functionality the system was to provide.

One senior design student was able to accompany the author and the faculty advisor to the presentation. The presentation was given jointly by the author and the senior design team member.

This presentation is described for the purpose of highlighting the difference between working on an academic project and developing software for an actual client. The senior design team member that participated in the presentation expressed the same sentiments as the author: the appreciation of the client organization for the team's efforts served to motivate them in a way that a purely academic project cannot.

The presentation also served to increase the pressure on the team; the West Virginia State Police indicated that the WVCRIME would be deployed in May, regardless of how ready the development team felt the system was.

SQL Injection and XSS

Through a course the team leader was taking during this time, she became aware of security vulnerabilities that had not been addressed. One major concern the author had for the WVCRIME was security; no team member had any experience of which to speak in this area.

SQL injection is a vulnerability created by the syntax inherent in SQL (Structured Query Language). Malicious users can manipulate responses in forms that take in free text from users to corrupt a database, up to and including deleting the database. The premise is that malicious users may type in text in such a way that the application will interpret the input as code and not data [10]. Releasing software to the client with such a vulnerability was unacceptable. Fortunately, the problem can be easily circumvented by replacing the offending characters with non-offending characters, or by throwing an error when any suspicious character is found [10]. The solution is not the most elegant or robust available, and this should be addressed by the next development group.

XSS (Cross-Site Scripting) is a security vulnerability that is similar in theory to SQL injection. XSS exploits vulnerabilities that are inherent in HTML (Hyper Text Markup Language). Information is entered in HTML forms, and the HTML code can be altered in much the same way as SQL code in regard to SQL Injection. This vulnerability can also be circumvented with the methods described above. It is also necessary to ensure that proper HTML encoding is enforced to prevent malicious users from circumventing the data sanitization [11].

At the end of the implementation phase is not the preferred time to discover potential security flaws, particularly in this project with a deployment date upon which

the client is adamant. The author educated the senior design team members that were developing affected modules and tested for these vulnerabilities.

It stands to reason that if these two vulnerabilities were uncovered, then there will be others. The client indicated that he was expecting to discover flaws and that they would be corrected as they were found.

Trademark Issues

The author was able to apply knowledge gained from courses other than the security class. The team had suggested that the notification that was displayed on the screen to the jurisdictions when they receive new crime complaints or tips should read "You've Got Crime." The client in particular seemed fond of the notification message.

However, America Online has trademarked "You've got mail", both as a trademark and as a service mark [12]. The United States Code provides that any imitation of a trademark that is likely to cause confusion among users is infringing upon the trademark, and those responsible will be held liable for damages [13]. It could be argued that the jurisdiction users of the WVCRIME would make an association with AOL's trademark. It is not the author's intent to argue the case here; the fact that it could foreseeably be an issue was enough for the author to recommend to the client that using the phrase might expose the client to a civil lawsuit. The client stated that the author should use her discretion in choosing a replacement.

Chapter 6 – Documentation Manuals and Testing

The author was responsible for compiling and editing the users and programmers manuals for the WVCRIME project. Each team member was to deliver information concerning his module for inclusion into the manuals. The author developed these documents because the team had lost much time in the spring semester, which adversely affected the implementation schedule. This was due to the loss of approximately a month of development time; integration testing and the development of the manuals were necessarily the last tasks to be done. As the deadline approached, the tasks had to be performed simultaneously. In order for the team to complete both tasks, the author had to compile the manuals and leave the majority of integration testing to the senior design team.

Chapter 7 – Suggestions for changes in Computer Science curriculum for senior design students

The requirements for the senior design students presented unnecessary obstacles during the development of the WVCRIME. Many of the assignments the students completed were not industry-standard documentation and could not be applied to the documentation required of a project of this nature. As a result, the senior design students developed nearly twice the documentation of any other group. This section proposes remedies to this and other problems encountered during the course of developing the WVCRIME.

The author formulated these suggestions on the basis of the following assumptions:

1. The senior design team will have an actual client for whom they are to specify and build a system.
2. The system will lend itself to an adaptation of the classic waterfall lifecycle software development lifecycle methodology.
3. The client will agree to not make any changes to the requirements after approving the requirements specification.

Even though the author assumed the employment of the waterfall method for the purposes of this discussion, she felt that further research into differing development

lifecycles was warranted to provide students the opportunity to select a lifecycle model best suited to their project.

A breakdown of proposed assignments and how they corresponded with senior design assignments is provided in Appendix D.

Requirements Elicitation

In order for students to receive the fullest software development experience from the senior design course, the benefit of requirements elicitation with an actual client cannot be overstated. For the senior design team with which the author worked, the experience transformed the project from an academic exercise into work experience. Having a client added pressure to the team, but the team met this additional stress with enthusiasm. Early in the development lifecycle, the team gained a sense of personal pride in the WVCRIME and the importance of the project. The author does not believe that such cohesion is as easily attained when the project remains in the realm of academia.

The senior design curriculum did not allocate time for the elicitation of requirements. This process forms the most vital part of the software development process. If this part is not completed correctly, all remaining phases in the lifecycle will be completed in vain. The majority of defects in software results from errors that occur in the requirements phase of the software development lifecycle [158]. More recent data indicate that while management and development tools have aided in the reduction of errors, little progress is being made in the validation of the requirements [159]. Errors that occur in this phase are also the most difficult and costly to correct.

Given this information, it is clear that senior design students will benefit from having experience in performing requirements elicitation. Indeed, depriving them of this experience with an actual client would be doing them a disservice.

Allow Time for Change

The most apparent difference in the proposed curriculum and the current curriculum is that there is no time allotted for revision and change. While the capacity of the system to change is somewhat restricted by the assumption that the client will agree to not request additional changes once the requirement specifications are agreed upon, time must be allotted for change before that point in the development lifecycle. Change is inherent in the nature of software development.

The senior design curriculum was very linear and did not provide for substantial changes once an assignment was completed. This added undue stress to the WVCRIME team. Not only were the team members producing two sets of documentation, they had little time to revise documentation when the client requested changes. Had the team not been completely dedicated to the project, this circumstance would have resulted in incorrect requirements. Time must be allotted for revision.

One Set of Documentation

This issue has been discussed at length elsewhere in this document. The author includes it here merely for completeness. The senior design students' assignments should correlate with industry-standard documentation for software development.

Timeline and Development Environment

The senior design curriculum mandated that design be done in the beginning of the second semester. The author did not believe that this allowed enough time to properly implement and test the system. The author proposes that by the end of the first semester design should be largely complete. The development team should also have a plan in place to set up a working environment by this time. The senior design team lost several weeks at the beginning of the second semester because there was no plan in place to obtain and make operational the requisite resources. As a result, the deadline for delivery approached and the system was not tested to the satisfaction of the author.

Not having the design completed by the end of the first semester translated into problems for the development team. The team was attempting to set up the development environment and complete the design documentation simultaneously. As a result, neither task was accomplished efficiently. This loss of time became a problem when the server was attacked later in the semester, which caused the team to lose another week. The team member who developed the web applications also had difficulties with new technology. Because of the amount of time that had been lost, this module was in danger of not being delivered by the deadline. For these reasons (foreseeable and otherwise), the author strongly recommends that the design phase of the lifecycle model be complete by the beginning of the second semester.

Security

The author feels that security should be addressed during all phases in the development lifecycle, beginning with the requirements phase. Software development projects present different issues than those covered by hardware-associated projects. The author recommends that faculty give a seminar on possible security issues that developers may need to consider. This should replace all assignments and information given to computer science senior design students regarding the physical safety of their projects. Each student should submit a short, well-researched report on the potential vulnerabilities to which his or her module may be susceptible and how the student plans to mitigate these circumstances. The student may have to prioritize the risks based upon feedback from the client.

The senior design team and the author failed to give adequate consideration to security vulnerabilities associated with the WVCRIME. The author discovered several vulnerabilities late in development; these were addressed as best as was possible in the time that remained. However, this only added to the frenzied atmosphere as the deadline approached. Had the suggested assignment been carried out before design, the team could have developed more elegant, robust solutions to the vulnerabilities.

Chapter 8 – Further Research

WVCRIME

The client insisted on deploying the system in May 2006, regardless of how the development team felt about the readiness of the WVCRIME. The team became aware of this in late March. This resulted in a scramble to make the system as ready as possible at the end of the implementation phase.

Due to the inexperience of the team in the security area, (including the author), further research remains to be done in this area. Through her coursework, the author became aware of some security vulnerabilities late in the development lifecycle concerning the web application. While the team did its best to alleviate the threats posed by SQL Injection and Cross-Site Scripting, the author recommends that the next development team research possible threats to the system and design and implement more robust security measures as necessary to the security of the system.

Another security issue that merits mention is the inability of the current team to secure relevant files that reporters and tipsters upload. This capability of WVCRIME was listed in the original proposal, and the system was designed to accommodate this functionality. However, the capability to upload files was not included with this project due to security concerns. The author recommends that the next development team research a method whereby this feature can be securely incorporated with the current system.

There remained room for expansion of the functionality of the WVCRIME. The interactive map can be improved to further narrow the jurisdictional areas, currently determined by county. The statistical reports module could be further refined to allow the user to select which statistics he or she would like to be included in the report.

In moving toward a completely electronic crime reporting system, a first step might involve allowing reporters to report crimes that they are reporting solely for insurance purposes. These reports would function as the official police report required by many insurance companies. These reports would involve crimes against property that have no known suspects, thereby freeing officers who respond to these crimes to expend their time and efforts on more urgent matters.

Graduate Team Leadership

The impact of the graduate student's leadership on the productivity and efficiency of the development team remains to be compared to the performance of teams with no graduate student involvement. In order to objectively make this comparison, the author recommends that several future senior design teams be paired with graduate students and their performance evaluated against teams with no graduate student involvement. The success of the team developing the WVCRIME was promising. However, without other like teams against which to compare it, it was impossible to determine if the success was due to the involvement of a graduate student with the team, or if it was a result of the individuals involved in the development effort.

WVU Curriculum

The author has presented high level suggestions for necessary changes to the undergraduate curriculum to better accommodate those projects that primarily involve software development. The author recommends further evaluation of these suggestions by faculty from both the computer engineering and computer science disciplines. If the suggestions are implemented, the author further recommends that the results of the implementation of these suggestions is reviewed by faculty after one iteration through the senior design course in order to refine the requirements for this course.

Chapter 9 – Conclusion

WVCRIME

The development team successfully developed an online crime reporting system for the West Virginia State Police. While further research and implementation remains to be done in the security arena, the end product can be considered a success in that it meets the requirements agreed upon by the development team and the client. The system benefits the state police by automating the process of receiving online tips and complaints, including those complaints that originate from the IC3.

Graduate Team Leadership

This part of the research was a success, although the reasons for this were not necessarily quantifiable. The author feels that it was beneficial to the WVCRIME to have a team member that was responsible for the completion of the project. The team leader was able to gauge the overall progress of the project without having to be concerned about the class-based deliverables required of the senior design team. The graduate student was able to apply advanced knowledge obtained in graduate-level courses to the project where appropriate.

The team benefited from the author's role as liaison to the client in two ways. The author's teaching skills allowed the author to communicate effectively with the client; she was able to identify possible instances of miscommunication that less-experienced team members may not have recognized. The author served as the voice for the client's concerns in regard to the project, and was able to negotiate solutions with the team and the client without impeding development progress.

The team also benefited from the author's continuing education throughout the development process. Examples of this included uncovering potentially dangerous security issues (albeit late in the development process) and considering the impact of laws and regulations on the product.

The team further benefited from the advocacy of the team leader in regard to the deliverable schedule for the senior design team. The author was able to work with the faculty that evaluated the senior design team to develop an ad-hoc deliverable schedule

that would serve as the deliverables for the senior design course and the software product. This eliminated the double set of documentation required of the senior design team.

The team was significantly harmed by the ambiguous role of the author in the first phase of the development. The author strongly advises future teams employing this team structure to clearly define the role of the team members as early in the development process as is possible.

In conclusion, the author feels that her participation in the project was beneficial to development for the aforementioned reasons. However, the experiment must be repeated and the results evaluated to determine if the success was due to graduate student involvement, or if the success was due to other factors. The author urges future teams of this nature to clearly delineate the role of the team leader before the team enters the software development process.

WVU Curriculum

The difficulties of the senior design team in completing two sets of documentation for software development highlighted the need for changes in the course requirements for senior design projects that dealt primarily with software. Some of the assignments required by the curriculum were not applicable to software development. Needs specific to software development projects were not being met by the senior design curriculum, specifically regarding education about the security aspects of software development. The senior curriculum should be altered in the key areas of scheduling phases, deliverables, planning a development environment, and addressing possible security issues to more adequately meet the needs of senior design students that develop software-oriented projects.

Appendices

Appendix A. Original Request for Proposal

WEST VIRGINIA UNIVERSITY

COLLEGE OF ENGINEERING AND MINERAL RESOURCES
LANE DEPARTMENT OF COMPUTER SCIENCE AND ELECTRICAL ENGINEERING

BIOM/CS/CPE/EE480 FALL 2005

==========

REQUEST FOR PROPOSALS **RFP 05-18**

West Virginia Crime Reporting Website and Database

Introduction: Crime reporting is accomplished in a number of ways, including calls directly to various law enforcement officers, through visits to police stations, and calls to 911. In most cases the verbal report of a crime by a victim or witness is transcribed onto a crime report form. The form is then used to initiate a response by patrol officers, detectives, or other appropriate responders. The information is also used to compile statistical crime reports for various purposes. West Virginia State Police wish to provide an alternative method to report non-emergency crime that increase convenience to the reporter as well as reduces the time involved and potential errors with human transfer of information.

The Concept: The idea to be pursued is to provide a user-friendly interactive web site that victims, witnesses, or informants could use to report non-emergency crime. The reporter would enter personal information (name, address, telephone number, etc) and then proceed to provide standardized answers to a series of simple questions that appear on the screen. At the end of the question and answer session, an appropriate completed crime report form would appear for review by the reporter for review and possible editing. Once the reporter was satisfied with the report, it would be submitted with a click, and the report would automatically forward to the appropriate jurisdiction for a response and the information would go to a central data base. An acknowledging e-mail would be returned to the reporter. The data base could be queried for victim names, perpetrator names, crimes of similar type or in a particular location, etc. In addition, the system would be set up to automatically query the database periodically in order to generate standard statistical crime reports that are required.

Functionality: The sponsor desires the following functionality:

Web Interface

- Compatible with multiple browsers
- Requires minimal data rates for download/upload
- Simple to use question and answer format
- Maximum use of yes/no and multiple choice questioning.
- Minimum use of free text responses.
- Ability to accept photograph or other relevant file.
- Display of appropriate completed form(s) for review
- Edit capability before submission'
- Automatic forwarding to appropriate jurisdiction
- Automatically generated acknowledgement

Data base

- Compatible with existing state and national systems
- Capable of automatically generating statistical reports
- Well documented
- Easily modifiable

Students participating on this project will work directly with staff from the West Virginia State Police, the Internet Crime Complaint Center, and the National White Collar Crime Center. The possibility exists to be appointed NW3C interns as part of the project.

Sponsor:
Sergeant C. M. Casto
Senior Investigator
West Virginia State Police
Rt. 3, Box 358A-2
Fairmont, WV 26554
304-367-2747 (w)
304-290-2741 (c)
castoc@wvnet.edu

Faculty Advisor:
Dr. Roy Nutter
933 ESB
293-0405 x 2510
Roy.Nutter@mail.wvu.edu

Appendix B. Database code test

ActivityLog Test

```
--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--ActivityLog,
BEGIN TRANSACTION;
SET IDENTITY_INSERT ActivityLog ON;
--Insert Good Data Into ActivityLog
--ActivityLog table fields verified 3-13-6
INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp, Actions,
tableAccessed)
      VALUES (400, 300, '152.21.2.1', current_timestamp, 'whatever goes
in here1', 'Users');
INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp, Actions,
tableAccessed)
      VALUES (401, 300, '152.21.2.1', current_timestamp, 'whatever goes
in here2', 'Users');
INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp, Actions,
tableAccessed)
      VALUES (402, 301, '152.21.2.3', current_timestamp, 'whatever goes
in here3', 'Users');
--Begin HotWord Table Test
BEGIN TRANSACTION ActivityLog_Table_Test WITH MARK 'begin-activitylog-
test';
--variable to hold error messages
DECLARE @actlog_err int
-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (400, 300, '152.21.2.1', current_timestamp,
'whatever goes in here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to insert dupe pks in ActivityLog';
END
-------------------------------------------------------------------
--try to insert without userID
BEGIN
      INSERT INTO ActivityLog(logID, IPaddress, timestamp, Actions,
tableAccessed)
            VALUES (403, '152.21.2.1', current_timestamp, 'whatever
goes in here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to insert without userID in
ActivityLog';
END
```

```
-----------------------------------------------------------------
--try to insert without IPaddress
BEGIN
      INSERT INTO ActivityLog(logID, userID, timestamp, Actions,
tableAccessed)
            VALUES (404, 300, current_timestamp, 'whatever goes in
here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to insert without IPaddress in
ActivityLog';
END
-----------------------------------------------------------------
--try to insert without timestamp
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, Actions,
tableAccessed)
            VALUES (400, 300, '152.21.2.1', 'whatever goes in here1',
'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - timestamp default not active in
ActivityLog';
END
-----------------------------------------------------------------
--try to insert without Actions
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
tableAccessed)
            VALUES (405, 300, '152.21.2.1', current_timestamp,
'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to insert without Actions in
ActivityLog';
END
-----------------------------------------------------------------
--try to insert without tableAccessed
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions)
            VALUES (406, 300, '152.21.2.1', current_timestamp,
'whatever goes in here1');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'FINE - able to insert without tableAccessed in
ActivityLog';
      ELSE
            PRINT 'ERROR - unable to insert without tableAccessed in
ActivityLog';
END
```

```
-------------------------------------------------------------------
--try to insert without a foreign key in Users
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (407, 3000, '152.21.2.1', current_timestamp,
'whatever goes in here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to insert without corresponding User in
ActivityLog';
END
-------------------------------------------------------------------
--try to insert without a corresponding foreign key to tableAccessed
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (408, 300, '152.21.2.1', current_timestamp,
'whatever goes in here1', 'yippee');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to insert without tableAccessed
corresponding fk in ActivityLog';
END
-------------------------------------------------------------------
--try to violate logID data type
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES ('ivy', 300, '152.21.2.1', current_timestamp,
'whatever goes in here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to violate logID data type in
ActivityLog';
END
-------------------------------------------------------------------
--try to violate userID data type
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (409, 'ivy', '152.21.2.1', current_timestamp,
'whatever goes in here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to violate userID data type in
ActivityLog';
END
```

```
--try to violate IPaddress data type
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (410, 300, 99, current_timestamp, 'whatever goes in
here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to violate IPaddress data type in
ActivityLog';
END
-----------------------------------------------------------------
--try to violate timestamp data type
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (411, 300, '152.21.2.1', 'ivy', 'whatever goes in
here1', 'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to violate timestamp data type in
ActivityLog';
END
-----------------------------------------------------------------
--try to violate Actions datatype
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (412, 300, '152.21.2.1', current_timestamp, 90,
'Users');
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to violate Actions data type in
ActivityLog';
END
-----------------------------------------------------------------
--try to violate tableAccessed data type
BEGIN
      INSERT INTO ActivityLog(logID, userID, IPaddress, timestamp,
Actions, tableAccessed)
            VALUES (413, 300, '152.21.2.1', current_timestamp,
'whatever goes in here1', 90);
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to violate tableAccesssed data type in
ActivityLog';
END
-----------------------------------------------------------------
--try to select
BEGIN
      SELECT * FROM ActivityLog;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to select in ActivityLog';
END
```

```
-----------------------------------------------------------------
--try to update logID
BEGIN
      UPDATE ActivityLog
            SET logID = 414
            WHERE logID = 400;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to update logID in ActivityLog';
END
-----------------------------------------------------------------
--try to update userID
BEGIN
      UPDATE ActivityLog
            SET userID = 301
            WHERE logID = 402;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to update userID in ActivityLog';
END
-----------------------------------------------------------------
--try to update IPaddress
BEGIN
      UPDATE ActivityLog
            SET  IPaddress = '152.21.5.6'
            WHERE logID = 402;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to update IPaddress in ActivityLog';
END
-----------------------------------------------------------------
--try to update timestamp
BEGIN
      UPDATE ActivityLog
            SET timestamp = current_timestamp
            WHERE logID = 402;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to update timestamp in ActivityLog';
END
-----------------------------------------------------------------
--try to update Actions
BEGIN
      UPDATE ActivityLog
            SET Actions = 'new actions'
            WHERE logID = 402;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to update Actions in ActivityLog';
END
```

```
-----------------------------------------------------------------
--try to update tableAccessed
BEGIN
      UPDATE ActivityLog
            SET tableAccessed = 'HotWord'
            WHERE logID = 402;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to update tableAccessed in
ActivityLog';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM ActivityLog;
      SET @actlog_err = @@ERROR;
      IF @actlog_err = 0
            PRINT 'ERROR - able to delete from ActivityLog';
END
ROLLBACK TRANSACTION;
```

Agency, Complaint and CrimeCategory Table Tests

```
--Linda Kress
--WVCRIME DB TEST
--Tables having no foreign keys as columns:
--Agency
--CrimeCategory
--Complaint
BEGIN TRANSACTION;
SET IDENTITY_INSERT Agency ON;
SET IDENTITY_INSERT CrimeCategory ON;
SET IDENTITY_INSERT Complaint ON;
--Insert Good Data Into Agency
--Agency table fields verified 3-13-6
INSERT INTO Agency (name, agencyType, status)
      VALUES ('CIA', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('FBI', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('ATF', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('DOJ', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('WVDMV', 'state', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('Governor', 'state', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('some local organization', 'local', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('Morgantown Mayor', 'local', 1);
--Begin Agency Table Test
BEGIN TRANSACTION Agency_Table_Test WITH MARK 'begin-agency-test';
--variable to hold error messages
DECLARE @agency_err int
-----------------------------------------------------------------
--Try to insert two seperate records with the same primary key
BEGIN
      INSERT INTO Agency (name, agencyType, status)
            VALUES ('Morgantown Mayor', 'federal', 1);
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - Primary key constraint not enforced on
Agency Table. Successful insertion of duplicates.';
      ELSE
            PRINT 'Agency primary key OK -- no dupes';
END
-----------------------------------------------------------------
--Try to insert agency with an invalid type
BEGIN
      INSERT INTO Agency (name, agencyType, status)
            VALUES ('Mon Sheriff', 'shakazulu', 1);
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - agencyType enumeration not enforced';
END
```

```
-----------------------------------------------------------------
--Try to insert agency with no name
BEGIN
      INSERT INTO Agency (agencyType, status)
            VALUES ('state', 1);
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - presence of name in Agency is not enforced';
END
-----------------------------------------------------------------
--insert with no agencyType
BEGIN
      INSERT INTO Agency (name, status)
            VALUES ('Department of Justice', 1);
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - presence of agencyType in Agency is not
enforced';
END
-----------------------------------------------------------------
--insert with no status
BEGIN
      INSERT INTO Agency (name, agencyType)
            VALUES ('Department of Homeland Security', 'federal');
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - presence of status in Agency is not
enforced';
END
-----------------------------------------------------------------
--delete from Agency
BEGIN
      DELETE FROM Agency
            WHERE name = 'CIA';
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - able to delete from Agency, and not supposed
to be';
END
-----------------------------------------------------------------
--illegal update of pk in Agency
BEGIN
      UPDATE Agency
      SET name = 'This should not happen'
      WHERE name = 'ATF';
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - able to update on primary key of Agency;
this shouldn't happen';
END
```

```
-----------------------------------------------------------------
--update status
BEGIN
      UPDATE Agency
      SET status = 0
      WHERE name = 'WVDMV';
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'successful update of status - ok';
      ELSE
            PRINT 'ERROR - unable to update status of an Agency -
problem';
END
-----------------------------------------------------------------
--violate status datatype
BEGIN
      INSERT INTO Agency (name, agencyType, status)
            VALUES ('Department of Natural Resources', 'federal',
'ivy');
      SET @agency_err = @@ERROR;
      IF @agency_err = 0
            PRINT 'ERROR - data type of status in Agency not enforced';
END
-----------------------------------------------------------------
--check logging functionality
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
--insert Good Data into CrimeCategory
--CrimeCategory test updated 3-13-6
--Table was previously "Category"
--added identity ID as primary key
--deleted column "crime"
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (11, 'Victim', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (12, 'Suspect', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (13, 'Victim2', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (14, ''Reporter', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (15, 'Victim3', 1);
--Begin Category Table Test
BEGIN TRANSACTION Category_Table_Test WITH MARK 'begin-category-test';
--variable to hold error messages
DECLARE @category_err int
```

```
-------------------------------------------------------------
--Try to insert two seperate records with the same primary key
BEGIN
      INSERT INTO CrimeCategory (ID, category, status)
            VALUES (11, 'arson', 0);
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'ERROR - Primary key constraint not enforced on
Category Table. Successful insertion of duplicates.';
      ELSE
            PRINT 'CrimeCategory primary key OK -- no dupes';
END
 -------------------------------------------------------------
--Try to insert category with no primary key
BEGIN
      INSERT INTO CrimeCategory (category, status)
            VALUES ('shakazulu', 1);
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'category ID presence not enforced; category ID
possibly created';
END
-------------------------------------------------------------
--Try to insert category with no category field
BEGIN
      INSERT INTO CrimeCategory (ID, status)
            VALUES (16, 1);
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'ERROR - presence of category in CrimeCategory is not
enforced';
END
-------------------------------------------------------------
--insert with no status
BEGIN
      INSERT INTO CrimeCategory (ID, category)
            VALUES (17, 'Reporter2');
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'ERROR - presence of status in CrimeCategory is not
enforced';
END
-------------------------------------------------------------
--delete from CrimeCategory
BEGIN
      DELETE FROM CrimeCategory
      WHERE category = 'Reporter';
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'ERROR - able to delete from CrimeCategory, and not
supposed to be';
END
```

```
-----------------------------------------------------------------
--illegal update of pk in CrimeCategory
BEGIN
      UPDATE CrimeCategory
      SET category = 'This should not happen'
      WHERE category = 'Victim';
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'ERROR - able to update on category of CrimeCategory;
this shouldn't happen';
END
-----------------------------------------------------------------
--update status
BEGIN
      UPDATE CrimeCategory
      SET status = 0
      WHERE category = 'Suspect';
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'successful update of status - ok';
      ELSE
            PRINT 'ERROR - unable to update status of a CrimeCategory -
problem';
END
-----------------------------------------------------------------
--violate status datatype
BEGIN
      INSERT INTO CrimeCategory (ID, category, status)
            VALUES (18, 'Suspect3', 'ivy');
      SET @category_err = @@ERROR;
      IF @category_err = 0
            PRINT 'ERROR - data type of status in CrimeCategory not
enforced';
END
-----------------------------------------------------------------
--check logging functionality
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
--insert Good Data into Complaint
--Complaint table test updated 3-13-6
--added fields text, county, dateOfCrime, locationOfCrime

INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('123', '192.68.12.1', 'text1', 'Mingo',
current_timestamp, 'my backyard');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('124', 192.68.12.1','I killed Barney', 'Monongalia',
current_timestamp, 'my backyard');
```

```
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('125', '192.68.12.11','I double parked', 'Wood',
current_timestamp, 'Main Street');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('126', '192.68.12.12','I littered', 'Hancock',
current_timestamp, 'in a van down by the river');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('127', '192.68.12.13', 'I infringed on a trademark',
'Ritchie', current_timestamp, 'Rt. 50');
--Begin Complaint Table Test
BEGIN TRANSACTION Complaint_Table_Test WITH MARK 'begin-complaint-
test';
--variable to hold error messages
DECLARE @complaint_err int
-----------------------------------------------------------------
--Try to insert two seperate records with the same primary key
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
            VALUES (127, '192.68.12.14', 'whatever dude', 'Kanawha',
current_timestamp, 'the capitol building');
      SET @Complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - Primary key constraint not enforced on
Complaint Table. Successful insertion of duplicates.';
      ELSE
            PRINT 'Complaint primary key OK -- no dupes';
END
-----------------------------------------------------------------
--Try to insert complaintID with no primary key
BEGIN
      INSERT INTO Complaint (IPaddress, text, county, dateOfCrime,
locationOfCrime)
            VALUES ('123.12.22.0', 'shakazulu', 'Wood',
current_timestamp, 'detroit');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'complaintID presence not enforced, or possibly it
was created';
END
-----------------------------------------------------------------
--Try to insert complaintID with no IPaddress field
BEGIN
      INSERT INTO Complaint (complaintID, text, county, dateOfCrime,
locationOfCrime)
            VALUES (128, 'blah', 'Upshur', current_timestamp, 'state');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - presence of IPaddress in Complaint is not
enforced';
END
```

```
-------------------------------------------------------------
--insert with timestamp
--expected: timestamp is supposed to default to the current time;
--this tests if you can enter your own
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, timestamp, text,
county, dateOfCrime, locationOfCrime)
            VALUES (129, '65.256.54.2', current_timestamp, 'blahblah',
'Upshur', current_timestamp, 'state');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'own timestamp field entered successfully';
      ELSE
            PRINT 'ERROR - not allowed to enter timestamp in
Complaint';
END
-------------------------------------------------------------
--Try to insert complaintID with no text field
--This should succeed
BEGIN
      INSERT INTO Complaint (complaintID, county, dateOfCrime,
locationOfCrime)
            VALUES (130, 'Upshur', current_timestamp, 'state');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'presence of text in Complaint is not enforced; this
is correct';
      ELSE
            PRINT 'ERROR - text field does not have to present to enter
a complaint - ERROR';
END
-------------------------------------------------------------
--Try to insert complaintID with no county field
BEGIN
      INSERT INTO Complaint (complaintID, text, dateOfCrime,
locationOfCrime)
            VALUES (131, 'text345', current_timestamp, 'state');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - presence of text in county is not enforced -
ERROR';
      ELSE
            PRINT 'county column in Complaint is being enforced ';
END
```

```
-------------------------------------------------------------
--insert with no dateOfCrime field
--this should be successful
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, timestamp, text,
county, locationOfCrime)
            VALUES (129, '65.256.54.2', current_timestamp, 'blahblah',
'Upshur', 'state');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT ' ';
      ELSE
            PRINT 'ERROR should be able to enter a complaint with no
dateOfCrime';
END
-------------------------------------------------------------
--insert with no locationOfCrime
--should be successful
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, timestamp, text,
county, dateOfCrime)
            VALUES (129, '65.256.54.2', current_timestamp, 'blahblah',
'Upshur', current_timestamp);
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT ' ';
      ELSE
            PRINT 'ERROR should be able to enter a complaint with no
locationOfCrime';
END
-------------------------------------------------------------
--delete from Complaint
BEGIN
      DELETE FROM Complaint
      WHERE complaintID = '124';
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - able to delete from Complaint, and not
supposed to be';
END
-------------------------------------------------------------
--illegal update of pk in Complaint
BEGIN
      UPDATE Complaint
      SET complaintID = '132'
      WHERE complaintID = '129';
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - able to update on primary key of Complaint;
this shouldn't happen';
END
```

```
-------------------------------------------------------------
--illegal update of text
BEGIN
      UPDATE Complaint
      SET text = 'booya'
      WHERE complaintID = '123';
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - successful update of text in Complaint -
ERROR';
      ELSE
            PRINT 'unable to update text of a Complaint - ok';
END
-------------------------------------------------------------
--illegal update of county
BEGIN
      UPDATE Complaint
      SET county = 'Morgan'
      WHERE complaintID = '122';
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - successful update of county in Complaint -
ERROR';
      ELSE
            PRINT 'unable to update county of a Complaint - ok';
END
-------------------------------------------------------------
--illegal update of locationOfCrime
BEGIN
      UPDATE Complaint
      SET locationOfCrime = 'booya'
      WHERE complaintID = '121';
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - successful update of locationOfCrime in
Complaint - ERROR';
      ELSE
            PRINT 'unable to update locationOfCrime of a Complaint -
ok';
END
-------------------------------------------------------------
--violate timestamp datatype
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, timestamp, text,
county, dateOfCrime, locationOfCrime)
            VALUES (133, '65.256.54.2', 'ivy', 'some sort of felony',
'Upshur', current_timestamp, 'your mom's house');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - data type of timestamp in Complaint not
enforced';
END
```

```
-------------------------------------------------------------
--violate text datatype
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
            VALUES (134, '65.256.54.2', 4, 'Upshur', current_timestamp,
'your mom's house');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - data type of text in Complaint not
enforced';
END
-------------------------------------------------------------
--violate county datatype
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
            VALUES (133, '65.256.54.2', 'some sort of felony', 6,
current_timestamp, 'your mom's house');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - data type of county in Complaint not
enforced';
END
-------------------------------------------------------------
--violate dateOfCrime datatype
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
            VALUES (133, '65.256.54.2', 'some sort of felony',
'Upshur', 'having fun at', 'blah');
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - data type of dateOfCrime in Complaint not
enforced';
END
-------------------------------------------------------------
--violate locationOfCrime datatype
BEGIN
      INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
            VALUES (133, '65.256.54.2', 'some sort of felony',
'Upshur', current_timestamp, 9);
      SET @complaint_err = @@ERROR;
      IF @complaint_err = 0
            PRINT 'ERROR - data type of locationOfCrime in Complaint
not enforced';
END
-------------------------------------------------------------
--check logging functionality
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
ComplaintToCrimeCategories Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--ComplaintToCrimeCategory
--fields verified 3-14-6
--changed name of table from ComplaintToCategory to
ComplaintToCrimeCategory

BEGIN TRANSACTION;

--Insert Good Data Into ComplaintToCrimeCategory
--ComplaintToCrimeCategory table fields verified 3-14-6
INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (123, 'Suspect');
INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (123, 'Victim');
INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (124, 'Suspect');
INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (124, 'Reporter');

--Begin ComplaintToCrimeCategory Table Test
BEGIN TRANSACTION ComplaintToCrimeCategory_Table_Test WITH MARK 'begin-
ComplaintToCrimeCategory-test';

--variable to hold error messages
DECLARE @ctcc_err int

-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (123, 'Suspect');
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - able to insert dupe pks in
ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--try to insert without complaintID
BEGIN
      INSERT INTO ComplaintToCrimeCategory(category)
      VALUES ('Victim2');
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - able to insert without complaintID in
ComplaintToCrimeCategory';
END
```

```
-------------------------------------------------------------------
--try to insert without category
BEGIN

      INSERT INTO ComplaintToCrimeCategory(complaintID)
      VALUES (126);
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - able to insert without category in
ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--try to update complaintID
BEGIN
      UPDATE ComplaintToCrimeCategory
            SET complaintID = 126
            WHERE complaintID = 124;
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - able to update complaintID in
ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--try to update category
BEGIN
      UPDATE ComplaintToCrimeCategory
            SET category = 'Reporter'
            WHERE complaintID = 124;
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - able to update catgory in
ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--try to insert with no matching fk in Complaint
BEGIN
      INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (130, 'Suspect');
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - fk complaintID  not enforced in
ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--try to insert with no matching fk in CrimeCategory
BEGIN
      INSERT INTO ComplaintToCrimeCategory(complaintID, category)
      VALUES (127, 'arson');
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - fk category  not enforced in
ComplaintToCrimeCategory';
END
```

```
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM ComplaintToCrimeCategory;
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in
ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM ComplaintToCrimeCategory;
      SET @ctcc_err = @@ERROR;
      IF @ctcc_err = 0
            PRINT 'ERROR - able to delete in ComplaintToCrimeCategory';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
ComplaintToHotWord Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--ComplaintToHotWord
--fields verified 3-14-6

BEGIN TRANSACTION;

--Insert Good Data Into ComplaintToHotWord
--ComplaintToHotWord table fields verified 3-14-6

INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (123, 'murder');
INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (123, 'death');
INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (124, 'kill');
INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (124, 'booya');

--Begin ComplaintToHotWord Table Test
BEGIN TRANSACTION ComplaintToHotWord_Table_Test WITH MARK 'begin-
ComplaintToHotWord-test';

--variable to hold error messages
DECLARE @cthw_err int
-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (123, 'death');
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - able to insert dupe pks in
ComplaintToHotWord';
END
-------------------------------------------------------------------
--try to insert without complaintID
BEGIN
      INSERT INTO ComplaintToHotWord(word)
      VALUES ('heroin');
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - able to insert without complaintID in
ComplaintToHotWord';
END
```

```
-------------------------------------------------------------------
--try to insert without word
BEGIN
      INSERT INTO ComplaintToHotWord(complaintID)
      VALUES (126);
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - able to insert without word in
ComplaintToHotWord';
END
-------------------------------------------------------------------
--try to update complaintID
BEGIN
      UPDATE ComplaintToHotWord
            SET complaintID = 126
            WHERE complaintID = 124;
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - able to update complaintID in
ComplaintToHotWord';
END
-------------------------------------------------------------------
--try to update word
BEGIN
      UPDATE ComplaintToHotWord
            SET word = 'die'
            WHERE complaintID = 124;
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - able to update word in ComplaintToHotWord';
END
-------------------------------------------------------------------
--try to insert with no matching fk in Complaint
BEGIN
      INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (130, 'kidnap');
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - fk complaintID  not enforced in
ComplaintToHotWord';
END
-------------------------------------------------------------------
--try to insert with no matching fk in HotWord
BEGIN
      INSERT INTO ComplaintToHotWord(complaintID, word)
      VALUES (127, 'arson');
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - fk word  not enforced in
ComplaintToHotWord';
END
```

```
-----------------------------------------------------------------
--select
BEGIN
      SELECT * FROM ComplaintToHotWord;
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in ComplaintToHotWord';
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM ComplaintToHotWord;
      SET @cthw_err = @@ERROR;
      IF @cthw_err = 0
            PRINT 'ERROR - able to delete in ComplaintToHotWord';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
ComplaintToIC3Agency Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--ComplaintToIC3agency
--fields verified 3-14-6
BEGIN TRANSACTION;

--Insert Good Data Into ComplaintToIC3agency
--ComplaintToIC3agency table fields verified 3-14-6

INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (123, 'CIA');
INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (123, 'FBI');
INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (124, 'ATF');
INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (124, 'DOJ');

--Begin ComplaintToIC3agency Table Test
BEGIN TRANSACTION ComplaintToIC3agency_Table_Test WITH MARK 'begin-
ComplaintToIC3agency-test';

--variable to hold error messages
DECLARE @ctic3_err int

-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (123, 'CIA');
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - able to insert dupe pks in
ComplaintToIC3agency';
END
-------------------------------------------------------------------
--try to insert without complaintID
BEGIN
      INSERT INTO ComplaintToIC3agency(agency)
      VALUES ('WVDMV');
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - able to insert without complaintID in
ComplaintToIC3agency';
END
```

```
-----------------------------------------------------------------
--try to insert without agency
BEGIN
      INSERT INTO ComplaintToIC3agency(complaintID)
      VALUES (126);
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - able to insert without agency in
ComplaintToIC3agency';
END
-----------------------------------------------------------------
--try to update complaintID
BEGIN
      UPDATE ComplaintToIC3agency
            SET complaintID = 126
            WHERE complaintID = 124;
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - able to update complaintID in
ComplaintToIC3agency';
END
-----------------------------------------------------------------
--try to update agency
BEGIN
      UPDATE ComplaintToIC3agency
            SET agency = 'CIA'
            WHERE complaintID = 124;
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - able to update agency in
ComplaintToIC3agency';
END
-----------------------------------------------------------------
--try to insert with no matching fk in Complaint
BEGIN
      INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (127, 'ATF');
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - fk complaintID  not enforced in
ComplaintToIC3agency';
END
-----------------------------------------------------------------
--try to insert with no matching fk in agency
BEGIN
      INSERT INTO ComplaintToIC3agency(complaintID, agency)
      VALUES (137, 'arson');
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - fk agency  not enforced in
ComplaintToIC3agency';
END
```

```
-----------------------------------------------------------------
--select
BEGIN
      SELECT * FROM ComplaintToIC3agency;
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in ComplaintToIC3agency';
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM ComplaintToIC3agency;
      SET @ctic3_err = @@ERROR;
      IF @ctic3_err = 0
            PRINT 'ERROR - able to delete in ComplaintToIC3agency';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
ComplaintToJurisdiction Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--ComplaintToJurisdiction
--fields verified 3-14-6
--added sentToJurisdiction

BEGIN TRANSACTION;

--Insert Good Data Into ComplaintToJurisdiction
--ComplaintToJurisdiction table fields verified 3-14-6
--added sentToJurisdiction

INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (123, 110, 0, current_timestamp);
INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (123, 111, 0, current_timestamp);
INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (124, 112, 0, current_timestamp);
INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (125, 113, 0, current_timestamp);

--Begin ComplaintToJurisdiction Table Test
BEGIN TRANSACTION ComplaintToJurisdiction_Table_Test WITH MARK 'begin-
ComplaintToJurisdiction-test';

--variable to hold error messages
DECLARE @ctj_err int

-----------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (123, 110, 0, current_timestamp);
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - able to insert dupe pks in
ComplaintToJurisdiction';
END
```

```
-----------------------------------------------------------------
--try to insert without complaintID
BEGIN
      INSERT INTO ComplaintToJurisdiction(jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (113, 0, current_timestamp);

      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - able to insert without complaintID in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to insert without jurisdictionID
BEGIN
      INSERT INTO ComplaintToJurisdiction(complaintID, urgent,
sentToJurisdiction)
      VALUES (126, 1, current_timestamp);
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - able to insert without jurisdictionID in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to insert without urgent
BEGIN
      INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
sentToJurisdiction)
      VALUES (126, 112, current_timestamp);
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - able to insert without urgent in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to insert without sentToJurisdiction
BEGIN
      INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent)
      VALUES (123, 110, 0);
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'FINE'
      ELSE        PRINT 'ERROR - default timestamp not operational in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to update complaintID
BEGIN
      UPDATE ComplaintToJurisdiction
            SET complaintID = 126
            WHERE complaintID = 124;
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - able to update complaintID in
ComplaintToJurisdiction';
END
```

```
-----------------------------------------------------------------
--try to update jurisdiction
BEGIN
      UPDATE ComplaintToJurisdiction
            SET jurisdictionID = '113'
            WHERE complaintID = 124;

      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update jurisdictionID in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to update urgent
BEGIN
      UPDATE ComplaintToJurisdiction
            SET urgent = 1
            WHERE complaintID = 125;
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update urgent in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to update timestamp
BEGIN
      UPDATE ComplaintToJurisdiction
            SET timestamp = current_timestamp
            WHERE complaintID = 124;
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update timestamp in
ComplaintToJurisdiction';
END
-----------------------------------------------------------------
--try to insert with no matching fk in Complaint
BEGIN
      INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (130, 113, 0, current_timestamp);
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - fk complaintID  not enforced in
ComplaintToJurisdiction';
END
```

```
-------------------------------------------------------------------
--try to insert with no matching fk in jurisdictionID
BEGIN
      INSERT INTO ComplaintToJurisdiction(complaintID, jurisdictionID,
urgent, sentToJurisdiction)
      VALUES (123, 66, 0, current_timestamp);
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'ERROR - fk jurisdictionID not enforced in
ComplaintToJurisdiction';
END
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM ComplaintToJurisdiction;
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in
ComplaintToJurisdiction';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM ComplaintToJurisdiction;
      SET @ctj_err = @@ERROR;
      IF @ctj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to delete in
ComplaintToJurisdiction';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
ComplaintToJurisdictionHistory Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--ComplaintToJurisdictionHistory
--fields verified 3-14-6
--added ID
BEGIN TRANSACTION;

--Insert Good Data Into ComplaintToJurisdictionHistory
--ComplaintToJurisdictionHistory table fields verified 3-14-6
--added sentToJurisdiction

INSERT INTO ComplaintToJurisdictionHistory(complaintID, jurisdictionID,
            viewed, viewedBy, sentToJurisdiction)
      VALUES (123, 110, current_timestamp, 300, current_timestamp);
INSERT INTO ComplaintToJurisdictionHistory(complaintID, jurisdictionID,
            viewed, viewedBy, sentToJurisdiction)
      VALUES (124, 111, current_timestamp, 301, current_timestamp);
INSERT INTO ComplaintToJurisdictionHistory(complaintID, jurisdictionID,
            viewed, viewedBy, sentToJurisdiction)
      VALUES (125, 110, current_timestamp, 302, current_timestamp);
INSERT INTO ComplaintToJurisdictionHistory(complaintID, jurisdictionID,
            viewed, viewedBy, sentToJurisdiction)
      VALUES (126, 112, current_timestamp, 303, current_timestamp);

--Begin ComplaintToJurisdictionHistory Table Test
BEGIN TRANSACTION ComplaintToJurisdictionHistory_Table_Test WITH MARK
'begin-ComplaintToJurisdictionHistory-test';

--variable to hold error messages
DECLARE @ctjh_err int

-------------------------------------------------------------------
--try to insert duplicate primary keys

BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID,
jurisdictionID, viewed, viewedBy, sentToJurisdiction)
      VALUES (123, 110, current_timestamp, 300, current_timestamp);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to insert dupes in
ComplaintToJurisdictionHistory';
END
```

```
-----------------------------------------------------------------
--try to insert without complaintID
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(jurisdictionID,
viewed, viewedBy, sentToJurisdiction)
      VALUES (113, current_timestamp, 300, current_timestamp);

      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to insert without complaintID in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert without jurisdictionID
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID, viewed,
viewedBy, sentToJurisdiction)
      VALUES (126, current_timestamp, 300, current_timestamp);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to insert without jurisdictionID in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert without viewed
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID,
jurisdictionID, viewedBy, sentToJurisdiction)
      VALUES (126, 112, 300, current_timestamp);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to insert without viewed in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert without sentToJurisdiction
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID,
jurisdictionID, viewed, viewedBy)
      VALUES (127, 113, current_timestamp, 300);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to insert without sentToJurisdiction in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to update complaintID
BEGIN
      UPDATE ComplaintToJurisdictionHistory
            SET complaintID = 126
            WHERE complaintID = 124;
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to update complaintID in
ComplaintToJurisdictionHistory';
END
```

```
-----------------------------------------------------------------
--try to update jurisdiction
BEGIN
      UPDATE ComplaintToJurisdictionHistory
            SET jurisdictionID = '113'
            WHERE complaintID = 124;

      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to update jurisdictionID in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to update viewed
BEGIN
      UPDATE ComplaintToJurisdictionHistory
            SET viewed = current_timestamp
            WHERE complaintID = 125;
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to update viewed in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to update viewedBy
BEGIN
      UPDATE ComplaintToJurisdictionHistory
            SET viewedBy = 305
            WHERE complaintID = 125;
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to update viewedBy in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to update sentToJursdiction
BEGIN
      UPDATE ComplaintToJurisdictionHistory
            SET sentToJursdiction = current_timestamp
            WHERE complaintID = 124;
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to update sentToJursdiction in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert with no matching fk in Complaint
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID,
jurisdictionID, viewed, viewedBy, sentToJurisdiction)
      VALUES (130, 113, current_timestamp, 300, current_timestamp);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - fk complaintID  not enforced in
ComplaintToJurisdictionHistory';
END
```

```
-----------------------------------------------------------------
--try to insert with no matching fk in jurisdictionID
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID,
jurisdictionID, viewed, viewedBy, sentToJurisdiction)
      VALUES (123, 20, current_timestamp, 300, current_timestamp);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - fk jurisdictionID not enforced in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert with no matching fk in Users
BEGIN
      INSERT INTO ComplaintToJurisdictionHistory(complaintID,
jurisdictionID, viewed, viewedBy, sentToJurisdiction)
      VALUES (123, 20, current_timestamp, 65, current_timestamp);
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - fk Users (viewedBy) not enforced in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--select
BEGIN
      SELECT * FROM ComplaintToJurisdictionHistory;
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM ComplaintToJurisdictionHistory;
      SET @ctjh_err = @@ERROR;
      IF @ctjh_err = 0
            PRINT 'ERROR - able to delete in
ComplaintToJurisdictionHistory';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

HotWord, IC3Agency, and Jurisdiction Table Tests

```
--Linda Kress
--WVCRIME DB TEST
--Tables having no foreign keys as columns:
--
--HotWord
--IC3Agency
--Jurisdiction

BEGIN TRANSACTION;
SET IDENTITY_INSERT HotWord ON;
SET IDENTITY_INSERT IC3agency ON;
SET IDENTITY_INSERT Jurisdiction ON;

--Insert Good Data Into HotWord
--HotWord table fields verified 3-13-6

INSERT INTO HotWord(word, priority, status)
      VALUES ('murder', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('death', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('kill', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('heroin', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('gun', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('kidnap', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('die', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('booya', 2, 1);

--Begin HotWord Table Test
BEGIN TRANSACTION HotWord_Table_Test WITH MARK 'begin-hotword-test';

--variable to hold error messages
DECLARE @hotword_err int
-----------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO HotWord(word, priority, status)
            VALUES ('murder', 3, 0);
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - Primary key constraint not enforced on
HotWord Table. Successful insertion of duplicates.';
      ELSE
            PRINT 'HotWord primary key OK -- no dupes';
END
```

```
-----------------------------------------------------------------
--try to insert record with no primary key
BEGIN
      INSERT INTO HotWord( priority, status)
            VALUES ( 3, 0);
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - Primary key constraint not enforced on
HotWord Table. Word field not required.';
      ELSE
            PRINT 'HotWord primary key OK -- no dupes';
END
-----------------------------------------------------------------
--try to insert record with no status
BEGIN
      INSERT INTO HotWord(word, priority)
            VALUES ('murder', 3);
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - status presence not enforced on HotWord
Table. ';
END
-----------------------------------------------------------------
--try to insert record with no priority
BEGIN
      INSERT INTO HotWord(word, status)
            VALUES ('linda', 1);
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - priority presence not enforced on HotWord
Table. ';
END
-----------------------------------------------------------------
--try to update primary key
BEGIN
      UPDATE HotWord
            SET word = 'kress'
            WHERE word = 'kill';
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - Able to update HotWord primary key';
      ELSE
            PRINT 'HotWord primary key OK -- no dupes';
END
-----------------------------------------------------------------
--try to update status
BEGIN
      UPDATE HotWord
            SET status = 0
            WHERE word = 'heroin';
      SET @hotword_err = @@ERROR;

      IF @hotword_err = 0
            PRINT '  '
      ELSE
            PRINT 'ERROR - unable to update status in HotWord.';
END
```

```
-----------------------------------------------------------------------
--try to update priority
BEGIN
      UPDATE HotWord
            SET priority = 13
            WHERE word = 'heroin';
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT '  '
      ELSE
            PRINT 'ERROR - unable to update priority in HotWord.';
END
-----------------------------------------------------------------------
--try to delete hot word
BEGIN
      DELETE FROM HotWord
            WHERE word = 'kill';
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - able to delete from HotWord';
END
-----------------------------------------------------------------------
--valid select
BEGIN
      SELECT * from HotWord;
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT '  '
      ELSE
            PRINT 'ERROR - unable to select from HotWord.';
END
-----------------------------------------------------------------------
--violate word data type
BEGIN
      INSERT INTO HotWord(word, priority, status)
            VALUES( 12, 3, 0);
      SET @hotword_err = @@ERROR;

      IF @hotword_err = 0
            PRINT 'ERROR - word data type violated in HotWord';
END
-----------------------------------------------------------------------
--violate status data type
BEGIN
      INSERT INTO HotWord(word, priority, status)
            VALUES( 'ivy', 3, 'bite me');
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - status data type violated in HotWord';
END
```

```
-----------------------------------------------------------------
--violate priority data type
BEGIN
      INSERT INTO HotWord(word, priority, status)
            VALUES( 'remedy', 'ivy', 0);
      SET @hotword_err = @@ERROR;
      IF @hotword_err = 0
            PRINT 'ERROR - priority data type violated in HotWord';
END
-----------------------------------------------------------------
--check activity log
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
BEGIN TRANSACTION;

--Insert Good Data Into IC3agency
--IC3agency table fields verified 3-13-6
INSERT INTO IC3agency(name, description)
      VALUES ('FBI', 'Federal Bureau of Investigation');
INSERT INTO IC3agency(name, description)
      VALUES ('NSA', 'We do not exist');
INSERT INTO IC3agency(name, description)
      VALUES ('NRA', 'We like guns');
INSERT INTO IC3agency(name, description)
      VALUES ('PETA', 'We like animals');
--Begin HotWord Table Test
BEGIN TRANSACTION IC3agency_Table_Test WITH MARK 'begin-ic3agency-
test';
--variable to hold error messages
DECLARE @ic3agency_err int
-----------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO IC3agency(name, description)
            VALUES('PETA', 'We do not much care for rabbits, though');
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'ERROR - able to insert dupe pks in IC3agency';
END
-----------------------------------------------------------------
--try to insert record with no name
BEGIN
      INSERT INTO IC3agency( description)
            VALUES('We do not much care for asparagus, though');
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'ERROR - able to insert without pk in IC3agency';
END
```

```
-----------------------------------------------------------------
--try to insert record with no description
BEGIN
      INSERT INTO IC3agency(name)
            VALUES('CORBA');
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'ERROR - able to insert without description in
IC3agency';
END
-----------------------------------------------------------------
--try to update primary key
BEGIN
      UPDATE IC3agency
            SET name = 'FIB'
            WHERE name = 'FBI';
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'ERROR - able update pks in IC3agency';
END
-----------------------------------------------------------------
--try to select
BEGIN
      SELECT * FROM IC3agency;
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in IC3agency';
END
-----------------------------------------------------------------
--valid update
BEGIN
      UPDATE IC3agency
            SET description = 'never heard of them'
            WHERE name = 'NSA';
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update description in IC3agency';
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM IC3agency;
      SET @ic3agency_err = @@ERROR;
      IF @ic3agency = 0
            PRINT 'ERROR - able to insert dupe pks in IC3agency';
END
-----------------------------------------------------------------
--check logging
SELECT * FROM ActivityLog;
ROLLBACK TRANSACTION;
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
```

```
BEGIN TRANSACTION;
--Insert Good Data Into Jurisdiction
--Jurisdiciton table fields verified 3-13-6
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (110, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'charlesincharge@cia.gov', 1);
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (111, 'state', 'WVSP - Wood', 'Chris Casto', '304-656-
8811',
            '99 his way drive', 'apt b', 'williamstown', 'WV', '66666',
'casto@wvnet.org', 1);
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (112, 'state', 'WVSP - Marion', 'Linda', '304-656-8812',
            '99 one way drive', 'apt c', 'fairmont', 'WV', '66666',
'linda@wvnet.org', 1);
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (113, 'local', 'Morgantown Sheriff', 'Bartoolo', '304-656-
8813',
            '99 no way drive', 'apt d', 'morgantown', 'WV', '66666',
'bartoolo@morg.org', 1);

--Begin Jurisdiction Table Test
BEGIN TRANSACTION Jurisdiction_Table_Test WITH MARK 'begin-
jurisdiction-test';
--variable to hold error messages
DECLARE @jurisdiction_err int
-----------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, name,
personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (110, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'charlieincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert dupe pks in Jurisdiction';
END
-----------------------------------------------------------------
--try to insert record without primary key
BEGIN
      INSERT INTO Jurisdiction(type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES ('federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'chuckincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert w/out pk in Jurisdiction or
it was auto-created';
END
```

```
-------------------------------------------------------------------
--violate data type of pk
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, name,
personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES ('ivy', 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'chuckieincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - pk data type violated';
END
-------------------------------------------------------------------
--try to insert without the type field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (114, 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'upchuckincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
type';
END
-------------------------------------------------------------------
--try to insert without the name field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (115, 'federal', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'degaulleincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
name';
END
-------------------------------------------------------------------
--try to insert without the personInCharge field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (116, 'federal', 'CIA', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'chuckwagonincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
personInCharge';
END
```

```
-----------------------------------------------------------------
--try to insert without the phone field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, name,
personInCharge,
            street1, street2, city, state, zip, email, status)
      VALUES (116, 'federal', 'CIA', 'Charles',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'mansonincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
phone';
END
-----------------------------------------------------------------
--try to insert without the street1 field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street2, city, state, zip, email, status)
      VALUES (117, 'federal', 'CIA', 'Charles', '304-656-8810',
            'apt a', 'hoboken', 'NJ', '66666',
'chocolatefactoryincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
street1';
END
-----------------------------------------------------------------
--try to insert without the street2 field
--should allow it
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, city, state, zip, email, status)
      VALUES (118, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'hoboken', 'NJ', '66666',
'norrisincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - able to insert in Jurisdiction without
name';
END
-----------------------------------------------------------------
--try to insert without the city field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, street2, state, zip, email, status)
      VALUES (119, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'NJ', '66666',
'kingincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
city';
END
```

```
-------------------------------------------------------------------
--try to insert without the state field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, street2, city,  zip, email, status)
      VALUES (120, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', '66666',
'charlotteincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
state';
END
-------------------------------------------------------------------
--try to insert without the zip field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, street2, city, state, email, status)
      VALUES (121, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ',
'raeincharge@cia.gov', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
name';
END
-------------------------------------------------------------------
--try to insert without the email field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, street2, city, state, zip, status)
      VALUES (122, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666', 1);
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
email';
END
-------------------------------------------------------------------
--try to insert without the status field
BEGIN
      INSERT INTO Jurisdiction(jurisdictionID, type, personInCharge,
            phone, street1, street2, city, state, zip, email)
      VALUES (123, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'charlotteswebincharge@cia.gov');
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to insert in Jurisdiction without
status';
END
```

```
-----------------------------------------------------------------
--try to delete from the Jurisdiction table
BEGIN
      DELETE * FROM Jurisdiction;
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to delete Jurisdiction';
END
-----------------------------------------------------------------
--try to update the primary key
BEGIN
      UPDATE Jurisdiction
            SET jurisdictionID = 124
            WHERE jurisdictionID = 111;
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to update pk in Jurisdiction';
END
-----------------------------------------------------------------
--try to select
BEGIN
      SELECT * FROM Jurisdiction;
      SET @jurisdiction_err = @@ERROR;
      IF @jurisdiction_err= 0
            PRINT 'ERROR - able to select from Jurisdiction';
END
-----------------------------------------------------------------
--check activity log
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
HotWordToJurisdiction Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--HotWordToJurisdiction
--fields verified 3-14-6

BEGIN TRANSACTION;

--Insert Good Data Into HotWordToJurisdiction
--HotWordToJurisdiction table fields verified 3-14-6
INSERT INTO HotWordToJurisdiction(word, jurisdictionID) VALUES
('murder', 110);
INSERT INTO HotWordToJurisdiction(word, jurisdictionID) VALUES ('kill',
111);
INSERT INTO HotWordToJurisdiction(word, jurisdictionID) VALUES
('booya', 112);
INSERT INTO HotWordToJurisdiction(word, jurisdictionID) VALUES
('heroin', 112);
--Begin HotWordToJurisdiction Table Test
BEGIN TRANSACTION HotWordToJurisdiction_Table_Test WITH MARK 'begin-
HotWordToJurisdiction-test';
--variable to hold error messages
DECLARE @hwtj_err int
-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO HotWordToJurisdiction(word, jurisdictionID) VALUES
('murder', 110);
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'ERROR - able to insert dupe pks in
HotWordToJurisdiction';
END
-------------------------------------------------------------------
--try to insert without word
BEGIN
      INSERT INTO HotWordToJurisdiction(jurisdictionID) VALUES (113);
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'ERROR - able to insert without word in
HotWordToJurisdiction';
END
-------------------------------------------------------------------
--try to insert without jurisdictionID
BEGIN
      INSERT INTO HotWordToJurisdiction(word) VALUES ('die');
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'ERROR - able to insert without jurisdictionID in
HotWordToJurisdiction';
END
```

```
-------------------------------------------------------------------
--try to update word
BEGIN
      UPDATE HotWordToJurisdiction
            SET word = 'kidnap'
            WHERE word = 'booya';
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to update word in
HotWordToJurisdiction';
END
-------------------------------------------------------------------
--try to update jurisdiction
BEGIN
      UPDATE HotWordToJurisdiction
            SET jurisdictionID = '113'
            WHERE word = 'heroin';
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update jurisdictionID in
HotWordToJurisdiction';
END
-------------------------------------------------------------------
--try to insert with no matching fk in hot word
BEGIN
      INSERT INTO HotWordToJurisdiction(word, jurisdictionID)
      VALUES ('hypodermic', 113);

      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'ERROR - fk word  not enforced in
HotWordToJurisdiction';
END
-------------------------------------------------------------------
--try to insert with no matching fk in jurisdictionID
BEGIN
      INSERT INTO HotWordToJurisdiction(word, jurisdictionID)
      VALUES ('gun', 66);
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'ERROR - fk jurisdictionID not enforced in
HotWordToJurisdiction';
END
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM HotWordToJurisdiction;
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in HotWordToJurisdiction';
END
```

```
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM HotWordToJurisdiction;
      SET @hwtj_err = @@ERROR;
      IF @hwtj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to delete in HotWordToJurisdiction';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
Location, Question, StatisticScripts Table Tests

--Linda Kress
--WVCRIME DB TEST
--Tables having no foreign keys as columns:
--
--Location
--QuestionTypes
--StatisticScripts

BEGIN TRANSACTION;
SET IDENTITY_INSERT Location ON;
SET IDENTITY_INSERT QuestionTypes ON;
SET IDENTITY_INSERT StatisticScripts ON;

--Insert Good Data Into Location
--Location table fields verified 3-13-6
INSERT INTO Location (location) VALUES ('here');
INSERT INTO Location (location) VALUES ('there');
INSERT INTO Location (location) VALUES ('yonder');
INSERT INTO Location (location) VALUES ('nowhere');

--Begin HotWord Table Test
BEGIN TRANSACTION Location_Table_Test WITH MARK 'begin-location-test';
--variable to hold error messages
DECLARE @location_err int
-----------------------------------------------------------------
--try to insert dupe pks
BEGIN
      INSERT INTO Location VALUES ('yonder');
      SET @location_err = @@ERROR;
      IF @location_err = 0
            PRINT 'ERROR - able to insert dupes in Location';
END
-----------------------------------------------------------------
--try to update pk
BEGIN
      UPDATE Location
            SET location = 'everywhere'
            WHERE location = 'here';
      SET @location_err = @@ERROR;
      IF @location_err = 0
            PRINT 'ERROR - able to update pk in Location';
END
-----------------------------------------------------------------
--try to select
BEGIN
      SELECT * FROM Location;
      SET @location_err = @@ERROR;
      IF @location_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in Location';
END
```

```
-----------------------------------------------------------------
--try to violate location data type
BEGIN
      INSERT INTO Location VALUES (123);
      SET @location_err = @@ERROR;
      IF @location_err = 0
            PRINT 'ERROR - able to violate pk data type in Location';
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM Location;
      SET @location_err = @@ERROR;
      IF @location_err = 0
            PRINT 'ERROR - able to delete in Location';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
--Insert Good Data Into QuestionTypes
--QuestionTypes table fields verified 3-13-6
INSERT INTO QuestionType(type) VALUES('Victim');
INSERT INTO QuestionType(type) VALUES('Reporter');
INSERT INTO QuestionType(type) VALUES('Suspect');

--Begin QuestionTypes Table Test
BEGIN TRANSACTION QuestionTypes_Table_Test WITH MARK 'begin-
questiontypes-test';
--variable to hold error messages
DECLARE @qtypes_err int
-----------------------------------------------------------------
--try to insert dupe pks
BEGIN
      INSERT INTO QuestionTypes(type) VALUES('Victim');
      SET @qtypes_err = @@ERROR
      IF @qtypes_err = 0
            PRINT 'ERROR - able to insert dupe pks in QuestionTypes.'
END
-----------------------------------------------------------------
--violate type data types
BEGIN
      INSERT INTO QuestionTypes(type) VALUES(213);
      SET @qtypes_err = @@ERROR
      IF @qtypes_err = 0
            PRINT 'ERROR - able to violate pk type data type in
QuestionTypes.'
END
```

```
-----------------------------------------------------------------
--select
BEGIN
      SELECT * FROM QuestionTypes;
      SET @qtypes_err = @@ERROR
      IF @qtypes_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in QuestionTypes.'
END
-----------------------------------------------------------------
--attempt update
BEGIN
      UPDATE QuestionTypes
            SET type = 'oogabooga'
            WHERE type = 'Victim';
      SET @qtypes_err = @@ERROR
      IF @qtypes_err = 0
            PRINT 'ERROR - able to update in QuestionTypes.'
END
-----------------------------------------------------------------
--attempt delete
BEGIN
      DELETE * FROM QuestionTypes;
      SET @qtypes_err = @@ERROR
      IF @qtypes_err = 0
            PRINT 'ERROR - able to delete from QuestionTypes.'
END
-----------------------------------------------------------------
--check logs
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
--Insert Good Data Into StatisticScripts
--StatisticScripts table fields verified 3-13-6
INSERT INTO StatisticScripts(path) VALUES('c:\script1');
INSERT INTO StatisticScripts(path) VALUES('c:\script2');
INSERT INTO StatisticScripts(path) VALUES('c:\script3');

--Begin QuestionTypes Table Test
BEGIN TRANSACTION QuestionTypes_Table_Test WITH MARK 'begin-
questiontypes-test';
--variable to hold error messages
DECLARE @sscripts_err int
-----------------------------------------------------------------
--try to insert dupe pks
BEGIN
      INSERT INTO StatisticScripts(path) VALUES('c:\script1');
      SET @sscripts_err = @@ERROR;
      IF @sscripts_err = 0
            PRINT 'ERROR - able to insert dupe pks in
StatisticScripts';
END
```

```
-----------------------------------------------------------------------
--violate path data type
BEGIN
      INSERT INTO StatisticScripts(path) VALUES(123);
      SET @sscripts_err = @@ERROR;
      IF @sscripts_err = 0
            PRINT 'ERROR - able to violate path data type in
StatisticScripts';
END
-----------------------------------------------------------------------
--try to update
BEGIN
      UPDATE StatisticScripts
            SET path = 'c:\scripts4'
            WHERE path = 'c:\scripts';
      SET @sscripts_err = @@ERROR;
      IF @sscripts_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to update in StatisticScripts';
END
-----------------------------------------------------------------------
--try to select
BEGIN
      SELECT * FROM StatisticScripts;
      SET @sscripts_err = @@ERROR;
      IF @sscripts_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in StatisticScripts';
END
-----------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM StatisticScripts;
      SET @sscripts_err = @@ERROR;
      IF @sscripts_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to delete from StatisticScripts';
END
-----------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
LocationToJurisdiction Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--LocationToJurisdiction
--fields verified 3-14-6

BEGIN TRANSACTION;

--Insert Good Data Into LocationToJurisdiction
--LocationToJurisdiction table fields verified 3-14-6

INSERT INTO LocationToJurisdiction(location, jurisdictionID) VALUES
('Mingo', 110);
INSERT INTO LocationToJurisdiction(location, jurisdictionID) VALUES
('Marion', 111);
INSERT INTO LocationToJurisdiction(location, jurisdictionID) VALUES
('Morgan', 112);
INSERT INTO LocationToJurisdiction(location, jurisdictionID) VALUES
('Monongalia', 112);
--Begin LocationToJurisdiction Table Test
BEGIN TRANSACTION LocationToJurisdiction_Table_Test WITH MARK 'begin-
LocationToJurisdiction-test';
--variable to hold error messages
DECLARE @loctj_err int

-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO LocationToJurisdiction(location, jurisdictionID)
VALUES ('Mingo', 110);
      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'ERROR - able to insert dupe pks in
LocationToJurisdiction';
END
-------------------------------------------------------------------
--try to insert without location
BEGIN
      INSERT INTO LocationToJurisdiction(jurisdictionID) VALUES (113);

      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'ERROR - able to insert without location in
LocationToJurisdiction';
END
```

```
-----------------------------------------------------------------
--try to insert without jurisdictionID
BEGIN
      INSERT INTO LocationToJurisdiction(location) VALUES
('Monongalia');

      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'ERROR - able to insert without jurisdictionID in
LocationToJurisdiction';
END
-----------------------------------------------------------------
--try to update location
BEGIN
      UPDATE LocationToJurisdiction
            SET location = 'Marion'
            WHERE location = 'Morgan';
      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to update location in
LocationToJurisdiction';
END
-----------------------------------------------------------------
--try to update jurisdiction
BEGIN
      UPDATE LocationToJurisdiction
            SET jurisdictionID = '113'
            WHERE location = 'Monongalia';
      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update jurisdictionID in
LocationToJurisdiction';
END
-----------------------------------------------------------------
--try to insert with no matching fk in Location
BEGIN
      INSERT INTO LocationToJurisdiction(location, jurisdictionID)
      VALUES ('hypodermic', 113);

      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'ERROR - fk location  not enforced in
LocationToJurisdiction';
END
```

```
-----------------------------------------------------------------
--try to insert with no matching fk in jurisdictionID
BEGIN
      INSERT INTO LocationToJurisdiction(location, jurisdictionID)
      VALUES ('Marion', 66);

      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'ERROR - fk jurisdictionID not enforced in
LocationToJurisdiction';
END
-----------------------------------------------------------------
--select
BEGIN
      SELECT * FROM LocationToJurisdiction;

      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in LocationToJurisdiction';
END
-----------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM LocationToJurisdiction;
      SET @loctj_err = @@ERROR;
      IF @loctj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to delete in LocationToJurisdiction';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
Mandatory First Inserts for Tests

--Linda Kress
--WVCRIME DB TEST
--inserts necessary to test tables with foreign keys

BEGIN TRANSACTION;
SET IDENTITY_INSERT Complaint ON;
SET IDENTITY_INSERT CrimeCategory ON;
SET IDENTITY_INSERT Jurisdiction ON;
SET IDENTITY_INSERT Tip ON;

--Agency inserts
INSERT INTO Agency (name, agencyType, status)
      VALUES ('CIA', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('FBI', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('ATF', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('DOJ', 'federal', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('WVDMV', 'state', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('Governor', 'state', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('some local organization', 'local', 1);
INSERT INTO Agency (name, agencyType, status)
      VALUES ('Morgantown Mayor', 'local', 1);
-----------------------------------------------------------------
--Location inserts
INSERT INTO Location (location) VALUES ('Mingo');
INSERT INTO Location (location) VALUES ('Marion');
INSERT INTO Location (location) VALUES ('Morgan');
INSERT INTO Location (location) VALUES ('Monongalia');
-----------------------------------------------------------------
--Complaint inserts
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('123', '192.68.12.1', 'text1', 'Mingo',
current_timestamp, 'my backyard');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('124', 192.68.12.1','I killed Barney', 'Monongalia',
current_timestamp, 'my backyard');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('125', '192.68.12.11','I double parked', 'Morgan',
current_timestamp, 'Main Street');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('126', '192.68.12.12','I littered', 'Marion',
current_timestamp, 'in a van down by the river');
INSERT INTO Complaint (complaintID, IPaddress, text, county,
dateOfCrime, locationOfCrime)
      VALUES ('127', '192.68.12.13', 'I infringed on a trademark',
'Marion', current_timestamp, 'Rt. 50');
```

```
-----------------------------------------------------------------
--CrimeCategory inserts
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (11, 'Victim', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (12, 'Suspect', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (13, 'Victim2', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (14, ''Reporter', 1);
INSERT INTO CrimeCategory (ID, category, status)
      VALUES (15, 'Victim3', 1);
-----------------------------------------------------------------
--HotWord inserts
INSERT INTO HotWord(word, priority, status)
      VALUES ('murder', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('death', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('kill', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('heroin', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('gun', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('kidnap', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('die', 2, 1);
INSERT INTO HotWord(word, priority, status)
      VALUES ('booya', 2, 1);
-----------------------------------------------------------------
--IC3agency inserts
INSERT INTO IC3agency(name, description)
      VALUES ('FBI', 'Federal Bureau of Investigation');
INSERT INTO IC3agency(name, description)
      VALUES ('NSA', 'We do not exist');
INSERT INTO IC3agency(name, description)
      VALUES ('NRA', 'We like guns');
INSERT INTO IC3agency(name, description)
      VALUES ('PETA', 'We like animals');
-----------------------------------------------------------------
--Jurisdiction inserts
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (110, 'federal', 'CIA', 'Charles', '304-656-8810',
            '99 my way drive', 'apt a', 'hoboken', 'NJ', '66666',
'charlesincharge@cia.gov', 1);
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (111, 'state', 'WVSP - Wood', 'Chris Casto', '304-656-
8811',
            '99 his way drive', 'apt b', 'williamstown', 'WV', '66666',
'casto@wvnet.org', 1);
```

```
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (112, 'state', 'WVSP - Marion', 'Linda', '304-656-8812',
            '99 one way drive', 'apt c', 'fairmont', 'WV', '66666',
'linda@wvnet.org', 1);
INSERT INTO Jurisdiction(jurisdictionID, type, name, personInCharge,
            phone, street1, street2, city, state, zip, email, status)
      VALUES (113, 'local', 'Morgantown Sheriff', 'Bartoolo', '304-656-
8813',
            '99 no way drive', 'apt d', 'morgantown', 'WV', '66666',
'bartoolo@morg.org', 1);
-----------------------------------------------------------------
--QuestionTypes inserts
INSERT INTO QuestionType(type) VALUES('Victim');
INSERT INTO QuestionType(type) VALUES('Reporter');
INSERT INTO QuestionType(type) VALUES('Suspect');
-----------------------------------------------------------------
--StatisticScripts inserts
INSERT INTO StatisticScripts(path) VALUES('c:\script1');
INSERT INTO StatisticScripts(path) VALUES('c:\script2');
INSERT INTO StatisticScripts(path) VALUES('c:\script3');
-----------------------------------------------------------------
--Tip inserts
INSERT INTO Tip (tipID, IPaddress, timestamp, text, county)
      VALUES (200, '21.544.324.2', current_timestamp,
            'I saw Mommy Kissing Santa Claus', 'Mingo');
INSERT INTO Tip (tipID, IPaddress, timestamp, text, county)
      VALUES (201, '21.544.324.3', current_timestamp,
            'A penny saved', 'Marion');
INSERT INTO Tip (tipID, IPaddress, timestamp, text, county)
      VALUES (202, '21.544.324.4', current_timestamp,
            'Sweet pickled are great', 'Mingo');
COMMIT TRANSACTION;
```

```
Question Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--Question
--fields verified 3-14-6
--removed category

BEGIN TRANSACTION;
SET IDENTITY_INSERT Question ON;
--Insert Good Data Into Question
--Question table fields verified 3-14-6

INSERT INTO Question(questionID, level, status, ask, type)
      VALUES (100, 1, 1, 'Were there any weapons involved?',
'Reporter');
INSERT INTO Question(questionID, level, status, ask, type)
      VALUES (101, 1, 1, 'Was anyone physically harmed?', 'Reporter');
INSERT INTO Question(questionID, parent, level, status, ask, type)
      VALUES (102, 100, 1, 1, 'What type of weapon was involved?',
'Reporter');
INSERT INTO Question(questionID, level, status, ask, type)
      VALUES (103, 1, 1, 'Do you know who committed the crime?',
'Reporter');

--Begin Question Table Test
BEGIN TRANSACTION Question_Table_Test WITH MARK 'begin-Question-test';
--variable to hold error messages
DECLARE @question_err int
-----------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO Question(questionID, level, status, ask, type)
      VALUES (100, 1, 1, 'Were there any weapons involved?',
'Reporter');
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to insert dupe pks in Question';
END
-----------------------------------------------------------------
--try to insert without questionID
BEGIN
      INSERT INTO Question(level, status, ask, type)
      VALUES (1, 1, 'Who's your daddy?', 'Reporter');
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to insert without questionID in
Question';
END
```

```
-----------------------------------------------------------------
--try to insert without level
BEGIN
      INSERT INTO Question(questionID, status, ask, type)
      VALUES (105, 1, 'Got milk?', 'Reporter');

      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to insert without level in Question';
END
-----------------------------------------------------------------
--try to insert without status
BEGIN
      INSERT INTO Question(questionID, level, ask, type)
      VALUES (106, 1, 'Who's on first?', 'Reporter');
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to insert without status in Question';
END
-----------------------------------------------------------------
--try to insert without ask
BEGIN
      INSERT INTO Question(questionID, level, status, type)
      VALUES (107, 1, 1, 'Reporter');
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to insert without ask in Question';
END
-----------------------------------------------------------------
--try to insert without type
BEGIN
      INSERT INTO Question(questionID, level, status, ask)
      VALUES (108, 1, 1, 'Got milk?');
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to insert without type in Question';
END
-----------------------------------------------------------------
--try to update questionID
BEGIN
      UPDATE Question
            SET questionID = 109
            WHERE questionID = 100;
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to update questionID in Question';
END
-----------------------------------------------------------------
--try to update ask
BEGIN
      UPDATE Question
            SET ask = 'To be or not to be'
            WHERE questionID = 102;
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - able to update ask in Question';
END
```

```
-------------------------------------------------------------------
--update parent
BEGIN
      UPDATE Question
            SET parent = 100
            WHERE questionID = 103;
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update parent in Question';
END
-------------------------------------------------------------------
--try to insert with no matching fk in QuestionTypes
BEGIN
      INSERT INTO Question(questionID, level, status, ask, type)
      VALUES (111, 1, 1, 'Were there any weapons involved?',
'rhetorical');
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'ERROR - fk questionID  not enforced in Question';
END
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM Question;
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in Question';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM Question;
      SET @question_err = @@ERROR;
      IF @question_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to delete in Question';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
QuestionToComplaint Table Test

--Linda Kress
--WVCRIME database test

-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--QuestionToComplaint
--fields verified 3-14-6

BEGIN TRANSACTION;
--Insert Good Data Into QuestionToComplaint
--QuestionToComplaint table fields verified 3-14-6
INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (123, 100, 'yes');
INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (123, 101, 'my imaginary friend');
INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (124, 102, 'ice pick');
INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (124, 103, 'no');

--Begin QuestionToComplaint Table Test
BEGIN TRANSACTION QuestionToComplaint_Table_Test WITH MARK 'begin-
QuestionToComplaint-test';

--variable to hold error messages
DECLARE @qtc_err int
-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (123, 100, 'yes');
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to insert dupe pks in
QuestionToComplaint';
END
-------------------------------------------------------------------
--try to insert without complaintID
BEGIN
      INSERT INTO QuestionToComplaint(questionID, answer)
      VALUES (102, 'heroin');
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to insert without complaintID in
QuestionToComplaint';
END
```

```
-----------------------------------------------------------------
--try to insert without questionID
BEGIN
      INSERT INTO QuestionToComplaint(complaintID, answer)
      VALUES (126, 'what was that question again?');

      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to insert without questionID in
QuestionToComplaint';
END
-----------------------------------------------------------------
--try to insert without  answer
BEGIN
      INSERT INTO QuestionToComplaint(complaintID, questionID)
      VALUES (126, 100);
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to insert without answer in
QuestionToComplaint';
END
-----------------------------------------------------------------
--try to update complaintID
BEGIN
      UPDATE QuestionToComplaint
            SET complaintID = 126
            WHERE complaintID = 124;
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to update complaintID in
QuestionToComplaint';
END
-----------------------------------------------------------------
--try to update questionID
BEGIN
      UPDATE QuestionToComplaint
            SET questionID =  100
            WHERE complaintID = 124;
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to update questionID in
QuestionToComplaint';
END
-----------------------------------------------------------------
--try to update answer
BEGIN
      UPDATE QuestionToComplaint
            SET answer =  'nobody knows the trouble I see'
            WHERE complaintID = 124;
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to update answer in
QuestionToComplaint';
END
```

```
-------------------------------------------------------------------
--try to insert with no matching fk in Complaint
BEGIN
      INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (130, 100, 'kidnap');
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - fk complaintID  not enforced in
QuestionToComplaint';
END
-------------------------------------------------------------------
--try to insert with no matching fk in Question
BEGIN
      INSERT INTO QuestionToComplaint(complaintID, questionID, answer)
      VALUES (127, 65, 'arson');
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - fk questionID not enforced in
QuestionToComplaint';
END
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM QuestionToComplaint;
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in QuestionToComplaint';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM QuestionToComplaint;
      SET @qtc_err = @@ERROR;
      IF @qtc_err = 0
            PRINT 'ERROR - able to delete in QuestionToComplaint';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
Tip Table Test

--Linda Kress
--WVCRIME DB TEST
--Tables having no foreign keys as columns:
--
--Tip

BEGIN TRANSACTION;
SET IDENTITY_INSERT Tip ON;

--Insert Good Data Into Tip
--Location table fields verified 3-13-6, added county
INSERT INTO Tip (tipID, IPaddress, timestamp, text, county)
      VALUES (200, '21.544.324.2', current_timestamp,
            'I saw Mommy Kissing Santa Claus', 'Mingo');
INSERT INTO Tip (tipID, IPaddress, timestamp, text, county)
      VALUES (201, '21.544.324.3', current_timestamp,
            'A penny saved', 'Marion');
INSERT INTO Tip (tipID, IPaddress, timestamp, text, county)
      VALUES (202, '21.544.324.4', current_timestamp,
            'Sweet pickled are great', 'Mingo');

--Begin HotWord Table Test
BEGIN TRANSACTION Tip_Table_Test WITH MARK 'begin-tip-test';
--variable to hold error messages
DECLARE @tip_err int
-----------------------------------------------------------------
--try to insert dupe pks
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text, county)
            VALUES (200, '21.544.324.5', current_timestamp,
                  'get organized', 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to insert dupe pks in Tip';
END
-----------------------------------------------------------------
--try to insert with no IP
BEGIN
      INSERT INTO Tip(tipID, timestamp, text, county)
            VALUES (203, '21.544.324.5', current_timestamp,
                  'get organized', 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to insert without IPaddress in Tip';
END
-----------------------------------------------------------------
--try to insert with no timestamp
BEGIN
      INSERT INTO Tip(tipID, IPaddress, text, county)
            VALUES (204, '21.544.324.5',
                  'get organized', 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to insert without timestamp in Tip';
END
```

```
-------------------------------------------------------------------
--try to insert without text
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, county)
            VALUES (205, '21.544.324.5', current_timestamp,
                  'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to insert without text in Tip';
END
-------------------------------------------------------------------
--try to insert without county
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text)
            VALUES (206, '21.544.324.5', current_timestamp,
                  'get organized');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to insert without county in Tip';
END
-------------------------------------------------------------------
--try to update
BEGIN
      UPDATE Tip
            SET text = 'it is a secret'
            Where tipID = 200;
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to update in Tip';
END
-------------------------------------------------------------------
--try to select
BEGIN
      SELECT * FROM Tip;
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'FINE'
      ELSE        PRINT 'ERROR - unable to select in Tip';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM Tip;
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to delete from Tip';
END
-------------------------------------------------------------------
--violate tipID data type
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text, county)
            VALUES ('ivy', '21.544.324.5', current_timestamp,
                  'get organized', 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to violate tipID data type in Tip';
END
```

```
-----------------------------------------------------------------
--violate IPaddress data type
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text, county)
            VALUES (207, 2, current_timestamp,
                  'get organized', 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to violate IPaddress data type in Tip';
END
-----------------------------------------------------------------
--violate timestamp data type
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text, county)
            VALUES (208, '21.544.324.5', 'ivy',
                  'get organized', 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to violate timestamp data type in Tip';
END
-----------------------------------------------------------------
--violate text data type
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text, county)
            VALUES (209, '21.544.324.5', current_timestamp,
                  123, 'Monongalia');
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to violate text data type in Tip';
END
-----------------------------------------------------------------
--violate county data type
BEGIN
      INSERT INTO Tip(tipID, IPaddress, timestamp, text, county)
            VALUES (210,'21.544.324.5', current_timestamp,
                  'get organized', 213);
      SET @tip_err = @@ERROR
      IF @tip_err = 0
            PRINT 'ERROR - able to violate tipID data type in Tip';
END
-----------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
TipToHotWord Table Test
--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--TipToHotWord
--fields verified 3-14-6
BEGIN TRANSACTION;

--Insert Good Data Into TipToHotWord
--TipToHotWord table fields verified 3-14-6
INSERT INTO TipToHotWord(tipID, word)
      VALUES (201, 'murder');
INSERT INTO TipToHotWord(tipID, word)
      VALUES (202, 'death');

--Begin TipToHotWord Table Test
BEGIN TRANSACTION TipToHotWord_Table_Test WITH MARK 'begin-
TipToHotWord-test';

--variable to hold error messages
DECLARE @tthw_err int
-----------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO TipToHotWord(tipID, word)
      VALUES (202, 'death');
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - able to insert dupe pks in TipToHotWord';
END
-----------------------------------------------------------------
--try to insert without tipID
BEGIN
      INSERT INTO TipToHotWord(word)
      VALUES ('heroin');
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - able to insert without tipID in
TipToHotWord';
END
-----------------------------------------------------------------
--try to insert without word
BEGIN
      INSERT INTO TipToHotWord(tipID)
      VALUES (200);
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - able to insert without word in
TipToHotWord';
END
```

```
-------------------------------------------------------------------
--try to update tipID

BEGIN
      UPDATE TipToHotWord
            SET tipID = 302
            WHERE tipID = 201;
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - able to update tipID in TipToHotWord';
END
-------------------------------------------------------------------
--try to update word
BEGIN
      UPDATE TipToHotWord
            SET word = 'die'
            WHERE tipID = 201;
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - able to update word in TipToHotWord';
END
-------------------------------------------------------------------
--try to insert with no matching fk in Tip
BEGIN
      INSERT INTO TipToHotWord(tipID, word)
      VALUES (130, 'kidnap');
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - fk tipID  not enforced in TipToHotWord';
END
-------------------------------------------------------------------
--try to insert with no matching fk in HotWord
BEGI
      INSERT INTO TipToHotWord(tipID, word)
      VALUES (202, 'arson');
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - fk word  not enforced in TipToHotWord';
END
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM TipToHotWord;
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in TipToHotWord';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM TipToHotWord;
      SET @tthw_err = @@ERROR;
      IF @tthw_err = 0
            PRINT 'ERROR - able to delete in TipToHotWord';
END
```

```
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
TipToJurisdiction Table Test

--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--TipToJurisdiction
--fields verified 3-14-6
--added sentToJurisdiction

BEGIN TRANSACTION;

--Insert Good Data Into TipToJurisdiction
--TipToJurisdiction table fields verified 3-14-6
--added sentToJurisdiction
INSERT INTO TipToJurisdiction(tipID, jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (200, 110, 0, current_timestamp);
INSERT INTO TipToJurisdiction(tipID, jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (201, 111, 0, current_timestamp);

--Begin TipToJurisdiction Table Test
BEGIN TRANSACTION TipToJurisdiction_Table_Test WITH MARK 'begin-
TipToJurisdiction-test';
--variable to hold error messages
DECLARE @ttj_err int
-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO TipToJurisdiction(tipID, jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (200, 110, 0, current_timestamp);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - able to insert dupe pks in
TipToJurisdiction';
END
-------------------------------------------------------------------
--try to insert without tipID
BEGIN
      INSERT INTO TipToJurisdiction(jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (113, 0, current_timestamp);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - able to insert without tipID in
TipToJurisdiction';
END
```

```
-----------------------------------------------------------------
--try to insert without jurisdictionID
BEGIN
      INSERT INTO TipToJurisdiction(tipID, urgent, sentToJurisdiction)
      VALUES (202, 1, current_timestamp);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - able to insert without jurisdictionID in
TipToJurisdiction';
END
-----------------------------------------------------------------
--try to insert without urgent
BEGIN
      INSERT INTO TipToJurisdiction(tipID, jurisdictionID,
sentToJurisdiction)
      VALUES (202, 112, current_timestamp);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - able to insert without urgent in
TipToJurisdiction';
END
-----------------------------------------------------------------
--try to insert without sentToJurisdiction
BEGIN
      INSERT INTO TipToJurisdiction(tipID, jurisdictionID, urgent)
      VALUES (202, 110, 0);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - default timestamp not operational in
TipToJurisdiction';
END
-----------------------------------------------------------------
--try to update tipID
BEGIN
      UPDATE TipToJurisdiction
            SET tipID = 214
            WHERE tipID = 201;
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - able to update tipID in TipToJurisdiction';
END
-----------------------------------------------------------------
--try to update jurisdiction
BEGIN
      UPDATE TipToJurisdiction
            SET jurisdictionID = '113'
            WHERE tipID = 200;
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update jurisdictionID in
TipToJurisdiction';
END
```

```
-----------------------------------------------------------------
--try to update urgent
BEGIN
      UPDATE TipToJurisdiction
            SET urgent = 1
            WHERE tipID = 200;

      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update urgent in
TipToJurisdiction';
END
-----------------------------------------------------------------
--try to update timestamp
BEGIN
      UPDATE TipToJurisdiction
            SET timestamp = current_timestamp
            WHERE tipID = 201;
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to update timestamp in
TipToJurisdiction';
END
-----------------------------------------------------------------
--try to insert with no matching fk in Tip
BEGIN
      INSERT INTO TipToJurisdiction(tipID, jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (130, 113, 0, current_timestamp);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - fk tipID  not enforced in
TipToJurisdiction';
END
-----------------------------------------------------------------
--try to insert with no matching fk in jurisdictionID
BEGIN
      INSERT INTO TipToJurisdiction(tipID, jurisdictionID, urgent,
sentToJurisdiction)
      VALUES (202, 66, 0, current_timestamp);
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'ERROR - fk jurisdictionID not enforced in
TipToJurisdiction';
END
```

```
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM TipToJurisdiction;
      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in TipToJurisdiction';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM TipToJurisdiction;

      SET @ttj_err = @@ERROR;
      IF @ttj_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - unable to delete in TipToJurisdiction';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
TipToJurisdictionHistory Table Test
--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPTS
-------------------------------------------------------------------------
--Contains tests for the following tables:
--TipToJurisdictionHistory
--fields verified 3-14-6
--added ID

BEGIN TRANSACTION;

--Insert Good Data Into TipToJurisdictionHistory
--TipToJurisdictionHistory table fields verified 3-14-6
--added sentToJurisdiction
INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
            viewed, viewedBy, sentToJurisdiction)
      VALUES (200, 110, current_timestamp, 300, current_timestamp);
INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
            viewed, viewedBy, sentToJurisdiction)
      VALUES (201, 111, current_timestamp, 301, current_timestamp);

--Begin TipToJurisdictionHistory Table Test
BEGIN TRANSACTION TipToJurisdictionHistory_Table_Test WITH MARK 'begin-
TipToJurisdictionHistory-test';
--variable to hold error messages
DECLARE @ttjh_err int
-------------------------------------------------------------------
--try to insert duplicate primary keys
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
viewed, viewedBy, sentToJurisdiction)
      VALUES (200, 110, current_timestamp, 300, current_timestamp);
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to insert dupes in
TipToJurisdictionHistory';
END
-------------------------------------------------------------------
--try to insert without tipID
BEGIN
      INSERT INTO TipToJurisdictionHistory(jurisdictionID, viewed,
viewedBy, sentToJurisdiction)
      VALUES (113, current_timestamp, 300, current_timestamp);
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to insert without tipID in
TipToJurisdictionHistory';
END
```

```
-----------------------------------------------------------------
--try to insert without jurisdictionID
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, viewed, viewedBy,
sentToJurisdiction)
      VALUES (202, current_timestamp, 300, current_timestamp);

      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to insert without jurisdictionID in
TipToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert without viewed
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
viewedBy, sentToJurisdiction)
      VALUES (202, 112, 300, current_timestamp);
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to insert without viewed in
TipToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to insert without sentToJurisdiction
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
viewed, viewedBy)
      VALUES (202, 113, current_timestamp, 300);
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to insert without sentToJurisdiction in
TipToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to update tipID
BEGIN
      UPDATE TipToJurisdictionHistory
            SET tipID = 555
            WHERE tipID = 201;

      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to update tipID in
TipToJurisdictionHistory';
END
-----------------------------------------------------------------
--try to update jurisdiction
BEGIN
      UPDATE TipToJurisdictionHistory
            SET jurisdictionID = '113'
            WHERE tipID = 201;
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to update jurisdictionID in
TipToJurisdictionHistory';
END
```

```
------------------------------------------------------------------
--try to update viewed
BEGIN
      UPDATE TipToJurisdictionHistory
            SET viewed = current_timestamp
            WHERE tipID = 201;

      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to update viewed in
TipToJurisdictionHistory';
END
------------------------------------------------------------------
--try to update viewedBy
BEGIN
      UPDATE TipToJurisdictionHistory
            SET viewedBy = 305
            WHERE tipID = 201;

      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to update viewedBy in
TipToJurisdictionHistory';
END
------------------------------------------------------------------
--try to update sentToJursdiction
BEGIN
      UPDATE TipToJurisdictionHistory
            SET sentToJursdiction = current_timestamp
            WHERE tipID = 201;
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to update sentToJursdiction in
TipToJurisdictionHistory';
END
------------------------------------------------------------------
--try to insert with no matching fk in Tip
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
viewed, viewedBy, sentToJurisdiction)
      VALUES (130, 113, current_timestamp, 300, current_timestamp);
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - fk tipID  not enforced in
TipToJurisdictionHistory';
END
------------------------------------------------------------------
--try to insert with no matching fk in jurisdictionID
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
viewed, viewedBy, sentToJurisdiction)
      VALUES (202, 20, current_timestamp, 300, current_timestamp);
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - fk jurisdictionID not enforced in
TipToJurisdictionHistory';
END
```

```
-------------------------------------------------------------------
--try to insert with no matching fk in Users
BEGIN
      INSERT INTO TipToJurisdictionHistory(tipID, jurisdictionID,
viewed, viewedBy, sentToJurisdiction)
      VALUES (202, 20, current_timestamp, 65, current_timestamp);

      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - fk Users (viewedBy) not enforced in
TipToJurisdictionHistory';
END
-------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM TipToJurisdictionHistory;
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in
TipToJurisdictionHistory';
END
-------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM TipToJurisdictionHistory;
      SET @ttjh_err = @@ERROR;
      IF @ttjh_err = 0
            PRINT 'ERROR - able to delete in TipToJurisdictionHistory';
END
-------------------------------------------------------------------
--check logging
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

```
User Inserts for Additional Testing
--Linda Kress
--WVCRIME database test
-------------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPT
-------------------------------------------------------------------------
--Contains tests for the Users table
BEGIN TRANSACTION;
SET IDENTITY_INSERT Users ON;
--Insert Good Data Into Users
--Users table fields verified 3-13-6
--added username and onlineTime
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName, middleInitial,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (300, 'ivycress', 'OU812', current_timestamp, 'agency',
            'Linda', 'K', 'Kress', 'XII', 'ATF', 'ivycress@gmail.com',
'304-555-5555',
            'Ms.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, jurisdictionID, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (301, 'dbaker', 'OU813', current_timestamp,
'jurisdiction',
            'Daniel', 'Baker', 'X', 113, 'dbaker@gmail.com', '304-555-
5555',
            'Mr.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (302, 'ngunn', 'OU814', current_timestamp, 'agency',
            'Nathan', 'Gunn', 'X', 'DOJ', 'ngunn@gmail.com', '304-555-
5555',
            'Mr.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (303, 'mizzi', 'OU815', current_timestamp, 'agency',
            'Michael', 'Izzi', 'X', 'WVDMV', 'mizzi@gmail.com', '304-
555-5555',
            'Mr.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, jurisdictionID, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (304, 'ojalali', 'OU816', current_timestamp,
'jurisdiction',
            'Omid', 'Jalali', 'X', 110, 'ojalali@gmail.com', '304-555-
5555',
            'Special Agent', current_timestamp, 1, current_timestamp);
```

```
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (305, 'nmoore', 'OU817', current_timestamp, 'agency',
            'Nathan', 'Moore', 'X', 'ATF', 'nmoore@gmail.com', '304-
555-5555',
            'Inspector', current_timestamp, 1, current_timestamp);
COMMIT TRANSACTION;
```

```
Users Table Test
--Linda Kress
--WVCRIME database test


-----------------------------------------------------------------------
--DO NOT RUN THIS SCRIPT WITHOUT FIRST RUNNING THE MANDATORY INSERT
SCRIPT
-----------------------------------------------------------------------
--Contains tests for the Users table
BEGIN TRANSACTION;
SET IDENTITY_INSERT Users ON:
--Insert Good Data Into Users
--Users table fields verified 3-13-6
--added username and onlineTime
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName, middleInitial,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (300, 'ivycress', 'OU812', current_timestamp,
'jurisdiction',
            'Linda', 'K', 'Kress', 'XII', 'ATF', 'ivycress@gmail.com',
'304-555-5555',
            'Ms.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, jurisdictionID, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (301, 'dbaker', 'OU813', current_timestamp, 'agency',
            'Daniel', 'Baker', 'X', 113, 'dbaker@gmail.com', '304-555-
5555',
            'Mr.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (302, 'ngunn', 'OU814', current_timestamp, 'agency',
            'Nathan', 'Gunn', 'X', 'DOJ', 'ngunn@gmail.com', '304-555-
5555',
            'Mr.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (303, 'mizzi', 'OU815', current_timestamp, 'agency',
            'Michael', 'Izzi', 'X', 'WVDMV', 'mizzi@gmail.com', '304-
555-5555',
            'Mr.', current_timestamp, 1, current_timestamp);
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, jurisdictionID, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (304, 'ojalali', 'OU816', current_timestamp,
'jurisdiction',
            'Omid', 'Jalali', 'X', 110, 'ojalali@gmail.com', '304-555-
5555',
            'Special Agent', current_timestamp, 1, current_timestamp);
```

```
INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (305, 'nmoore', 'OU817', current_timestamp, 'agency',
            'Nathan', 'Moore', 'X', 'ATF', 'nmoore@gmail.com', '304-
555-5555',
            'Inspector', current_timestamp, 1, current_timestamp);
--Begin Users Table Test
BEGIN TRANSACTION Users_Table_Test WITH MARK 'begin-users-test';
--variable to hold error messages
DECLARE @users_err int
-------------------------------------------------------------------
--try to insert duplicate pks
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (305, 'nmoore', 'OU817', current_timestamp, 'agency',
            'Nathan', 'Moore', 'X', 'ATF', 'nmoore@gmail.com', '304-
555-5555',
            'Inspector', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to enter dupe pks in Users';
END
-------------------------------------------------------------------
--try to insert without username
BEGIN
      INSERT INTO Users(userID, password, dateCreated, type, firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (306, 'OU818', current_timestamp, 'agency',
            'Halle', 'Green', 'X', 'ATF', 'hgreen@gmail.com', '304-555-
5555',
            'Miss', current_timestamp, 1, current_timestamp);

      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without username in Users';
END
-------------------------------------------------------------------
--try to insert without password
BEGIN
      INSERT INTO Users(userID, username, dateCreated, type, firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (307, 'lila', current_timestamp, 'agency',
            'Lila', 'McCann', 'X', 'ATF', 'lmccann@gmail.com', '304-
555-5555',
            'Inspector', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without password in Users';
END
```

```
-------------------------------------------------------------------------
--try to insert without dateCreated
BEGIN
      INSERT INTO Users(userID, username, password, type, firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (308, 'kldjf', 'OU8110', 'agency',
            'ooogieboogie', 'man', 'X', 'ATF', 'oman@gmail.com', '304-
555-5555',
            'Sergeant', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without dateCreated in
Users';
END
-------------------------------------------------------------------------
--try to insert without type
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (309, 'never', 'sleep_again', current_timestamp,
            'Insom', 'Niac', 'X', 'ATF', 'iniac@gmail.com', '304-555-
5555',
            'Inspector', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without type in Users';
END
-------------------------------------------------------------------------
--try to insert duplicate pks
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (310, 'procrasinator', 'OU8rrsf', current_timestamp,
'agency',
            'Moore', 'X', 'ATF', 'nonamemoore@gmail.com', '304-555-
5555',
            'Inspector', current_timestamp, 1, current_timestamp);

      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without firstName in Users';
END
```

```
-------------------------------------------------------------------------
--try to insert without lastName
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (311, 'mustard', 'OU8alot', current_timestamp, 'agency',
            'Nathan', 'X', 'ATF', 'nathan@gmail.com', '304-555-5555',
            'Inspector', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without lastName in Users';
END
-------------------------------------------------------------------------
--try to insert without email
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (312, 'ore', 'OU8more', current_timestamp, 'agency',
            'Nathaniel', 'hawthorne', 'X', 'ATF', '304-555-5555',
            'Dead Author', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without email in Users';
END
-------------------------------------------------------------------------
--try to insert without phone
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, title,
            datePasswordChange, status, onlineTime)
      VALUES (313, 'eapoe', 'OU81nevermore', current_timestamp,
'agency',
            'Ed', 'Poe', 'X', 'ATF', 'eapoe@gmail.com',
            'Dead Author', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without phone in Users';
END
-------------------------------------------------------------------------
--try to insert without title
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, status, onlineTime)
      VALUES (314, 'carve', 'OU81', current_timestamp, 'agency',
            'Raymond', 'Carver', 'X', 'ATF', 'rcarver@gmail.com', '304-
            555-5555', current_timestamp, 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to insert without title in Users';
END
```

```
-------------------------------------------------------------------------
--try to insert without datePasswordChange
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            status, onlineTime)
      VALUES (315, 'mmoore', 'OU817whome', current_timestamp, 'agency',
            'Michael', 'Moore', 'X', 'ATF', 'mmoore@gmail.com', '304-
            555-5555', 'Director', 1, current_timestamp);
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'FINE'
      ELSE
            PRINT 'ERROR - datePasswordChange default not working in
Users';
END
-------------------------------------------------------------------------
--try to insert without status
BEGIN
      INSERT INTO Users(userID, username, password, dateCreated, type,
firstName,
            lastName, suffix, agencyName, email, phone, title,
            datePasswordChange, onlineTime)
      VALUES (316, 'dmoore', 'OI8dgdf', current_timestamp, 'agency',
            'Demi', 'Moore', 'X', 'ATF', 'dmoore@gmail.com', '304-555-
5555',
            'Inspector', current_timestamp, current_timestamp);

      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - status default not working in Users';
END
-------------------------------------------------------------------------
--test to make sure an inactive agency's users are also inactive
BEGIN
      UPDATE Agency
            SET status = 0
            WHERE name = 'ATF';
      PRINT 'Start here.'
      SELECT firstName, lastName, agencyName
            FROM Users
            WHERE status = 0;
      PRINT 'If there are no records returned, ERROR - users from an
inactive';
      PRINT 'agency are still able to obtain access.';
--put things back the way they were (all agency users should have
status 1)
      UPDATE Agency
            SET status = 1
            WHERE status = 0;
      UPDATE Users
            SET status = 1
            WHERE status = 0;
END
```

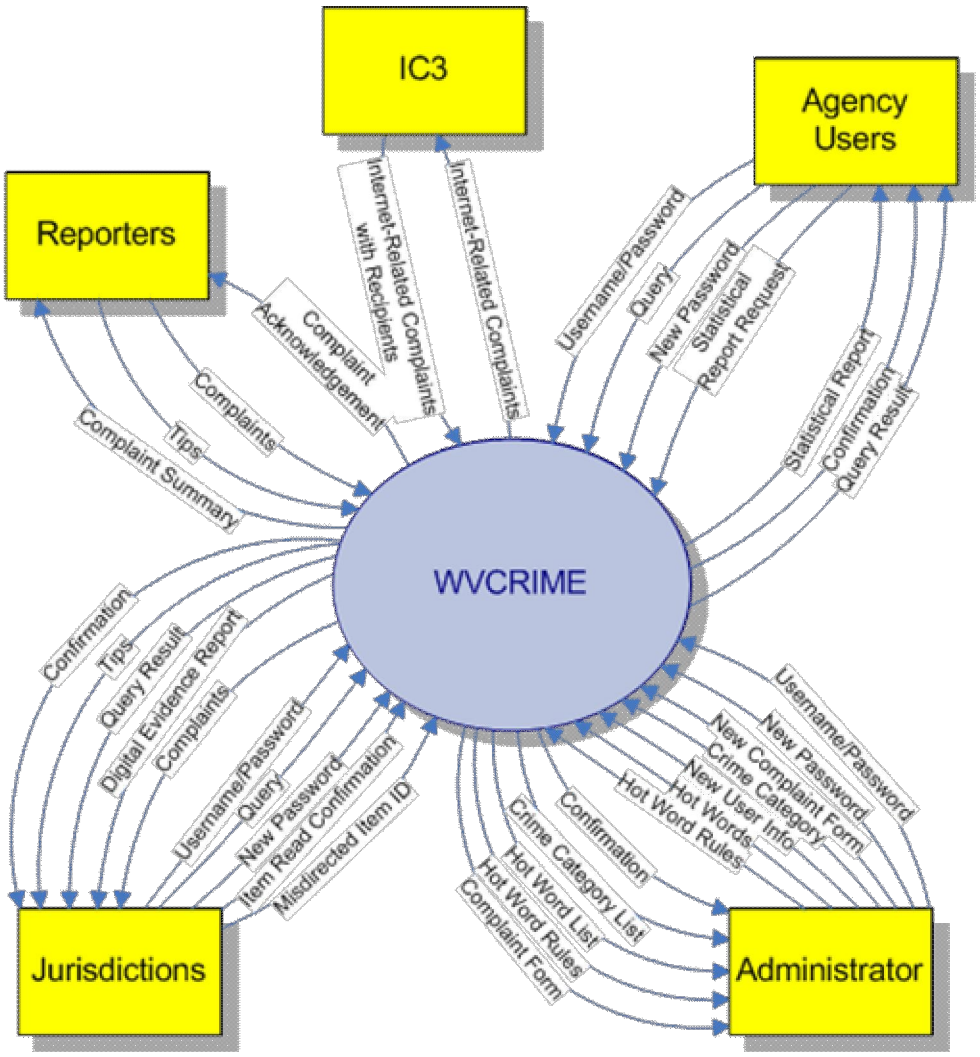
```
-------------------------------------------------------------------------
--try to update userID
BEGIN
      UPDATE Users
            SET userID = 317
            WHERE userID = 301;
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to update userID in Users';
      IF @users_err = 0
            UPDATE Users SET userID = 301 WHERE userID = 317;
END
-------------------------------------------------------------------------
--try to update dateCreated
BEGIN
      UPDATE Users
            SET dateCreated = current_timestamp
            WHERE userID = 302;
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to update dateCreated in Users';
END
-------------------------------------------------------------------------
--try to update firstName
BEGIN
      UPDATE Users
            SET firstName = 'Ivy'
            WHERE userID = 300;
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to update firstName in Users';
      UPDATE Users
            SET firstName = 'Linda'
            WHERE userID = 300;
END
-------------------------------------------------------------------------
--try to update lastName
BEGIN
      UPDATE Users
            SET lastName = 'Ivy'
            WHERE userID = 300;
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to update lastName in Users';
      UPDATE Users
            SET lastName = 'Kress'
            WHERE userID = 300;
END
```

```
-------------------------------------------------------------------------
--ensure date password is changed is being accurately reflected
BEGIN
      UPDATE Users
            SET password = 'nowwhat'
            WHERE userID = 304;
      PRINT current_timestamp;
      PRINT 'If the time immediately preceding is not close to the one
following,
            then ERROR - datePasswordChange is not being updated';
      SELECT datePasswordChange
            FROM Users
            WHERE userID = 304;

END
-------------------------------------------------------------------------
--select
BEGIN
      SELECT * FROM Users;
      SET @users_err = @@ERROR;
      IF @users_err = 0
            PRINT 'FINE';
      ELSE
            PRINT 'ERROR - unable to select in Users';
END
-------------------------------------------------------------------------
--try to delete
BEGIN
      DELETE * FROM Users;
      SET @users_err = @@ERROR
      IF @users_err = 0
            PRINT 'ERROR - able to delete from Users';
END
-------------------------------------------------------------------------
--check logs
BEGIN
      SELECT * FROM ActivityLog;
END
ROLLBACK TRANSACTION;
```

Appendix C. High-level diagram of the WVCRIME

Context Diagram



Appendix D. Proposed Assignment Breakdown

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
1	Referring to table on pg. 10: In preparation for the project team and selection, develop a personal inventory that includes a list of five favorite technologies or engineering subjects that you are interested in pursuing. Also, list the strengths and weaknesses that you bring to a project team. P.13	Monday Week 2	Monday Week 2	**Keep original assignment, add team formation**
N/A	N/A	N/A	Friday Week 2	**Brainstorm for interview questions with client; decide upon one team member (if no grad student involved) to serve as the single point of contact for the client**
			By Friday Week 3	**Interview Client (remember to provide an agenda to client at least 48 hours in advance and to send a thank you letter w/ meeting summary afterwards)**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
2	Extended Problem Statement: A clear Problem Statement should be developed after the needs are identified, the initial research complete, and a project concept selected. The Problem Statement should identify the needs and objectives of the project. Need, Objective, Background, Marketing Requirements, Objective Tree, Ranking of the Needs	Monday Week 5	Monday Week 4	**Introduction, Scope, Purpose, Overall Description, Product Perspective RQ Def.**
3	Web Page	Monday Week 6	Monday Week 5	**Keep original assignment, would also add team management part of assignment 9; also additional client interviews if necessary**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
4	Develop a complete Requirements Specification document for your project that includes the requirements, tradeoffs, constraints, and standards that are applicable to the design. Make sure that the requirements, tradeoffs, and constraints are relevant for the particular problem. A format for documentation is provided in Section 3.8, along with a self-assessment checklist in table 3.7. Also be sure to provide a justification that supports the realism of the requirements.	Monday Week 7	Monday Week 6	**Requirements Definition Due; include context diagram, level one data flow diagrams of individual modules, use-case diagram** **Suggest Following IEEE standards 830 & 1233** **Keep self-assessment checklist if desired, but for purely software projects, leave out the safety, connector standards, meta standards**
5	Individual mid-semester report	Friday Week 7	Friday Week 7	**Keep original assignment; attempt to get feedback from client regarding correctness of requirements by this date**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
6	Concept generation and evaluation	Monday Week 9	Monday Week 7	**This should be a discussion of which analysis and design methodology to follow (e.g. structural analysis or object oriented). Preliminary implementation issues may also be discussed here, but the focus should be on choosing an appropriate design methodology. Documentation of this process may remain in tabular format. It should specify which methodology was selected, and why it is superior to the others in the context of the project. This time should also be used to revise any changes to the requirements definition requested by the client.**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
7	Functional Design Decomposition Design Level 0 Design Level 1 Design Level N Design Alternatives	Monday Week 10	Monday Week 9	**By this time, the team should have obtained feedback from the client on the req def and should be well on the way to developing the requirements specification. The team is not ready to do design. The requirements will differ based on the analysis and design methodology chosen in the previous assignment. Team members should know by this time for which modules they are responsible. Coupling and cohesion discussion remains applicable.**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
8	System Behavior Models	Monday Week 12	Monday Week 10	**The requirements specification should be delivered to the client. Components specific to structural analysis and design methodology: ER Diagram, Level 2&3 data flow diagrams, data dictionary, state transition diagrams, Warnier-Orr diagrams. Components specific to object-oriented analysis and design: UML diagrams (use case diagram, sequence diagrams, activity diagrams, collaboration diagrams, deployment diagrams, state chart diagrams, object diagrams, component diagrams, package diagrams).**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
9	Acceptance Test Plan and Management Documents	Monday Week 14	N/A	**The acceptance test plan should be delivered to the client with the requirements specification (previous assignment). Team management (Gantt Chart specifically) should have been developed previously.**
N/A	N/A	N/A	Week 12	**Non-functional prototype of user interfaces due; gather feedback from client and make any changes necessary to rq def and rq specs; client agrees to not change requirements from here on out (once agreed upon)**

Assignment	Current 480 Assignment	Due Date	Recommended Due Date	**Recommended 480 Assignment**
N/A	N/A	N/A	Week 14	**Work Breakdown Structure part of assignment 9, plan/budget for development environment (specify all software needed, space needed, machines, Internet access, configuration management tools, printers, etc. needed).**
10	Proposal	Monday Week 15	Week 15	**Design Specification due**
11	Detail Design Review	Week 3 of second semester	Friday Week 15	**Keep original assignment**
12	Oral Presentation	Week 15	Week 15	**Keep original assignment**
13	**Individual Final Reports**	**Friday Week 15**	**Friday Week 15**	**Keep original assignment**

Assignment	Current 481 Assignment	Due Date	Recommended Due Date	**Recommended 481 Assignment**
1	N/A	N/A	Week 1	**Set up development environment (configure all software, routers, machines, Internet connections, etc.)**
N/A			Week 2 – Week 7 (inclusive)	**Development of individual modules**
2	Demonstration of Individual Modules	Week 7	Week 8	**Keep original assignment**
			Week 12	**Integration Testing Completed**
			Week 13	**Updated documentation; those portions of the users and programmers manuals that can be completed (i.e. non-volatile functions and functionality) should be completed by this date**
3	Compact disc with documentation	Week 16	Week 14	**Programmers and users manuals completed, system packaged for delivery**
4	Senior design fair	Week 15	Week 15	**Keep original assignment**
5	Individual Reports	Week 15	Week 15	**Keep original assignment**
6	**N/A**		**Week 16**	**Delivery to client**

Bibliography

[1] R. Pressman, Software Engineering: A Practitioner's Approach, 6th ed., New York: McGraw-Hill, 2005, pp. 47-49.

[2] W. Royce, "Managing the Development of Large Software Systems." *Proceedings, IEEE WESCON*, [Online serial] (1970 Aug.), [21 Mar. 2006], Available at HTTP://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf.

[3] B. Boehm, "A spiral model of software development and enhancement." *Computer* 21 (5), (May 1988), [21 Mar. 2006], pp. 1-9 Available at HTTP://ieeexplore.ieee.org/.

[4] A. Mortimer, "Project management in rapid application development." in Project Management for Software Engineers, IEE Colloquium 11 Dec 1995 pp. 5/1 - 5/3.

[5] F. Baker, "Chief Programmer team management of production programming", 1972, IBM Systems Journal, vol. 11, no. 1, pp. 56-73 (1972) [24 March 2006] Available at HTTP://www.research.ibm.com/journal/sj/111/ibmsj1101E.pdf.

[6] R. Pressman, Software Engineering: A Practitioner's Approach, 5th ed., New York: McGraw-Hill, 2001, pp. 59, 62 – 63.

[7] R. Nutter, Roy, "West Virginia Crime Reporting Website and Database," RFP 05-18, West Virginia University, Fall 2005.

[8] M. Jackman, "Homeopathic Remedies for Team Toxicity," Software, IEEE vol. 15, no. 4, July-Aug. 1998 pp. 43 – 45.

[9] F. Brooks, The Mythical Man-Month. Reading, Massachusetts: Addison-Wesley Publishing Company, 1975.

[10] J. Held, "Preventing SQL Injection Security Vulnerabilities through Data Sanitization," in Information Security Management Handbook, 5th ed., vol. 2, H. Tipton and M. Krause, Eds. Boca Raton: Auerbach Publications, 2003. pp. 311 – 325.

[11] J. Held, "Cross-Site Scripting (XSS)," in Information Security Management Handbook, 5th ed., vol. 2, H. Tipton and M. Krause, Eds. Boca Raton: Auerbach Publications, 2003. pp. 279-289.

[12] United States Patent and Trademark Office, "You've Got Mail Typed drawing serial #: 75487493, registration #: 2921866," [25 Mar. 2006], Available at HTTP://tess2.uspto.gov/.
[13] Lanham Act § 32, 15 U.S.C.S. § 1114.
[14] Northern Arizona University Police Department, "Online Reporting," [Online document], [2006 March 23], Available at HTTP://www4.nau.edu/police/Online_Reporting.htm.
[15] University of Nevada, Reno – Police Services, "Online Crime Report Form," [Online document], [2006 March 23], Available at HTTP://www.unr.edu/police/reports/reportform.html.
[16] Ohlone College Campus Security, "Online Confidential Crime Report Form," [Online document], [2006 March 23], Available at HTTP://www.ohlone.edu/org/security/form-confidcrimereport.html.
[17] University of Oregon Department of Public Safety, "Crime Reporting," [Online document], [2006 March 23], Available at HTTP://safetyweb.uoregon.edu/safety/crime_report.htm.
[18] Colorado State University Police Department, "Online Crime Reporting Form," [Online document], [2006 March 23], Available at HTTP://www.colostate.edu/Depts/CSUPD/crimereport.htm.
[19] Indiana University Police Department, "Online Forms," [Online document], [2006 March 23], HTTP://www.indiana.edu/~iupd/forms.htm.
[20] Virginia Tech Police, "Online Reporting," [Online document], [2006 March 23], Available at HTTP://www.police.vt.edu/reporting/reporting.htm.
[21] Tidewater Community College, "Anonymous Witness Form for Security @ TCC," [Online document], [2006 March 23], Available at HTTP://www.tcc.edu/security/witness.htm.
[22] Hampton University Police, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.hamptonu.edu/police/silent_witness.cfm.
[23] Christopher Newport University Police, "Report a Crime," [Online document], [2006 March 23], Available at HTTP://police.cnu.edu/silentwitness/silent.html.

[24] Bridgewater State College Police Department, "Crime Reporting," [Online document], [2006 March 23], Available at HTTP://www.bridgew.edu/Police/crimereporting.cfm.
[25] San José State University Police Department, "On-line Confidential Crime Reporting," [Online document], [2006 March 23], Available at HTTP://www2.sjsu.edu/police/pages/resources/onlinereport2.html.
[26] Old Dominion University, "Crime Report – Silent Witness Form," [Online document], [2006 March 23], Available at HTTPs://jasper.ts.odu.edu/Apps/ODUPolice/silentwitness.nsf/crimereport?OpenForm.
[27] University of Maryland Eastern Shore Public Safety & Police Services, "UMES Public Safety – 'Silent Watch' Anonymous Crime Reporting," [Online document], [2006 March 23], Available at HTTP://www.umes.edu/police/silentwatch.cfm.
[28] Claremont Colleges Department of Campus Safety, "Silent Witness Program," [Online document], [2006 March 23], Available at HTTP://www.cuc.claremont.edu/cs/silent_witness_program.htm.
[29] Weber State University State Police, "Silent Witness and Crime Reporting," [Online document], [2006 March 23], Available at HTTP://community.weber.edu/police/witness.html.
[30] Itawamba Community College, "Silent Witness – Crime Reporting," [Online document], [2006 March 23], Available at HTTP://www.iccms.edu/info/silent_witness.asp.
[31] Radford University Police, "Silent Witness Crime Report," [Online document], [2006 March 23], Available at HTTP://www.radford.edu/~police/silent.htm.
[32] James Madison University Public Safety, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.jmu.edu/pubsafety/SilentWitness.shtml.
[33] University of South Alabama Police Department, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.southalabama.edu/police/submitcrime.html.

[34] University of Arkansas Police Department, "If you have questions, information or tips send them to us.," [Online document], [2006 March 23], Available at HTTP://uapd.uark.edu/uapd-feedback.html.
[35] Eastern Connecticut State University, "Anonymous Crime Reporting Form," [Online document], [2006 March 23], Available at HTTP://cit3.easternct.edu/websurvey/TakeSurvey.asp?SurveyID=5KHnn321mn72G.
[36] American University Public Safety, "Crime Stoppers," [Online document], [2006 March 23], Available at HTTP://www.american.edu/finance/dps/crmstop.html.
[37] Florida State University Police, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.police.fsu.edu/silentwitness.cfm.
[38] Augusta State University Department of Public Safety, "Augusta State University Public Safety," [Online document], [2006 March 23], Available at HTTP://www.aug.edu/public_safety/Silent_witness_report.html.
[39] University of Hawaii Campus Security, "Anonymous Tip Form," [Online document], [2006 March 23], Available at HTTP://www.hawaii.edu/uhmle/tip.html.
[40] Northwest Nazarene University, "Incident Report," [Online document], [2006 March 23], Available at HTTP://www.nnu.edu/1633/.
[41] Governors State University Campus Police, "Report an Incident," [Online document], [2006 March 23], Available at HTTP://www.govst.edu/sas/t_police.asp?id=1728.
[42] Butler University Police Department, "Silent Watch," [Online document], [2006 March 23], Available at HTTP://www.butler.edu/bupd/silentwatch.asp.
[43] University of Iowa Department of Public Safety, "Report Crime on Campus," [Online document], [2006 March 23], Available at HTTP://www.uiowa.edu/~pubsfty/witness.htm.
[44] Kansas City Kansas Community College, "Crime Report Form," [Online document], [2006 March 23], Available at HTTP://www.kckcc.edu/cgi-bin/police/crimereportform.pl.
[45] University of Louisville Department of Public Safety, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.louisville.edu/admin/dps/police/witness.htm.

[46] University Police/Department of Public Safety – Loyola University New Orleans, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.loyno.edu/police/.
[47] Bates College Security and Campus Safety, "Anonymous Reporting Form," [Online document], [2006 March 23], Available at HTTP://www.bates.edu/x57504.xml.
[48] Loyola College in Maryland Department of Public Safety, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.loyola.edu/publicsafety/.
[49] Department of Public Safety University of Massachusetts Boston, "Confidential Witness," [Online document], [2006 March 23], Available at HTTP://www.publicsafety.umb.edu/GRAPHIC/MAIN/witness.html.
[50] Delta College Department of Public Safety, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.delta.edu/cops/witness.htm.
[51] Ethics Point – University of Minnesota, "Confidential Reporting Service for University of Minnesota," [Online document], [2006 March 23], Available at HTTPS://secure.ethicspoint.com/domain/en/report_custom.asp?clientid=9167.
[52] Meridian Community College, "Silent Witness Crime Report," [Online document], [2006 March 23], Available at HTTP://www.mcc.cc.ms.us/housing/silent_witness_crime__report.htm.
[53] Northwest Missouri State University, "Report Crime," [Online document], [2006 March 23], Available at HTTP://www.nwmissouri.edu/safety/reportcrime.htm.
[54] University of Montana Public Safety, "Crime Report Form," [Online document], [2006 March 23], Available at HTTP://www.umt.edu/publicsafety/crimereport.html.
[55] University of Nebraska Kearney, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.umt.edu/publicsafety/crimereport.html.
[56] University of New Hampshire Police Department, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.unh.edu/upd/Contact-Us.html.

[57] Ramapo College of New Jersey, "Ramapo College Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.ramapo.edu/studentlife/security/silentWitness.html.
[58] New Mexico Military Institute Police Department, "Voluntary Statement," [Online document], [2006 March 23], Available at HTTP://www.nmmi.edu/police/voluntary_statement.htm.
[59] Skidmore College, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.skidmore.edu/administration/safety/silent_witness.htm.
[60] Elon Campus Safety and Police Tip Line, "Tip Line – The Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.elon.edu/forms/safety/tipline.aspx.
[61] North Dakota State College of Science, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.ndscs.nodak.edu/police/silentform.html.
[62] Myers University Campus Safety, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.dnmyers.edu/witness.htm.
[63] Southwestern Oklahoma State University Department of Public Safety Campus Police/Risk Management, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.swosu.edu/safety/campus-police/witness.asp.
[64] Pennsylvania College of Technology Penn College Police, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.pct.edu/police/silent_witness.htm.
[65] Providence College Safety and Security, "Silent Witness Program," [Online document], [2006 March 23], Available at HTTP://www.providence.edu/security/silent-witness.html.
[66] Wofford College Public Safety, "Campus Safety – Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.wofford.edu/publicSafety/formSilentWitness.asp.

[67] University of South Dakota Public Safety, "Silent Witness Form," [Online document], [2006 March 23], Available at HTTP://www.usd.edu/publicsafety/crimeinfo/witness.cfm.
[68] Walters State Crime Watch/Silent Witness, "Crime Watch Form," [Online document], [2006 March 23], Available at HTTP://www.wscc.cc.tn.us/campuspolice/crimewatch.asp.
[69] University of Texas at Brownsville and Texas Southmost College, "Silent Witness," [Online document], [2006 March 23], Available at HTTPs://www.utbtsc.edu/form/campolice/silentwitness.htm.
[70] Utah State University Police Department, "Report a Crime," [Online document], [2006 March 23], Available at HTTP://www.usu.edu/usupd/reportcrime/.
[71] Saint Michael's College, "Saint Michael's Crime Prevention Silent Witness," [Online document], [2006 March 23], Available at HTTP://www2.smcvt.edu/security/prevention/silent_witness.htm.
[72] Whitman College, "Hate Crime/Incident Report Form," [Online document], [2006 March 23], Available at HTTPS://www.whitman.edu/intercultural_center/aah/ReportForm.htm.
[73] Marshall University Police Department, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.marshall.edu/mupd/SILENT_WITNESS.HTML.
[74] St. Norbert College Campus Safety, "Silent Witness," [Online document], [2006 March 23], Available at HTTP://www.snc.edu/security/silentwitness.htm.
[75] Western Wyoming Community College, "Silent Witness Program," [Online document], [2006 March 23], Available at HTTP://www.wwcc.wy.edu/pplant/silent_witness.htm.
[76] City of Albuquerque Independent Review Office of the Police Oversight Commission, "On-line Police Complaint," [Online document], [2006 March 23], Available at HTTP://www.cabq.gov/iro/complaints.html.

[77] Albuquerque Police Department, "Report Abandoned Vehicle," [Online document], [2006 March 23], Available at HTTP://www.cabq.gov/police/abandonedvehicleform.html.
[78] Vancouver Police Department, "Welcome to Online Citizen Reporting," [Online document], [2006 March 23], Available at HTTPS://vancouver.ca/citizenreport_wa/.
[79] Salinas Police Department, "Online Crime Reporting," [Online document], [2006 March 23], Available at HTTP://www.salinaspd.com/Forms/Report.html.
[80] City of Wichita, "Police Department WPD Main," [Online document], [2006 March 23], Available at HTTP://www.wichita.gov/CityOffices/Police/.
[81] City of Wichita, "Motor Vehicle Accident On-line Report," [Online document], [2006 March 23], Available at HTTPS://www.wichita.gov/CityOfWichita/Templates/Form.aspx?NRMODE=Published&NRORIGINALURL=%2fCityOffices%2fPolice%2fForms%2fAccidentReport%2ehtm&NRNODEGUID=%7bFDABF1E6-1714-454D-96C5-A8D9A0AAA3F4%7d&NRCACHEHINT=Guest.
[82] Charlotte-Mecklenberg Police Department, "CMPD Online Crime Reporting System," [Online document], [2006 March 23], Available at HTTP://ww.charmeck.org/online_reporting/.
[83] Spokane Crime Reporting Center, "Spokane Crime Reporting Center Online Reports," [Online document], [2006 March 23], Available at HTTP://www.spokanecounty.org/crimereportingcenter/report.asp.
[84] City of Fresno Police Department, "Crime Reporting Page," [Online document], [2006 March 23], Available at HTTP://www.ci.fresno.ca.us/fpd/crime_reporting.asp.
[85] Anchorage Police Department, "Online Reporting System," [Online document], [2006 March 23], Available at HTTP://www.muni.org/apd1/ereportstart.cfm.
[86] City of Palo Alto Police Department, "Crime/Incident Reporting," [Online document], [2006 March 23], Available at HTTP://www.papd.org/onlinereporting/crimeincidentform.html.
[87] City of Stockton, CA – Police Department, "File a non-emergency Police Report online," [Online document], [2006 March 25], Available at HTTP://www.stocktongov.com/policereport/.

[88] Tulsa Police Department, "Online Citizens Crime Report," [Online document], [2006 March 25], Available at HTTP://www.tulsapolice.org/records/online_report.html.
[89] Menlo Park Police, "Online Crime Report Introduction," [Online document], [2006 March 25], Available at HTTP://www.menloparkpolice.org/online_intro.html.
[90] Sacramento Police Department, "Online Crime Report Instructions," [Online document], [2006 March 25], Available at HTTP://www.sacpd.org/crm_rpt_instructions.asp.
[91] Sacramento Police Department, "Crime Report Form," [Online document], [2006 March 25], Available at HTTP://www.sacpd.org/crime_rpt.asp.
[92] City of San Rafael, "On-line Crime Report," [Online document], [2006 March 25], Available at HTTP://www.cityofsanrafael.org/Government/Police_Department_Home_Page/On-line_Crime_Report.htm.
[93] City of Joplin Police Department, "Online Reporting," [Online document], [2006 March 25], Available at HTTP://www.joplinpolice.org/onlinereporting.cfm.
[94] Peoria Police Department, "Online Crime Reporting Instructions," [Online document], [2006 March 25], Available at HTTP://www.peoriaaz.com/PoliceDept/OnlineReporting/online_report_main.htm.
[95] Peoria Police Department, "Online Crime Reporting Form," [Online document], [2006 March 25], Available at HTTP://www.peoriaaz.com/PoliceDept/OnlineReporting/online_report.asp.
[96] Napa Police Department, "Online Crime Reports," [Online document], [2006 March 25], Available at HTTP://www.cityofnapa.org/departments/police/crimeform2_v2.asp.
[97] Porterville Police Department, "Online Crime Report," [Online document], [2006 March 25], Available at HTTP://www.ci.porterville.ca.us/forms/crimereport.cfm.
[98] City of Lakewood, Colorado, "Welcome to the Lakewood Police Department Online Crime Reporting System," [Online document], [2006 March 25], Available at HTTP://www.lakewood.org/index.cfm?&include=/PD/OnlineForms/OnlineTheftForm_redesign.cfm.
[99] Layfayette Police Department, "Online Crime Reporting," [Online document], [2006 March 25], Available at HTTP://lafayettepolice.us/programs/crimereport/.

[100] Tucson Police Department, "TPD Online Crime Reporting," [Online document], [2006 March 25], Available at HTTP://www.ci.tucson.az.us/police/online/index.html.
[101] Charlottesville Police, "Report Incident Instruction Page," [Online document], [2006 March 25], Available at HTTP://www.charlottesville.org/default.asp?pageid=391C21F6-D6F3-4CE4-8ADA-BBF88FBF6261.
[102] Charlottesville Police, "Online Incident Report Form," [Online document], [2006 March 25], Available at HTTP://www.charlottesville.org/default.asp?pageid=A3FEFFE1-79B2-44FD-BE2A-7E006FF2B505.
[103] Town of Windsor Police Department, "Online Crime Reporting Form," [Online document], [2006 March 25], Available at HTTPS://ci.windsor.co.us/forms/frmwpdcrimereport.html.
[104] Philadelphia Police Department, "Submit Reports," [Online document], [2006 March 25], Available at HTTP://www.ppdonline.org/rpts/.
[105] City of Turlock Police Services, "Online Crime Report," [Online document], [2006 March 25], Available at HTTP://www.turlock.ca.us/citydepartments/policeservices/.
[106] City of Turlock Police Services, "Welcome to Online Reporting," [Online document], [2006 March 25], Available at HTTPS://ci.turlock.ca.us/citydepartments/policeservices/crimereport.asp.
[107] Denver Police Department, "Online Police Reporting," [Online document], [2006 March 25], Available at HTTP://www.denvergov.org/Police/240forms835.asp.
[108] Webmaster, Personal Electronic Communication, webmaster@ci.denver.co.us, March 28, 2006.
[109] Manteca Police, "Manteca Police Online Reporting," [Online document], [2006 March 25], Available at HTTP://www.ci.manteca.ca.us/police/reporting/.

[110] Hurst Police Department, "Crime Reporting Form," [Online document], [2006 March 25], Available at HTTP://www.ci.hurst.tx.us/Safety/Police/FormFiles/crime_form.htm.
[111] Fairfield Police Department, "Online Crime Reporting," [Online document], [2006 March 25], Available at HTTP://www.fairfieldnj.org/police/report.html.
[112] Lake Wales Police Department, "Lake Wales Police Department Online Reporting System," [Online document], [2006 March 25], Available at HTTP://lwpd.cityoflakewales.com/bins/site/templates/pages.asp?area_2=pages/onlinereporting.
[113] Redmond Police Department, "Online Reporting System," [Online document], [2006 March 25], Available at HTTP://rdmpd.smartz.com/Online_Report/.
[114] Chicago Crime Commission, "Anonymous Crime Reporting Hotline," [Online document], [2006 March 25], Available at HTTP://www.chicagocrimecommission.org/reportcrime.html.
[115] Virginia Beach Police Department, "Police Department - Send a message to Captain D.T. McGratten," [Online document], [2006 March 25], Available at HTTP://www.vbgov.com/dept/police/contacts/mailform.asp?name=mcgrattan.
[116] Wabash Valley Crime Stoppers, "Online Crime Reporting Completely Anonymously," [Online document], [2006 March 25], Available at [Online document], [2006 March 25], Available at HTTP://vigocountysheriff.com/crime_reports/cra.sh.
[117] Parke County Sheriff's Department, "Online Crime Reporting," [Online document], [2006 March 25], Available at HTTP://www.parkecountysheriff.com/crime_reports.htm.
[118] Mobile County Sheriff's Office, "Tip Line," [Online document], [2006 March 25], Available at HTTP://www.mobileso.com/MobileSO/Templates/MobileSOTipLine.aspx.
[119] Maricopa County Sheriff, "Techno Cops Sheriff Joe Arpaio's Search For Arrest Warrants," [Online document], [2006 March 25], Available at HTTP://www.mcso.org/TechnoCops/index.php, HTTP://www.mcso.org/TechnoCops/index.php?m=Info&id=145470#tip.

[120] Office of the Sheriff County of Monterey, "Submit Crime Tips Online – Anonymously or Otherwise," [Online document], [2006 March 25], Available at HTTP://www.co.monterey.ca.us/sheriff/rep_crime.htm.
[121] Eagle County Sheriff, Colorado, "Crime Stoppers," [Online document], [2006 March 25], Available at HTTP://www.eaglecounty.us/sheriff/crimestoppers.cfm.
[122] New Castle County, Delaware, "Please assist us by sending a TIP," [Online document], [2006 March 25], Available at HTTP://www.co.new-castle.de.us/police/nccpolice/ci/tipsadd.asp.
[123] TipSoft Secure Web Tip Submission Form, "Southwest Florida C.S.", [Online document], [2006 March 25], Available at HTTPS://www.tipsubmit.com/WebTips.aspx?AgencyID=104.
[124] Chatham County, Georgia, "Crimestoppers Tip," [Online document], [2006 March 25], Available at HTTP://www.savannahga.gov/cityweb/crimetips.nsf.
[125] St. Clair County Sheriff's Department, "File a Report," [Online document], [2006 March 25], Available at HTTP://www.sheriff.co.st-clair.il.us/onlcrime.asp.
[126] Fountain County Sheriff's Department Online Crime Reporting, "Crime Report Form," [Online document], [2006 March 25], Available at HTTP://www.fcsd.com/crime_reporting.htm.
[127] Sedgwick County Sheriff, "Felon Tip Sheet," [Online document], [2006 March 25], Available at HTTP://www.sedgwickcounty.org/Sheriff/felon_tip_sheet.htm.
[128] St. Charles Parish Sheriff's Office, Louisiana, "Report Criminal Activity," [Online document], [2006 March 25], Available at HTTP://www.stcharlessheriff.org/report.php.
[129] Anne Arundel County, Maryland, "Online Crime Reporting," [Online document], [2006 March 25], Available at HTTP://www.aacounty.org/Police/crimeReportForm.cfm.
[130] Monroe County Sheriff, "On-Line Crime Reporting," [Online document], [2006 March 25], Available at HTTP://www.co.monroe.mi.us/sheriff/crimereporting.html.
[131] Scott County Sheriff's Department, "Scott County Sheriff's Department Crime Tip Form," [Online document], [2006 March 25], Available at HTTP://www.showme.net/scottcountysheriff/crime_tips.html.
[132] Hall County, Nebraska, "Warrant/Fugitive Tip Form," [Online document], [2006 March 25], Available at HTTP://www.hallcountyne.gov/content.lasso?page=7353.

[133] Nye County Sheriff's Office, South Area Command, "ePolice Report," [Online document], [2006 March 25], Available at HTTPS://www.epolicereport.com/asp/pdweb.asp?pdkey=NV89060NCSO&it=2.
[134] Morris County Sheriff, "Morris County Sheriff's CrimeStoppers, Inc.", [Online document], [2006 March 25], Available at HTTP://www.morriscrimestoppers.org/tipform.htm.
[135] Cattaraugus County, NY Sheriff's Office, "Contact Us," [Online document], [2006 March 25], Available at HTTP://www.sheriff.cattco.org/.
[136] Catawba County Sheriff's Office, "Welcome to the Catawba County Sheriff's Office Tip Line," [Online document], [2006 March 25], Available at HTTP://www.catawbadetectives.com/giveatip.asp.
[137] Gallia County Sheriff's Office, "Tips," [Online document], [2006 March 25], Available at HTTP://www.galliasheriff.org/tips.htm.
[138] Marion County Sheriff Cyberwatch, "Cold Case Squad Tip Form," [Online document], [2006 March 25], Available at [Online document], [2006 March 25], Available at HTTP://sheriff.co.marion.or.us/ccs/ccsquad%20tip%20form.html.
[139] Cumberland County Crime Stoppers, "CCPA: Fraud and Abuse Tip Submission Form," [Online document], [2006 March 25], Available at HTTP://www.ccpa.net/cumberland/cwp/view.asp?a=1136&q=513474.
[140] Pickens County Crime Stoppers, "Tips," [Online document], [2006 March 25], Available at HTTP://www.pickenscosheriff.org/crimestoppers.htm.
[141] Harris County Constables Office, "Traffic Report," [Online document], [2006 March 25], Available at HTTPS://www.hcco4.com/traffic_report.html.
[142] Salt Lake County, Utah, "E-mail Info on Cold Cases or Missing Persons," [Online document], [2006 March 25], Available at HTTP://www.slsheriff.org/forms/mail_coldcaseinfo.cfm.
[143] Lamoille County Sheriff's Department, "Lamoille County Sheriff's Department Complaint Form," [Online document], [2006 March 25], Available at HTTP://www.lamoillesheriff.com/.

[144] Fairfax County, Virginia, "Crime Reporting – Citizen Contact – Fairfax County, Virginia," [Online document], [2006 March 25], Available at HTTPS://www.fairfaxcounty.gov/police/CRS/CRSDiscl.htm.
[145] Snohomish County, Washington, "Online Crime Report," [Online document], [2006 March 25], Available at HTTPS://web5.co.snohomish.wa.us/Sheriff/CrimeReport/.
[146] Oneida County Sheriff's Department, Wisconsin, "Tipline," [Online document], [2006 March 25], Available at HTTP://www.oneidasheriff.org/tipline.htm.
[147] Central Wyoming Crime Stoppers, "Tip Hotline," [Online document], [2006 March 25], Available at HTTP://www.crime-stoppers.com/tips.htm.
[148] RECOL: Reporting Economic Crime Online, "File a Complaint," [Online document], [2006 March 25], Available at HTTP://www.recol.ca/howtofile.aspx.
[149] Internet Watch Foundation, "Report Illegal Content," [Online document], [2006 March 25], Available at HTTP://www.iwf.org.uk/reporting.htm.
[150] Police UK, "Non-Emergency Minor Crime and Hate Crime / Incident Report," [Online document], [2006 March 25], Available at HTTP://www.online.police.uk/english/when_to_use.asp.
[151] Australian High Tech Crime Centre, "Report a High Tech Crime Online," [Online document], [2006 March 25], Available at HTTP://www.ahtcc.gov.au/ocrmain.aspx.
[152] Human Rights and Equal Opportunity Commission, "Lodging Your Complaint Online," [Online document], [2006 March 25], Available at HTTP://www.hreoc.gov.au/complaints_information/online_form/index.html.
[153] New South Wales Police, "Reporting Knowledge of Criminal Activity Online," [Online document], [2006 March 25], Available at HTTPS://secure.nsw.gov.au/.
[154] FBI – NW3C, "Internet Crime Complaint Center," [Online document], [2006 March 25], Available at HTTP://www.ic3.gov/.
[155] Cybersnitch – Cyber Enforcement Resources Incorporated, "Cybersnitch – Online Crime Reporting System," [Online document], [2006 March 25], Available at HTTP://www.cybersnitch.net/cybersnitch/cybersnitch.asp.
[156] United Stated Federal Bureau of Investigation, "FBI Tips and Public Leads," [Online document], [2006 March 25], Available at HTTPS://tips.fbi.gov/.

[157] America's Most Wanted, "Report a Tip," [Online document], [2006 March 25], Available at HTTP://www2.amw.com/report_tip/.
[158] Young, R. "Recommended Requirements Gathering Practices." *Crosstalk-The Journal of Defense Software Engineering*, [Online Serial] (2002 Apr.), Available at HTTP://www.stsc.hill.af.mil/crosstalk/2002/04/young.html.
[159] Marasco, J. "Software development productivity and project success rates: Are we attacking the right problem?" [Online document], [2006 Apr. 20], Available at HTTP://www-128.ibm.com/developerworks/rational/library/feb06/marasco/index.html.