EVALUATION OF COMMERCIAL OFF-THE-SHELF SOFTWARE: CRIME ANALYSIS SOFTWARE.

Omar .I. Lasassmeh[1],Wafa Tarawneh[2], Asma Nawaisah[3]
[1,2,3] IT, Department Mutah University, P.O. Box 7, Mutah 61710, Karak, Jordan
E-Mail:Omarjo@mutah.edu.jo

ABSTRACT : *Law enforcement agencies faced with the aims of building major Crime Analysis Software (CAS) from Commercial Off-The-Shelf (COTS) software products .Thus, they must study and examine the available (CAS) commercial products to determine their suitability for use in a particular system. The number of Commercial Off-The-Shelf (COTS) based crime analysis systems being built continues to increase. Consequently, the need for a model that ensures quality characteristics of such systems becomes a necessity. The Commercial Off-The-Shelf (COTS) approach changed the focus of software engineering from one traditional system specification and construction to one requiring simultaneous consideration of the system context (system characteristics such as requirements, cost, schedule, and operating and support environments).*
The general Commercial Off-The-Shelf (COTS) based crime analysis systems evaluation approaches are criticized for labour-intensive activities to define evaluation criteria. Further, none of the approaches are specifically targeted towards requirement tools. Therefore, the evaluation criteria, the measurement, and the process definition are time consuming and domain knowledge demanding.
Consequently, a new model specialized in evaluating COTS-components based crime analysis systems is needed to overcome the problems encountered with general quality models and current COTS selection methods.

Keywords: Crime Analysis Systems, Evaluation Methods, COTS, Framework, Social-Technical Approach.

1. INTRODUCTION

Crime analysis is a law enforcement tool that involves systematic analysis for recognizing and analyzing patterns and trends in crime and disorder. It provides a set of quantitative and qualitative techniques that are used to analyze data valuable to police agencies and their communities. Information on crime patterns can help law enforcement agencies deploy resources in a more effective manner, and assist detectives in identifying and apprehending suspects. Crime analysis also plays a role in devising solutions to crime problems, and formulating crime prevention strategies. Quantitative social science data analysis methods are part of the crime analysis process; though qualitative methods such as examining police report narratives also play a role[23].

Crime analysis software provides crime-prediction solution; however, crime analysis system automates the tasks associated with tactical and strategic crime analysis. The system automates the discovery of existing crime patterns and crime series in addition to providing predictive analytics regarding future crime occurrences.

large systems of readymade commercial off-the-shelf (COTS) software products have many distinguished characteristics that are clearly pointed by Dean, John et al(john ,2004).They argued that such products are sold, leased, or licensed to the general public, offered by a vendor trying to profit from it, supported and evolved by the vendor, available in multiple identical copies and used without modification of the internals. Moreover, it provides many benefits, including the potential of rapid delivery to end users, shared development costs with other customers, reusability of the final application due to the reuse of software components already tested and validated, and the opportunity to expand capacity and performance as improvement are made in the products. For systems that depend on COTS products, the evaluation and selection of appropriate products is essential to the success of the entire system. Yet many firms struggle during the evaluation and selection process.

Off-the- shelf software products is increasingly an "acquire and glue" process[9] . In 1997 an estimated 25.5 percent of typical corporation's software portfolio was commercial off-the-shelf software. Forecasts had that figure rising in 1998 to around 28.5 percent. These figures also supported[3].Boehm confirms that the usage of COTS products has increased significantly in building software systems during the last decade. The empirical results at the Center for Systems and Software Engineering (CSSE) reveal that the percentage of COTS-Based Application (CBA) in CSSE e-services projects increased from 28% in 1997 to 70% in 2002**.**

Building software systems using reusable components had introduces the discipline of component-based software engineering (CBSE). Such discipline has the obvious advantage of reducing the amount of software to be developed and so reducing cost and risks. However, requirements compromises are inevitable and this may lead to a system that does not meet the real needs of the users. Furthermore, some control over the system evolution is lost as new versions of the reusable components are not under the control of the organization using them.

The most common practice in CBSE is COTS. Over the past decade, the use of COTS products to implement significant portions of a software system has grown in both government and industry. The use of COTS products emphasizes buying commercial capabilities rather than building unique ones from scratch. Organizations that adopt a COTS-based system approach generally expect either more rapid or less costly system construction. These organizations also hope to stay in step with the technological advancements happening in the competitive marketplace. For example, NASA (National Aeronautics and Space Administration) successfully employed COTS products in reengineering the Hubble Space Telescope Command and Control system [14]. The COTS approach change the focus of software engineering from one traditional system specification and construction to one

requiring simultaneous consideration of the system context (system characteristics such as requirements, cost, schedule, operating and support environments).

2. Crime analysis software(CAS)

Crime analysis software is a performance management system that is used to reduce crime and achieve other police department goals. It emphasizes information-sharing, responsibility and accountability, and improving effectiveness. It includes four generally recognized core components: (1) Timely and spatial accurate information or intelligence; (2) Rapid deployment of resources; (3) Effective tactics; and (4) Relentless follow-up[24].

One of the most important functions of crime analysis software is crime mapping. The use of maps to project crimes data for spatial analysis purpose is crucial facility provided by such systems. Thus, law enforcement agencies looking for complete solution based software to analyze crime patterns. Geographic information system applications are widely used as main part of crime analysis based computer systems. This implies that the decision of selecting COTS based crime analysis system involves many criteria of different perspectives. For the purpose of selection COTS based crime analysis system; experts of GIS and spatial analysis are the stakeholder of the process.

Crime analysis systems are considered as one of the most effective application of artificial intelligence. The use of classification and clustering techniques are the key input of crime analysis process. The ability of crime prediction based on time and spatial features is the core of crime analysis, and then in crime investigation process. Figure 1 illustrates the crime analysis hierarchy.

Crime analysis software products just like any tools can be seen as a kind of commercial off-the-shelf (COTS) software product. First, tools are ready made products and potential users can select them from vendor product lists[13]. Second, tools are sold in many copies and users are neither controlling

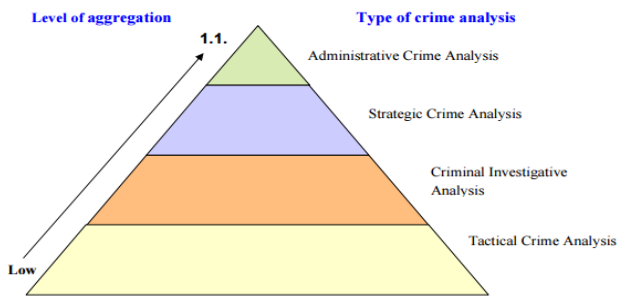

Fig 1crime analysis hierarchy

the tools specifications nor development processes[16].Finally, tool users do not get access to tool source code (except in case of open source code), and vendors are responsible for tool maintenance and improvement [2].

There two basic actors during the software procurement: evaluation team and tool users or customers. *Evaluation team* plans, organizes and executes the tool evaluation process, coordinates the evaluation actions and proposes a tool after evaluation result analysis. Tool *users* or customers have intention to acquire tool, evaluates the tool suitability, and after the tool selection, use the tool to support their work processes. The tool selection process typically consists of four phases [5,8,12]: 1) user requirements specification; 2) understanding of the available tools; 3) assessment of the tool compatibility with the requirements; 4) and selection of the "best" tool (Fig 2).

Requirements specification is based on the working domain knowledge and existing manual systems. *Understanding of the available tools* involves the experience. During the *assessment of the tool compatibility* the user has to assess the *package* depends on the compatibility with the requirements and the priorities of these requirements. The user may have to compromise on requirements not satisfied by any of the tools. Then the user reconsiders the requirements and iterates the selection or reorganizes his working practices in order to fit the best tool. The following subsections describe an existing COTS evaluation approaches in detail, and outline the disadvantages of each technique.

2.1 Criminal analysis software functions:

The b
asic crime analysis software functions are :

1. Maintains basic criminal Details:
2. information and the criminal profiles.
3. Biometrics manipulation: This shows if there is or isn't a biometric match (finger print, face recognition, iris recognition and …etc) in the local database.
4. Criminal profiling.
5. Crime mapping using spatial information.
6. Crime and criminal classifications
7. Tactical and strategic analysis.
8. Support decision makers
9. Crime prediction.

3. COTS-based Integrated System Development (CISD) Method

Tran and Liu propose a two-stage COTS selection process[16]. First stage is product identification, where candidates are identified and classified. The data for this stage is gathered via vendor documentation, personal experience or other means. The results are a list of potential candidates. The second is evaluation, where the final candidates are chosen (and unsuitable candidates eliminated). In this stage the authors depend on concrete techniques. They state that the COTS evaluation phase requires the extensive use of prototyping techniques. They argue that prototyping is the only way to practically evaluate a COTS candidate within the system context. They define three critical stages of the evaluation phase: *functionality*, *interoperability*, and *performance*. In the functionality stage candidates are tested in isolation to confirm that the functionality of the COTS product is applicable to the current application. In the interoperability stage, the candidates are evaluated to ensure their ability to co-exist with other components of the system, both COTS-based and custom developed. The performance evaluation stage consists of a quantitative analysis of the

effect of the COTS component on the overall performance of the system.

The final aspect of the methodology is a management evaluation that considers the less tangible aspects of integration the COTS product. These include such things as training, cost, vendor capability, etc. At the end of this process a final selection of COTS products is made. The authors also discuss different approaches to evaluation based on constraints such as development time and cost. This methodology depends on having a relatively complete predefined set of requirements since the product identification stage is dependent on COTS candidates meeting the requirements. The methodology in general is a waterfall-style process in that each stage depends on the results of its predecessor, and does not provide an analysis of the evaluation product using criteria hierarchy.

3.2. Off-The-Shelf-Option (OTSO)

Kontio presents a multi-phase approach to COTS selection, which begins during requirement solicitation, [3]. With his approach the decision to incorporate COTS into the system has been predetermined and thus the COTS method is only concerned with the actual selection process, not with implementation. The phases are the search phase, the screening and evaluation phase and the analysis phase. In the search phase, the goal is to identify potential candidates for further study. The screening phase selects the most promising candidates for detailed evaluation. In the analysis phase, the results of product evaluations are consolidated, and a decision about reuse is made.

The central theme to the OTSO method is the construction of a "product evaluation criteria hierarchy". This hierarchy serves as a template for situation specific criteria definition.

The disadvantages of this approach is that it can be very sensitive to bias or the experience of the personnel. The OTSO method does not address what method or model is used for COTS software reuse cost estimation. Whatever approach is used, the OTSO method extends the finical COTS software evaluation by allowing the consideration of other factors that may influence the decision. Example of such factors include the consideration of features that exceed the requirement specification, quality characteristics that are not included in the cost estimation model (e.g., reliability, maintainability, portability, efficiency), and business or strategic issues that may influence the decision. These issues can sometimes be decisive in COTS software selection and cost estimation alone cannot effectively cover these aspects.

3.1 Social-Technical Approach to COTS Evaluation Framework (STACE)

The Social Technical Approach to COTS Evaluation (STACE) framework[12] comprises four interrelated processes (Fig 3): 1) requirement elicitation; 2) social-technical criteria definition; 3) alternatives identification; and 4) evaluation. In the *elicitation*, tool requirements are discovered through consultation with users, from domain knowledge and market studies. In the *social-technical criteria* definition the elicited tool requirements are decomposed into hierarchical set, and each branch in this hierarchy ends in a measure or metric. *Alternative identification* includes searching for tool candidates. *Evaluation* involves ranking of identified tools against the social-technical criteria by examining capabilities, reading documentation, and experimentation.

Fig 4 illustrates the main COTS evaluation process in STACE. Step1 selects the underlying technology or other keystone issues. The selection process involves 1) define the evaluation criteria, 2) search and screen for available alternatives; 3) revising the criteria and requirements based Available technologies; 4) assessing and selection the best nology among alternative. Step 2 defines social-technical uation criteria for COTS products based on the selected nology. The criteria should include functionality issues nology and interface issues, quality characteristics and -technical issues. Step 3 searches and screens for lable COTS products. Search the marketplace to identify lidate COTS components through market surveys and r techniques like internet search. The search criteria at stage should be limited to functionality issues. Step 4 ses tool requirements and social-technical criteria based vailable COTS products. The selection of the best among packages available depends on the assessment of their patibility with the requirements specification and the rities of these requirements. Step 5 evaluates the lidate components and selects the best COTS product. evaluation should include ranking of the candidate lucts according to the evaluator's preferences.

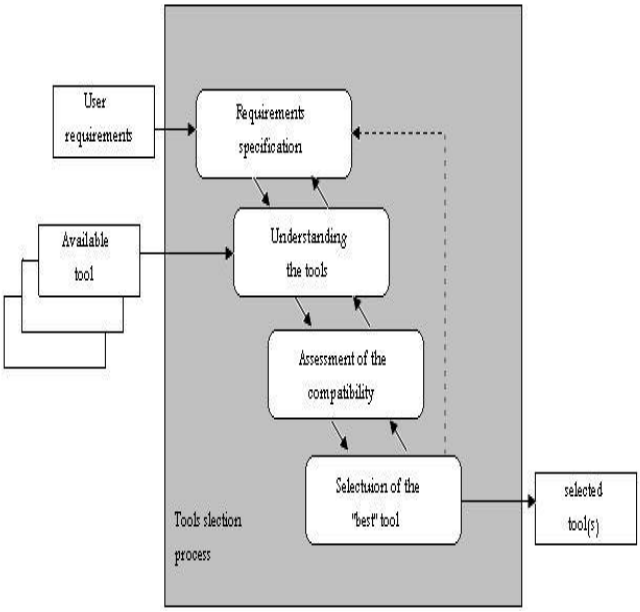

Fig 2 COTS Selection Process

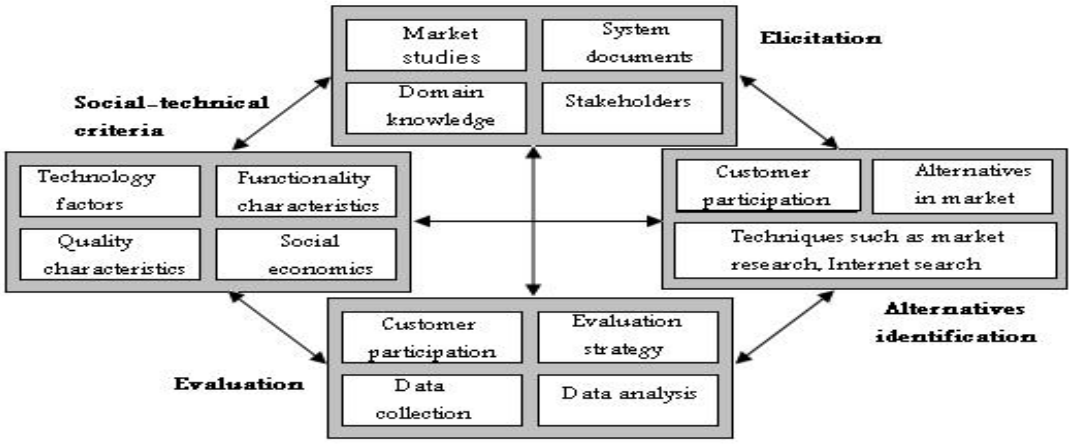

Fig 3 STACE Framework, (Kunda,1999)

An approach that emphasizes social and organization issues related to COTS selection. The main limitation of this approach is lack of a process of requirements gathering and specification. Moreover, the STACE does not provide an analysis of the evaluated products using a decision-making technique.

3.3 Procurement-Oriented Requirements Engineering (PORE)

The Procurement-Oriented Requirements Engineering (PORE) method integrates techniques for requirements acquisition and COTS selection with process guidance for choosing and using each technique,(Vijayalakshmi et al ,2008) .It is template-based and advocates a parallel and iterative requirements acquisition and tool-candidate selection/ rejection (Fig 5).

The PORE supports engineering team to acquire, describe and analyze customer requirements at the same time as acquiring, modeling and analyzing candidate COTS. The main steps for PORE are (Fig 6):

- acquire information about user requirements, COTS, suppliers, and procurement contract from customers;
- analyze information for completeness and correctness;
- *use decision-making* techniques to analyze and determine requirement compliance;
- *reject (select)* one or more *candidate* COTS that are non-compliant with customer requirements defined by the evaluation goals. The main limitation of this approach is lack of clear how requirements are specified in the evaluation process and how products are eliminated (do not capture the decision relational).

3.4 Scenario-based Selection

The scenario-based selection[6] proposes a comparison between *baseline scenarios* which describe how organization operates, and *tool scenarios*, which are baseline scenario projections into a future, where a tool is applied. Fig 7 depicts an analyst who rewrites the baseline scenarios adapting it to represent scenarios with tool 'A'. once the baseline scenario to determine: 1) what differences are between the baseline and tool scenarios; 2) what the impact of changes are; 3) what changes to propagate to other parts of the organizational processes; and 4) what the impacts of the propagated changes are. The main limitation of this approach does not analyze the tool coverage of the required functionality, the non-functional requirements, tool interoperability with the systems used in the organization, fiscal health of the tool vendor, or its ability to provide support of the tool

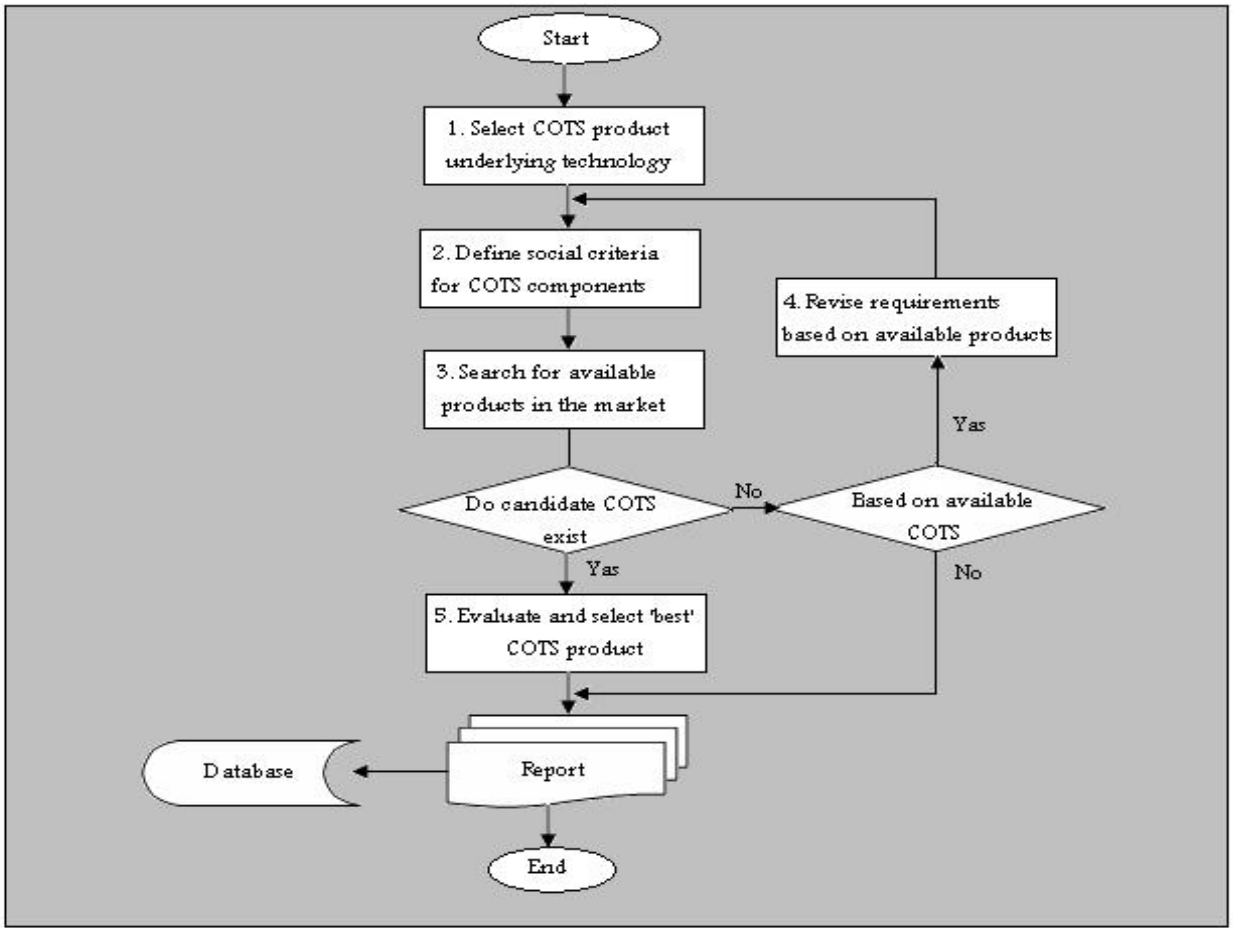

Fig 4 Flowcharting the STACE Evaluation Process (Kunda.1999)

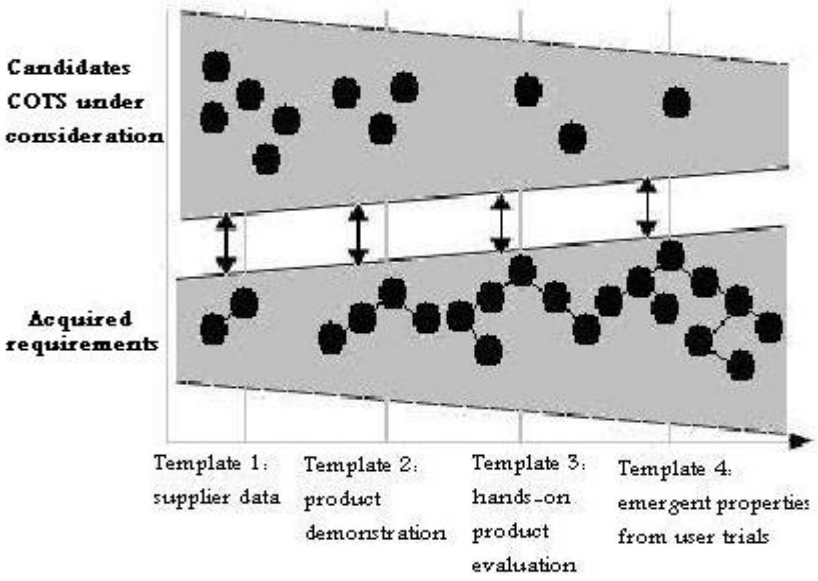

Fig 5 PORE Process for COTS Selection(Couts and Gerdes,2010)

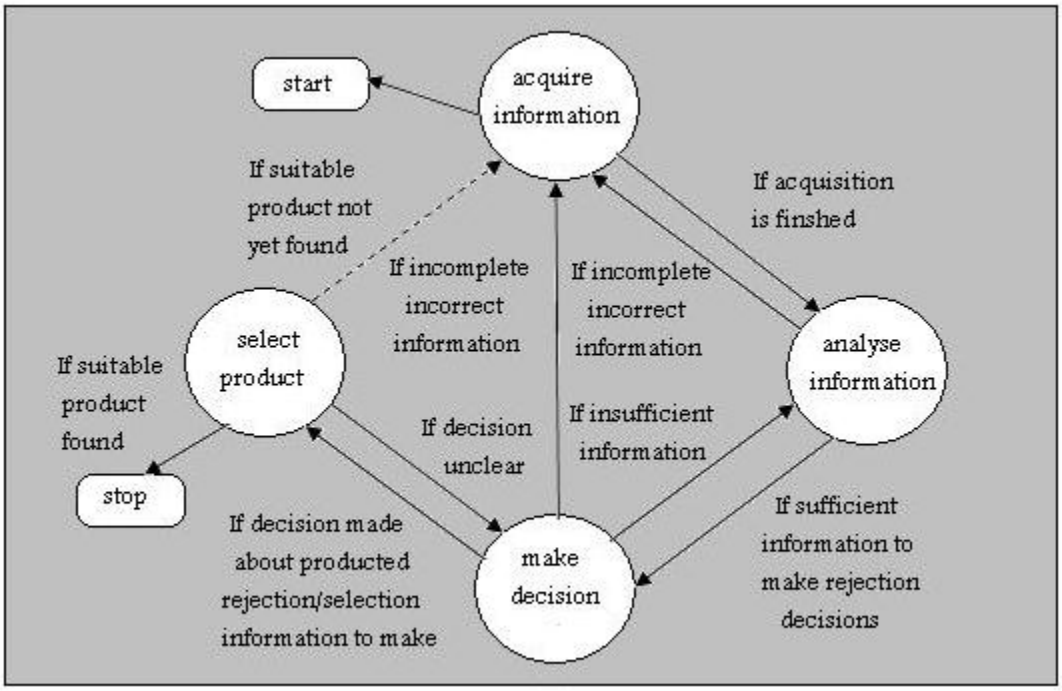

Fig 6 PORE Processes, (Achour & Ncube ,2000)

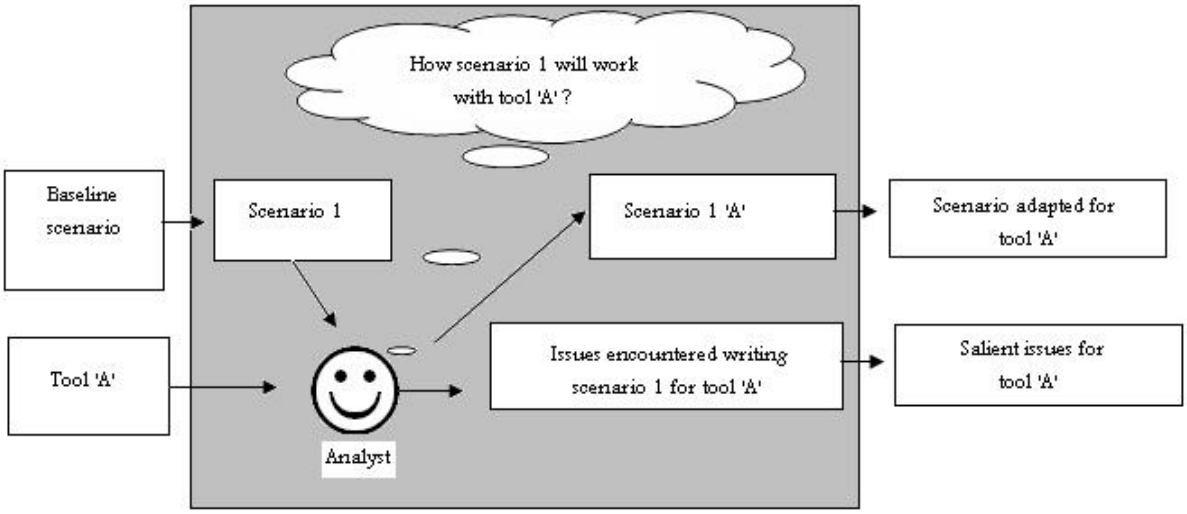

Fig 7 How Scenario Will Work Candidate 'A'(Feblowitz & Greenspan ,1998).

4. Comparison of Existing Approaches

The Crime Analysis Software (CAS) market is crowded with the development and emergence of new software. Looking for new and upgraded tools is demand, but it creates problems in the minds of the potential stakeholder to take right decision. Investment in (CAS) that is most appropriate for the law enforcement agencies can lead to improved management and customer service. Multiple criteria of selection and evaluation must be considered. Evaluation and selection of (CAS) systems that can meet the requirements of an organization is a very time consuming and difficult task. Thus, there is a need to study the various (CAS) tools available in the market so that the most appropriate tool can be selected for the organization as per the requirement.

All approaches correspond to the general COTS selection process in Fig 2. However each of them highlights different techniques. CISD describes two-stage COTS selection process, first stage is product identification, and second stage is evaluation. The PORE describes template-based tool procurement. The scenario-based selection maps baseline

scenario and tool scenario. OTSO, scenario-based selection and STACE assume that evaluation criteria are defined in advance. The approaches describe only general categories like functional, non-functional tool requirements, tool architecture, vendor and business requirements, which are standard, the category definition resemble to construction of an evaluation framework. The evaluation could be divided into two types: *one-round* and *multi-round* evaluation. In one-round the evaluation is executed one time taking in account all the candidate tools. After each evaluation step, the tool list is reduced and the tool requirements are expanded (e.g. PORE). But most of the approaches (OTSO, scenario-based, CISD, and STACE) apply multi-round evaluation, when the full evaluation cycle is repeated with each individual tool. The decision about tool suitability is made after testing all tools.

5. Addressing the Problems

The OTSO method proposed by Kontio has the disadvantage of being very sensitive to bias or the experience of the personnel. In addition, it does not address what model can be used for COTS software reuse to estimate cost. The STACE method proposed by Kunda emphasizes only social and organizational issues related to COTS selection. The main limitation of this approach is the lack of a process for requirements gathering and specification. Moreover, it does not provide an analysis of the evaluated products using a decision-making technique. With regard to PORE method by Ncube, it is not clear how requirements are specified in the evaluation process and how products are eliminated. Finally, Tran and Liu method does not provide an analysis of the evaluation product using criteria hierarchy.

The general COTS evaluation approaches are criticized for labour-intensive activities to define evaluation criteria. Further, none of the approaches are specifically targeted towards requirement tools. Therefore, the evaluation criteria, the measurement, and the process definition are time consuming and domain knowledge demanding. Consequently, a new model specialized in evaluating COTS-components is needed to overcome the problems encountered with general quality models and current COTS selection methods.

6. The Proposed Approach

Steps that are used to build COTS evaluation methodology.

Step 1: Identify a small set of agreed-upon, high-level quality attributes, and then, in a top-down fashion decompose each attribute into a set of subordinate attributes.

Step 2: Distinguish between internal and external metrics. For COTS components, it is essential to observe such distinctions, specifically; the internal metrics measure the internal attribute of a product (e.g. specification or source code) during the design and coding phases, known as 'white box' metrics. Whereas external metrics specialize in the system behavior during testing and component operation, from an outsider view. In fact, external metrics, known as 'black-box', are more appropriate for COTS components.

Step 3: Identify Stakeholders (type of users) for each high-level quality attribute.

Step 4: Put the pieces together; constructing the new model that implements ideas from international standards: ISO-9126, and accordingly recognize appropriate Stakeholders for each set of attributes.

Step 5: Identify attributes for every sub-characteristic at product level.

Step 6: Identify attributes for every sub-characteristic at process level

Step 7: Finds a way to measures (weights) these attributes

Step 8: Draw a framework (shows COTS evaluation approach)

Step 9: Apply Data Analysis techniques: Our selection would be the Analytic Hierarchy Process (AHP), and it will be used to consolidate data evaluation in order to select the best COTS among several alternatives.

7. Executing the Methodology to Build COTS Evaluation Approach

The objective of creating our new approach is to build one suitable to work for a variety of COTS-based systems. The starting point for building our model is the ISO 9126, simply because it includes the common software attributes, as shown in Table 1.

The next step is to apply some tailoring on the ISO 9126 that harness COTS evaluation requirements. The six areas of importance for software evaluation, as proposed by ISO 9126 as a standard, are shown in Table 2.

The following is the evaluation discussion of the high-level set of characteristics, along with their associated sub-characteristics; the implementation of step 1 of our approaFunctionality is the capability of the software product to provide functions, which meet stated and implied needs when the software is used under specified conditions. Functionality is a set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs. Functionality is assessed by three things: (i) evaluate the set of features and capabilities of the program, (ii) the generality of functions that are delivered, and (iii) the security of the overall system. The sub-characteristic Compatibility has been added to our model. The purpose of Compatibility is to reflect the degree to which a component can be used and function correctly in different environments, and that is consistence with evaluating COTS components.

Reliability is the capability of the software product to maintain a specified level of performance when used under specified conditions. Reliability is the extents, to which a program can be expected to perform its intended function with required precision, usually evaluated by measuring the frequency and severity of failure, the accuracy of output result, the ability to recover from failure and the predictability of the program, because unreliable programs fail frequently, or produce incorrect data. Also, Reliability is a set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time. Reliability is the degree to which a work product operates without failure under given conditions during a given time period.

Table 1: Common software attributes for the ISO 9126

Software Attributes	Description
Testability	is the ease with which an application or component facilitates the creation and execution of successful tests
Efficiency	is the capability of the software product to provide appropriate performance, relative to the amount of resources used under stated conditions
Understandability	the capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of uses
Reliability	is the capability of the software product to maintain a specified level of performance when used under specified condition
Functionality	is the capability of the software product to provide functions, which meet stated and implied needs when the software is used under specified condition
Integrity	is the assurance that information can only be accessed or modified by those authorized to do so
Interoperability	is the capability of the software product to interact with one or more specified system
Mainability	is the capability of the software product to be modified
Portability	is the capability of the software product to be transferred from one environment to another
Usability	Is the capability of the software product to be understood, learned, used and attractive to the user when used under specified conditions

The Maturity sub-characteristic is measured in terms of the number of commercial versions and the time interval between them. The Recoverability sub-characteristic is a capability of the software product to re-establish a specified level of performance and recover the data directly affected in the case of failure. Therefore, it tries to measure whether the component is able to recover from unexpected failures, and how it implements these recovery mechanisms. The Fault Tolerance tries to measure the capability of the software product to maintain a specified level of performance in cases of software faults. The COTS-based system that supports Recoverability feature is in a subsequent stage of passing a Fault Tolerance stage, thus Recoverability implies Fault Tolerance and not vice versa. For this reason Fault Tolerance is omitted from our approach, while the emphasis remains on Recoverability.

Usability is the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions. Usability is related to the set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users. In addition, Usability is the effort required to learn, operate, prepare input, and interpret output of a program. In COTS, most stakeholders of components are the application developers, designers that have to build applications with them, and end-users who interact with COTS. Thus, the Usability of a component should be interpreted as its ability to be used by the application developer and designer when constructing a new software product. Under this characteristic we must add an attribute that measures the component's Usability during the design of application. Therefore, Complexity is a new sub-characteristic that is added to provide a measure of the components complexity when integrating and using it within a software product or system. This characteristic aims to measure the complexity of using and integrating the component into the final system.

Table 2: Characteristics and sub characteristics for the ISO 9126

Characteristics	Sub-characteristics
Functionality	Suitability, Accuracy, Interoperability, Compliance, Security
Reliability	Maturity, Recoverability, Fault tolerance
Usability	Learnability, Understandability, Operability
Efficiency	Time behavior, Resource behavior
Maintainability	Stability, Analyzability, Changeability, Testability
Portability	Installability Conformance, Replaceability, Adaptability,

Efficiency is the capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions. Efficiency is the degree to which something effectively uses (i.e., minimizes its consumption of) its resources. These may include all types of resources such as computing (hardware, software, and network), machinery, facilities, and personnel. In fact, Efficiency will used in our new model as it is described in the ISO.

Maintainability is the capability of the software product to be modified. Modifications may include corrections, improvements or adaptations of the software to change in an environment, and in requirements and functional specifications. Also, the effort required to locate and to fix an error in an operational program. Maintainability is the ease

with which an application or component can be maintained between major releases. Also, a set of attributes that bear on the effort needed to make specified modifications, the degree of changing or modifying the components to correct errors, to improve performance, or to adapt for changing the environment. The user of a component (i.e. the developer) does not need to do the internal modifications but he does need to adapt it, re-configure it, and perform the testing of the component before it can be included in the final product. Thus, Stability and Analyzability are omitted from our approach.

Portability is the capability of the software product to be transferred from one environment to another. Also, the effort required transferring a program from one hardware configuration and/or software system environment to another. Portability is the ease with which an application or component can be moved from one environment to another. In COTS, Portability is an important property in the nature of components, which are in principle designed and developed to be re-used in different environments (it is important to note that in COTS, re-use means not only to use more than once, but also to use in different environments). Thus, Portability is omitted from our approach.

Manageability; in order to empower our model with new a feature, the characteristic Manageability has been added. Manageability is concerned with developing and refining estimates of effort and deadlines for the project as a whole, and with gathering any data that might be needed for such estimates. The sub-characteristics Quality Management has been added to the model , which indicates the people within the law enforcement agencies , who are constantly monitoring what they do to find ways to improve quality of operation, product, budgets, schedule, services, and everything else about the firm. Table 3 shows the characteristics and sub characteristics we propose for selecting COTS components.

The next step in the proposed methodology, distinction between internal and external metrics, is already described and reasoning led us to consider the external metrics 'black-box' as more appropriate for COTS components.

Stakeholders: the term stakeholder is used to refer to any person or group who will be affected by the system, directly or indirectly. Stakeholders include the end-user who interacts with the system and everyone else in an organization that may be affected by its installation. Other system stakeholders may be engineers who are developing or maintaining a related system, and business managers. From our experience, end-users are concerned with observable attributes (such as Functionality, Reliability, Availability, and Efficiency). BO (Business Owner) is concerned with Maintainability, while system administrators are concerned with Scalability, Portability, and Manageability.

In this work, a typical set of stakeholders has been adopted in order to name the appropriate category of evaluators for each quality characteristic. Starting with the analyst who produces the business model, the end-user who interacts with the system, QA (Quality Assurance) officer who tests the product, and the PM (Project Manager) who constructs and manages the process.

Table 3: Characteristics and subcharacteristics for COTS Components

Characteristics	Sub-characteristics (Product)	Sub-characteristics (Process)
Functionality	Accuracy, Security	Suitability, Interoperability, Compliance, Compatibility
Reliability	Recoverability	Maturity
Usability	{Non Applicable }	Learnability, Understandability,
Efficiency	Time behavior, Resource behavior	{Non Applicable }
Maintainability	{Non Applicable }	Changeability, Testability
Manageability	Quality management	Quality Management

- the solution verifiability satisfies the requirement, both functional and non-functional, this should be verifiable by the analysts and the QA (Quality Assurance) professionals
- The developers can build the solution. This implies partitioning the solution into comprehensible pieces, with clear interface and definitions, and explicit mapping of dependencies among pieces.
- The product can be tested by QA (Quality Assurance). This relies on the mentioned partitioning (to plan unit testing) and traceability (to verify deployed functionality and properties).
- The process can be managed by PM (Project Manager). This relies on partitioning (to determine work units for teams and individuals) and on dependencies (to schedule work); thus, the project manager must be able to determine "intermediate deliverables" that are usable, testable and allow to show working progress.

The domain of the above classification of stakeholders can be re-organized as follows; which implements the third step of our methodology:

* The solution must offer the Functionality, observable attributes (Reliability, Usability, and Efficiency) specified requirements according to end-users, verified by analysts and QA (Quality Assurance).
* The solution must be maintainable according to the future PM (Project Manager), verified by the BO (Business Owner).
* The construction process must be manageable according to the project manager.

Table 4 shows the components that constitute our new model. Consequently, I have adapted to our model the common characteristics that are found and agreed upon by the majority of the existing models, and these are consistent with COTS component evaluation criteria. However, some of the characteristics that are inconsistent with the new model requirements. New characteristics are added, and these are necessary to empower our new model. Accordingly any modification step, including removal or additions has been justified above.

Table 4 : Tabular illustration of the new model components

Stakeholders (Professional Parties)	Characteristics	Product Sub-characteristics	Process sub-characteristics
End user, analysts, quality assurance	Functionality	Accuracy, Security	Suitability, Interoperability, Compliance, Compatibility
End user, analysts, quality assurance	Reliability	Recoverability	Maturity
End user, analysts, quality assurance	Usability	{Non Applicable}	Learnability, operability understandability, complexity
End user, analysts, quality assurance	Efficiency	Time behavior, Resource behavior	{Non Applicable}
Project manager or business owner	Maintainability	{Non Applicable}	Changeability, Testability
Project manager	Manageability	Quality management	Quality management

Next, a new set of sub-characteristics has been defined and associated with each high-level characteristic that is supported by the new model, this was done by breaking down the characteristics into two categories; one set supports the development process (the process) and the second one supports the operational state on the production area (the product).

Finally, stakeholders, the members of the team responsible for developing, maintaining, interacting with and/or using the information system have been categorized then matched accordingly with the appropriate characteristics throughout the entire system development life-cycle, including operational and maintenance phases, A major advantage of the new model is the addition of stakeholders, the members of the team responsible for developing, maintaining, interacting and/or using the COTS-based system. End-users, analysts, QA, PM and BO categories are matched with appropriate characteristics that each one is concerned about .As Figure 8 shows, the pieces are put together constructing the new model. Although our proposed model features specialization and improvement over existing models, it lacks the ability to measure the internal quality characteristics. This can be accomplished in future research work by applying one of the evaluation techniques such as AHP (Analysis Hierarchy Process).

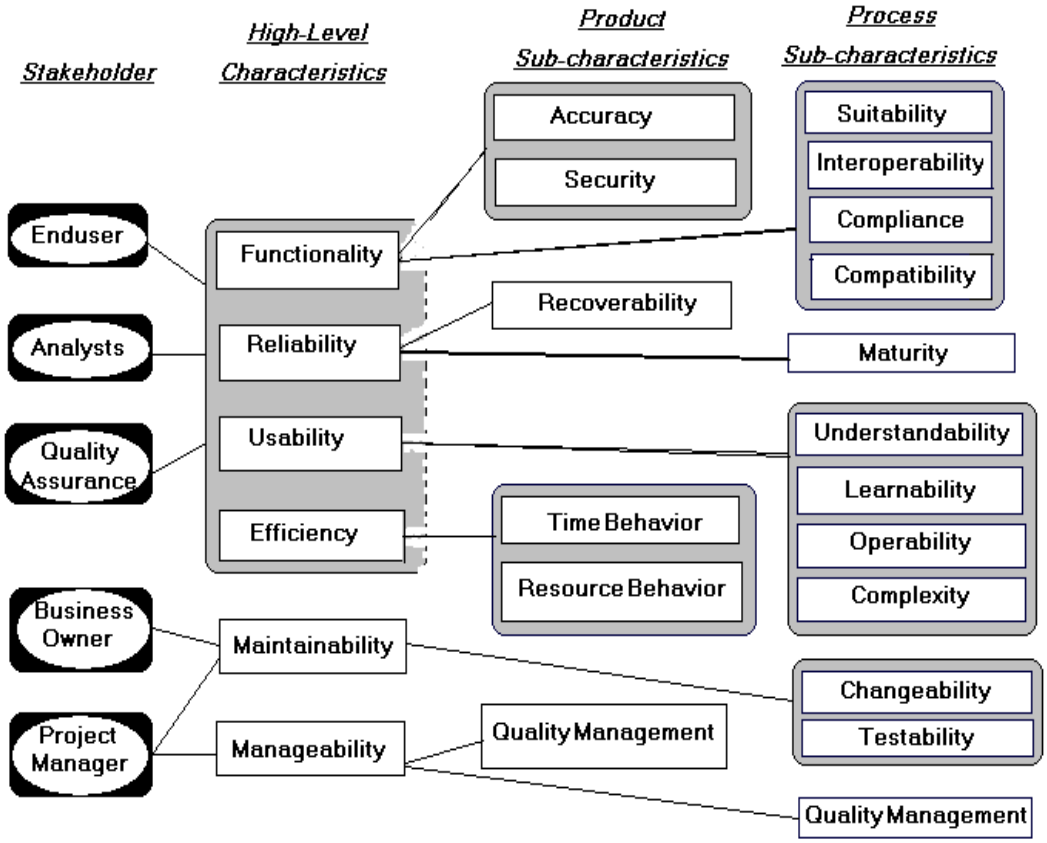

Figure 8: The new quality model for COTS-based systems

6. CONCLUSION AND FUTURE WORK

Criminal Analysis Software (CAS) package if wrongly chosen would lead to loss of money and time. This makes a priority for law enforcement agencies to be aware of best suited CAS software package in order to extract benefits as possible from CAS and able to improve. The generic software

evaluation criteria proposed above can be used effectively in evaluating (CAS) system package.

The vision of component-based software engineering means development of software systems by composing reusable components. COTS-based Crime Analysis Software (CAS) is live example of such vision. Using COTS based CAS products in large system provides many benefits, including the potential of rapid delivery to end user, shared development costs with other customers, reusability of the final application due to the reuse of software components already tested and validated, and the opportunity to expand capacity and performance as improvement are made in the products. However, for system that depends on COTS products, the evaluation and selection appropriate is essential to the success of the entire system.

Further, none of the approaches are specifically targeted towards requirement tools. Therefore, the evaluation criteria, the measurement, and the process definition are time consuming and domain knowledge demanding. I defined an eight-step methodology to guide the process of building the new model that is specialized in evaluating COTS components. The analysis step assisted us to benefit from existing general quality models and simultaneously avoiding repetition of such limitations.

Subsequently, justified high-level characteristics have been projected and new set of sub-characteristics has been defined for each one. This is accomplished by breaking down the characteristics into two categories; 'the process' and 'the product'.

The distinction between internal and external metrics led us to realize the external metrics is more appropriate for COTS components.

A major advantage of the new model is the addition of stakeholders, the members of the team responsible for developing, maintaining, interacting and/or using the COTS-based system. End-users, analysts, QA, PM, and BO categories are matched with appropriate characteristics that each one is concerned about. Finally, the pieces are put together to construct the new model.

7. Future Work

The AHP uses pairwise comparison to setting the priorities to choose the best alternative based on trade off between several levels, the possible future work would be the integration of AHP with the Linear Programming (LP), and then apply such integrated method to our model for selecting the best Crime Analysis Software(CAS) product among several alternatives. To elaborate on this point, currently, AHP is applied to calculate the rating of alternatives; the resulting weights can be used as function-coefficients in LP.

LP can act as a complement to AHP. A possible outline of the suggested integrated method will consistent of the following steps:

Step 1: Define goals, levels, and sub-levels of the undertaken problem.

Step 2: Use AHP to calculate the weight of characteristics, sub-characteristics, attributes, and alternatives.

Step 3: Build linear model (equation)

Step 4: Perform the sensitive analysis (Sensitive analysis identifies if the weight of one criterion has changed what happens to the alternatives**)**

REFERENCES

[1] Achour C. B, & Ncube C. Engineering the PORE Method for COTS Selection and Implementation with the MAP Process Meta-Model. Proceeding of the Sixth International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ 2000). Stockholm, Sweden.2000.

[2] Basili V. R. & Boehm B. COTS-based System Top 10. IEEE Computer, 34(5) 91-93.2001.

[3] .Boehm, B and L. Huang, Value-Based Software Engineering: A Case Study, IEEE Computer, March 2003, pp. 21-29.

[4] K. Vijayalakshmi, N. Ramaraj, and R. Amuthakkannan, "Improvement of Component Selection Process Using Genetic Algorithm for Component-Based Software Development," International Journal of Information Systems and Change Management, vol. 3, pp. 63-80, 2008.

[5] Couts,C and Gerdes,P. "Integrating COTS Software: Lessons from a Large Healthcare Organization," IT Professional, vol. 12, pp. 50-58, 2010.

[6] Feblowitz M. D., & Greenspan S. J. (1998). Scenario-Based Analysis of COTS Acquisition Impacts. Requirements Engineering, 3, 182-201.

[7] Finkelstein A., & Ryan M. Software Package Requirements and Procurement. Proceedings of the 8th International Workshop Software Specification and Design. IEEE Comp. Soc. Press, USA, 141-145.

[8] Jeffrey, V., 1999. Certification: Reducing the hidden costs of poor quality. Reliable Software Technologies, IEEE Software, 0740-7459/99.1996.

[9] John, D., G. Lewis, E. Morris, P. Oberndorf and E. Harper, 2004. A Process for COTS Software Product Evaluation. Technical Report CMU/SEI-2003-TR-017, ESC-TR-2003-017.

[10] Jvrki ,K. A Case Study in Applying a Systematic Method for COTS Selection. In 18th International Conference on Software Engineering, pp201-209, 1996.

[11] Jyrki ,K , Gianluigi ,C and Victor, R.. Defining Factors., Goals and Criteria for Reusable Component Evaluation. In Proceedings of CASCON '96, pp17- 28, Nov 1996.

[12] Kunda, D and Brooks, L. Applying Social Technical Approach for Cots Selection. Proceeding of the 4th UKAIS Conference, University of York, April 1999 9 [online]. Available: http://www.cs.york.ac.uk/mis

[13] Oberndorf , T. COTS and Open System – an Overview [Online].1997 Available: http://www.sei.cmu.edu/str/descriptions/cots.html1#ndi

[14] Pfarr, T. and J.E. Reis, The integration of COTS/GOTS within NASA's HST command and control system. Proc. First Intl. Conf. COTS-Based Software System, Systems, Orlando, FL, Feb. 4-6,. In Lecture Notes in Computer Science 2255, Berlin: Springer-Verlag, .2002.pp: 209-221.

[15] Tran V, and Dar-Biau Liu. A Risk-Mitigating Model for the Development of Reliable and Maintainable Large-Scale Commercial-Off-The-Shelf Integrated Software

[16] Vigder M. R., & Dean J. (1997). An Architectural Approach to Building System from COTS Software

Components. Proceeding of the 1997 Center for Advanced Studies Conference (CASCON 97). Toronto.

[17] Dhananjay Gade(2013).The Evaluation of Software Quality ,University of Nebraska-Lincoln.

[18] Rawashdeh A and Matalkah B, A New Software Quality Model for Evaluating COTS Components, Journal of Computer Science, 2 (2006) 373-381.

[19] Minkiewicz, A. Six steps to a successful COTS Implementation, The Journal of Defense Software Engineering, 18(8),17-21(2005).

[20] Gnanasankaran, S. Natarajan, K. Alagarsamy and K. Iyakutti, Application of COTS Components: A Software Package for Crystallography, British Journal of Mathematics & Computer Science, 3(2):99-107,(2013).

[21] Boba, R. Crime Analysis and Crime Mapping. Sage Publications. 2005.pp. 5–6.

[22] Commonwealth . National Crime Information Center retrieved from http://www.mass.gov/eopss/law-enforce-andcj/cjis/national-crime-information-centerncic.html.(2013).

[23] Levine,N .CrimeStat: A Spatial Statistics Program for the Analysis of Crime Incident Locations (v 3.3). Ned Levine & Associates, Houston, TX, and the National Institute of Justice, Washington, DC. July.2010.

[24] Gorr, W.L., & Kurland, K.S. GIS Tutorial for Crime Analysis. Redlands, CA:ESRI Press.2012.