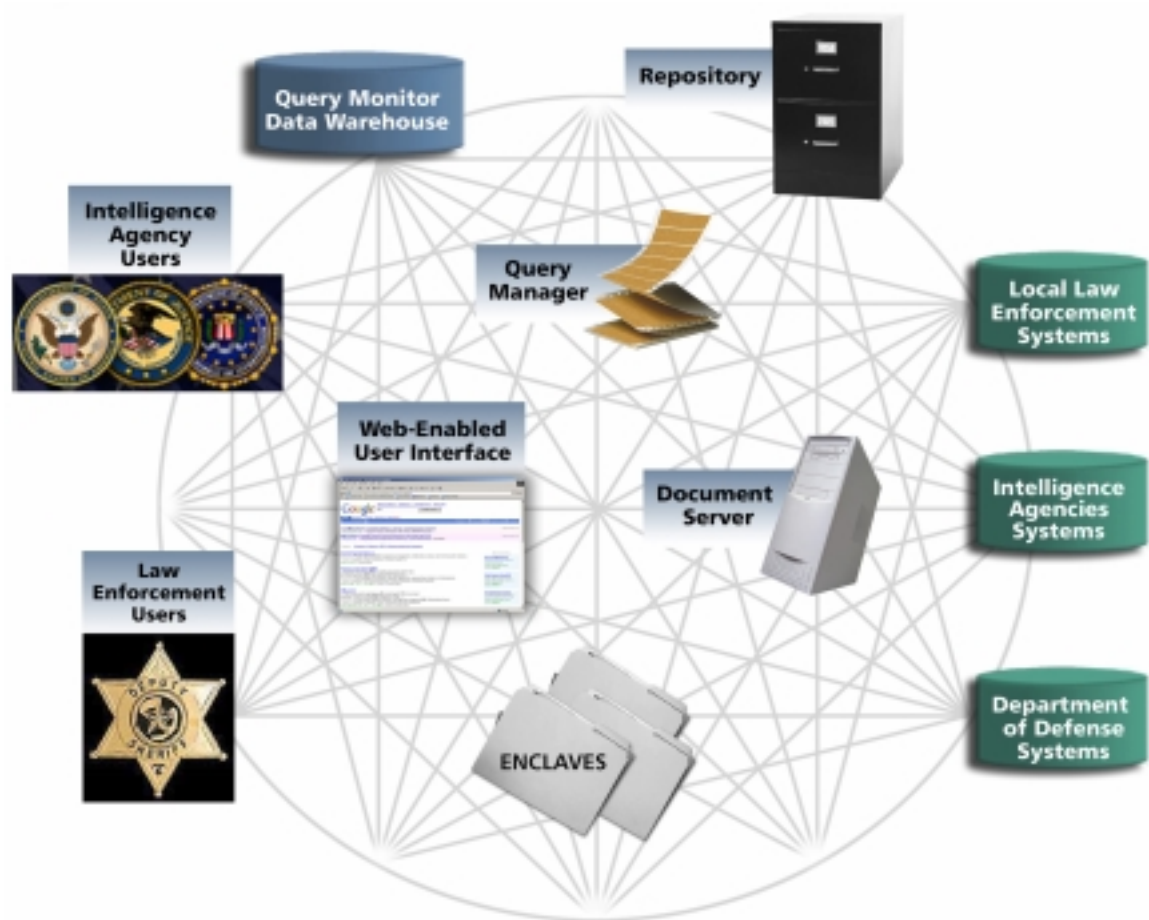


## Information Sharing for Homeland Defense

### High-level Functional Description of Architecture Proposed in October 2002

Kay Hammer  
Evolutionary Technologies International

The purpose of this document is to describe the high-level components called out in the following figure which was used in the October 2002 paper entitled “Information Sharing for Homeland Defense: Concerns and Suggestions.”



ETI believes that the above architecture could be built from proven and tested products, with a minimum of custom coding, and that this architecture could meet the criteria laid out in the “Risks and Concerns” paper.

This document calls out specific products—such as ETI Solution™ Version 5, including ETI's patented Dialogue Coach™ technology—as well as classes of products that could serve as the basis of such a solution. *(For ease of reference, we will call the proposed architecture “Homeland Defense Enterprise Architecture” – or HDEA.)*

*NOTE: The following may be a bit “ETI-centric” as that is where our expertise lies. Having additional vendors flesh out other aspects of the architecture will help better determine where the holes in functionality are, thereby establishing which requirements cannot be met by currently available commercial products. (Or, perhaps, this will identify requirements that need to be met by products not specifically mentioned.)*

### **Enabling Technology**

Web services, based on SOAP and WSDL, provide the core communication capabilities for HDEA, as they allow access to a rich user interface, providing a portal to the information and applications. However, HDEA would not use the UDDI structure provided by Web services on its own for two reasons:

- The UDDI concept was designed to allow discrete organizations a data-driven way of publishing instructions for accessing their applications. While this open architecture is well-suited to B2B applications, it does not address many of the requirements for providing access and authorizations in order to provide security.
- Likewise, the UDDI has no concept of such things as enforcing policy, etc. These matters are left to the internal applications of the organizations using Web services. Such an approach is unacceptable if the HDEA is to meet the goal of minimizing impact on the systems currently in use by various agencies and departments.

A secure mechanism for storing and validating the passwords assigned to and provided by the user. While each user of the HDEA needs an individual password, it may be that a single password could be used to access the source system since the system will be recording who asked for what at any particular time. There are a number of emerging standards that may provide a means to create and enforce security policies and levels of access.

An rdmbs database server capable of efficiently processing potential petabytes of data and supporting storage and interpretation of special data types, such as geospatial data. Teradata and DB2 are two commercially available databases with such capabilities, although Teradata is commercially more widely used in client-server architectures and might be preferable unless a mainframe is chosen for the HDEA architecture. The database server supports four types of databases:

- The Query Warehouse that is used to detect suspicious patterns of usage
- Any enclaves or caches of information which provide realtime access to such information as the names and identifying IDs of individuals or suspicious parties

- The Document Server which is used to manage and display query outputs
- Temporary databases used to support case management investigations.

The various descriptions below include additional capabilities required by the database server.

### ***Repository***

The repository serves as the traffic cop for the HDEA. The repository contains the following information:

1. Metadata descriptions of the information available from the various agencies and sources. These descriptions would be represented in terms of a generic metamodel that would be used by the GUI to give the various types of interfaces a consistent look and feel. These metadata descriptions must contain any system-specific information that must be known to the user in order to be able to obtain the desired information.
2. User access and profiles. The access would include a representation of which sources a user has access to and a representation of the types of queries he or she would be expected to ask. This information will be used by the Query Warehouse to determine potential misuse of the system. (See the description of the Query Warehouse below.) Passwords would not be stored in the repository, but elicited at log-on and passed to the secure system referenced in the enabling technology section.
3. A versionable history of the queries made by a particular user so the HDEA can provide proof that the information sharing policies enacted by Congress have been enforced.
4. Environment and system access information that tells the Query Manager how to map user requests to system access calls in order to obtain and return the desired information.

The ETI MetaStore™, which serves as the repository for the ETI data integration management solution, has an embedded object-oriented database and already contains much of the information outlined. The one extension required in order to handle the full range of information outlined above would be to enrich the representation of user. The current MetaStore contains definitions of user and access – information that is used by the ETI GUI in restricting what the user can see. However, it does not have the concept of user profile – in terms of who should be asking for what or at what frequencies. Yet, this capability might be critical in detecting when the security of the system has been breached. For example, suppose local sheriffs are allowed to query the system to see if a particular individual is “of interest” to the intelligence community or other law enforcement agencies in the event that they have apprehended a suspicious individual. If a particular sheriff issues thirty or forty such queries within a matter of days and the names all share a particular pattern, it could suggest that someone had obtained the sheriff’s ID and was using the system to determine how “safe” an individual was with respect to detection. One could argue that this information need only be stored in the Query Warehouse, as that is the application that uses the profile information; however, having a central repository that feeds this information to the various servers should

reduce the complexity of system administration and provide a single place to view a full information map of how the HDEA is being used.

### ***Graphical User Interface (GUI)***

The GUI serves as the user's portal to the HDEA, providing an intuitive way to allow users to accurately and efficiently access the various data sources, regardless of their physical locations and underlying technology. The GUI uses the information regarding the users' access rights to determine what they actually see on the screen. The GUI must include some means of letting the user specify complex search conditions, part of which may be specific to a particular database, and do so in such a way that the request is "correct by construction" in order to minimize unnecessary processing on the source systems and to provide timely results to the user. Once the user has issued the query, the GUI hands the query specifications to the Query Manager for processing.

The user interface for ETI Solution is data-driven and includes a patented Dialogue Coach for creating context-sensitive, menu-driven dialogues for leading users through specifying business rules and test and transformation logic. (For more information about how this component works, see Appendix A.) The ETI GUI was created for interface developers and may contain too much information about physical aspects of the data, such as data type, that would not be appropriate for the HDEA. However, the ETI GUI does allow users to dynamically create the definition of the target report or database, so aspects of it might be well-suited to the HDEA task at hand, e.g., in one case the user may be asking for a collection of documents and in another, want to create a case management database.

### ***Query Manager***

The Query Manager uses the environment and system access information stored in the repository to determine how to decompose the query and establish which actions are required in order to consolidate the information to create the desired report or database. Note: if the user is requesting that an "enclave" be created, he or she must be able to specify who has access to the enclave in addition to him/herself. The Query Manager maps this information both into the native access calls/code required to retrieve the desired information and to the representation of the query for the Query Warehouse to monitor their execution.

These above capabilities are what lie at the core of ETI Solution – its extensible code generation capabilities. It then stores a representation of the request in the repository so that a full metadata audit trail of everything requested is retained for audit and reporting purposes. However, the Executive, ETI's internal component that monitors the execution of the interface project or conversion, is really better suited to a development environment. There are mechanisms in the databases to handle and schedule local and highly distributed queries. There would not be any mechanism, including EAI products, to handle this kind of scheduling and workload management. Therefore, this is another area where a custom solution may be required.

### ***Query Warehouse***

The Query Warehouse is a data warehouse that it is mined to look for patterns of usages – either to refine the access protocols defined to the repository or to detect improper patterns of usage. The Query Warehouse is one of five types of security provided by HDEA. The others include password encryption, the user access-driven GUI, the security on the source system, and the document server. While data warehousing and data mining are well understood, the notion of user profile and the associated query protocols remain areas for research and design.

### ***Enclaves***

Enclaves is the term associated with creating a database to be used by a team for the purpose of case management. In these situations, the amount and variety of data requested is likely to be large. As a result, the data will probably be delivered at different times, and the HDEA will not want to wait until the entire query set has been fulfilled to make the data available to the investigators. Thus, one of the areas to investigate regarding the database servers is how easily they can be used to stage the building of a database, making it available in piecemeal fashion as the data becomes available. It is likely that a combination of triggers and alerts – available in many rdbms products – would be sufficient.

### ***Document server***

The Document server is a Web server application that controls access to the data returned from fulfilling a query set. The data is stored on the document server or in an enclave and – depending upon the rules associated with the data itself or how the enclave was set up – it may prevent downloading, printing, or copying the data in question. Likewise, it may destroy certain query results after they have been viewed. This is an area that requires further research as the author has limited experience with these products.

## Appendix A

### Dialogue Coach™ technology

A primary goal in the development of ETI Solution™ Version 5, including ETI's patented Dialogue Coach™ technology, was to enable non-technical users to specify complex data manipulations. To meet this challenge, ETI created a patented context-sensitive process that enables users to specify data transformations as English-language business rules. To specify a transformation, users select options one by one from menus; these menus are context-sensitive and change with each new selection. Transformations are always correct by construction and will execute with precision since only appropriate options are made available. For example, if a user selects a numeric function (e.g., “sum”), the user can then only select numeric fields to which to apply the function (e.g., “salary” or “bonus,” not “department.”)

Dialogue Coach is driven by a set of grammars specified through the use of the IDS “Grammar Editor.” These grammars include the ability to obtain user-specific information both from the vendor’s data stores and the user’s own local system to dynamically lead the user through a menu-driven dialogue that lets the user say what he wants while providing all the technical parameters needed by the back-end system. The grammars also contain “translation rules” that are used in conjunction with programming language templates to instantiate the functional logic.

Dialogue Coach interfaces offer the following benefits:

- This approach allows people without in-depth IT experience to perform sophisticated operations with little or no formal training.
- It allows the user to specify arbitrarily complex commands, which are dynamically table driven and “correct by construction.”
- The sequence of menu choices and value specifications creates a natural language description of what the user has specified.

ETI’s DSLs are all shipped with a broad range of filter grammars that perform common test and transformation logic. The “out of the box” capabilities supported include aggregation; alpha checks; assembler subroutine reformat; table lookup; database lookup; date reformatting; header/trailer processing; hierarchy inversion; simple calculation (ex: trunc, round); complex calculations; logical expression; string concatenate; unstring; substring; normalize; numeric checks; insert; update; delete; replace; multiple cursor processing; null processing; SQL functions (including inner and outer joins); Unit of Work control logic; virtual elements; constant value, bulk load and IF-THEN-ELSE logic.

The IDS also allows users to copy business rules from one object to another, where a wizard will substitute the appropriate field names – or warn the user that the rule needs to be modified. The natural language description of all business rules, search conditions, or transformation logic is stored with the conversion specification. This information is used in generating documentation and reports and can be exported to a customer’s repository of choice.

Customers regularly extend these grammars to support the specification of industry-specific computations such as return on a bond, maximum velocity, etc.

Components of Dialogue Coach include:

- The Grammar Editor, an interactive tool which aids in the specification and testing of the grammars used by ETI Solution Parser/Translator. In addition to error checking to insure that the grammar is well formed, the user can specify the attributes of lexical items and rules for how to translate each rule expansion.
- The Parser/Translator that:
  1. Accepts arbitrarily ambiguous grammars, thereby simplifying the task of grammar writing. To contrast, YACC, a popular UNIX shareware utility, accepts only grammars that are LALR1 – i.e., only tolerates ambiguity which can be resolved with one-token look-a-head.
  2. Is tightly coupled to a Token Recognizer that serves as the interface between the parser and the menu system. According to the instructions specified by the grammar writer, the Token Recognizer can deploy user functions, which query the parse state by means of ETI-provided functions before and/or after attempting to match the next. These user functions allow the grammar writer to do such things as the following:
    - Use context sensitivity to insure the correctness of each option presented as the next token. (For example, to insure that the second value specified in a range is greater than the first or to “turn off” an option.)
    - Validate any type-in value presented by the user.
    - Add a list of dynamic values to the menus.

These capabilities allow grammar writers to build a great deal of “intelligence” into the dialogues permitted in the interface. By contrast, there is no such dialogue between the parser and lexer generated by YACC and LEXX, another widely used UNIX shareware utility.

3. Provides three declarative, plus one procedural (through a function call) mechanisms for translating each expansion of each rule.
4. Can operate in both batch and interactive mode so that a string representing a statement consisting of menu choices can be presented to the Parser as a string for parsing such that all user functions apply while the menu-interaction is ignored.

The following figure illustrates the flow of control between these components.

