# IC.NET – Incident Command "Net"

## A System Using EDXL-DE for Intelligent Message Routing

Mr. Donald P. McGarry
The MITRE Corporation
Rome, NY 13441
dmcgarry@mitre.org

Dr. C.Y. Roger Chen
Syracuse University
Syracuse, NY 13244
crchen@syr.edu

***Abstract*—Traditionally, during a disaster response, primary reliance is on voice over radio communication along with pen and paper notes for situational awareness. This paper explores our research regarding emergency data interoperability, seeks to connect decision makers, operators/responders, and additional stakeholders through the development and application of standardized data messaging formats. In addition, we have investigated ways to route and expose emergency message data in ways that are complimentary to the current business processes of emergency response.**

**This paper will discuss the use and implementation of data interoperability standards in this system, focused primarily on the use of a single top-level loose coupler used for dynamic routing and exposure of operational level Emergency Services / First Responder. Our IC.NET prototype implements payloads that include data specific to Emergency Medical Services such as incident representation, unit tasking, and triage, treatment, and transport tracking of emergency patients.**

***Keywords- Emergency Preparedness and Response; DHS; FEMA; EDXL; OASIS; Loose Coupler; ICS; Incident Command System; NIMS; National Incident Management System; EMS; Emergency Medical Services; First Responder; Emergency Management; Disaster Management; EDXL; Emergency Data Exchange Language***

## I. INTRODUCTION

The Incident Command System (ICS) [1] as defined through the National Incident Management System [2] describes a "standardized on-scene emergency management construct specifically designed to provide for the adoption of an integrated organizational structure that reflects the complexity and demands of single or multiple incidents, without being hindered by jurisdictional boundaries" [3]. The ICS is designed such that it can be used "for all kinds of emergencies and is applicable to small as well as large and complex incidents" [3].

Through analysis of NIMS [2] and ICS training courses [3,4] as well as direct engagement with First Responders, we have determined that during medium to large scale disaster responses that involve mutual aid from neighboring counties, states, or regions, Incident command relies primarily on voice of radio communication along with pen and paper notes for situational awareness. This leads to cluttered radio chatter, necessitates the need for dedicated staff to transfer notes to a common operational picture, and can result in the unreliable transfer of information to and from the Incident Command Post.

The authors have developed a comprehensive data model for the First Responder Domain by engaging directly with First Responders and Incident Managers. During the first phase of our research we focused on providing a general data model for the ICS as well as a detailed data model for Emergency Medical Services (EMS). These data models will drive application of existing data standards such as the Emergency Data Exchange Language (EDXL) data standard and have also identified gaps in the existing standards model. Identified gaps were be filled with placeholder data structures called „Loose Couplers' and these findings are being transitioned to the Department of Homeland Security (DHS) and the Organization for the Advancement of Structured Information Standards (OASIS) for further development. Finally, this research is investigating ways through prototyping efforts to appropriately route and deliver emergency messages to their proper destination using an interface that can be modified by the user without significant IT expertise.

Driven from the data model and implementation of existing EDXL standards, IC.NET is a prototype of an Emergency Data Exchange Language Distribution Element (EDXL-DE) based messaging platform designed for the exchange of Emergency Data at the operational level. We have defined operational level data as the message data generated by all components contained within the ICS organizational structure. This includes command staff [3], general staff [3], task forces [3], and single resources [3]. This prototype leverages the encapsulation and routing capabilities of the Emergency Data Exchange Language's Distribution Element (EDXL-DE) [6] as the system's top-level loose coupler and allows for processing of both non-XML and XML content payloads including Common Alerting Protocol (CAP) [7], EDXL Hospital Availability Exchange (EDXL-HAVE) [8], EDXL Resource Messaging (EDXL-RM) [9], and two placeholder standards for Situation Reporting and Tracking of Emergency Patients. We refer to EDXL-DE as a loose coupler because EDXL-DE provides a data structure that provides the capability to package XML and non-XML payloads called content objects and provides a standard set of header information that systems can use to make delivery decisions on the information contained in the content objects without having to examine the data in the content objects directly. EDXL-DE is considered top-level because all interfaces and services within a messaging system can simply pass a single EDXL-DE message between them to perform data exchanges. For example, IC.NET routes and

exposes message data based on dynamically maintained lists of terminology, called ValueLists [6] that represent operational roles of users within the ICS command structure and recognized keywords that the routing system can use to make delivery decisions. This allows for message data to be exposed automatically to end users based on their specified role within ICS. The architecture of the IC.NET prototype is designed to provide bi-directional application of the EDXL data standards to allow end user applications to both visualize data and to preform Command and Control operations on field units. Through this prototyping effort we sought to provide a common messaging platform to experiment with connecting of emergency systems such as Computer Aided Dispatch (CAD), Mobile Data Terminals (MDT), incident command software, and field sensor data using the EDXL data standards.

This paper will discuss the use and implementation strategy of data interoperability standards in the IC.NET prototype, focused primarily on the application and use of EDXL-DE as a top-level loose coupler used for delivery and exposure of operational level Emergency Services / First Responder data. This includes data specific to Emergency Medical Services such as incident representation, unit tasking, and triage, treatment, and transport tracking of emergency patients.

## II. BACKGROUND

### A. User Requirements

Through direct engagement with First Responders, a number of user requirements were identified that dictated the feasibility of any proposed architecture or prototype. These included that any interoperable messaging solution should support use of local terminology for individuals, facilities, resources, and organizations. be lightweight, agile, simple to implement, and rapidly deployable in order to support the dynamic environment of practicing First Responders. Additionally it should be extendable to support unanticipated users, units, roles, or data; not relying on region-specific tools or systems in order to facilitate information sharing.

Another commonly identified operational level user requirement was to have incident management tools that were applicable to local and regional MCI's but were also integrated into their regular day-to-day operational systems. What was informally expressed to the authors by the operational level users can be illustrated through breaking down figure 1, in which the authors of [13] compared and contrasted relative complexity of an incident response versus predictability, situational awareness, and decision making ability.

Operational level First Responders and Emergency Managers expressed that although incidents at the bottom-left to middle region of the plot represented incidents that could / are relatively manageable compared to those incidents represented on the top-right region of the plot, that there was a significant degradation of services that occurred in the less complex type of incidents. These issues, related to communications needs, situational awareness, lack of a common operating picture, and ad-hoc information transmission to third and fourth responder scale directly to incident complexity. The need for integration into day-to-day systems along with the identification of information sharing issues that scale directly with incident complexity led the authors to an architectural requirement that any proposed system must integrate with existing systems and allow for scalability through federation with additional systems in the event of a large scale disaster.

### B. Domain Overview

#### 1) Defining the Emergency Management Domain

The principle functions of emergency management include identifying events that require response, tasking existing resources to events, tracking events to completion, determining additional resource needs, maintaining a chain of command, and disseminating information to field resources and other ICS

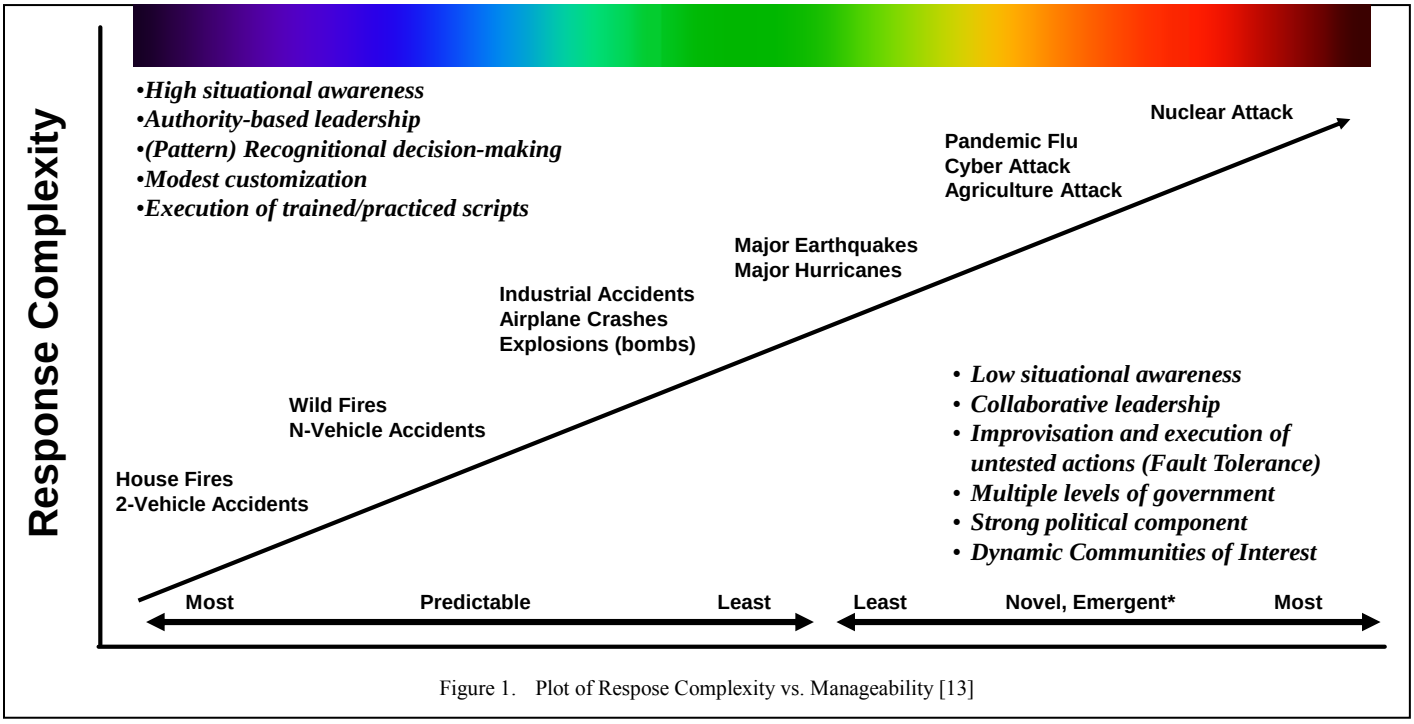


Figure 1. Plot of Respose Complexity vs. Manageability [13]

stakeholders [3]. In order to define the emergency response we have divided a large disaster response into four phases of response including: local or First Responders, regional or second responders, state or third responders, and federal or forth responders. After breaking down the phases of response we analyzed each phase with the following key considerations for each:

- Who are the participants at each level?
- What activities generate information to be shared?
- Where do standards or information gaps exist?
- What implications exist for a messaging architecture?

Our breakdown of the phases of an Incident Response of an emergency response include: Primary (Local responders), Secondary (Regional, Tertiary (State), and Fourth (Federal).

*2) The Incident Command System*

The Incident Command System as defined by FEMA [1,3,4] is a standardized on-scene emergency management construct derived to ensure the safety of responders and others, the achievement of tactical objectives, and the efficient use of resources. It is an integrated organizational structure to handle the complexities of incident(s) without the hindrance of jurisdictional boundaries [3]. The command structure represents a top down modular design dependent on the size, complexity and hazard environment caused by the incident that includes facilities, equipment, personnel, procedures and communications. The breadth of this modular command structure is represented in figure 2, which illustrates a small example of the modular structure of the ICS by providing a simple breakdown of the operations section into four branches.

One of the advantages of this standardized command structure with clearly defined roles is that the current state or context of the ICS structure can be used for message delivery. One of the main focuses of using the EDXL-DE standard was to provide message delivery for consumers of data that defined their role using the terminology definitions contained within the ICS. This strategy aims to provide the "right" data at the "right" level of detail to the "right" user at the "right" time. This focus is to use the ICS role-based context to allow end users to determine what information users will be interested in what information and at what level of detail and deliver it to them without the need for significant IT expertise. This creates an environment where end users could stand up a unified command post and start serving immediately in a time of need without the need for complex configuration. As the incident expanded and the ICS structure grows, the system can be continually updated based on the context of the ICS structure and user subscriptions to data feed will be updated appropriately in real-time. This allows for easy continuity of incident management, without the need for an entire IT department staff managing complex enterprise architecture in the incident command post.

## III. IMPLEMENTATION

### *A. Data Standards*

*1) EDXL-DE*

The EDXL-DE message was used as our top-level loose coupler in the IC.NET implementation. All content message data, referred to as content objects are carried within an EDXL-DE message. The EDXL-DE standard was developed by OASIS and "allows an organization to wrap separate but related pieces of emergency information into a single "package" for easier and more useful distribution" [15]. Although there are a number of data standards that implement the compositional design pattern [16] with a defined set of "header" information, EDXL-DE provides a few differentiating features which makes it ideal for emergency data. These features [6] include:

- Classification of payloads based on geographic area
- Data structures to use region specific terminology to make message delivery decisions
- Providing the capability for both explicit and implicit

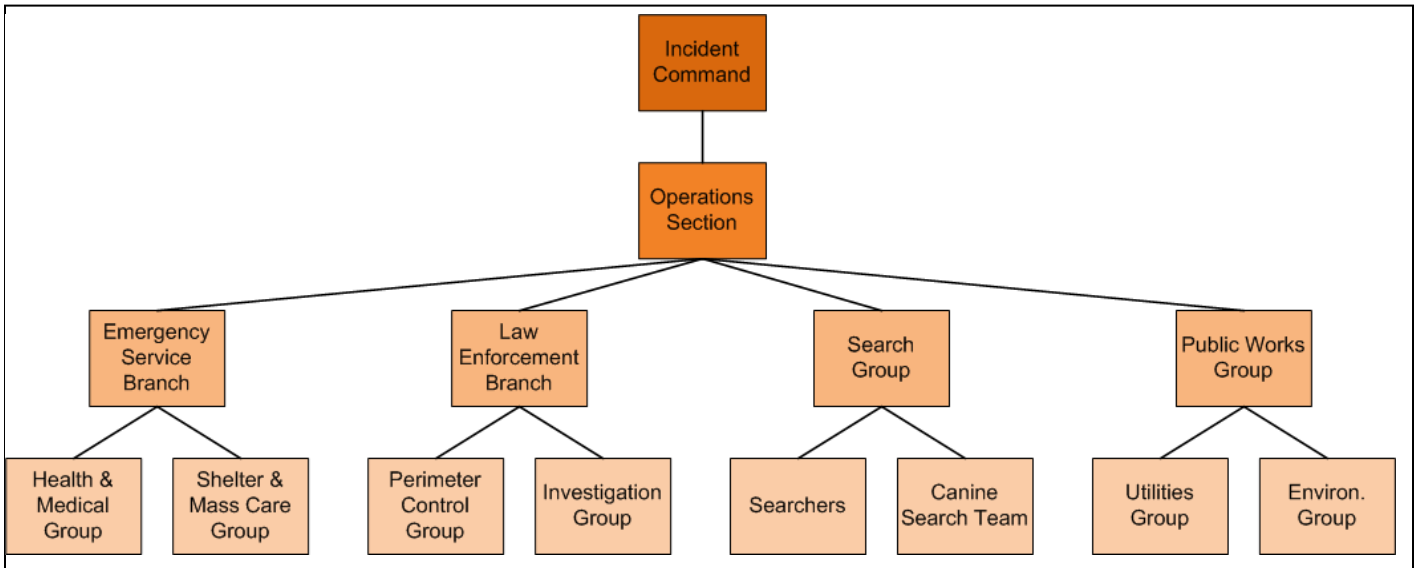


Figure 2. Breakdown of the ICS Operations Section

delivery of messages
- Encapsulation of both XML and non-XML payloads
- Encapsulation of multiple payloads within a single message

The details of routing, delivery, and exposure decision making facilitated by use of EDXL-DE will be discussed later in this paper.

*2) EDXL-CAP EDXL-HAVE and EDXL-RM*

CAP [7], EDXL-HAVE [8], and EDXL-RM [9] were implemented as XML payloads to be carried in an EDXL-DE message within the IC.NET system. CAP was used for its traditional purpose of public alert and warning messages as well as chemical and radiological sensor data from the Integrated CBRNE (iCBRNE) pilot program through SPAWAR Systems Pacific and DHS. EDXL-HAVE was used for its traditional purpose of carrying hospital status or availability information. Integration work was done with the Sahana Open Source Tool [17] with their response to the earthquake in Haiti to integrate their field EDXL-HAVE reports into IC.NET. A subset of EDXL-RM messages were used for unit status reporting similar to a simple automatic vehicle location, unit tasking, and unit status update reporting.

*3) Situation Reporting and Tracking of Emergency Patients*

Two key areas that were identified as gaps within the data model were incident / event representation and EMS-specific field reporting. Although there are already comprehensive standards for representing Pre-Hospital Care Reports [18] in their entirety, the ability to represent simple and unique patient data that followed the EMS workflow of triage, treatment, and transport, center around lightweight patient status updates with the ability to track a patient's status was missing. We are working closely with the OASIS Emergency Management Technical Committee (EM-TC) on their current work with EDXL Situational Representation (EDXL-SitRep) to incorporate gaps in the data model that exist for field units to report information about a current event to ICS. EDXL-SitRep is primarily focused on ICS reports to document ICS operations, however we feel in light of our findings within the data model a message set can be developed for field unit reporting that would allow auto-creation of EDXL-SitRep reports for ICS. We will also be transferring our findings on the EMS specific activities for the EDXL Tracking of Emergency Patients (EDXL-TEP) to the EM-TC once this effort is transferred to OASIS for implementation.

## B. *Using EDXL-DE for Delivery Decisions*

With a preliminary messaging data structures defined we investigated architectural concepts for a message delivery system. The architecture was focused around support for making automated delivery decisions based on the header information in the EDXL-DE messages and rules or policies that are applied within the architecture to deliver the messages to the appropriate clients. Delivery is accomplished through the application of the Service Oriented Architecture (SOA) style of design that incorporates mechanisms to route messages based on the application of rules and policies stored within the system that map to the business rules and processes of the operational organization. The architectural concept was to combine a single public endpoint for each application layer protocol that then access a set of backend services. The architecture of the IC.NET prototype implementation is represented in figure 3.

The message routing and exposure architecture illustrated in figure 3 is designed to be transport layer agnostic so that the end user can seamlessly connect to the system regardless of the underlying transport protocol. Although our public services, which are depicted below the ".NET 4.0 Routing Service" currently only support Representational State Transfer (REST) over Hypertext Transfer Protocol (HTTP) and publication / subscribe over advanced queuing message protocol (AQMP), these services are not strongly bound to the underlying application layer protocol, so that additional application or transport layer protocols can be added by alteration of a configuration file and addition of a plug-in that implements a standard interface.

Additionally the endpoints allow for abstraction at the presentation / user interface level. By supporting a number of standard translations such as GeoRSS and Keyhole Markup Language (KML) in addition to raw XML feeds of EDXL-DE messages and content payloads we are able to rapidly integrate with a number of visualization platforms including Google Earth and ESRI ArcGIS. This allows the users the freedom to use whatever user interface was best for their specific use case.

## C. *ValueLists & Local Terminology*

One of the common problems in the Emergency Management domain is the need to support local or jurisdictional terminology. Although there are efforts underway for common code-lists through NIMS [2] and the National Information Exchange Model (NIEM), these efforts have not been widely adopted at the local level. In order to facilitate the use of local terminology, the EDXL data standards contain data objects called ValueLists.

ValueLists are "lists maintained by a Community of Interest (COI)" [6] that contain a unique name, called the ValueListURN, which is a unique string representing the list name, and a list of associated values. The IC.NET implementation uses the REST concept of unique Uniform Resource Identifiers (URI) for list names. The ValueList Exposure service depicted in figure 3 contains a flexible URI template structure that is linked to the underlying Value List data store. This allows both the data store to be dynamically updated and client applications to access IC.NET's terminology lists to take advantage of the EDXL-DE routing capabilities of the messaging system.

For example, given the base URI: http://icnet.mitre.org/ValueLists/{R1}/{R2}, where {R1} and {R2} are optional REST parameters to the ValueList exposure service, we can derive the implementation specified in Figure 4.

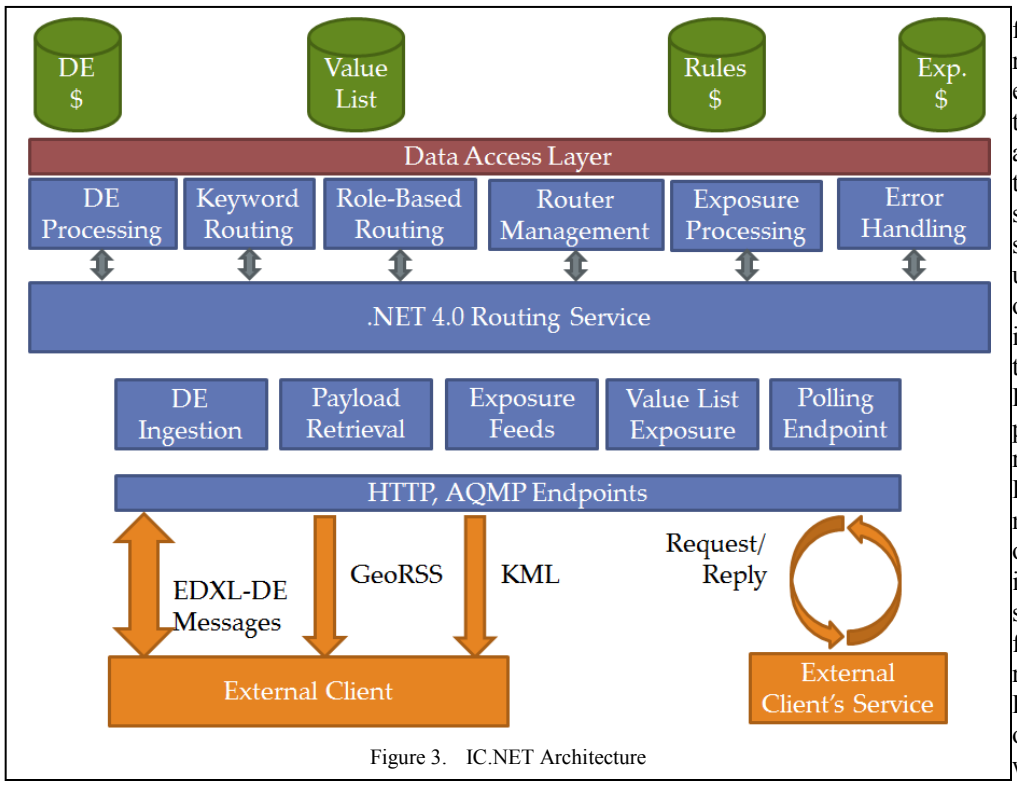


Figure 3. IC.NET Architecture

From this we can determine the possible values for {R1} and {R2} and therefore query for an individual list of values that is depicted in Figure 5.

*D. Example Message Routing & Delivery*

In order to illustrate how messages are routed in the IC.NET architecture, we have constructed a simple use case where an EMT in a field treatment sector has completed treating a patient in the field and will be arranging their transport to a hospital. This scenario involves the field EMT, the treatment sector supervisor, the medical control hospital, the transport officer, the transporting unit, the receiving hospital, and the supporting ICS command staff.

We will begin where the message is generated by the field EMT's client and placed on the network transport layer. Messages that are transmitted on the transport layer can be done through a variety of protocols such as TCP, UDP, etc. At the application layer of the network architecture, a variety of protocols can be tied into the layered architecture such as HTTP and AQMP, which represent the endpoints depicted in figure 3. These endpoints are representative of both points of entry and exit for messages from the architecture. The endpoints also provide the capability for translation from legacy or non-standard formats into the standardized format that the underlying architecture functions on. This capability allows for integration with older systems or those with non-xml data formats. From the endpoints, messages are placed on the common message routing platform through our public DE ingestion service. Although messaging oriented middleware is often realized in actual implementations as an enterprise service bus (ESB), many of the features of an ESB were not required in this implementation. Keeping with the user requirement of a simple and lightweight system, we chose to use the .NET 4.0 Routing Service which allows for translation across messaging protocols, and routing of messages based on built-in or custom filters through a single endpoint. This common interface provides a layer of abstraction among services, so that services are not tied directly to one another. This allows for services to be added, deleted, and modified without affecting other key components or services within the architecture. The .NET Routing Service runs on the Windows Communication Foundation (WCF) Platform and can be both scaled horizontally and load balanced depending on the hosting environment.

Once a message is transmitted to the routing service, the message is sent to the DE Processing Service which will perform the archiving of the message. The message will be passed to the data access layer, which acts as a broker between services and the underlying databases to allow for the database structures underneath to be changed without affecting code within the services / business processes that drive them. The data access layer will then archive the current message in the message archive database.

*When {R1}=={R2}==null => return unique List Names:*
*<ValueListURIs>*
*<ValueListURI>http://icnet.mitre.org/ValueLists/ContentKeywords</ValueListURI>*
*<ValueListURI>http://icnet.mitre.org/ValueLists/Roles/Facilities</ValueListURI>*
*<ValueListURI>http://icnet.mitre.org/ValueLists/Roles/ICS</ValueListURI>*
*<ValueListURI>http://icnet.mitre.org/ValueLists/Roles/Responders</ValueListURI>*
*<ValueListURI>http://icnet.mitre.org/ValueLists/Roles/Units</ValueListURI>*
*</ValueListURIs>*

Figure 4. Getting Supported ValueList Types

```
When {R1}=="Roles" && {R2}=="ICS" => return values:
<ICS>
<ValueInfo>
   <Value>TreatmentGroupSupervisor</Value>
   <CommandStructure>/IncidentCommander/OperationsSectionChief/TreatmentGroupSupervisor</CommandStructure>
</ValueInfo>
<ValueInfo>
   <Value>IncidentCommander</Value>
   <CommandStructure>/IncidentCommander</CommandStructure>
</ValueInfo>
<ValueInfo>
   <Value>OperationsSectionChief</Value>
   <CommandStructure>/IncidentCommander/OperationsSectionChief</CommandStructure>
</ValueInfo>
</ICS>
```

Figure 5. Retrieving a Specific ValueList's Values

Simultaneously the message is sent to both the keyword and role-based routing services. The EDXL-DE message header is examined by these services to determine the content types, their detail levels, and the roles of the senders associated with the message. Additionally the role of the sender will be examined to determine the potential general recipient roles for the message. These services make routing decisions based on linking information from both the underlying ValueLists and Rules Cache data store. When combined, this allows for the recipient roles to be assigned to the message which is added to the EDXL-DE message and transmitted back to the routing service. The message is then transmitted to the exposure processing service. This service inspects the individual roles exposes them through the underlying message feed and exposure rules to the exposure cache through the data access layer. The message is then exposed through the exposure data feeds on the various endpoints. If there are for the pub/sub clients, the message will be forwarded to those appropriate endpoints. Otherwise, the next time a client request is made to the endpoints for data (in a request / response system) the new data will be polled through the data exposure service through the appropriate endpoint to the client.

To elaborate on this general message flow, a specific message delivery example will be given. A patient treatment message is generated from a field EMS provider that contains three types of data:

- The unit location, transport information, and sender role of the transporting unit
- Basic patient information
- Detailed medical / treatment information.

This XML message is transmitted along with a binary payload of vital signs and an EKG image into the pub/sub endpoint of the layered architecture. The endpoint will transmit the message to the routing service where it will then be ingested by the DE Processing service. The DE Processing service will push the message through the data access layer into the message archive. The message is also sent to the keyword and role-based routing services where they will examine the message keywords and sender roles to determine the categories or keywords of the data contained within the message by utilizing the underlying value lists through the data access layer. The two routing services will add this information (keywords: transport message, unit update, patient update, patient treatment message, patient transport message, vital signs data, EKG data). The router rules state what recipient roles should receive the data. In this case the treatment officer, medical control hospital, transport officer, incident commander, EMS group supervisor, operations section chief, emergency medical services branch director, receiving hospital, and transporting unit are all defined as roles that should receive the message. The keyword and role-based routing services will enter these values in the recipient role fields and transmit the message back to the routing service. From there the message will be ingested by the exposure processing service which will examine both the keywords and recipient roles to assign the cached exposure data to the various roles and will publish the messages to the various endpoints. In this case all message recipients poll RESTful web services, so the exposure processing service will publish the message data to the REST endpoints, where they will published to the clients through the exposure feeds.

## IV. Results/Future Work

During the first year of this research effort, we have presented [12] our initial results of the successful implementation of the data model for ICS and EMS operations. These efforts are being deployed as a prototype on the public internet for testing and use by the general public as well as a closed prototyping effort with the Los Angeles Fire Department (LAFD). Our work with LAFD has been to integrate with their existing legacy CAD software, their new MDT software, and a number of legacy systems in the City of Los Angeles. We plan to publish results from our findings regarding translation of legacy binary and free-text data formats for CAD and MDT systems in future papers. Additional research for event representation is underway to connect the work with EDXL-DE and Situation Reporting to the National Emergency Numbers Association's (NENA) Next-Gen 911 project. We also plan to abstract the software design patterns used in this software-based router into a

common set of layers for implementation of enterprise architecture, along with their associated loose couplers, similar to the network layered architecture.

## V. CONCLUSIONS

IC.NET is a work in progress prototype designed to bring focus on the need for interoperable systems and data standards for operational-level first responder systems such as CAD and MDT software. This work has developed a data model for the First Responder Domain, extracted common data exchange patterns for two EMS use cases, married existing data standards where applicable, and identified gaps in the data model. By combining these loose couplers with a dynamic and intelligent message routing system that is both content and context aware, we have created an environment that captures comprehensive field data and routes it to the appropriate parties within ICS and their stakeholders.

During the preliminary prototyping effort, we were able to successfully create a lightweight software-based EDXL-DE router that was able to route, deliver, and expose messages based on payload content type and sender role information. We will continue to explore both the operational and software design implications for our initial findings.

## REFERENCES


[1] FEMA (2008). Incident Command System (ICS), Review Materials. Retrieved from the FEMA Independent Study Web site: http://training.fema.gov/EMIWeb/IS/ICSResource/assets/reviewMaterials.pdf

[2] FEMA, "National Incident Management System (NIMS)," December 2008. Retrieved from:
http://www.fema.gov/pdf/emergency/nims/NIMS_core.pdf

[3] FEMA (2008). IS-100 Introduction to Incident Command System ICS-100 Instructor & Student Guide. Retrieved from the FEMA Independent Study Web site: http://training.fema.gov/EMIWeb/IS/is100alst.asp

[4] FEMA (2008). IS-200 ICS for Single Resources and Initial Action Incidents ICS-200 Instructor & Student Guide. Retrieved from the FEMA Independent Study Web site: http://training.fema.gov/EMIWeb/IS/is200alst.asp

[5] FEMA (2009). *IS-700 NIMS An Introduction Instructor & Student Guide.* Retrieved from the FEMA Independent Study Web site: http://training.fema.gov/EMIWeb/IS/is700alst.asp

[6] Organization for the Advancement of Structured Information Standards (OASIS) (2006). *Emergency Data Exchange Language Distribution Element (EDXL-DE) 1.0 OASIS Standard.* Retrieved from the OASIS Web site: http://docs.oasis-open.org/emergency/edxl-de/v1.0/EDXL-DE_Spec_v1.0.pdf

[7] OASIS (2007). *Common Alerting Protocol (EDXL-CAP) 1.1 Errata. OASIS Standard.* Retrieved from the OASIS Web site: http://docs.oasis-open.org/emergency/cap/v1.1/errata/approved/CAP-v1.1-errata.pdf

[8] OASIS (2008). *Emergency Data Exchange Language Hospital Availability Exchange (EDXL-HAVE) 1.0 Errata. OASIS Standard.* Retrieved from the OASIS Web site: http://docs.oasis-open.org/emergency/edxl-have/v1.0/errata/edxl-have-v1.0-os-errata-os.pdf

[9] OASIS (2008). *Emergency Data Exchange Language Resource Messaging (EDXL-RM) 1.0 Errata. OASIS Standard.* Retrieved from the OASIS Web site: http://docs.oasis-open.org/emergency/edxl-rm/v1.0/errata/EDXL-RM-v1.0-OS-errata-os.pdf

[10] OASIS (2009). Emergency Data Exchange Language (EDXL) Requirements Statement and draft Messaging Specification for the Situation Reporting Messaging Standard (EDXL-SitRep). Retrieved from the OASIS Web site: http://www.oasis-open.org/apps/org/workgroup/emergency-msg/download.php/32999/EDXL-SitRep-Rqmts-MsgSpec020209.pdf

[11] United States Department of Homeland Security [DHS] (2009). Emergency Data Exchange Language (EDXL) Project Initiation Document (PID) For the Tracking of Emergency Victims (EDXL-TEP) Messaging Standard PHASE I - Tracking of Emergency Patients (EDXL-TEP). Retrieved from the Evotec inc. Web site: http://www.evotecinc.com/TEP/index.php?dir=&file=EDXL-TEP%20Project%20Initiation%20Document%20%28PID%29%20v4%201.pdf

[12] McGarry, D.P. and Chen, C.Y. (2010). MITRE IC.NET. Conference presentation at the Emergency Services Workshop 7. College Park, Maryland, USA 11-13 May 2010.

[13] Leonard, H. B. and Howitt, A. M., *Against Desperate Peril: High Performance in Emergency Preparation and Response*, forthcoming in *Communicable Crisis*, edited by D. E. Gibbons. Leonard and Howitt, professors at Harvard's Kennedy School of Government, have researched the characteristics of novel crises and have distinguished sudden crises from emergent crises.

[14] Bledsoe, Porter, Cherry. (2006). Paramedic Care Principles & Practices: Special Considerations Operations. Second Edition. Upper Saddle River, NJ: Pearson Education.

[15] OASIS (2009). *The Distribution Element:The Basic Steps to Package and Address Your Emergency Information.* Retrieved from the OASIS Web site:
http://www.oasis-open.org/apps/org/workgroup/emergency/download.php/34264/EDXL-DE-Basics-WhitePaper-18Aug09-r2.doc

[16] Gamma, Helm, Johnson, Slissides (1995). *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley.

[17] Sahana Foundation. (2010). *Haiti 2010 Sahana Disaster Response Portal.* Retrieved From: http://haiti.sahanafoundation.org/prod/

[18] National EMS Information System (NEMSIS). (2010). *NEMSIS v 2.2.1 Data Standard.* Retrieved from NEMSIS Website:
http://www.nemsis.org/softwareDevelopers/downloads/datasetDictionaries.html