An Open-System Transportation Security Sensor Network: Field-Trial Experiences

Daniel T. Fokum, *Student Member, IEEE*, Victor S. Frost, *Fellow, IEEE*, Martin Kuehnhausen, *Student Member, IEEE*, Daniel DePardo, *Senior Member, IEEE*, Angela N. Oguna, *Student Member, IEEE*, Leon S. Searl, Edward Komp, Matthew Zeets, Daniel D. Deavours, *Member, IEEE*, Joseph B. Evans, *Senior Member, IEEE*, and Gary J. Minden, *Senior Member, IEEE*

***Abstract*—Cargo shipments are subject to hijack, theft, or tampering. Furthermore, cargo shipments are at risk of being used to transport contraband, potentially resulting in fines to shippers. The Transportation Security Sensor Network (TSSN), which is based on open software systems and service-oriented architecture principles, has been developed to mitigate these risks. Using commercial off-the-shelf hardware, the TSSN can detect and report events that are relevant to appropriate decision makers. However, field testing is required to validate the system architecture and to determine if the system can provide timely event notification. Field experiments were conducted to assess the TSSN's suitability to monitor rail-borne cargo. Log files were collected from these experiments and were postprocessed. We present the TSSN architecture and results of field experiments, including the time taken to report events using the TSSN and the interaction between various components of the TSSN. These results show that the TSSN architecture can be used to monitor rail-borne cargo.**

***Index Terms*—Cargo security, mobile rail network (MRN), service-oriented architecture (SOA), trade data exchange (TDE), transportation security, virtual network operations center (VNOC).**

I. Introduction

In 2006, the Federal Bureau of Investigation (FBI) estimated that cargo theft costs the U.S. economy between $15 billion and $30 billion per year [1]. Cargo theft affects originators, shippers, and receivers as follows. Originators and receivers need a reliable supply chain to deliver goods in a timely and cost-effective manner. Shippers hold liability and insurance costs for shipments, which are proportional to the rate of theft. Finally, receivers are affected by out-of-stock and scheduling issues due to cargo theft. Most nonbulk cargo travels in shipping containers. Container transport is characterized by complex interactions between shipping companies, industries, and liability regimes [2]. Deficiencies in the container transport chain expose the system to attacks such as the commandeering of a legitimate trading identity to ship an illegitimate or dangerous consignment, hijack, or the theft of goods. Insufficiencies in these areas can be overcome by creating secure trade lanes or trusted corridors, particularly at intermodal points, e.g., at rail or truck transitions. Research and development is under way to realize the vision of trusted corridors.

This paper focuses on advanced communications, networking, and information technology applied to creating trusted corridors. The objective of this paper is to provide the basis needed to improve the efficiency and security of trade lanes by combining real-time tracking and associated sensor information with shipment information. One crucial research question that must be answered to attain this objective is how we can create open technologies that will allow continuous monitoring of containers by integrating communications networks, sensors, and trade and logistics data. This integration must occur within an environment composed of multiple enterprises, owners, and infrastructure operators.

To achieve improved efficiency and security of trade lanes, we have developed the Transportation Security Sensor Network (TSSN) architecture, which uses service-oriented architecture (SOA) [3] principles, to monitor the integrity of rail-borne cargo shipments. The TSSN is an open system where different components can be provided by different vendors. The TSSN is composed of a trade data exchange (TDE) [4], a virtual network operations center (VNOC), and a mobile rail network (MRN). The functions of each of these components are discussed in more detail in Section II. The TSSN detects events, integrates the event type from the train in the field with logistics information, and then reports events that are important to decision makers by using networks with commercial links. Decision makers want to be notified of events within 15 min [5] so that they can take effective action. For the TSSN to be deployed, we need to validate its architecture and understand the timeliness of the system response. The works of Arsanjani *et al.* [6] and Saiedian and Mulkey [7] show that SOAs introduce overhead. As a result, we want to determine whether an SOA-based system, e.g., the TSSN, provides timely event notification. TSSN event notification is also affected by unpredictable packet latency on commercial networks and the use of e-mail and Short Message Service (SMS) [8] for event notification. Thus, we

Manuscript received October 9, 2009; revised March 10, 2010 and July 3, 2010; accepted July 16, 2010. Date of publication July 23, 2010; date of current version October 20, 2010. This work was supported in part by Oak Ridge National Laboratory (ORNL)—Award Number 4000043403. This material is also based in part upon work supported while V. S. Frost was serving at the National Science Foundation. This work was conducted in collaboration with EDS, an HP company, and by Kansas City SmartPort. The review of this paper was coordinated by Dr. L. Chen.

The authors are with the Information and Telecommunication Technology Center, The University of Kansas (KU), Lawrence, KS 66045 USA (e-mail: fokumdt@ittc.ku.edu; frost@ittc.ku.edu; mkuehnha@ittc.ku.edu; ddepardo@ittc.ku.edu; oguna@ittc.ku.edu; searl@ittc.ku.edu; komp@ittc.ku.edu; mzeets@ittc.ku.edu; deavours@ittc.ku.edu; evans@ittc.ku.edu; gminden@ittc.ku.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2010.2060504

have designed and implemented hardware and software needed to realize a prototype of the TSSN and carried out experiments [9] to characterize the system, particularly the end-to-end time between event occurrence and decision-maker notification using SMS or e-mail, as well as the impact of SOA overhead.

In this paper, we provide a high-level description of the TSSN architecture and document two field experiments that were carried out to demonstrate that sensors and software based on an open SOA can be used to monitor cargo in motion. Our experiments focused on determining the time from event occurrence to decision-maker notification and on testing functionality between the component services of the prototype TSSN. Our experimental results show that decision makers can be notified of events on the train in a timely manner by using the prototype TSSN architecture. The rest of this paper is organized as follows. In Section II, we describe the TSSN architecture, including the components and the prototype hardware implementation. Section III discusses experiments that were conducted to assess the suitability of the TSSN system for cargo monitoring. In Section IV, we discuss the framework used to postprocess the log files from our experiments. Section V presents empirical results that show the interaction between various components of the TSSN. Refinements to the TSSN, given our field-trial experiences, are discussed in Section VI. In Section VII, we present related research on monitoring trains and securing shipping containers in motion. Section VIII provides concluding remarks.

II. System Architecture

To achieve the vision of a trusted corridor, we have designed and implemented a prototype of the TSSN architecture. The detailed architecture of the TSSN, including system extensibility, is found in [10], whereas this section gives an overview of the TSSN. The architectural details discussed here are important in understanding the experiments and results presented in Sections III and V, respectively.

The SOA and web services used in the TSSN enable the integration of different systems from multiple participating partners. Moreover, the use of SOA and web services enables us to enter data once and use those data many times. Using commercial off-the-shelf (COTS) hardware and networks, as well as an open systems approach, the TSSN can detect events and report events that are relevant to shippers and other decision makers as alarms. Furthermore, the TSSN supports multiple methods for notifying decision makers of system events.

The TSSN uses web service specification standards, e.g., Web Services Description Language (WSDL) 2.0 [11], Simple Object Access Protocol (SOAP) 1.2 [12], Web Services (WS)-Addressing [13], WS-Security [14], and WS-Eventing [15], which are implemented through Apache Axis2 [16] and associated modules. These standards are used to exchange structured information between a web service and a client. The use of SOAP allows the deployment of platform-independent interfaces and, thus, a heterogeneous network of web service platforms. On the other hand, because SOAP and web services are based on Extensible Markup Language (XML), which is verbose, there is communication and processing overhead related to SOAP messages.

The TSSN supports wireless and satellite communication technologies, e.g., High-Speed Downlink Packet Access (HSDPA) [17] and Iridium [18]. The TSSN uses the Hypertext Transfer Protocol (HTTP) for message transport over wired and wireless links. Finally, the TSSN prototype uses sensors and readers from Hi-G-Tek [19]. There is also a need to gather log files to enable system debugging and to capture metrics that can be used to evaluate system performance. Logging is currently done at the MRN, VNOC, and TDE using Apache log4j [20].

As shown in Fig. 1, the TSSN system is composed of three major geographically distributed components: 1) TDE; 2) VNOC; and 3) MRN. Wired links are used between the TDE and the VNOC, whereas MRN-to-VNOC communications are done using networks with commercial wireless link components. The TDE, VNOC, and MRN are examined in more detail in the following sections.

A. TDE

The TDE contains shipping data, and it interconnects commercial, regulatory, and security stakeholders. The TDE is based on a technology-neutral standards-based SOA [4]. The TDE is hosted on a server with a wired connection to the Internet. The TDE is geographically separated from the VNOC and responds to queries from the VNOC. The TDE also stores event messages that the VNOC sent. Finally, the TDE sends commands to start and stop monitoring at the MRN and to get the train's current location.

In addition to the aforementioned functions, the TDE monitors the progress of shipment and other logistics information. The TDE captures commercial and clearance data, including the shipping list, bill of lading, commercial invoice, Certificate of Origin [e.g., the North American Free Trade Agreement (NAFTA) Letter], and shipper's export declaration. It also validates and verifies data to ensure accuracy, consistency, and completeness. The TDE monitors the progress of the documentation and notifies responsible parties when errors or incompleteness pose the threat of delaying a shipment. The TDE forwards notification to the customs broker to request verification of the trade origination documents. The customs broker accesses the TDE through the same portal to review and verify the trade documentation. Finally, the TDE allows for collaboration between participating shippers, third-party logistics providers, carriers, and customs brokers to define and document business requirements and risk assessment requirements. Real-time cargo-sensing capability is provided to the TDE through the TSSN. Data from the TDE are combined with event data from the MRN to provide the decision-maker complete information with regard to the alarm, e.g., cargo information, location, and nature of the event.

B. VNOC

The VNOC is the management facility of the TSSN [10], and it is also the shipper's interface to the TDE. The VNOC

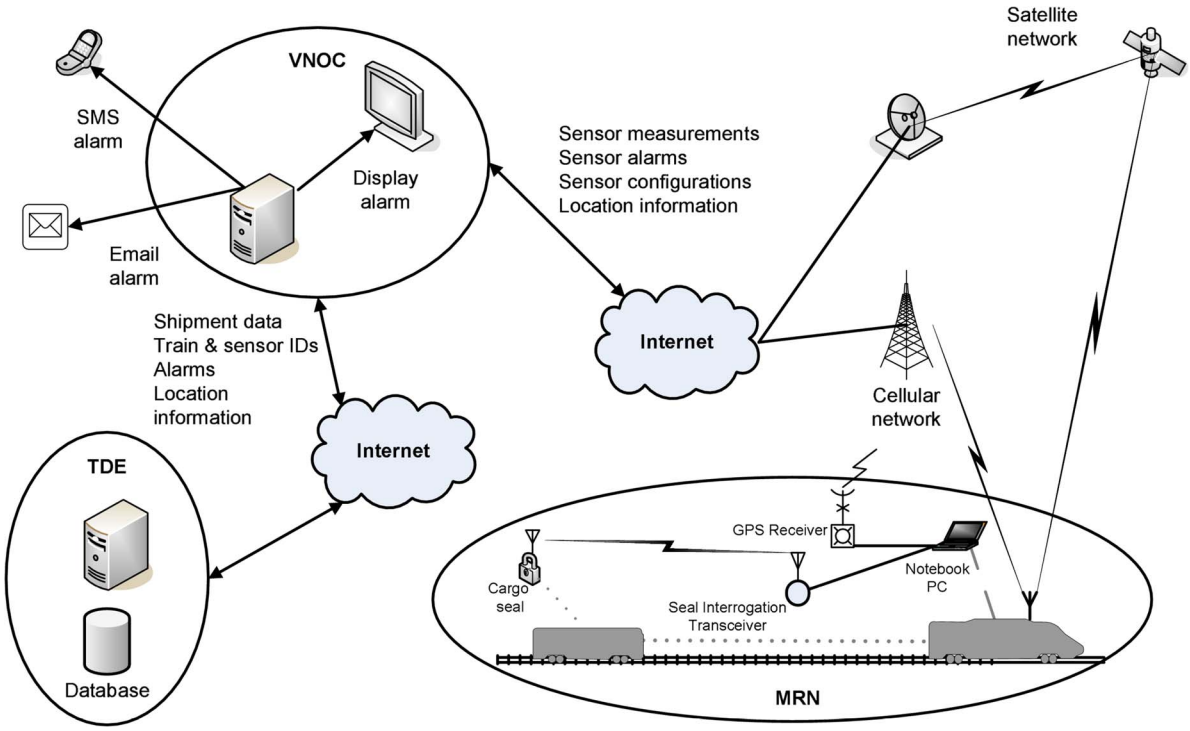


Fig. 1. TSSN architecture.

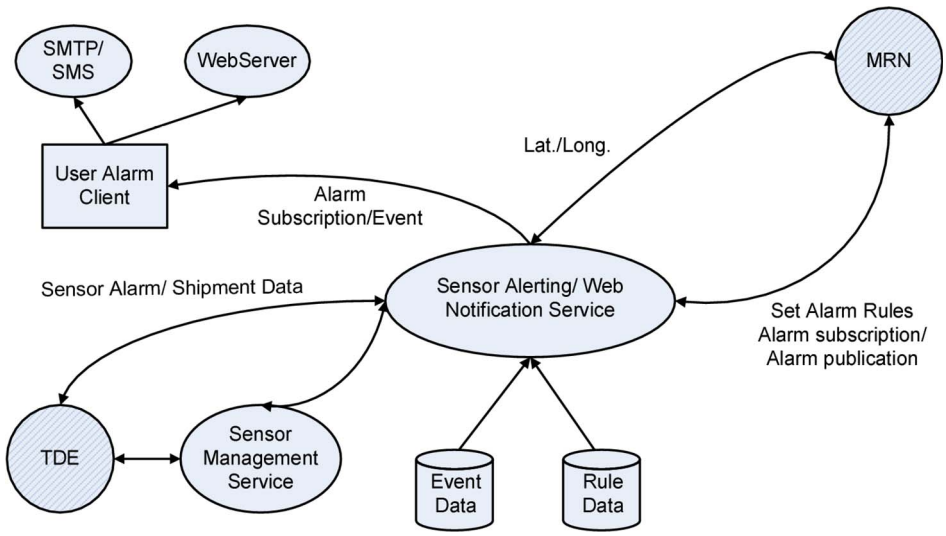


Fig. 2. VNOC architecture.

can serve as the central decision and connection point for multiple MRNs. The VNOC consists of services that receive and process alarms from the MRN and services that notify decision makers of events. Fig. 2 summarizes the VNOC and its components.

The functions of the VNOC include forwarding commands from a client to the MRN to start and stop sensor monitoring, as well as to get the MRN's current location, receiving *MRN_Alarms* from the MRN, obtaining event-associated cargo information from the TDE in real time, and combining cargo information obtained from the TDE with an *MRN_Alarm* to form a *VNOC_Alarm* message that is sent by SMS or e-mail to decision makers, as shown in Figs. 9 and 10. One key role of the VNOC is to get the right alarm information to the right personnel in a timely manner and also to prevent personnel from being overwhelmed with event messages. An AlarmProcessor service in the VNOC makes decisions, using rules, on which *MRN_Alarms* are forwarded to decision makers. For example, a low-battery alarm is sent to technical staff, whereas an unexpected open/close event is sent to system security personnel. These decisions are made using a complex event processor, Esper [21], which takes into account shipping information and data (e.g., geographical location) from current and past *MRN_Alarms*.

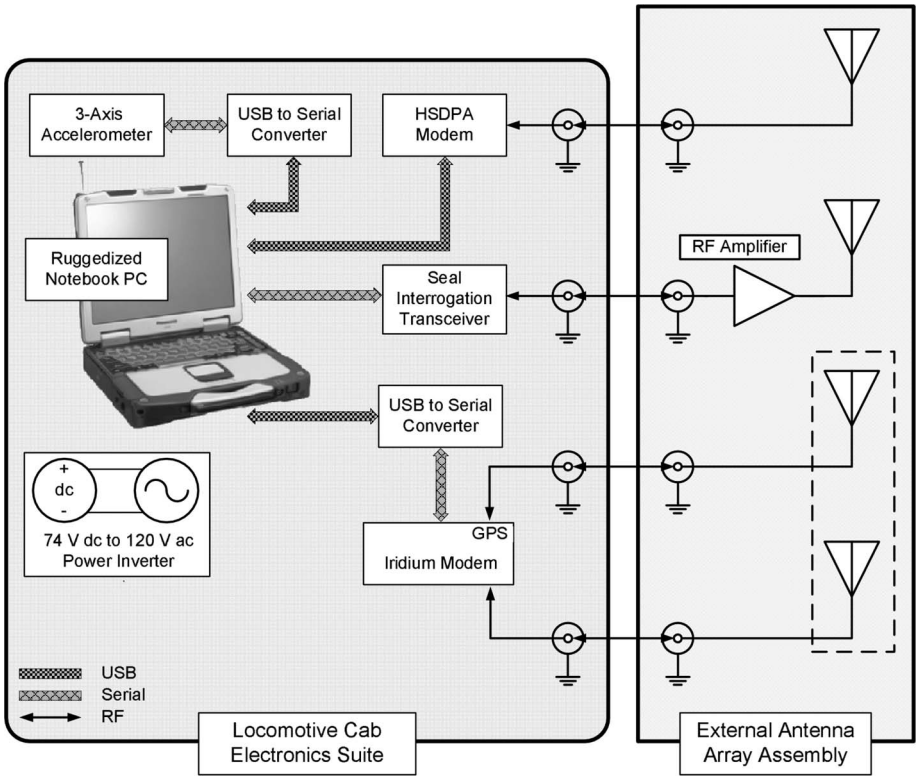


Fig. 3. TSSN collector node hardware configuration.

C. MRN

The MRN subsystem is located on the train, and it consists of hardware and software. The prototype hardware and software architecture is described as follows.

1) MRN Hardware: The MRN subsystem hardware consists of a set of wireless shipping container security seals and a TSSN collector node. The collector node is composed of two major sections: 1) an electronics suite mounted in the locomotive cab and 2) a remote antenna assembly that is magnetically attached to the exterior of the locomotive. Fig. 3 summarizes the key components of the TSSN collector node.

The electronics suite contains a power inverter, a security seal interrogation transceiver (SIT), a computing platform, wireless data modems, a three-axis accelerometer, and a Global Positioning System (GPS) receiver. The antenna assembly consists of three communications antennas, a GPS receiver antenna, and a bidirectional RF amplifier. Coaxial cables connect electronics suite devices to corresponding antennas.

Container physical security is monitored using a system that was originally designed for tanker truck security [19]. Container security is monitored with active and battery-powered container seals (sensors) equipped with flexible wire lanyards that are threaded through container keeper bar lock hasps, as shown in Fig. 4. These seals had no support for multihop communications. The TSSN is designed to monitor and report security seal events, including seal opened, seal closed, tampered seal, seal armed, seal missing, and low-battery warnings. The SIT communicates with the container seals over a wireless network, whereas the interrogation transceiver communicates with a notebook computer through a serial data connection. Each container seal contains a clock that is periodically updated from the SIT, whereas the time on the SIT is updated from the notebook computer. The mechanism for time synchronization of the seals is outside the scope of this paper.

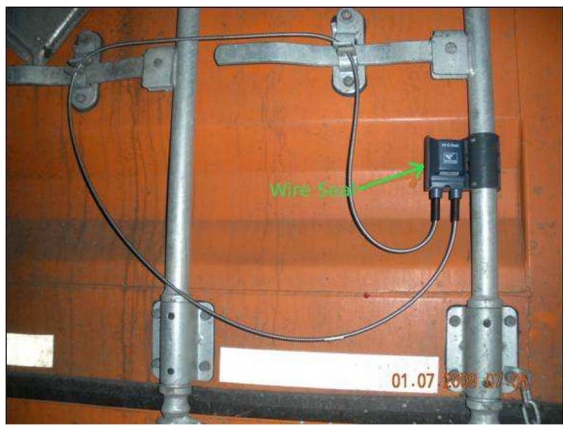


Fig. 4. Container seal.

To conserve energy, the container seals are asleep most of the time [22]. About every 3 s, the seals listen for commands from the interrogation transceiver; however, the frequency at which the seals listen for commands is configurable. If the sensors are instructed to more frequently listen for commands, then their battery lifetimes are reduced, whereas longer intervals between interrogations result in longer battery lifetimes [22].

Communication between the MRN and the VNOC is accomplished using an HSDPA cellular data modem. An Iridium satellite modem is also available and is intended for use in remote locations that lack cellular network coverage. The Iridium modem is a combination unit that includes a GPS

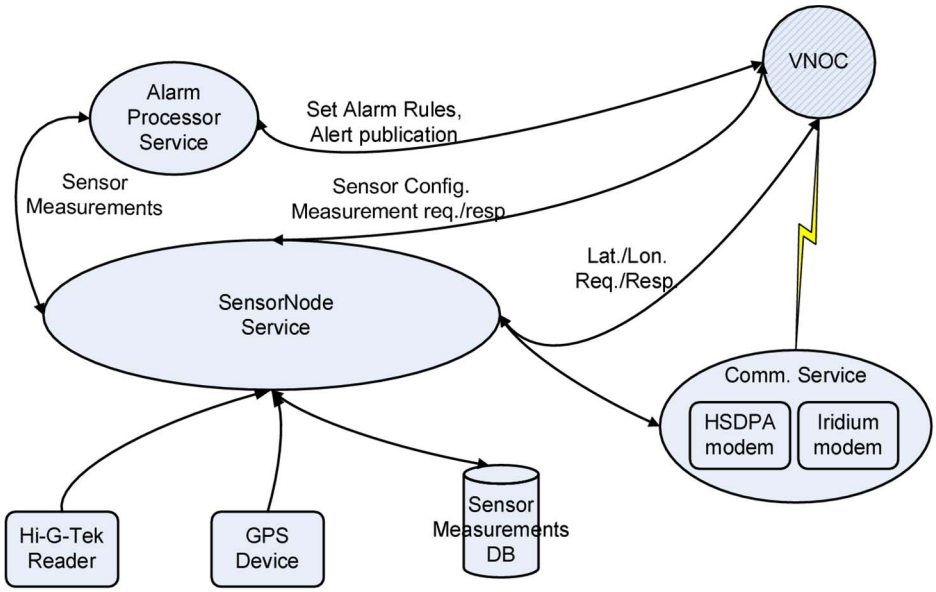


Fig. 5. MRN collector node architecture.

receiver, which is used to provide the MRN with position information.

2) MRN Software: The prototype MRN software was implemented using the SOA approach. The software consists of a SensorNode service, an AlarmProcessor service, and a Communications service. The SensorNode service finds and monitors sensors that have been assigned to its control. The SensorNode service manages several sensor software plug-ins, e.g., a SIT plug-in and a GPS device plug-in, that do all the work on behalf of the SensorNode service. During a typical operation, each container seal listens for interrogation command signals at regular intervals from the interrogation transceiver. In case that a seal is opened, closed, or tampered with, the seal immediately transmits a message to the SensorNode service that runs on the collector node. The message contains the seal event, a unique seal ID, and the event time. The SensorNode service passes the seal message as an *Alert* message to the service that has subscribed for this information.

The AlarmProcessor service determines which messages from the SensorNode service require transmission to the VNOC as *MRN_Alarms*. Alarm messages include the seal event, the event time, the seal ID, and the train's GPS location. The Communications service uses either HSDPA or Iridium to report events through the Internet to the VNOC. Fig. 5 shows the key software functions of the MRN.

III. Experiments

This section presents two experiments—a road test and the short-haul rail trial—that were conducted to assess the suitability of the TSSN architecture for cargo monitoring and to collect data that would be used to guide the design of future cargo-monitoring systems. It is nontrivial to carry out experiments on moving freight trains; furthermore, as part of this effort, we were limited to one chance to carry out experiments from a train. As a result, the TSSN architecture was tested in several static and some mobile tests, including the road test with trucks and the short-haul rail trial. In this section, we present the experimental objectives, configuration, data collected during the tests, and issues that were encountered during the tests. The overarching goals of these experiments are listed as follows:

- to demonstrate the concept of using sensors, communications, and SOA to monitor cargo in motion using the TSSN architecture;
- to determine the time from event occurrence to decision-maker notification in a real field experiment;
- to verify proper operation of the prototype TSSN in a field environment (proper operation means that all messages were transmitted, received, and processed as expected and that decision makers received all correct notification).

Thus, the following items were within the scope of our experiments: 1) the stability of the communications protocols between TSSN component services and 2) their timely performance. On the other hand, the following items were out of the scope of this paper:

1) overall system robustness;
2) whole-train monitoring;
3) energy consumption of the sensors;
4) comprehensive security[1] issues, e.g., message spoofing;
5) decision-maker response time, given that event notification had been delivered.

A. Road Test With Trucks

The first experiment was conducted with two pickup trucks on local roads to validate the system operation and to determine if the TSSN collector node reports correct information, including valid GPS coordinates. One of the pickup trucks used in the test had the locomotive cab electronics suite in the truck bed, whereas both trucks had seals in their truck cabins so that seal open and close events could be emulated and reported.

[1]Comprehensive security issues will be addressed in the next version of the prototype.

Fig. 6. Map of road test with event annotations.

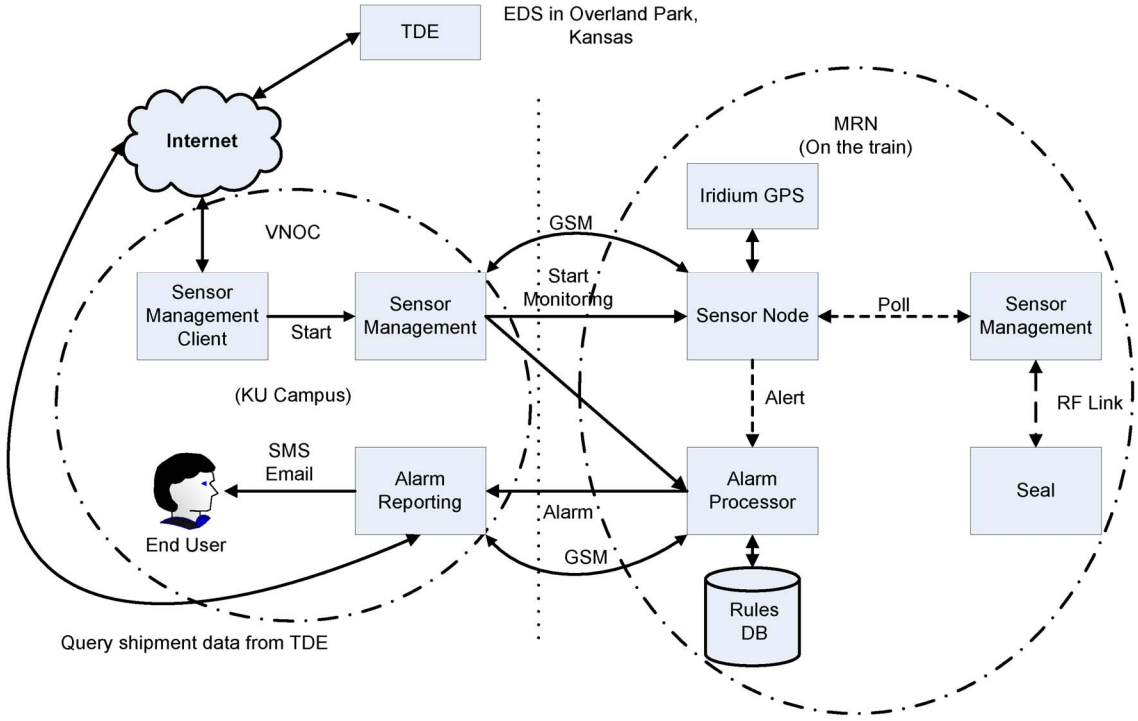


Fig. 7. Logical short-haul rail trial configuration.

The VNOC was located in Lawrence, KS, whereas the TDE was located in Overland Park, KS. The trucks were driven for approximately 1.5 h over a 90-km route that began and ended in Lawrence. The experiment route covered suburban and rural roads, as well as state highways. During the experiment, the seals were opened and closed at selected intersections along the test route that were easily identifiable on Google Maps [23]. Fig. 6 shows a trace of our route and the events overlaid on a Google map.

B. Short-Haul Rail Trial

Another experiment was carried out using a freight train that travels from an intermodal facility to a rail yard. Our objectives in this experiment are listed as follows:

- to determine the performance of the prototype TSSN architecture when detecting events on intermodal containers in a real rail environment;
- to investigate if decision makers could be informed of events in a timely manner using SMS messages and e-mails;
- to collect data that will be used in a model to investigate system tradeoffs and the design of communications systems and networks for monitoring rail-borne cargo;
- to evaluate the overall system performance to guide the future development of the TSSN architecture.

Fig. 7 shows the logical system configuration used in the short-haul rail trial. In this experiment, the VNOC was located in Lawrence, the TDE was located in Overland Park, and the MRN was placed on the train. Within the MRN, the TSSN collector node was placed in a locomotive and was used to monitor five seals. All communications between the MRN and the VNOC were passed through a virtual private network (VPN) for message security. Prior to the start of the experiment, prototype logistics data were added to the TDE to facilitate testing.

Fig. 8. Collector node and sensor deployment during short-haul rail trial.

```
NOC_AlarmReportingService:
 Date-Time: 2009.01.07 07:12:17 CST /
    2009.01.07 13:12:17 UTC
 Lat/Lon: 38.83858/-94.56186,
          Quality: Good
 http://maps.google.com/maps?q=38.83858,-94.56186
 TrainId=ShrtHaull
 Severity: Security
 Type: SensorLimitReached
 Message: SensorType=Seal
           SensorID=IAHA01054190
           Event=Open Msg=
 NOC Host: laredo.ittc.ku.edu

Shipment Data:
 Car Pos: 3
 Equipment Id: EDS 10970
 BIC Code: ITTC054190
 STCC: 2643137
```

Fig. 9. E-mail message received during short-haul trial.

During the short-haul trial, the train traveled for approximately 5 h over a 35-km (22-mi) route. The route, which traversed both rural and urban areas, was relatively flat, with a total elevation change of about 100 m. Fig. 8 shows the train used in the short-haul rail trial, along with the arrangement of the sensors (wire seals). As shown in Fig. 8, the short-haul trial train was composed of well cars with a mixture of empty cars, cars with a single container, and cars with double-stacked containers. Because we demonstrate a proof of concept and the sensors in use for this test were COTS devices with no support for multihop communications, three sensors were placed on containers on three of the five railcars nearest the locomotive so that they could be within the radio range of the SIT. One sensor was placed on the front of the locomotive, whereas the fifth sensor was kept in the locomotive and was manually opened and closed while the train was in motion to create events.

During the experiment, the VNOC reported events to decision makers by using e-mail and SMS messages. The e-mail messages also include a link to Google Maps so that the exact location of the incident could be visualized. Fig. 9 shows the content of one of the e-mail messages that was sent to the decision makers, and Fig. 10 presents one example of an SMS message.

In Figs. 9 and 10, the sensor ID, latitude and longitude data, and event type come from the MRN, whereas the shipment data come from the TDE. The VNOC combines these pieces

```
NOC_Alarm:
Time:2009.01.07 07:12:49 CST
GPS:38.83860/-94.56186
Trn:ShrtHaull
Sev:Security
Type:SensorLimitReached
Msg:Sensortype=Seal SensorID=IAHA01054190
    Event=Close
```

Fig. 10. SMS message received during short-haul trial.

of information into an e-mail message that also includes a link to Google Maps so that the exact location of the incident can be visualized.

During the test, the interrogation transceiver lost communication with the seals for a brief period along the route, whereas the train was stationary and then regained communications once the train started moving. We believe that this loss of communication was due to electromagnetic interference. However, further investigations are needed to validate this claim.

The short-haul rail trial was a success, because all seal events were detected and reported to decision makers using both e-mail and SMS messages. Extensive log files were collected during the test, and they were postprocessed to obtain data on the TSSN system performance. The results from postprocessing, which are reviewed in Section V, show that the prototype system functioned as expected.

Following this experiment, analysis of event logs obtained from the MRN, VNOC, and TDE revealed that there was a significant amount of clock drift on the TSSN collector node during this relatively short trial. The time recorded at the VNOC for the receipt of a message, in some cases, was earlier than the time recorded at the TSSN collector node for when the message was sent. Because time at the VNOC is controlled by a Network Time Protocol (NTP) [24] server, we conclude that the clock drift occurs on the TSSN collector node. The clock drift problem was resolved in the next version of the TSSN by using a high-performance GPS receiver to get high-quality local time. Pulse-per-second (PPS) output from the GPS receiver was used as an input to the NTP server that runs on the TSSN collector node. Note that, in spite of the clock drift in the TSSN collector node, we corrected for it in our data analysis by assuming that data from different parts of the TSSN are independent, e.g., the time taken to break a seal and generate an alert message is independent of the time taken to transfer

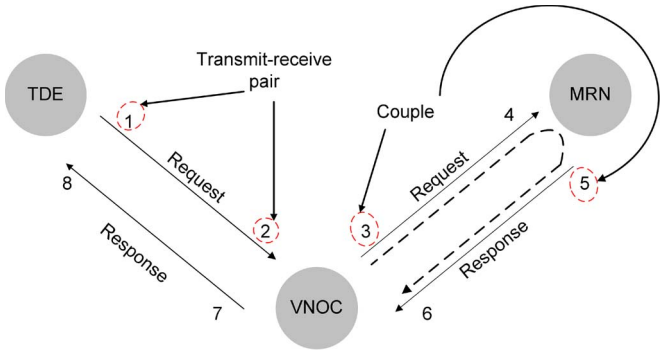


Fig. 11. LogParser framework that shows message couples and transmit–receive pairs.

a message from the MRN to the VNOC. As a result, we can separately measure the elapsed time in different epochs and characterize the performance of the TSSN prototype.

IV. Postprocessing of Experimental Data

In this section, we discuss the framework for postprocessing the results of our experiments. During the short-haul rail trial, we recorded events in log files at the geographically distributed VNOC, MRN, and TDE. These log files contained data on message sizes, timestamps, event type, and message type (incoming/outgoing), among other data elements. Our objective was to postprocess these files to evaluate the performance of the prototype TSSN.

Postprocessing of log files was accomplished using a Java library (LogParser) that was developed in-house. First, the library read in all available information in each log file, including time, message size, from and to addresses, and the original SOAP message. Information from the MRN, VNOC, and TDE log files in this experiment was combined into a single collection of log entries. We expect that every message transmitted in the TSSN should result in at least two log entries: 1) a transmit log entry (at the originating entity) and 2) a received log entry (at the receiving entity). The LogParser library identified log entries as follows:

- transmit–receive pairs, i.e., the outgoing and incoming log entries with the same SOAP WS-Addressing [13];
- couples, i.e., SOAP request–response message pairs.

Fig. 11 shows the relationship between log entry couples and transmit–receive pairs. Suppose that the TDE sends a message to the VNOC, requesting the current MRN location. The circled 1 and 2 in Fig. 11 denote the log entries that represent message transmission from the TDE and receipt of this same message at the VNOC. Much of the communication between the client and the server is based on a request–response model. As a result, there are two related messages that contain additional information to establish their relationship:

1) REQUEST — from the client to the server, asking for something;
2) RESPONSE — from the server back to the client with the response.

Log entry couples are marked by the records for the outgoing request and response messages. Consequently, the circled 3 and 5 in Fig. 11 constitute the log entry couple for the VNOC, forwarding the location request message to the MRN and the MRN's origination of a response, respectively. Using the receive pairs for records 3 and 5, we can also identify entries 4 and 6.

With this framework, programs were written against the log entry collection to extract the number of messages sent by each service, the request–response time for messages, the processing time at either the MRN, VNOC, or TDE, the time that messages were carried by the network, and message sizes. Additional information, e.g., latitude, longitude, sensor IDs, and event timestamps, is extracted from the SOAP message using XPath expressions. XML Path Language (XPath) is used to extract information from XML by using path expressions that traverse the XML tree. Because SOAP is based on XML and the elements that we use, e.g., *Alerts*, *MRN_Alarms*, and *VNOC_Alarms*, are also based on XML, the use of XPath is appropriate. XPath also provides basic facilities for manipulation of strings, numbers, and Booleans [25].

V. Results

This section discusses the results of the experiments presented in Section III. Most of the results shown here are based on the short-haul rail trial, because we had more data to analyze. The results presented here are selected to the following two test claims.

- All messages between the component services of the TSSN were transmitted, received, and processed, as expected.
- Decision makers can be notified of events on the train in a timely manner.

The rest of this section is organized as follows. Sections V-A and B present results on message counts for the road test and short-haul rail trial, respectively. These results test the claim that all messages between component services of the TSSN are transmitted, received, and processed as expected. The rest of the results are based on the short-haul rail trial. Sections V-C–E study different portions of the time from event occurrence to decision-maker notification to verify the claim that the TSSN can notify decision makers of events in a timely manner. Probability distributions are used in Section V-F to determine the likelihood of timely decision-maker notification.

Note that, due to significant clock drift in the TSSN collector node, we can only present an estimate of the time taken for an event report to travel from the MRN to the VNOC. However, observed time values can directly be used for other TSSN component interactions. These results show how the aggregate time from event detection to decision-maker notification is distributed among the various services and communication links in the TSSN. With this information, we can guide system refinements to further reduce the overall time. Suppose that T_n indicates when log entry n is made. Then, we can compute the following metrics:

- **Service request processing time**. This metric is the time between when a service receives a request and when a response message is composed. Using Fig. 11, this time is $T_5 - T_4$.

- **Request–response time**. This metric is the time taken to get a response from a remote service, including the processing time. Using Fig. 11, this time is $T_6 - T_3$.
- **Network time**. This metric is the time taken to get a response from a remote service, excluding the processing time. Using Fig. 11, this time is computed as $T_6 - T_3 - (T_5 - T_4)$.

Our time analysis in Section V-G examines request–response messages from VNOC → MRN → VNOC, TDE → VNOC → TDE, and VNOC → TDE → VNOC.

The last objective of the short-haul rail trial was to collect data that will be used in a model [26] to support the future design of systems for monitoring rail-borne cargo and to determine tradeoffs. Message size is one component of this model. As a result, Section V-H presents a table that summarizes the message size statistics between different components of the TSSN. Note that message sizes can be computed *a priori*; however, the distribution of these messages cannot be determined beforehand.

A. Road Test: Message Counts

The primary goal of the road test was to validate the TSSN prototype operation and to determine if correct information is reported by the TSSN collector node, including valid GPS coordinates. During the road test, a manual record was made of all seal events, and this written record was compared with the information generated from the TSSN. This comparison revealed that all open and close events were correctly propagated. During the approximately 1.5 h-long road test, 76 messages (72 *Alarms*, two *StartMonitorSensors*, and two *StopMonitorSensors* commands) were exchanged on the VNOC-to-MRN link, and these messages corresponded with the events that were recorded in the experiment log. Based on the analysis of these messages, we conclude that the system operated as expected. In addition, the experiment revealed that the TSSN recovered from a dropped HSDPA connection. However, note that the SIT could not read the sensors when the trucks were more than 400 m apart on a hilly road. Based on the road test, we conclude that the TSSN prototype worked as expected in a mobile scenario, and we combined sensor data from the MRN in a moving vehicle, with shipment information obtained from the TDE to generate e-mail messages that were sent to distributed decision makers. Results from the road test showed that the TSSN prototype was ready for evaluation in a real rail environment.

B. Short-Haul Trial: Message Counts

One objective of our postprocessing was to determine if messages were correctly passed between the TSSN components. During the short-haul trial, 203 messages (two *StartMonitorSensors*, two *StopMonitorSensors*, four *SensorNodeStatus*, four *SetMonitoringState* commands, 30 *getLocation* queries, 30 *Location* responses, and 131 *MRN_Alarms*) were passed over the VNOC-to-MRN link. Full details on the messages exchanged are found in [27]. All of the *MRN_Alarms* that the VNOC AlarmProcessor received met the necessary rules so that they could be forwarded to decision makers as SMS or e-mail messages. The test users who were designated to receive all event notifications from the TSSN received 131 e-mail messages each.

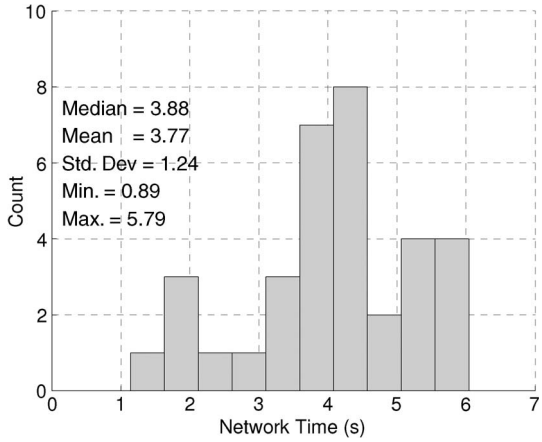


Fig. 12. Network times from the VNOC → MRN → VNOC.

C. Network Time From the VNOC to the MRN to the VNOC

The network time statistics from the VNOC to the MRN to the VNOC allow us to draw conclusions on the time taken to transfer request and response messages from the VNOC to the MRN, and vice versa. These statistics also allow us to gain insight into the one-way network delay from the TSSN collector node on the train to the VNOC in Lawrence—a delay that is one component of sending an *MRN_Alarm* message—which indicates an event at a sensor—from the MRN to the VNOC. Due to clock drift in the TSSN collector node, we could not obtain statistics on the one-way network delay from MRN → VNOC. However, it is reasonable to assume that the MRN ↔ VNOC links are symmetric; thus, the average one-way delay from the MRN to the VNOC is approximately 1.89 s. Fig. 12 is a histogram that shows the network time for messages from the VNOC to the MRN and back to the VNOC.

D. Elapsed Time From Alert Generation to AlarmReporting Service

The target notification time of security seal events is 15 min [5]. Thus, demonstrating that the elapsed time from alert generation to the AlarmReporting service is of the order of several seconds shows that the time taken to process events within the TSSN is not an impediment to timely notification. Fig. 13 shows the messages involved in notifying a decision maker of an event at a seal. This section deals with epochs 2, 3, and 4. Exact values can be computed for the time taken to propagate *Alert* and *VNOC_Alarm* messages, whereas we can use the 1.89 s estimate from the previous section as a reasonable value for the time taken to transfer a *MRN_Alarm* message from the MRN to the VNOC.

By analyzing the log files, we see that, on the average, it takes about 2 s for messages to get from the MRN SensorNode service to the VNOC AlarmReporting service. Thus, we conclude that the time taken to process events in the TSSN is not an impediment to timely notification of decision makers.

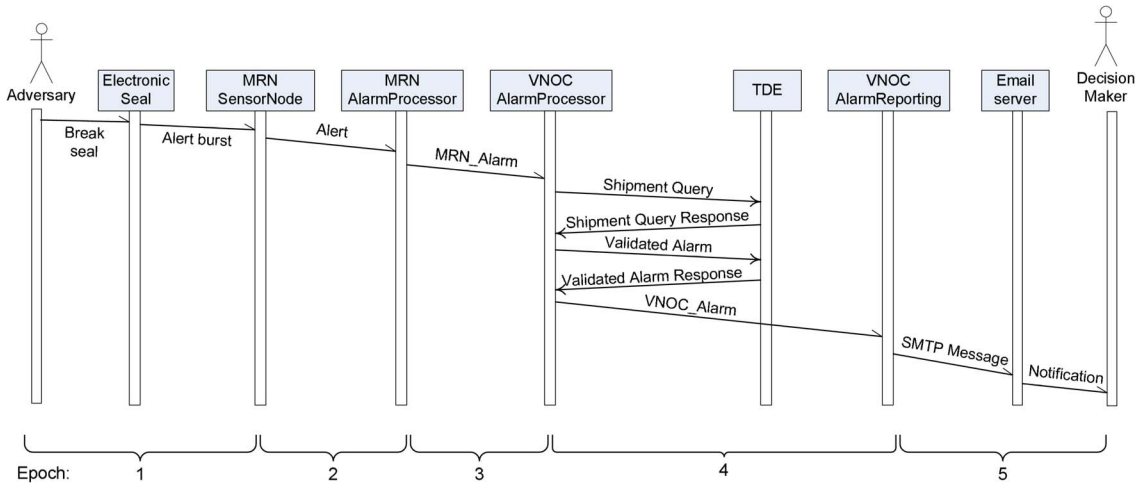


Fig. 13. Sequence diagram with messages involved in decision-maker notification.

TABLE I
SUMMARY OF TIME STATISTICS FOR DECISION-MAKER NOTIFICATION

Epoch	Description	Min./s	Max./s	Mean/s	Median/s	Std. Dev./s
1	Event occurrence to *Alert* generation	0.81	8.75	2.70	2.13	1.86
2	*Alert* generation to MRN AlarmProcessor service	0.01	0.08	0.02	0.01	0.01
3	One-way delay from MRN AlarmProcessor to VNOC AlarmProcessor	0.45	2.90	1.89	1.94	0.62
4	*MRN_Alarm* arrival at VNOC AlarmProcessor to AlarmReporting service	0.01	3.01	0.17	0.05	0.32
5	Elapsed time from VNOC AlarmReporting service to mobile phone	5.2	58.7	11.9	9.8	7.4

E. End-To-End Time From Event Occurrence to Decision-Maker Notification

In this section, we study the end-to-end system time between event occurrence and decision-maker notification. The components of the end-to-end time include epochs 1–5 in Fig. 13. Decision makers are notified of events using SMS or e-mail. In the case of SMS notification, a short Simple Mail Transfer Protocol (SMTP) message is sent to an e-mail-to-SMS gateway on a carrier's network, whereas with e-mail notification, the SMTP message length is unrestricted, and a message is sent to an e-mail server. The primary performance metric for prototype TSSN performance is the time between event occurrences until a decision maker is notified using an SMS message.

To gain an understanding of the end-to-end system time and to overcome any clock errors in the MRN subsystem, we set up a laboratory experiment to determine the elapsed time between an event occurrence and the TSSN generation of the related event alert. In this experiment, a stopwatch was started when a seal was either broken or closed. When the MRN SensorNode service generated an *Alert* message, the stopwatch was stopped. In Table I, we see that the longest observed time in epoch 1 is about 8.8 s, whereas the mean is about 2.7 s.

Because the commercial wireless networks used for decision-maker notification are outside the TSSN control, a second laboratory experiment was carried out to determine the elapsed time in epoch 5. In this experiment, a client program was written to send messages to the VNOC AlarmReporting service. A stopwatch was started when the VNOC sent an alarm to a decision maker, and the stopwatch was stopped when the decision maker's phone received an SMS message. This experiment was repeated for four different carriers, resulting in the data shown in Table I, row 5.

In Table I, we see that, although SMS was not designed as a real-time system, it provides excellent notification for this application, because most of our messages were delivered within 1 min. Combining all of these results, we see that, in these experiments, the longest observed end-to-end system time was just more than 1 min[2] to notify decision makers of events. Most of this time is spent delivering an SMS message to the decision maker; therefore, we conclude that the TSSN provides a mechanism for timely notification of decision makers.

F. Modeling of Decision-Maker Notification Time

In this section, we determine the likelihood of timely event notification. To determine the likelihood of timely event notification, a probabilistic model is needed for the time epochs shown in Fig. 13. The observed histograms for each epoch visually resembled a Gamma distribution. Thus, in this analysis, we assume that the times in each epoch followed a Gamma probability density function. Although the number of observations (less than 130) was insufficient to statistically validate this assumption, this postulate allows us to probabilistically determine if the TSSN prototype can provide event notification within 15 min [5], as required. The parameters for the distributions are estimated from the collected data and shown in Table II, where $\hat{\alpha}$ and $\hat{\theta}$ represent the shape and scale parameters of the associated Gamma random variable. Let τ, which is composed of each of the epochs presented in Sections V-C–E, represent

[2]This time is broken out as follows: 1) in the longest observed times in our experiments, it took approximately 8.8 s between event occurrence and the TSSN to generate an alert; 2) it took approximately 4.91 s for an alert message to go through the TSSN until notification was sent to decision makers; and 3) it took up to 58.7 s to deliver an SMS message to decision makers.

TABLE II
ESTIMATED GAMMA DISTRIBUTION PARAMETERS FOR THE TIME TAKEN BETWEEN SEAL EVENTS AND DECISION-MAKER NOTIFICATION

Epoch	Symbol	Description	$\hat{\alpha}$	$\hat{\theta}$
1	E_1	Event occurrence to *Alert* generation	4.01	0.60
2 + 4	$E_{2,4}$	*Alert* generation to MRN AlarmProcessor and *MRN_Alarm* arrival at VNOC AlarmProcessor to AlarmReporting service	1.13	0.13
3	E_3	One-way delay from MRN AlarmProcessor to VNOC AlarmProcessor	13.95	0.14
5	E_5	Elapsed time from VNOC AlarmReporting service to mobile phone	10.44	1.00

TABLE III
SUMMARY OF TIME STATISTICS FOR OTHER TSSN INTERACTIONS

Description	Min./s	Max./s	Mean/s	Median/s	Std. Dev./s
Request-response times from VNOC → MRN → VNOC	0.90	10.96	4.39	3.95	2.40
Network times from VNOC → MRN → VNOC	0.89	5.79	3.77	3.88	1.24
Processing times from VNOC → MRN → VNOC	0.00	5.21	0.61	0.01	1.69
Request-response times from TDE → VNOC → TDE	0.34	11.03	4.29	3.94	2.51
Network times from TDE → VNOC → TDE	0.00	4.00	0.14	0.04	0.64
Processing times from TDE → VNOC → TDE	0.29	10.98	4.15	3.85	2.45
Request-response times from VNOC → TDE → VNOC	0.02	0.41	0.12	0.07	0.11
Network times from VNOC → TDE → VNOC	0.01	0.08	0.05	0.07	0.02
Processing times from VNOC → TDE → VNOC	0.01	0.38	0.07	0.01	0.10

TABLE IV
SUMMARY OF MESSAGE-SIZE STATISTICS

Description	Min./bytes	Max./bytes	Mean/bytes	Median/bytes	Std. Dev./bytes
TDE → VNOC	846	1278	874.7	848	96.8
VNOC → TDE	968	975	971.5	971	2.6
VNOC → MRN	650	1036	690.8	650	101.5
MRN → VNOC	799	1560	1419.2	1536	237.1

the total time taken from event occurrence on the train to decision-maker notification on a mobile phone. Then, $\tau = E_1 + E_{2,4} + E_3 + E_5$, and we use the results in [28] to show that $\Pr[\tau \leq 240 \text{ sec}] = 99.9\%$. These results indicate that the prototype TSSN can notify decision makers in a timely manner with very high probability.

G. Timing Analysis of Other TSSN Interactions

Table III summarizes request–response, processing, and network time statistics for interaction between various TSSN components. The statistics on the VNOC → MRN → VNOC interaction allow us to draw conclusions on request–response and processing times for certain (start or stop monitoring at the MRN and get current MRN location) VNOC commands. The TDE → VNOC → TDE interaction statistics give us insight into the time taken to initiate and process commands to start or stop monitoring at the MRN, get the MRN's current location, or to process the SetAlarmSecure command. The VNOC forwards these commands to the MRN and returns the MRN response to the TDE. To the TDE, all the elapsed time from when the VNOC receives a message from the TDE until the VNOC sends a response is processing time at the VNOC, although part of that time is spent forwarding a response to the MRN and waiting for a response. Finally, the statistics on VNOC → TDE → VNOC interactions allow us to draw conclusions on request–response, processing, and network times for the TDE to store alarm messages and execute shipment queries. Both of these actions are carried out when the VNOC AlarmProcessor service is about to send an alarm to the VNOC AlarmReporting service. Note that there are no results for the MRN → VNOC → MRN interaction. This condition is due to two reasons: 1) The clock drift in the MRN prevents us from computing a one-way network delay, and 2) the MRN only generates response messages. As expected, there are no request messages that originate in the MRN that could be used in a log entry couple to calculate request–response or processing times.

H. Message Sizes

Table IV summarizes the message size statistics for all the messages exchanged in the TSSN. Message size data are needed for a model [26] that is under development to determine system tradeoffs and the optimal or near-optimal sensor locations when using a rail-borne cargo monitoring system. The cost of transmitting a message from the train to an operations center is one component of this model. This transmission cost, in turn, depends on the average message length transmitted from the train and the frequency at which these messages are generated.

VI. REFINEMENTS BASED ON EXPERIMENTAL RESULTS

This section proposes refinements to the TSSN based on experimental results. Recall from Section III-B that we have corrected the clock drift problem by using a high-performance GPS receiver to get high-quality local time on the TSSN collector node. In addition, postprocessing of the log files also indicated that a unique identifier—perhaps composed of a timestamp and counter—is needed in the *Alert*, *MRN_Alarm*, and *NOC_Alarm* messages to trace an *Alert* message through the TSSN. This identifier can also be used in the future to locate *MRN_Alarm* messages that need to be retransmitted to the VNOC following a loss of connectivity. Finally, the identifier

can be used to mark previously processed messages so that the VNOC does not process the same message more than once.

Additional TSSN enhancements include the following approaches:

- redesigning the MRN hardware so that the TSSN collector node has redundant backhaul communication capabilities, e.g., multiple satellite and cellular modems, each with a different provider;
- creating a comprehensive security framework for the TSSN (ongoing research addresses this issue [29]);
- enhancing sensor capabilities so that sensors can engage in multihop communications to enable whole-train monitoring.

The desired result of this paper is a standards-based open environment for cargo monitoring with low-entry barriers to enable broader access by stakeholders while showing a path to commercialization.

VII. Related Work

In this section, we provide a brief overview of related research to monitor trains and to secure cargo in motion. In 2005, Edwards *et al.* [30] presented a prototype system for monitoring and controlling various sensors and actuators on a freight train. The prototype uses a controller area network (CAN) bus to collect data from the sensors. The data are then coupled with GPS information and reported to a web server through a code-division multiple access (CDMA)-based transmitter. Edwards *et al.* [30] argue that on-board sensing of mechanical defects enables car owners to track defects and proactively schedule maintenance at a time and location that makes economic sense.

The Transf-ID system [31], which was proposed in 2009, uses radio frequency identification (RFID) tags and an SOA to track cargo, railcars, and frequently serviced parts. The authors in [31] argue that the use of the Transf-ID system improves rail freight safety, because part maintenance schedules are now based on actual use.

In 2007, Lauf and Sauff [32] proposed a security protocol for transmitting information from sensors within a shipping container to a trusted third party. Such a protocol permits tracing liability for cargo theft and damage while minimizing the risk that shipping containers can be used for terrorism or shipment of contraband. The protocol was successfully deployed to test hardware; however, additional research is needed to create tamper-resistant units for monitoring container security [32]. In addition, in 2007, Ruiz-Garcia *et al.* [33] argued that the technology for developing a monitoring system for refrigerated containers already exists. They added that sensor readings and GPS information can be combined to track a shipping container through different stages of the supply chain.

The review of related work presented in this section shows that other researchers have monitored train equipment using an SOA and developed security protocols to communicate with sensors inside shipping containers. To the best of our knowledge, the TSSN is the first effort that uses sensors and an open SOA to monitor freight in motion.

VIII. Conclusion

In this paper, we have presented results from field trials of the prototype TSSN. The TSSN is an open system where different vendors can supply different components of the system. Within the TSSN framework, we have successfully combined sensor and shipment information to provide event notification to distributed decision makers. This paper has shown results that document the interactions between the different components of the TSSN. Based on our experiments and evaluations with the prototype, the TSSN architecture is viable for monitoring rail-borne cargo. We have successfully demonstrated that alert messages can be sent from a moving train to the VNOC and combined with cargo information that is forwarded to geographically distributed decision makers using either SMS or e-mail. Furthermore, based on the experiments reported here, we detected events and notified decision makers in just more than 1 min. Thus, we conclude that the TSSN architecture provides a mechanism for timely notification of decision makers. However, additional development and testing are needed before the TSSN architecture can be deployed in production systems.

Acknowledgment

The authors would like to thank A. Francis for reading and commenting on previous versions of this paper, Kansas City Southern Railway for their participation in the short-haul rail trial, and L. Sackman of EDS, an HP company, for assisting with the short-haul rail trial.

References

[1] Cargo Theft's High Cost—Headline, Fed. Bur. Investigation, Washington, DC, Jul. 21, 2006. [Online]. Available: http://www.fbi.gov/page2/july06/cargo_theft072106.htm

[2] Organisation Economic Cooperation Develop., Paris, France, Eur. Conf. Ministers Transport Container Transport Security Across Modes, 2005.

[3] Reference Model for Service-Oriented Architecture 1.0, Oct. 12, 2006, [Online]. Available: http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf

[4] Trade Data Exchange—Nothing Short of a Logistics Revolution, KC SmartPort, Nov. 10, 2008. [Online]. Available: http://www.joc-digital.com/joc/20081110/?pg=29

[5] Private Communication, 2007.

[6] A. Arsanjani, J. Martin, P. Tarr, and B. Hailpern, "Web services: Promises and compromises," *Queue*, vol. 1, no. 1, pp. 48–58, Mar. 2003.

[7] H. Saiedian and S. Mulkey, "Performance evaluation of eventing web services in real-time applications," *IEEE Commun. Mag.*, vol. 46, no. 3, pp. 106–111, Mar. 2008.

[8] J. Brown, B. Shipman, and R. Vetter, "SMS: The short message service," *Computer*, vol. 40, no. 12, pp. 106–110, Dec. 2007.

[9] D. T. Fokum, V. S. Frost, D. DePardo, M. Kuehnhausen, A. N. Oguna, L. S. Searl, E. Komp, M. Zeets, D. Deavours, J. B. Evans, and G. J. Minden, "Experiences from a transportation security sensor network field trial," in *Proc. 3rd IEEE Workshop EFSOI: Towards Socially Aware Netw.*, Honolulu, HI, Dec. 2009, pp. 1–6.

[10] M. Kuehnhausen, "Service-oriented architecture for monitoring cargo in motion along trusted corridors," M.S. thesis, Univ. Kansas, Lawrence, KS, Jul. 2009.

[11] R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana, Web Services Description Language (WSDL) Ver. 2.0 Part 1: Core Language, Jun. 2007. [Online]. Available: http://www.w3.org/TR/wsdl20

[12] M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, H. F. Nielsen, A. Karmarkar, and Y. Lafon, SOAP Ver. 1.2 Part 1: Messaging Framework (Second ed.), Apr. 2007. [Online]. Available: http://www.w3.org/TR/soap12-part1/

[13] D. Box, E. Christensen, F. Curbera, D. Ferguson, J. Frey, M. Hadley, C. Kaler, D. Langworthy, F. Leymann, B. Lovering, S. Lucco, S. Millet, N. Mukhi, M. Nottingham, D. Orchard, J. Shewchuk, E. Sindambiwe, T. Storey, S. Weerawarana, and S. Winkler, Web Services Addressing

(WS-Addressing), Aug. 10 2004. [Online]. Available: http://www.w3.org/Submission/ws-addressing/
[14] Web Services Security: SOAP Message Security 1.0, Mar. 2004. [Online]. Available: http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf
[15] D. Box, L. F. Cabrera, C. Critchley, F. Curbera, D. Ferguson, S. Graham, D. Hull, G. Kakivaya, A. Lewis, B. Lovering, P. Niblett, D. Orchard, S. Samdarshi, J. Schlimmer, I. Sedukhin, J. Shewchuk, S. Weerawarana, and D. Wortendyke, Web Services Eventing (WS-Eventing), Mar. 2006. [Online]. Available: http://www.w3.org/Submission/WS-Eventing/
[16] Apache Axis2 Project Documentation, The Apache Software Foundation, Aug. 24, 2008. [Online]. Available: http://ws.apache.org/axis2/
[17] D. Mulvey, "HSPA," *Commun. Eng.*, vol. 5, no. 1, pp. 38–41, Feb./Mar. 2007.
[18] C. E. Fossa, R. A. Raines, G. H. Gunsch, and M. A. Temple, "An overview of the IRIDIUM (R) low-Earth-orbit (LEO) satellite system," in *Proc. IEEE NAECON*, Dayton, OH, Jul. 1998, pp. 152–159.
[19] Hi-G-Tek, Hi-G-Tek, Rockville, MD, Mar. 17, 2009. [Online]. Available: http://www.higtek.com/
[20] Apache log4j Project Documentation, The Apache Software Foundation, Sep. 1, 2007. [Online]. Available: http://logging.apache.org/log4j/
[21] Esper: Complex Event Processing—Project Documentation, EsperTech, Wayne, NJ, Feb. 11, 2009. [Online]. Available: http://esper.codehaus.org/
[22] Hi-G-Tek Ltd., Rockville, MD, DataReader and DataSeal: User's Manual, UM4710, 2001.
[23] Google Maps: Web Mapping Service, Google, May 6, 2009. [Online]. Available: http://maps.google.com
[24] D. L. Mills, "Internet time synchronization: The network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.
[25] J. Clark and S. DeRose, XML Path Language (XPath), Nov. 16, 1999. [Online]. Available: http://www.w3.org/TR/xpath
[26] D. T. Fokum, Optimal communications systems and network design for cargo monitoring, 10th Workshop Mobile Comput. Syst. Appl., Santa Cruz, CA, Feb. 2009.
[27] D. T. Fokum, V. S. Frost, D. DePardo, M. Kuehnhausen, A. N. Oguna, L. S. Searl, E. Komp, M. Zeets, D. D. Deavours, J. B. Evans, and G. J. Minden, "Experiences from a transportation security sensor network field trial," Univ. Kansas, Lawrence, KS, ITTC Tech. Rep. ITTC-FY2009-TR-41420-11, Jun. 2009.
[28] S. Nadarajah, "A review of results on sums of random variables," *Acta Applicandae Mathematicae*, vol. 103, no. 2, pp. 131–140, Sep. 2008.
[29] E. Komp, V. Frost, and M. Kuehnhausen, "Implementing web services: Conflicts between security features and publish/subscribe communication protocols," Univ. Kansas, Lawrence, KS, ITTC Tech. Rep. ITTC-FY2010-TR-41420-19, Feb. 2010.
[30] M. C. Edwards, J. Donelson, III, W. M. Zavis, A. Prabhakaran, D. C. Brabb, and A. S. Jackson, "Improving freight rail safety with on-board monitoring and control systems," in *Proc. ASME/IEEE Joint Rail Conf.*, Pueblo, CO, Mar. 2005, pp. 117–122.
[31] J. Fernandez, J. C. Y. Garcia, Y.-S. M. Garcia, and J. Santos, "Transf-ID: Automatic ID and data capture for rail freight asset management," *IEEE Internet Comput.*, vol. 13, no. 1, pp. 22–30, Jan./Feb. 2009.
[32] J. O. Lauf and H. Sauff, "Secure lightweight tunnel for monitoring transport containers," in *Proc. 3rd SecureComm*, Nice, France, Sep. 2007, pp. 484–493.
[33] L. Ruiz-Garcia, P. Barreiro, J. Rodriguez-Bermejo, and J. I. Robla, "Review: Monitoring the intermodal, refrigerated transport of fruit using sensor networks," *Spanish J. Agricultural Res.*, vol. 5, no. 2, pp. 142–156, 2007.

Daniel T. Fokum (S'07) received the B.A. degree in computer science from Park University, Parkville, MO, in 2000 and the M.S. degree in computer science from the University of Missouri, Kansas City, in 2005.

He is currently a Graduate Research Assistant with the Information and Telecommunication Technology Center while pursuing the Ph.D. degree in computer science with the Department of Electrical Engineering and Computer Science, University of Kansas (KU), Lawrence. Prior to joining KU in 2006, he worked in industry for six years. His research interests include sensor and wireless networks, information security, and concurrence control in databases.

Mr. Fokum is a member of the IEEE Computer Society, the IEEE Communication Society, and the Association for Computing Machinery.

Victor S. Frost (S'75–M'82–SM'90–F'98) received the B.S., M.S., and Ph.D. degrees from the University of Kansas (KU), Lawrence, in 1977, 1978, and 1982, respectively.

In 1982, he joined the faculty of KU and was the Dan F. Servey Distinguished Professor of Electrical Engineering and Computer Science. He was the Director of the KU Telecommunications and Information Technology Center (ITTC) for more than ten years. From 1987 to 1996, he was the Director of the KU Telecommunications and Information Sciences Laboratory. He is currently a Program Director with the Computer and Network Systems (CNS) Division, Computer and Information Science and Engineering (CISE) Directorate, National Science Foundation (NSF). He is currently the Area Editor for Communications Simulation of the *ACM Transactions on Simulation and Modeling of Computer Systems*. He has been involved in research on several national-scale high-speed wide-area testbeds. He was an Investigator on a gigabit testbed (MAGIC) research effort and the ACTS ATM Internetwork. His research has been sponsored by government agencies, including the NSF, the Defense Advanced Research Projects Agency, the Rome Laboratory, and the National Aeronautics and Space Administration. He has been involved in research for numerous corporations, including Sprint, NCR, Nortel, Telesat Canada, AT&T, McDonnell Douglas, DEC, and COMDISCO Systems. He has been a Principal Investigator on more than 35 research efforts and has been involved as a Coinvestigator on more than 40 projects. As a result of those efforts, he has published more than 100 journal articles and conference proceedings. His current research interests include communications systems and networks, networking testbeds, Internet quality of service, traffic management, and integrated broadband communication networks.

Dr. Frost received the Presidential Young Investigator Award from the NSF in 1984. He was listed in the Kansas City Star Tech 50 in 2000 and 2002. He is a Member-at-Large of the IEEE Communications Society Board of Governors for the term 2008–2011.

Martin Kuehnhausen (S'10) received the Dipl.Inf. (B.A.) degree from the Berufsakademie Stuttgart, Stuttgart, Germany, in 2006 and the M.S. degree from the University of Kansas (KU), Lawrence, in 2009. He is currently working toward the Ph.D. degree in computer science with the Department of Electrical Engineering and Computer Science, KU.

From 2003 to 2007, he worked on various projects with IBM. He was part of the Secure Trade Lane Team, creating an enterprise real-time container-tracking solution. He is currently working on the SensorNet Project with the Information and Telecommunication Technology Center, KU. His research interests include intelligent systems, information and data management, web service technologies, and software engineering.

Mr. Kuehnhausen is a member of the IEEE Computer Society, the IEEE Communications Society, and the IEEE Intelligent Transportation Systems Society.

Daniel DePardo (SM'00) is a graduate of the U.S. Army Intelligence Center School.

He is currently a Research Engineer with the Information and Telecommunication Technology Center (ITTC), University of Kansas (KU), Lawrence, and primarily supports the electronic hardware needs of ITTC laboratories. He has extensive test and measurement, hardware design, and prototype fabrication experience, and his research interests include radio transceiver and antenna design.

Mr. DePardo is a KU Staff Fellow.

Angela N. Oguna (S'09) is working toward the B.S. degree in electrical engineering with the University of Kansas (KU), Lawrence. For the last two years, she has been an undergraduate Research Assistant with the KU Information and Telecommunication Technology Center.

She presented her work at the 2009 KU Undergraduate Research Symposium and at the 2010 IEEE Region V Student Paper Contest. Her research interests include sensor networks and integrated communications for smart grid technology.

Ms. Oguna is a Student Member of the IEEE Power Engineering Society.

Leon S. Searl received the B.S. and M.S. degrees in electrical engineering from the University of Kansas, Lawrence, in 1985 and 1987, respectively.

He worked for 12 years in private industry, architecting and developing software for communication system simulation, satellite test set control, and electronic design automation tools. Since 2000, he has been a Research Engineer with the Information and Telecommunication Technology Center, University of Kansas, designing and developing hardware and software for research projects in dynamically configurable radios and radio networks, ambient computing, space-based computer networks, and wireless sensor networks.

Edward Komp received the B.A. degree in mathematics and the M.S. degree in computer science from the University of Kansas (KU), Lawrence, in 1976 and 1979, respectively.

After more than 15 years of designing, implementing, and managing commercial software development, he joined the research staff of the KU Information and Telecommunications Technology Center, where he is currently a Research Engineer. His primary research interests include specialized computer language design for application-specific domains, functional programming, software development environments, and networking.

Matthew Zeets received the B.S. degree in computer science from the University of Kansas (KU), Lawrence, in 2008, where he is working toward the M.S. degree in computer science.

In 2009 and 2010, he was a Graduate Teaching Assistant for a programming class. In 2008 and 2009, he was a Graduate Research Assistant with the Information and Telecommunication Technology Center. From 2006 to 2008, he was the President of the Association for Computing Machinery Chapter at KU. His research interests include information security and the use of Web 2.0 technologies in the supply chain industry.

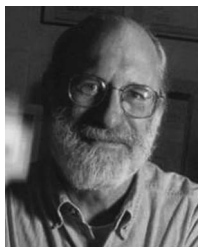
Daniel D. Deavours (M'01) received the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in electrical engineering from the University of Illinois at Urbana–Champaign.

In August 2001, he joined the University of Kansas (KU), Lawrence, where he is currently a Research Associate Professor. He has worked with the Bluetooth SIG in developing a Bluetooth interoperability test program, started and directs the RFID Alliance Laboratory, and has been active in the SensorNet Initiative. His primary research interests include developing rigorous methods for analyzing and developing radio frequency identification antennas, particularly microstrip antennas. He is the holder of three patents. He has published more than 20 technical papers and five journal articles.

Joseph B. Evans (SM'01) received the B.S.E.E. degree from Lafayette College, Easton, PA, in 1983 and the M.S.E., M.A., and Ph.D. degrees from Princeton University, Princeton, NJ, in 1984, 1986, and 1989, respectively.

He is the Deane E. Ackers Distinguished Professor of Electrical Engineering and Computer Science with the University of Kansas (KU), Lawrence. From 2008 to 2010, he was the Director of the KU Information and Telecommunication Technology Center, and from 2005 to 2008, he was the Director of Research Information Technology. He is currently on a partial leave of absence from KU to perform research on TIGR, which is a tactical information system that he helped develop and has extensively been deployed in Iraq and Afghanistan for the Defense Advanced Research Projects Agency and the U.S. Army. From 2003 to 2005, he was a Program Director with the National Science Foundation. He has been a Researcher with the Olivetti and Oracle Research Laboratory, Cambridge University Computer Laboratory, the United States Air Force Rome Laboratories, and AT&T Bell Laboratories. He has co-founded several technology companies, including a network gaming company acquired by Microsoft in 2000 and a defense-oriented venture recently acquired by General Dynamics. His research interests include cognitive radio networking, spectrum technology and policy, sensor networking, adaptive systems, and network testbeds.

Dr. Evans is currently a member of the IEEE Communications Society Board of Governors.



Gary J. Minden (S'73–M'81–SM'97) received the B.S. and Ph.D. degrees in electrical engineering from the University of Kansas (KU), Lawrence, in 1973 and 1982, respectively.

From 1978 to 1980, he was a Vice President of CHILD, Inc., where he was a codesigner of the LIGHT-50 computer graphic terminal. In 1981, he joined the Faculty of Electrical Engineering, KU, where he led the implementation of a new computer engineering program. In 1991, he completed a sabbatical with Digital's System Research Center, working on gigabit local area networks. From June 1994 to December 1996, he was a Program Manager with the Information Technology Office, Defense Advanced Research Projects Agency, working on high-performance networking systems. He is currently a Professor of electrical engineering and computer science with KU. He initiated a new research program in active networking. He has led several research projects in high-performance wide-area networks, mobile wireless systems, adaptive computational systems, and innovative networking protocols. He has served on three Defense Science Board Task Forces: 1) Tactical Battlefield Communications; 2) Spectrum Management, and 3) the Wideband RF Modulation task force, for which he was a Chair. He has served on a National Research Council (NRC) review panel for the Army Research Laboratory, has contributed to several DDR and E review panels, and has contributed to an NRC report on future tactical radio systems. His research interests include large-scale distributed systems that encompass high-performance networks, mobile wireless networks, software-defined radios, computing systems, and distributed software systems.

Dr. Minden is a member of the Association for Computing Machinery and the American Association for the Advancement of Science.