Interoperation of Organizational Data, Rules, Processes and Services for Achieving Inter-Organizational Coordination and Collaboration

Stanley Y. W. Su
Xuelian Xiao
Jeff DePree
Dept. of Computer & Information Science & Engineering
University of Florida
su@cise.ufl.edu

Howard W. Beck
Dept. of Agriculture & Biological Engineering
University of Florida
hwb@ufl.edu

Andrew Coggeshall
Richard Bostock
Dept. of Plant Pathology
University of California, Davis
cthomas@ucdavis.edu

Interoperation of Organizational Data, Rules, Processes and Services for Achieving Inter-Organizational Coordination and Collaboration

Abstract

Solutions to many complex problems that government organizations all over the world face today require these organizations to share, not only data and computing resources, but also policies, constraints, regulations, processes and services in order to achieve inter-organizational coordination and collaboration. This paper presents an integrated specification language and a user interface for collaborating government organizations to specify events of common interest, policies, constraints and regulations in the form of different types of knowledge rules, manual and automated services, and sharable workflow processes. A network system infrastructure for dynamic processing and interoperation of distributed rules and processes, and an event-triggered rule processing and process enactment technique are also described.

1. Introduction

Government organizations worldwide are facing many complex problems such as disease detection and control, border control, immigration, disaster assistance and recovery, and terrorism. The solutions to these and other problems require such organizations to collaborate, share their resources, and coordinate their activities in a timely manner. Resource sharing should not be restricted to the sharing of data and computing resources. The sharing and interoperation of organizational policies, constraints, regulations, processes, manual services and automated services are also important for achieving inter-organizational collaboration and coordination. Traditionally, organizational policies, constraints and regulations have been specified in terms of different types of business rules [15], such as logic-based derivation rules, condition-action rules and constraint rules. Manual services, automated services and processes are specified using a workflow process definition language [17]. Data, business rules and workflow processes are managed by different types of software systems such as database management systems, rule processing systems and workflow management systems. These systems do not interoperate; nor do the data, rules, services and processes they manage. There is a need for a high-level specification facility for collaborating organizations to independently define their sharable data, business rules, services and workflow processes, and a way to make these resources interoperate.

In our work, we have developed a user-friendly interface installed at the network sites of all the organizations of a collaborative federation to define the sharable rules, events, manual and automated services, and workflow processes. Distinct from a traditional workflow specification, a workflow process in our work can be defined as a structure of activities that invoke different types of rules, manual services and automated services. A condition-action rule can enact a workflow process, which may contain rules to enact other processes, making rules and processes interoperable. Rules and workflow processes defined through the user interface tool are formally specified in an integrated rule and process specification language for further processing. The specified rules and processes are first translated into code automatically and then packaged as Web services for their uniform discovery, processing, sharing and interoperation in a Web service infrastructure. There are two main advantages with the approach we take. First, a government organization no longer has to treat rules that capture organizational policies, constraints, and regulations, and workflow processes as separate entities. It can define them in an integrated manner, and the defined rules and processes can interoperate. Second, rules and processes do not
have to be separately processed by a rule processing system and a workflow management system, which are costly to purchase, install and maintain. They can be uniformly processed as Web services.

Another important issue that needs to be addressed is the dynamic nature of e-government. Government organizations' data, computing resources, policies, constraints, regulations, services and processes can change at any time. This means that the rules and workflow processes which capture these resources can be modified by these organizations at any time. New rules and processes can also be independently added by any of these organizations. A system that makes use of these rules and processes to achieve inter-organizational collaboration and coordination needs to be agile, capable of reacting and adapting to changes. Inter-organizational workflow processes should not be pre-defined because, once defined, they are very difficult to change. Rather, they should be dynamically constructed and run to reflect and adapt to the changing nature of organizations' resources and services. There is a need for an infrastructure and a processing technique to process these organizational rules and processes.

In our work, we allow collaborating organizations to define different types of events through the user interface. An event is anything of common interest to these organizations, such as the announcement of a government program or procedure, a state of a database, a real world incident, a signal from a sensor, or a request from an individual. Like the Web services generated from rules and processes, events are registered in a registry at a host site. Thus, rules, processes and events are searchable and accessible by users, and by the network system presented in this paper. We further allow users to subscribe to events, and define triggers, which link events to rules. When an event occurs (i.e., an event instance), data relevant to that event instance (i.e., the event data) is transferred to all the subscribers through a notification mechanism, and relevant rules installed at various network sites as Web services are activated. These distributed rules may in turn enact distributed processes that contain rules and services. The data produced by rules and services may in turn activate other rules and processes. There are two main advantages with the above event-triggered rule processing and process enactment technique. First, instead of transferring a query to different network sites for processing against their databases (the approach taken by traditional distributed database management systems), data sharing is achieved in our system by transferring only those data that are relevant to an event instance. The collaborating organizations do not have to open their entire databases to other organizations' users; thus, the privacy and security of their databases can be more easily protected. Second, most up-to-date rules and processes are activated at run-time. Inter-organizational processes are dynamically constructed and processed, and event data, rules, services and processes can interoperate.

This paper is organized as follows. Section 2 acknowledges some related works and points out the main differences between them and our work. Section 3 presents the integrated rule and process specification language and the user interface. Section 4 presents the architecture of our distributed network system and the event-triggered rule processing and process enactment technique. Section 5 covers the application of the developed technology and some scenario tests and results. Section 6 summarizes the key features of our language, user interface, processing technique and network system.

2. Related works

In the e-government area, several works on inter-organizational collaboration deal with the sharing of data/information resources. For example, [14] addresses research issues related to the integration of heterogeneous databases and the mappings between a virtual global database and various data sources. The work reported in [11] deals with a study on the causes and effects of adopting information sharing systems among counter-terrorism and disaster management agencies. [10] looks at the issue of information quality in the context of cross-agency information sharing. The sharing and interoperability of organizational processes has gained a lot of attention in the e-government area. Peristeras, et al. [13] gives an extensive review of existing research efforts in modeling data, processes and services, and organization integration for achieving e-government interoperability. The article points out the importance of orchestration and execution of complex processes that consist of a number of autonomous Web services offered by different organizations, which is one of the focuses of our paper. Charalabidis, et al. [4] proposed an approach of extending the functionality of interoperability registries through the use of business process management and XML authoring tools for supporting service composition, automated execution and optimization. Unlike ours, this work does not focus on representation of organizational knowledge using different types of rules and sharing these rules among business and government organizations. Overbeek, et al. [12] presented an event-driven, service oriented architecture for orchestrating service delivery. There are
rules and processes defined by collaborating triggered processing and interoperation of distributed specifications. Our system enables the event-inclusion of these rules in workflow process specification of different types of rules and the specification language allows for the integrated from CrossWork and IBM’s contribution, our multi-organizational business processes. Distinct system for modeling and controlling the execution of \( WEM \) and a dynamic workflow management effort in developing a Workflow Extension Model (WEM). The IBM research report \([1]\) presents IBM’s research selecting some collaborating partners in the market. operational business goals, which are fulfilled by system, a high-level goal is decomposed into processes among collaborating organizations. In this the composition, setup and enactment of business infrastructure, CrossWork \([9]\) provides support for domain have widely adopted workflow technology. CrossFlow \([8]\) uses an electronic contract as the basis for inter-organization collaboration. \([2]\) provides an agent-based middleware architecture for the composition and enactment of cross-organizational workflows. However, it requires that workflow designers first create a set of cross-organizational process models, which are used by agents for workflow enactment. Based on the agent-oriented and service-oriented technologies and the Internet infrastructure, CrossWork \([9]\) provides support for the composition, setup and enactment of business processes among collaborating organizations. In this system, a high-level goal is decomposed into operational business goals, which are fulfilled by selecting some collaborating partners in the market. The IBM research report \([1]\) presents IBM’s research effort in developing a Workflow Extension Model (WEM) and a dynamic workflow management system for modeling and controlling the execution of multi-organizational business processes. Distinct from CrossWork and IBM’s contribution, our specification language allows for the integrated specification of different types of rules and the inclusion of these rules in workflow process specifications. Our system enables the event-triggered processing and interoperation of distributed rules and processes defined by collaborating organizations. Inter-organizational workflow processes are not predefined at build-time, but are dynamically constructed and enacted at runtime.

The work reported here is the continuation of our R&D efforts reported in \([18]\). A major departure from our earlier work is the inclusion of different types of rules in workflow process specifications. In our earlier work, we separate rule structure from operation structure. The former has a very limited number of structural constructs used to specify the sequential and parallel execution and synchronization of rules. Whereas the latter is used to define a workflow process and has an extensive set of constructs for specifying the structural relationships of manual and automated operations. By including rules in a workflow process specification, we allow 1) workflow processes to be flexibly defined at a high-level by structures of rules, services, or a mixture of both, using a full set of structural constructs found in traditional workflow and business process specifications, and 2) the interoperation of rules, services and processes as well as the interoperation of different types of rules.

### 3. Integrated knowledge rule and process specification language and user interface

The language and the user interface presented in this section, and the network system and the processing technique to be presented in the next section are being developed for deployment in USDA’s National Plant Diagnostics Network (NPDN) environment. To demonstrate the utility of the developed knowledge and process sharing technology, the research team at the University of Florida and the collaborators at the University of California, Davis are using it to implement the NPDN Chain of Custody and Chain of Communication Standard Operating Procedure (SOP) defined by NPDN organizations. The developed technology will enable these organizations to effectively share their event data, knowledge rules, and processes, provide a timely exchange of information about any disease or pest outbreak, and receive assistance on appropriate emergency responses, thus establishing better communication, coordination and collaboration among them. It will strengthen the security of US agriculture. The developed language, user interface tool and network system have been successfully tested by the research team as well as with the NPDN research partners and members. The rule and process examples given in this paper are taken from the implementation of the SOP. Details on the application and test results will be given in Section 5.
3.1. Specification language

A knowledge rule is a declarative statement that specifies a governmental (and inter-governmental) policy, constraint or regulation. There are three popular types of knowledge rules that are commonly used in business applications. Integrity constraint rules specify the condition(s) to which some data must adhere as required by an application. Derivation rules are also known as inference rules or deductive rules. These infer new knowledge based on current knowledge. Action-oriented rules define the action that an organization must perform under certain conditions. These are commonly used in an event-condition-action (ECA) system. In our work, the action clause of an action-oriented rule can enact a governmental workflow process. We further extend the traditional workflow process specification by including not only manual services and automated services, but also the three types of knowledge rules. Thus, rules of different kinds as well as workflow processes can interoperate; for example, an action-oriented rule may enact a process, which may invoke a derivation rule to derive some data that can serve as the input to another action-oriented rule or service.

3.1.1. Integrity constraint. The enforcement of an integrity constraint ensures that any change to data does not break the data consistency requirement imposed by an application. Since we use an object-oriented model to model the event data exchanged between organizations, a constraint on the event data can be specified either by the constraints on the values of an object’s attributes (i.e., attribute constraint) or the relationship between attributes (i.e., inter-attribute constraint). The former limits the acceptable values that an attribute may have. Examples for this type of rule are *Shipping.estShipTime < 2* and *Sample.class in {suspected, positive, negative}*. The latter states the relationship between attributes modeled either in a mathematical expression (e.g., *Result.confDiagDate – General.estConfirmDate < 3*) or a conditional expression (e.g., *if Sample.class = ‘positive’ then Shipping.method is ‘UPS’*).

We define the input of a constraint rule as all of the attributes referenced in the rule. The output is the truth value indicating whether the constraint is satisfied or not. The input and output information of this and other rules as well as processes are used by the system to determine the applicability of rules and processes to an event occurrence (see Section 3.1.5).

3.1.2. Derivation rule. A derivation rule produces new data if some preconditions on existing data are satisfied. This kind of rule has the following form: \( P \rightarrow Q \), which means that, given that the head of the implication \( P \) is true, the body or conclusion \( Q \) is also true. One example of this rule type is \( (Lab.secLevel = 2) \rightarrow (Sample.status="destroyed") \).

The attributes of objects referenced in \( P \) are this rule’s input, and those referenced in \( Q \) are its output.

3.1.3. Action-oriented rule. An action-oriented rule is generally expressed in the form of event-condition-action (ECA) rules, which state that when an event \( E \) occurs, the action \( A \) should be executed if the condition \( C \) is satisfied. In our work, we separate the event specification from the CA rule specification to allow events and CA rules to be independently defined by different organizations. As mentioned in the introduction section, an event defined by one organization can be linked to the rules defined by the same or other organizations using triggers (see Section 3.1.5 for more details). We also use the format of condition-action-alternative-action (CAA) to specify action-oriented rules. The action part of our action-oriented rule is not limited to a single rule or service as shown in the following example: *Condition: none; Action: PrelimSampleDiag*, which unconditionally performs a preliminary diagnosis on a sample. It is extended to include the enactment of a workflow process (See Section 3.1.4 for details).

The input of a CAA rule is defined as the union of the attributes of objects referenced in the condition part, the inputs of the rule, service or process in the action clause, and the inputs of the rule or service or process in the alternative action clause. The output of a CAA rule is the union of the outputs of the rule, service or process referenced in the action clause, and the outputs of the rule, service or process referenced in the alternative action clause.

3.1.4. Workflow process. A workflow process is modeled as a structure of activities, each of which can invoke a rule, manual service or automated service. It is enacted by a CAA rule. A manual service is assigned by the network system to users who play a specific role. Those users will receive notifications with instructions about the assignment. One of them can choose to perform the service. After performing the service, this user needs to log on to the system to report its completion and provide the data resulting from the service. An automated service is executed automatically by the system. It can be a Web service invocation, a call to a local program procedure or application operation, an activation of a rule, or a posting of an event to indicate its occurrence. By integrating the process patterns defined in WPDL [17] and BPEL [3], we include the following constructs in the specification of a process:
sequential, switched, unordered, selective, repeated, split (AND, OR and XOR), join (AND and OR). These constructs are well-known in the workflow community and are thus not explained here.

A process can thus be viewed as a directed graph, with knowledge rules, automated and/or manual services as its nodes, which are connected by the constructs mentioned above. These rules and services can be defined by the same organization that defines the process or by different organizations. The inputs of a process are the union of the inputs of its rules and services (i.e., data needed to process the rules and services), and its output is the union of the output of these rules and services (i.e., data produced by the rules and services).

Figure 1 shows an example process. In the example, we avoid using the identities of the labs and persons involved in the development of the National Plant Diagnosis Network (NPDN) for confidentiality reasons. In this example, a sample is submitted to Lab A for diagnosis. A person in Lab A would first acknowledge the receipt of the sample and then assign it a unique id. A diagnostician would manually conduct a diagnostic procedure on the submitted sample depending on whether it is a plant or insect sample. Based on the diagnosis result, the sample is classified. If it is classified as presumptive positive or positive and is of suspect regulatory significance, then the need for further confirmation is set to true and the security level of the sample is set to 3 by a derivation rule. After that, three parallel activities that invoke three condition-action rules would take place as specified in an AND-split construct. The first one unconditionally notifies the State Plant Health Director and the Campus Safety Officer. The second one stores the sample in a place appropriate for its security level. The third one requests that another lab conduct a confirming diagnosis on the sample. As shown in the example, rules and services can be intermingled to flexibly define a process.

3.1.5. Trigger. A trigger is a specification, which conditionally links an event to a rule of any type. It states that if the event data associated with an event occurrence satisfy a certain data condition, the specified rule should be processed. An organization can explicitly define a trigger to link a local event to a rule defined by itself or by some other organization. It can also define a trigger to link a global event defined by another organization with a local rule. Triggers defined by different organizations can link the same event to different rules. The occurrence of an event will invoke all local and global rules. The above approach allows for the maximal sharing of events and rules.

Defined triggers are called explicit triggers and the organizations that define the rules are explicit subscribers. Triggers can also be automatically and dynamically generated by the system. If the event data associated with an event occurrence is a superset of the input data of a rule, we regard this rule as applicable to the event. The processing of an applicable rule may change the event data or produce new data, thus adding new information to the event data. System-defined triggers are called implicit triggers and the organizations that have applicable rules are implicit subscribers. When an event occurs, the event data will be sent to all explicit and implicit subscribers’ sites to trigger the processing of their applicable rules.

3.2. User interface

To make it possible for collaborating organizations to define events, rules, triggers, services and processes in an easy, straightforward manner, and to allow users within each organization to post events and carry out manual services, a comprehensive and intuitive interface has been developed. Each site has its own instance of the interface, but a global view of the system is available at every site, and users can examine shared specifications from any site. This software can be accessed from any web browser, without the need for the user to install plug-ins or other software. Depending on the access rights of each user and the current state of the system, only those features and notifications that are appropriate for a given role and phase are displayed, and all features are accompanied by a help system to explain their usage.
The system’s users are divided into administrators, knowledge engineers, and end users. The administrators define the structure of the collaborative federation, declaring which site will be the host and establishing the communication links between the host and the other sites. They also determine what roles users will fill in the system. The knowledge engineers at each site are responsible for using the interface to define events, rules, services, triggers and processes, which allow the system to react to events of interest, evaluate appropriate rules, and execute the workflow processes. These defined elements can be declared as either local to one site or global, meaning that they can be incorporated into the response mechanisms of any of the sites within the federation. End users then use the elements defined by the knowledge engineers to post events of interest and complete the manual services that have been assigned to them or to their roles. The interfaces for these latter two classes of users are described in more detail below.

3.2.1 Knowledge engineering interface. Often in a collaborative setting, something will happen at one site that may be of interest both to it and to the other sites in the federation. In order to capture and react to these occurrences, organizations define “events” that capture the time that an event of interest occurs, as well as any data associated with the occurrence. In order to prepare such an event, a knowledge engineer must specify the data attributes that are associated with the event, as well as a user-friendly description of the meaning of the event, and one or more roles that determine which users are capable of firing the event. Once an event is defined, it is registered with the host so that any interested sites can subscribe to it and be notified when it occurs.

Since there are three different types of rules, and an instance from any of these types can be of arbitrary complexity, the interface for defining rules changes at runtime to accommodate the rule that is being defined. All rules have a name, a user-friendly description, and flags for whether the rule is enabled and whether it should be shared with the other sites. What shows up in the rest of the interface is dependent on which type of rule is selected; when the user chooses CAA Rule, the interface allows the user to select a process for the primary action, and optionally an alternative action, along with a condition specification to determine which of these two actions should be executed. Each expression in the condition specification can include any number of sub-expressions or terms, and as each new piece is added, the interface opens a new section to allow the user to define that piece. When the user selects to save the rule, type checking is performed to make sure the rule is valid. Once defined, a rule is translated into a Web service and registered with the Web service registry at the host; if sharing is enabled, the rule may be accessed by any of the other sites in the federation and incorporated into their processes.

The interface also enables the knowledge engineer to define a process in the form of a diagram. He/she selects each defined service or rule and clicks on the diagram where it should be inserted. He/she can then add connections between the rules and services, select what types of splits or joins should be used, and specify conditions for the connections. The diagram will automatically expand to support a process of any size. As with rules, processes are translated into Web services and registered with the host; any of the collaborating sites may use event-triggered, action-oriented rules to enact the sharable processes.

3.2.2 End user interface. End users can perform two functions; they can either post events or perform manual services. Any event that has been assigned to a given user’s role can be posted at any time by that user; the user needs only to select the event from the list of available events and supply the appropriate data for the data objects that define the event data. A manual service can only be performed by a user when it has been assigned to the role that the user plays. Depending on the user’s preferences, he/she will be notified by email and/or text message of a service that fits his/her role, and be provided with a link to log into the system. Once logged in, the user will be presented with a list of assigned services that fit his/her role. He/she can then select the one that he/she wants to perform. Upon the selection, the system will reject other users' selection of the same service, and a list of tasks associated with the service will be presented to the user. When the boxes next to these tasks have been checked to indicate that all manual tasks have been completed, and all the data produced by these tasks have been supplied to the system, the user can signal to the system that the service has been completed.

For many applications, it is critical that a comprehensive audit trail be provided for any process. Because of this, the system records all events that are posted and services that have been carried out, along with the appropriate timestamps and accompanying data elements. In order to keep the end users apprised of the current state of the system, the interface maps this log data to an easy-to-follow flowchart which clearly shows what has been completed and what still remains to be completed.

There are times when a user has agreed to carry out a manual service, but is not able to complete one or more of its tasks in a reasonable amount of time. For this
reason, the system uses a time-out mechanism, which, after a specified interval, will send a reminder email to the user; the user may then log into the system and request more time to perform the service. However, if this is not done, the system will cancel the lock on the service and allow other users within the assigned role(s) to access it once again. Often, it is also appropriate to specify a backup role, so that if no one within the primary role elects to complete the assigned service in a reasonable amount of time, then an administrator can step up to fill the void and keep the process running.

4. System architecture and processing technique

4.1. System architecture

The ETKnet system has a peer-to-peer server architecture as shown in Figure 2. All of the collaborating organizations have identical software components installed. The software components include a User Interface, a Rule Server, an Event Server, and a database system. Each collaborating site defines its own events, event data, rules, services, processes, and triggers through the User Interface. The Rule Server component is responsible for managing the services, rules and processes defined at that site, and for invoking the applicable rules and processes after receiving an event notification with event data. The Event Server component is responsible for managing the events and triggers defined at that site, and acts as the coordinator in a knowledge sharing and collaboration session initiated by an event occurred at that site. One of the collaborating sites plays the role of the Host of the collaborative federation. The Host has an additional software component: a Web Service Registry. In a distributed, collaborative environment, services, knowledge rules and processes are defined independently by each collaborating site to express its organizational policies, regulations, constraints and processes. These are first converted into Java code, and then packaged and deployed as Web services. Their metadata, including name, input data items, output data items and WSDL file addresses are also stored on the local database system. If they are sharable, the metadata will then be registered in the WS Registry at the Host. The metadata of sharable events and triggers are also registered with the WS Registry.

We have implemented the ETknet system in the Java language and have hosted it on the Sun Application Server Platform Edition 9. The event and rule server software components are implemented using the Enterprise JavaBean 2.1 framework. We publish the Web services in a private registry at the Host. This registry makes use of a MySQL database to store the Web service information. The database is also used by the Rule Server and the Event Server to store the information about events, rules, processes, services and triggers.

![Figure 2. System architecture](image)

4.2. Event-triggered processing of rules and enactment of processes

The ETKnet system processes distributed rules and processes using an event-triggered mechanism. When an event occurs at a site, the data values assigned to the data objects and attributes that define the event form the event data, and the site becomes the coordinator of a knowledge sharing and collaboration session initiated by an event occurred at that site. One of the collaborating sites plays the role of the Host of the collaborative federation. The Host has an additional software component: a Web Service Registry. In a distributed, collaborative environment, services, knowledge rules and processes are defined independently by each collaborating site to express its organizational policies, regulations, constraints and processes. These are first converted into Java code, and then packaged and deployed as Web services. Their metadata, including name, input data items, output data items and WSDL file addresses are also stored on the local database system. If they are sharable, the metadata will then be registered in the WS Registry at the Host. The metadata of sharable events and triggers are also registered with the WS Registry.

We have implemented the ETKnet system in the Java language and have hosted it on the Sun Application Server Platform Edition 9. The event and rule server software components are implemented using the Enterprise JavaBean 2.1 framework. We publish the Web services in a private registry at the Host. This registry makes use of a MySQL database to store the Web service information. The database is also used by the Rule Server and the Event Server to store the information about events, rules, processes, services and triggers.

![Figure 2. System architecture](image)
updated event data to the coordinator. When the coordinator receives these data from the subscribers, it merges the returned data with the original event data to form a new version of the event data. The coordinator site’s rule that is linked to the same event can then be activated to process this new version. The new version of event data may cause some other rules at the same and/or different sites to become applicable. In that case, the coordinator will send the new version to all the explicit and implicit subscribers to start a new round of event notification, rule processing and process enactment. Multiple rounds can take place until no new or updated data have been generated and no applicable rules can be found. At that time, all sites would have received all the data that are relevant to the event occurrence, and processed all the applicable rules, services and processes. The knowledge sharing and collaboration session can then terminate.

The above technique achieves the interoperation of shared data, rules, services and processes without the need to set up different kinds of systems to process them. It also enables the dynamic construction and enactment of inter-organizational processes by run-time activation of the most up-to-date services, rules and processes independently defined by collaborating organizations. Furthermore, the technique also enables applicable rules, which are not explicitly linked to an event by triggers, to be activated. This is important because an organization may not know the existence of the event or may not know that the event data can serve as input to its rules to produce useful data and/or to enact relevant processes.

5. Application and evaluation

We have applied the developed technology in the management and processing of the NPDN Chain of Custody and Chain of Communication Standard Operating Procedure (SOP) defined by USDA’s National Plant Diagnostic Network. The procedure involves sample transfers, diagnoses, and communications as a sample of regulatory significance obtains official diagnostic confirmation. To evaluate the event, rule and process specifications, the distributed processing technique and the system architecture presented in this paper, we first implemented the SOP using 19 events, 19 rules, 27 automated operations, 75 manual operations, 21 organizational processes and 19 triggers. There are 11 unique roles assigned to end-users of NPDN labs and organizations. We then developed test scenarios for exercising each of the 11 roles and all of the steps in the SOP. The test runs resulted in several system improvements including the refinement of notification messages, the creation of a system log for audit trail purposes, and the display of a flowchart to show the progression of the process initiated by an event.

To further evaluate the processing technique and the system, researchers at UC-Davis, who are NPDN members, designed another test scenario based on the SOP and invited 4 additional NPDN members at the Kansas State University to participate in a field test over the Internet on June 30, 2010. This was done without the active participation of researchers at UF to ensure that NPDN members could independently operate the system and run through the entire scenario. The exercise was completed successfully in 5 days. The majority of SOP steps were completed during the first 24 hours. The reason that it took 5 days to run the scenario is partially due to manual operations, which take time to perform, and partially because some participants were not clear as to when the exercise was completed and thus terminated their assigned tasks.

The test run was followed by a debriefing conference call on July 26 for all participants to discuss the pros and cons of the system and the lessons learned. It was the participants' opinion that, overall, the system provides an efficient way to keep people engaged in the SOP by facilitating and guiding the communication channels. It helps each participant know who should do what and when. It provides a good framework for communication and helps participants understand how the SOP works. It can be a good training tool as well as a useful tool for exercises and/or the management of actual events. Participants indicated that the system has several areas which are improvements over the current exercise system developed and used by NPDN and several areas where functionality could be improved. These are described below:

Successes:

- This is a good start and demonstrates proof of concept. Its core functionalities are in place and working as intended. There is good potential for helping organizations carry out their steps of the SOP.
- The flow diagram and checklists are a significant improvement over the earlier version of the system. They allow the user to see what has been done, what is actively being done and what is yet to be done.
- Email prompts are very helpful. These serve to avoid fears that something should be done that isn’t getting done.
- The process requires very little training of participants and is self-managing and clear enough that participants can follow it without much administrative support.
The system did not get stuck in the process, but flowed smoothly from step to step. There wasn't a major stopping point where we had to restart. The Florida team did well in taking a very complicated process and making it efficient and relatively user-friendly.

Improvements:

- Some of the screens have several check boxes that may have long gaps in time between completion. It may be useful to reorganize some of the windows by combining or separating them to make more consistent with time lines.
- The user needs to be able to indicate that an activity will not be completed and provide an explanation as to why (i.e. someone else did it, it is not necessary or is inappropriate in this scenario, or unable to do it). Currently it is necessary to check all activity boxes as completed in order to move to the next step.
- The log tool should display all steps and inputs in real time.
- The flowchart could be more user-friendly and easier to read.
- Email prompts did not always arrive in a timely manner. This could be an end user issue that needs a work-around since it happened with both KSU and UC-Davis servers.
- It would be helpful to get an email that tells each participant when their activities for that event have been completed, and again when the event is completed for all roles. There was some confusion as to when the exercise was completed.
- The completed activity boxes should be made more distinguishable from those that are not in the displayed flowchart.
- The ability to start a new sample thread should be removed, except at the entry page.

It was agreed by the participants of the field test that there is value in exploring the adoption of the software into NPDN IT support services. A plan has been drawn up to present the system and the test results to NPDN's IT committee and then to other committees for NPDN's further development and deployment.

6. Summary and conclusion

In this paper, we have pointed out the need and importance for collaborating government organizations to share not only data and computing resources, but also their policies, constraints and regulations that can be specified in terms of knowledge rules, services and workflow processes. We also stressed the need for an infrastructure and a processing technique to enable the interoperation of these resources, and the dynamic construction and enactment of inter-organizational processes in order to achieve inter-organizational collaboration and coordination. To meet these needs, we have developed and presented an integrated rule and process specification language and its accompanying user interface for collaborating organizations to define events, three types of knowledge rules, processes, manual services, automated services, and triggers. We have also developed and presented an event-triggered knowledge network system, and showed how event data are transferred to explicit and implicit subscribers when an event of interest occurs, how distributed, organizational knowledge rules, processes and services can be processed in a Web service infrastructure, and how event data, different types of knowledge rules, processes and services can interoperate.

The key features of our work (also our intended contributions) are as follows. The integrated rule and process specification language and the user interface enable high-level specifications of events, knowledge rules, services, processes and triggers, which are easier for collaborating organizations to define, understand, modify and use than the program code that implement them. Organizations do not have to write application code to implement them because they can be automatically translated into Web services. The distributed, event-triggered rule processing and process enactment technique and the network system are important because different types of rules, and complex workflow processes can be uniformly processed as Web services in a Web service infrastructure by using replicas of a single rule server without having to use multiple rule engines and a workflow management system to process them. Event data, different types of knowledge rules, processes and services can interoperate because the event data can automatically trigger the processing of applicable rules, which can produce new data (by derivation rules) or enact processes (by action-oriented rules). These processes would activate their rules, perform automated services, and instruct users to carry out manual services. Through event notification, event data transmission and interoperation of distributed resources, collaborating organizations can “connect the dots”, receive guided assistance on appropriate emergency response, and establish better communication, coordination and collaboration among them.
Acknowledgements

This R&D work is supported by NSF under Grant IIS-0534065. The authors would like to express their deep appreciation to Jim Stack, Sharon Dobesh, Judy Omara and Elizabeth Schrum of the Kansas State University for their participation in the field test of ETKnet and their contributions in evaluating the system. The authors would also like to acknowledge the support from the National Plant Diagnostic Network, USDA-NIFA cooperative agreement no. 2007-37620-18230 to the University of California, Davis.

7. References


