RC24251 (W0705-009) May 1, 2007
Computer Science

IBM Research Report

An Enterprise Electronic Contract Management System Based on Service-Oriented Architecture

Trieu C. Chieu, Thao Nguyen, Sridhar Maradugu, Thomas Kwok
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

An Enterprise Electronic Contract Management System Based on Service-Oriented Architecture

Trieu C. Chieu, Thao Nguyen, Sridhar Maradugu and Thomas Kwok
IBM T. J. Watson Research Center
19 Skyline Drive, Hawthorne, NY 10532, USA
{tchieu, tnnguyen, sridharm, kwok}@us.ibm.com

Abstract

Electronic contract systems deal with the lifecycle automation and management of contract documents from their establishment to expiry. Many existing contract systems are built as monolithic, vertical stand-alone applications that are inflexible and difficult to scale and interoperate with other enterprise systems. This paper proposes a service-oriented electronic contract system that leverages a number of common middleware services to provide an open, extensible and interoperable solution. The basic common services include the content repository services, the workflow and document routing services, the security services, and the notification services. A standard data model is defined to specify the metadata properties for managing, tracking and storage of contract documents in a central repository. Integration services are also introduced to enable the system to collaborate and share contract data with external applications such as contract authoring and fulfillment tracking tools. The implementation of an enterprise multi-party electronic contract system using standard commercial middleware components will be presented to illustrate the simplicity, reusability, interoperability and flexibility of this service-based approach.

1. Introduction

Contracts constitute the binding relationship between organizations and their customers or suppliers, and capture the essential agreement recognized by law on key terms and conditions for trading or service, such as pricing, payments, liability, etc. In most large enterprises, a large number of contracts are created, executed, and managed daily via a traditional paper-based process that involves many manual steps for drafting, review, revision, signing, approval, completion or termination. However, the manual contracting process is inefficient, cumbersome, costly and time consuming because there are no standardized process to follow and no convenient way to access and manage the relevant documents. Thus, automation of the contract lifecycle presents a substantial value creation opportunity for the enterprises. The value spans from accelerated contract lifecycle processes, improved productivity, reduced costs, minimized potential contractual errors and faults, as well as better compliance enforcement [1].

With the proliferation of Internet technology and electronic commerce, many enterprises are adopting electronic contract systems to streamline their contracting processes. Much implementation efforts are centered on systems that deal with contract creation and lifecycle management [2-6]. Today, most of these electronic contract systems are designed as monolithic, vertical stand-alone systems and are not easily integrated with other business applications. As long as business processes remain confined within the systems, the lack of interoperability with other systems is of secondary concern.

Lately, in order to improve efficiency, productivity and agility in responding to changing business requirements, enterprises are pursuing new initiatives and objectives that increasingly involve multiple organizations, business processes, and data repositories, as well as making content and applications more accessible to external customers. The need for a more flexible approach to integrate various business applications to share data across the enterprise becomes increasingly evident. Separate silos of data and content duplication increase development and maintenance costs of business applications, and require custom code and extra efforts to bridge different APIs, interfaces, and platforms. Particularly, for the case of contract systems, enterprises will see limited benefits and returns on investment unless the systems are connected with the business as a whole, and leverage existing content management resources.

Recently, Service-Oriented Architecture (SOA) and Web services are increasingly recognized as a suitable architectural principle for developing modern enterprise applications [7]. SOA concept is to share and exchange the runtime results of executing software as services rather than the code itself that produces these results. The sharing and exchange is effectively achieved using standard XML format and HTTP communication protocol. By combining many low-level services, one can easily handle higher level business services or processes. Legacy business applications can also be service-enabled

to participate in these processes by introducing a standard service interface without the need to rewrite existing business logics of the application itself. The simplicity, reusability, interoperability and flexibility are just a few benefits provided by this SOA approach.

In this paper, we propose an enterprise electronic contract system with a service-oriented architecture that leverages a number of common middleware services. This service-oriented approach shows promise as an improved way to realize contract management systems with better interoperability and agility while minimizing development and maintenance costs. The lifecycles of contracts, the architecture of the contract system, and the common middleware services will be described in details in the following sections.

2. Automation of Contract Lifecycles across Multiple Enterprises

An electronic contract is the reification of paper contract in software that can be instantiated as a set of legal documents specifying the trading relationship and the legal terms and conditions that are fulfilled between organizations. A paper contract generally starts from an initial "draft" created by a contract representative in an organization that offers certain trade or service. It then goes through numerous rounds of internal reviews and revisions before it gets the final approval and is sent to its contracting parties. The contracting parties on the other hand have to go through similar internal process to review and approve the contract. Subsequently, the contract is sent back to the originating party to be counter-signed and finalized. The final approved copy of the contract is eventually forwarded to all involved parties for record keeping and execution. The number of intermediate stages during the contract lifecycles depends on the complexity of the contracting process among the involved parties. An example of a dual-signature contract processing flow involving three contracting parties (a supplier, a solution provider and a buyer) for contract drafting, submission, revision, signing, counter-signing, approval and completion is illustrated in Figure 1.

Conceptually, the lifecycles of contracts can be categorized into three stages of contract drafting, formation, and fulfillment as described below:

1. Contract drafting phase - This is the initial phase where a contract is drafted and created from an instance of a particular abstract contract flow. Contract rules and constraints can be added or modified, and should be adhered to during the contract fulfillment phase.
2. Contract formation phase – This is the phase where the contract is submitted for processing and the abstract contract flow processes are bound to concrete business interactions. The relationships between contract parties are defined and bound, and contracting parties assume their contract roles to perform the required actions of their responsibilities. The business interactions include actions such as review, revision, signing and approval of the contract.
3. Contract fulfillment phase – This is the final phase where the approved contract is delivered for implementation. Typically this phase constitutes the delivery of goods or service, invoicing, and bill presentment and payment. The interactions between the contracting parties can be monitored for their compliance or violation to the terms and conditions of the contract.

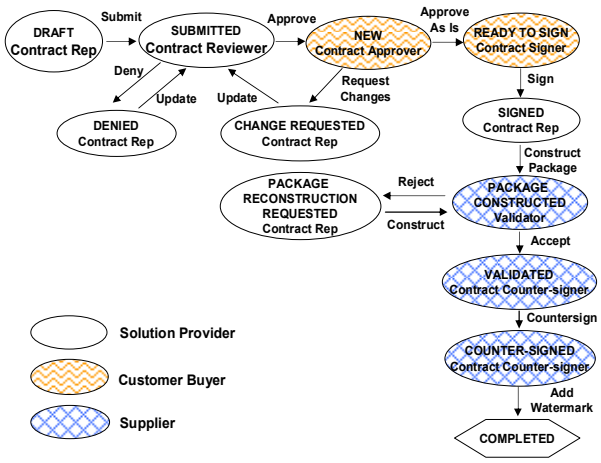

Figure 1. Three-party dual-signature contract processing flow

With the rapid development of computer technologies and the acceptance of the World Wide Web, enterprises are moving more and more of their business process online. There are existing technologies that allow enterprises to define their processes and assign the appropriate employees to their processes. An enterprise using such technologies to define a workflow for their internal contract process can scan their paper contracts or upload their electronic contract files onto their computer systems, route them through their intranet network to allow their authorized employees to review, revise and approve the contracts online. However, these workflows are intended to support an enterprise internal business process and often do not extend to support the workflow of the other enterprises that they engage contracts with. Once the contracts are approved and signed, they are very likely to be downloaded, printed on hard copy, faxed or sent to the next involved enterprise to be processed. This process may have to repeat as the contracts go through their entire contracting cycle across different enterprises.

To transform the manual process and automate the contract lifecycles across multiple enterprises, a hosted contract management system with service-oriented architecture is an essential component for the end-to-end contracting process. The system should provide the capabilities of defining and selecting different contract flows for contract drafting, and is capable of efficiently managing the many collateral documents such as the master and customer agreements, supplements, addenda and the like that are created, attached and associated with the contract formation. The ability to store, update, retrieve and share the pertinent information by users of the enterprises from a searchable content repository is also an important feature of the system. The architecture of such contract system will be given in details in the following section.

3. System Architecture

The architecture of an electronic contract management system for business-to-business interactions across multiple enterprises is shown in Figure 2. Basically, the system is architected as a web application provided by a host contracting organization, and can be accessed by the registered users of different organizations including customers, business partners, distributors and suppliers. The system consists of a number of user-facing modules including a system and user administration module, a contract authoring module, a contract management module, a contract search module, and a number of common management services. The user-facing modules are essentially the consumers of the common services.

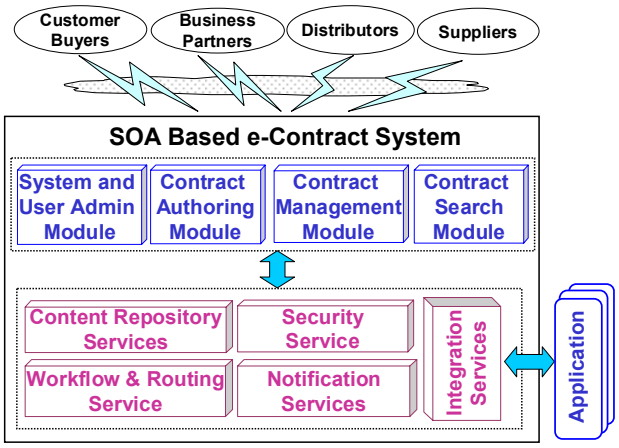

Figure 2. Architecture of an enterprise electronic contract management system

Briefly, the system and user administration module is used by different administrators to perform various administrative functions depending on their roles. The administration functions include system configuration, workflow creation and configuration, user and organization registration, and user role assignment and entitlement. The contract authoring module is used to create the drafts of the contracts for negotiation. The contract management module is used to initiate or terminate the workflow and routing of the draft contracts, display and track their lifecycle stages during the workflow process, and organize and manage the collateral documents. The contract search module is used to perform secure search and retrieval of the active and archive contracts stored in the system.

To support the basic functions of the user-facing modules, a number of common middleware services are implemented or provided by the architecture platform. The common services include the content repository services for supporting the storage, search and retrieval of documents, the workflow and document routing services for automating the lifecycles and workflow execution steps of contracts, and performing routing of documents to different user roles for subsequent task execution, the security services for supporting user authentication and authorization, document access control, and electronic signature for document signing and approval, the notification services for notifying and reminding users of the work tasks waiting in workflow steps, and finally the integration services for enabling the system to connect and interact with external applications.

4. Common Middleware Services

A number of common middleware services are provided by the service platform to support the various modules and management functions of the contract system. These services may be implemented as local or remote services, and can be accessed by consumer applications via service API calls. Detailed descriptions of these common services are given in the following sections.

4.1. Content Repository Services

During the workflow cycles of the contract process, users of different enterprises may create, upload or add a large number of contract documents into the system. The content repository services are used in conjunction with the contract management module to manage the storage, search and retrieval of these documents. As shown in Figure 3, the system is allowed to access the services only through a set of common content repository service interfaces that provide the standard CRUD operations as well as full-text search.

Different implementations of the repository interfaces may be plugged into the service platform. For instance, an administrator of the services can configure the services to use an implementation based on a file system or

another implementation based on enterprise content repository database middleware API. Both implementations basically can support the same exact functions of storage and retrieval of structured and unstructured content, full text search, versioning, and transactions. The client application that uses the services will not realize much functional difference. The only issue appears in the performance and scalability of the underlying repository support.

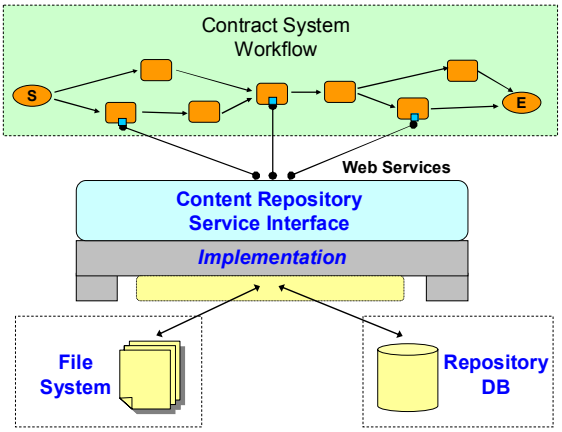

Figure 3. Content Repository Service Interfaces

In the core of the repository services, the documents are typically organized and stored following a hierarchical data model as shown in Figure 4. In this model, the actual stored objects are represented as items. Each item has an "*URI*" element that specifies its location on a particular branch of the repository tree, a unique "*Item ID*" element, a "*Name*" element, and a number of attributes or metadata describing its properties ("*Description*", "*Create_Date*", etc). An item can be a document or a folder. A document can contain zero or more document parts that correspond to the actual objects (textual files, images, etc.) stored in the repository. A folder can have zero or more items which can be documents or other folders. Items are allowed to be copies or moved from one folder to another.

The content repository Web services use an XML document messaging model to exchange the whole documents between client and server. One benefit provided by this messaging model is that it allows a business document to become self-describing and self-validating with its XML schema. Another benefit is that the client application will not need to be modified even though enhancements and changes are made to the XML schema. Furthermore, the document messaging model makes object exchange more flexible, because the design of a business document is often well suited to object-oriented architectures. As a result, two applications can be designed to exchange the state of an object by using XML while each application is free to design the object. An example of a SOAP "*RetrieveItemRequest*" service to retrieve a contract document from the content repository is shown in Table 1. The "*RetrieveItemRequest*" element includes the "*URI*" attribute of the requested "*Item*" and the "*Credential*" token representing the identity of the requester for service authentication.

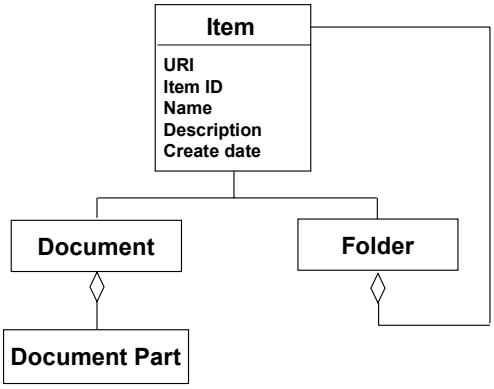

Figure 4. Document data model for content repository

Table 1. SOAP request of RetrieveItemRequest service

```
<soapenv:Envelope
    xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body  soapenv:encodingStyle=
     "http://schemas.xmlsoap.org/soap/encoding/">
  <processXMLRequest>
   <xmlRequestText xsi:type="xsd:string">
    <RetrieveItemRequest contentOption='ATTACHMENTS'
        retrieveOption='CONTENT'>
      <AuthenticationData
          connectString='SCHEMA=CONTRACT'>
        <ServerDef>
          <ServerType>Windows</ServerType>
          <ServerName>eContract</ServerName>
        </ServerDef>
        <Credential>h8p8sibYuMDnXZq-L2fh4Rt</Credential>
      </AuthenticationData>
      <Item URI='/ContractWebService/GetPIDUrl?
          url=CONTRACT_DOC_A07B26B42509H668151' / >
    </RetrieveItemRequest>
   </xmlRequestText>
   <mp xsi:type="apachesoap:Multipart" xsi:nil="true"
       xmlns:apachesoap="http://xml.apache.org/xml-soap"/>
  </processXMLRequest>
 </soapenv:Body>
</soapenv:Envelope>
```

4.2. Workflow and Document Routing Services

Most business operations can be characterized as a set of interrelated workflow processes. Work usually flows from one user to another and from one organization to another. Some simple processes might require only a few

work steps, while more complex processes involve a number of steps and users in different organizations. Document routing is a document-based workflow service specifically designed so that each unit of work is created and associated with a document or a folder of documents, along with other necessary information for carrying out the work. The sequence of business processes can be structured as a graph of interconnected steps known as a workflow template. Documents can be routed through the process that allocates work to individuals or group of users for action and processing. Document routing is typically integrated with access control checking, user management, and general system management to facilitate document management and processing in a business environment. Actions such as "submit", "approve", "sign", "reject" or any other meaningful selections can be assigned by administrator to each routing path to allow transition from a node to another node. An access control list can also be assigned to each work step to control which user role can perform the action. Based on their roles and privileges, authorized users can then access and examine the documents, and select an action from an action list to act on them.

Our workflow and routing services support the creation, definition and modification of user configurable workflow templates through a set of Web service interfaces. Workflow creation services allows user to produce an actual contract flow as a runtime instance of an abstract template, and perform the binding of enterprise users to the corresponding parties and roles defined in the template. Query services are also provided to facilitate the retrieval of available contract workflow templates stored in the system.

4.3. Security Services

Many users, each designated to perform the same or different predefined tasks on an electronic contract, can access simultaneously to the contract system. The security services are responsible to authenticate and authorize the user privileges of accessing the system for performing specific tasks on any given contract document during the contracting lifecycles. They also support the access control of contract documents and the verification of electronic signature for signing and approval.

To access the contract system, users need to perform log-in for authentication. The security services facilitate the log-in process by providing a single sign-on capability through Web services to offer a seamless and easy single-sign-on experience for users. In any client-server relationship, single sign-on is a user session authentication process that permits a user to enter a username and password in order to access multiple applications or services. The single sign-on, which is requested at the initiation of the session, authenticates the user to access all the applications they are entitled to use, and eliminates further authentication prompts when the user switches applications or services during that particular session. The user is only challenged once during a session for a username and password. Once authenticated, the user identity is securely associated with the execution context using a Kerberos credential of the user, and the Web service run-time component will perform authorization based on the asserted user credential containing in subsequent requests. After a fixed period of time of session inactivity, this credential cache will expire and the session will be invalidated by the service runtime.

At an approval step within the contract workflow, documents may need to be signed. For an electronic document, the signing is typically carried out using an electronic signature of a signer. The process usually involves the authentication of the identity of the signer, and the generation of a signed form indicating the signer's intent to sign the document. In our work, to support the signing of contract documents, an electronic signing Web service is provided. This service takes a username and password and a document as inputs, verifies the authenticity of the user's credential, signs the document, generates and returns a signed document to the requester. Detailed description of such a document signing system that superimposes signing information on the signed document was given elsewhere [8].

4.4. Notification Services

Notification mechanism is used to send reminders or other messages to users, or inform users about specific issues. Notification services allow users to continue working on other tasks, rather than constantly monitoring the progress of a process. They decouple systems producing notifications from applications displaying them. This allows existing notifications to be modified or new notifications to be generated with no impact on the applications using them.

The notification services are supported by a Java mailer program implemented with standard *JavaMail* API technology. A set of e-mail services are defined for the system to send outbound e-mail using SMTP to notify the responsible users when a contract workflow or a task has started, completed, or encountered an error, or is waiting for user's action at a particular work step. The system typically sends the messages to users using the e-mail address stored in the user's profile. In order to standardize the notification or reminder messages, the services also provide a number of configurable notification templates with pre-defined contents. Users of the contract system can use the e-mail services and select a notification template to attach and forward contract documents to other users of the system on-demand.

4.5. Integration Services

Integration is the process of linking disparate systems in order to enable effective sharing and seamless movement of information for operational advantages. While each type of integration solution involves the same basic technology, such as message bus, common data format and transformation, distributed transaction, and application interfaces or adaptors, each business scenario has unique requirements. In order to minimize development costs while maximizing reuse and agility, integration using Web services provides an optimal solution for flexibility and interoperability among diverse systems and platforms.

In this work, the contract management system is designed with a set of integration Web services to facilitate other business applications to connect and interact with the system. Particularly, two set of contract import and export Web services are implemented to enable contract information sharing with legacy contract management tools. The contract import Web services are provided for a contract authoring application to forward draft contract documents directly to the system to simplify the initiation of contract formation. The contract export Web services are provided to allow other contract fulfillment tools to dynamically extract and share the completed and active contracts in the system.

Figure 5 shows the various data models and elements used to hold the contract information during import and export service operations. The global element "*Application*" constitutes the root element of the import XML message. It consists of a "*Client_App_Id*" element, a "*Client_App_Password*" element, and a complex type "*Contract*" element that encapsulates the details of the contract information in its complex type child elements such as "*Attributes*", "*Parties*", "*Doc_Infos*", "*Doc_Info*", and "*Org_Access*" elements. An example of an XML message to forward a draft contract from a contract authoring tool to the contract system is given in Table 2.

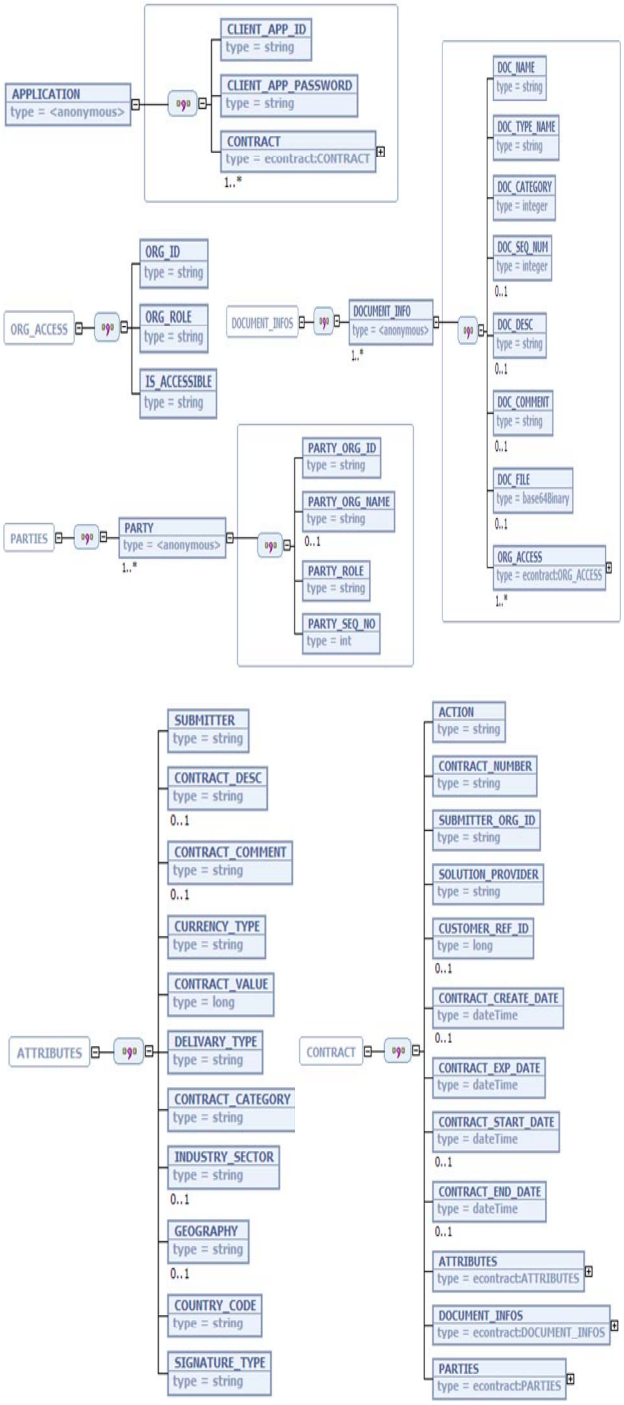

Figure 5. Data models and elements used for importing contract information

Table 2. Example of an XML message containing the draft contract information

```
<econtract:Application xmlns:econtract="http://www.ibm.com"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.ibm.com contract.xsd ">
 <Client_App_Id>App_00001</Client_App_Id>
 <Client_App_Password>6/x5EAd3cMg0D2PNLcoqwfEgRE8=
 </Client_App_Password>
 <Contract>
   <Action>create</Action>
   <Contract_Number>Contract_10001</Contract_Number>
   <Submitter_Org_Id>ORG_ID_1</Submitter_Org_Id>
   <Solution_Provider>Supplier_A</Solution_Provider>
   <Contract_Create_Date>2006-11-14 21:15:43.569
   </Contract_Create_Date>
   <Contract_Exp_Date>2007-06-21 23:59:00.000
   </Contract_Exp_Date>
   <Attributes>
      <Submitter>Submitter_A</Submitter>
      <Currency_Type>USD</Currency_Type>
      <Contract_Value>10,000</Contract_Value>
      <Delivery_Type>Online electronic</Delivery_Type>
      <Contract_Category>Master</Contract_Category>
      <Country_Code>USA</Country_Code>
      <Signature_Type>2</Signature_Type>
   </Attributes>
   <Document_Infos>
      <Document_Info>
         <Doc_Name>Master Agreement</Doc_Name>
         <Doc_Type_Name>Negotiable contract
         </Doc_Type_Name>
         <Doc_Category>1</Doc_Category>
         <Org_Access>
            <Org_Id>ORG_ID_1</Org_Id>
            <Org_Role>S</Org_Role>
            <Is_Accessible>Y</Is_Accessible>
         </Org_Access>
         <Org_Access>
            <Org_Id>ORG_ID_2</Org_Id>
            <Org_Role>C</Org_Role>
            <Is_Accessible>Y</Is_Accessible>
         </Org_Access>
      </Document_Info>
   </Document_Infos>
   <Parties>
      <Party>
         <Party_Org_Id>ORG_ID_1</Party_Org_Id>
         <Party_Role>S</Party_Role>
         <Party_Seq_No>1</Party_Seq_No>
      </Party>
      <Party>
         <Party_Org_Id>ORG_ID_2</Party_Org_Id>
         <Party_Role>C</Party_Role>
         <Party_Seq_No>2</Party_Seq_No>
      </Party>
   </Parties>
 </Contract>
</econtract:Application>
```

5. Prototype Implementation

We have successfully implemented an enterprise electronic contract management prototype system on a service hosting platform supported by a number of IBM WebSphere software products [9]. The platform services include a WebSphere portal server and application server, a DB2 server and a SMTP server. The common middleware services include a DB2 Content Manager [10] that offers both a configurable document workflow engine and a scalable content repository database, and an IBM Tivoli Access Manager WebSeal server that offers a single sign-on security service. A number of user-facing modules are developed as parts of the contract application to provide user interactions. These modules expose their user interfaces through Web pages with Ajax technologies to enable rich and dynamic interactions.

An example of a Web page that lists all the contract transactions and Web links to their details for a given user in our electronic contract prototype system is shown in Figure 6. By clicking on any contract transaction link, the contract details of that particular contract will then be displayed. Figure 7 shows another Web page that displays the details of a particular contract transaction.

The left navigation bar of the Web pages displays the accessible sub-modules and functions of the system. The "*Imported Draft Contracts*" module handle all draft contracts imported from any external contract authoring applications that are integrated with the system using the Integration services. The "*My Task list*" module lists all personalized contract transactions that are waiting for actions by the current logon user. Contract creation, submission and management functions are provided in the "*Contracts In Progress*" module. User and system administration functions are provided in the "*Administration*" module.

The right navigation bar of the Web pages optionally lists some additional function links that a user can choose to perform depending on his/her role and the current work step and situation. For instance, when displaying the "*Contract details*" page of a contract in "REVIEWED" status that is awaiting the action by a reviewer of another organization, the logon user who is the originator of the contract can have access to links such as "*Make correction*", "*Expire this contract*" to modify the details or cancel the contract before any action taken by other users.

Our prototype system provides many configurable options that a system or organization administrator can choose to enable or disable. These functions were supported and readily composed from the available common services. The implementation of the prototype system illustrated that the rapid development of an electronic contract system using service-oriented architecture to leverage available service resources is not only feasible but much cost advantageous over other approaches.

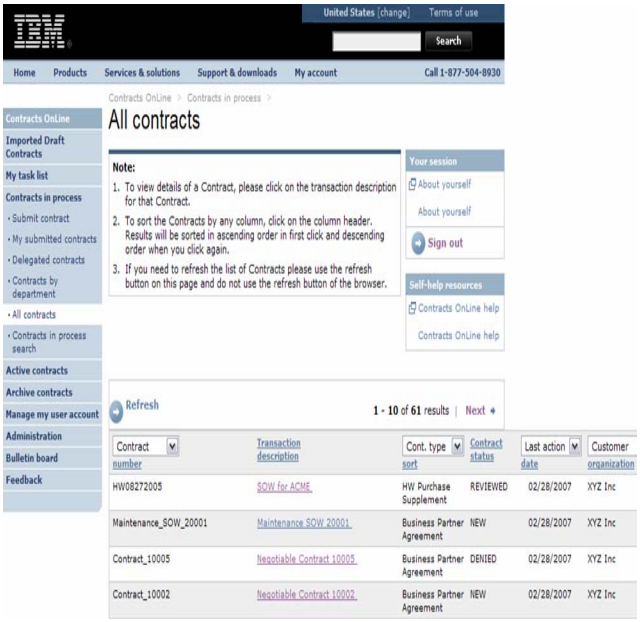

Figure 6. Web page showing lists of all contracts for a user in the contract system

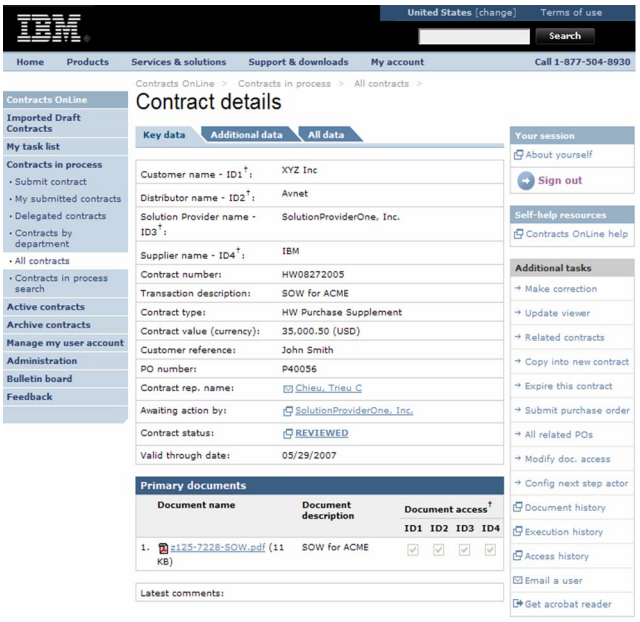

Figure 7. Web page showing details of a contract in the contract system

6. Concluding Remarks

In this paper, we introduced an electronic contract system based on a service-oriented architecture. The system is structured as a Web application on a common service platform that provides a number of common middleware services. The basic common services include the content repository services, the workflow and document routing services, the security services, the notification services and the integration services. These services are conveniently provided as Web services, and can be implemented using commercial available middleware. A service-based electronic contract prototype system has been implemented on a service platform that is supported by a number of IBM WebSphere middleware products. The prototype system has demonstrated the benefits of simplicity, reusability, interoperability and agility of our service-based design approach.

7. References

[1] International Association of Contract and Commercial Managers, http://www.iaccm.com.

[2] F. Griffel, M. Boger, H. Weinreich, W. Lamersdorf and M. Merz, "Electronic contracting with COSMOS – how to establish, negotiate and execute electronic contracts on the Internet", Proc. 2nd Int'l Enterprise Distributed Object Computing Workshop, 1998, pp. 46-55.

[3] L. Xu, "Monitorable electronic contract", Proc. IEEE Int'l Conf. on E-Commerce, 2003, pp. 92-99.

[4] M. Iwaihara, H. Jiang and Y. Kambayashi, "An integrated system for supporting problem solution in e-contract execution", Proc. IEEE Int'l Conf. on E-Commerce, 2004, pp. 9-16.

[5] O. Perrin and C. Godart, "An approach to implement contracts as trusted intermediaries", Proc. 1st IEEE Int'l Workshop on Electronic Contracting, 2004, pp. 71-78.

[6] T. Kwok and T. Nguyen, "A Secure Electronic Contract Management and Process System Automated with Predefined Tasks", 7th IEEE International Conference on e-Commerce Technology (CEC 2005), IEEE Computer Society 2005, Munich, Germany, pp. 479-502.

[7] L. Cherbakov, G. Galambos, R. arishankar, S. Kalyana and G. Rackham, "Impact of Service Orientation at the Business Level", IBM System Journal, Vol. 44, no. 4, 2005, pp. 653-667.

[8] T. Kwok and T. Nguyen, "An Automatic Electronic Contract Document Signing System in a Secure Environment", 2005 IEEE International Conference on e-Technology, e-Commerce, and e-Services (EEE 2005), IEEE Computer Society 2005, Hong Kong, China, pp. 276-281.

[9] IBM WebSphere software, http://www.ibm.com/software/websphere/.

[10] IBM DB2 Content Manager, http://www.ibm.com/software/data/cm/cmgr/.