Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

ELSEVIER

Decision Support Systems 40 (2005) 107–127

Decision Support Systems

www.elsevier.com/locate/dsw

On-demand business collaboration enablement with web services

John Y. Sayah[a], Liang-Jie Zhang[b,*]

[a] *IBM Software Group, 17 Skyline Drive, Hawthorne, NY 10532, USA*

[b] *IBM T.J. Watson Research Center, 1101 Kitchawan Road, Route 134, Yorktown Heights, NY 10598, USA*

Available online 19 June 2004

Abstract

Businesses are increasingly outsourcing key operations and interacting with ever extending nets of partners. Running extended business-to-business (B2B) operations creates the need for more advanced human interaction while advancing the automation base of B2B functions. In this paper, we introduce a model for on-demand business process-based collaboration, namely, Extended Business Collaboration (eBC), and its major elements of modeling, and a configurable business protocol-enabling framework. We discuss some of the major research issues associated with facilitating extended business collaboration, and present our proposed Annotated Business HyperChain technology leveraging Web services and semantic annotation model. A research prototype is described, followed by some observations and discussion of open research issues requiring further exploration.

Keywords: Extended business collaboration; Web services; Semantic annotation model; Business process; Annotated Business HyperChain

1. Introduction

Enterprises are not standalone anymore. They need to work with their value net of partners and customers. Whether we are considering, logistics, financial services, suppliers, handling customer orders, or marketing programs, large or small enterprises operate and interact in variety of forms within a complex global web of collaborating entities. A 21st century enterprise, not requiring all of its needs to be fulfilled by internal groups, leverages some well-proven services and available products in its daily operation processes and in product designs. Taking a new product design as an example, a consumer electronics company has to work with component suppliers, electronic manufacturing services (EMS), or contract manufactures to design a product collaboratively. Some component design work may be outsourced to design partners who are specializing in special components such as Application Specific Integrated Circuit (ASIC) chips, batteries, or motherboards. Outsourcing non-core-competency services has become a popular trend fueled by business transformation drivers. Information technology services and high-tech product developments are at the forefront of this trend that is becoming the dominant business model in the era of globalization. It is the outsourcing model that enables disaggregated businesses to form a value chain for creating more innovative and higher quality products or services than that they would have accomplished by their own.

In a typical business value chain, the trading partners or design partners could be dynamically added or removed in the lifecycle of a business solution, operation or during product development

* Corresponding author.

E-mail addresses: john_sayah@us.ibm.com (J.Y. Sayah), zhanglj@us.ibm.com (L.-J. Zhang).

0167-9236/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2004.04.011

or servicing thereafter. All the resources including business entities, services provided by business entities, documents and messages would be activated and accessed in what could be dubbed as an ‘‘on-demand model’’. In this on-demand model, a service is ‘‘brought-on-line’’ if and when is needed and solely for the duration of its need. In this model, the goal is to maximize efficiency and productivity. A possible transformation can now take place, from entities interacting in a disaggregated setting to one of a virtual on-demand enterprise operating in extended business collaboration mode. In this mode, requesting components for a product from a supplier would seamlessly locate the best suppliers but would concurrently enable and manage the related financial, logistics, or legal services. In this mode, we would also move from the paradigm of granular commercial transactions, such as issuing a purchase order (PO), to a paradigm of integrated business processes. These extended business and collaboration processes that engage and involve multiple business entities concurrently now have an extended lifecycle that is dynamic and cannot be confined and solely defined by a simplistic model of transactional handshakes or totally depend on the human intervention to manage it.

Fully automated business collaboration remains a goal on the horizon. However, experience leads us to believe that people-assisted enabling technologies can pave a way to this realization. Leaving aside practical business and geo-political considerations, preventing some companies from fully embracing extended business collaboration, we will attempt in this paper to address some of the key technical obstacles. We then present a methodology, supporting framework and techniques that would enable the practical realization of this on-demand collaborative business vision.

From a technical viewpoint, the real obstacles go beyond business process representation and data transformation techniques. The real problem arises from the fact that we are dealing with interactions between two or more business entities and their loosely coupled business processes. These business processes could be private business processes in some enterprises or public processes crossing the boundary of multiple enterprises. In this environment, the workflow is non-deterministic and you have projects and operations running across multiple companies with a mix of automation and human-driven actions. Timetables, project schedules, and response times are equally fluid. Leveraging emerging and evolving standards is a key starting point to help address the aforementioned challenges and problems.

Web Services [18] are network-enabled reusable components that conform to an interface with standard description format and access protocols. The basic enabling infrastructure of Web Services consists of UDDI registries, Simple Object Access Protocols (SOAP), Web Services Definition Language (WSDL), Business Process Execution Language (BPEL4WS), Web Services Inspection Language (WSIL), and so forth. Web Services provide the means to enable the integration in a standard way.

In this paper, we outline a new Extended Business Collaboration (eBC) model first. We then present, in Section 3, an enablement framework based on Web Services, followed by an illustration of a design collaboration scenario in a distributed industrial environment. Some related works are given in Section 4. The paper concludes with some observation and discusses further research topics.

2. Extended business collaboration (eBC) model

Several approaches have been proposed to represent business behaviors and a variety of ‘‘layer’’ models circulate in the business modeling domains. All models, independent of the adopted layering approach, do basically include a higher business layer and a lower Information Technology (IT) infrastructure layer. Typical business models are: Business-to-Customer (B2C), Application Service Provider (ASP), Application to Application (A2A) also called Enterprise Application Integration (EAI), and Business to Business (B2B). All of these ‘‘classic’’ models strictly differentiate between intra-enterprise interactions and inter-enterprise interactions. In support of these enterprise-based interaction models, different interaction techniques (business portals, e-mail, fax, etc.) and general and vertical industry standards (EDI, ebXML, RosettaNet [14], etc.) have emerged over the latter part of the last century providing various levels of business interaction and connectivity deployment.

As outsourcing and on-demand operation models are becoming more and more popular, the boundary between enterprises is gradually bypassed or eliminated. Enterprises are collaborating by different degrees and levels. The environment—the extended enterprise, the information sharing, the functions and the processes that enable two or more parties, within and cross-business entities, to interact in the context of business activities and events to deliver business results, are all being transformed. Hence, the representation needs evolve to capture and cover this emerging business paradigm.

We believe that an Extended Business Collaboration that dissolves or rather bypasses this enterprise boundary-based distinction is a factual representation of what is evolving in the real business world. This model aims at reducing the artificial elements at the boundary inherent in the enterprise-based interaction models. The diffusion of the boundary implies that new techniques are required in order to evolve converge the various converge interaction methods. Introducing business semantic computing techniques, creating pluggable business collaboration protocols, proposing dynamic activity chain representation, and leveraging distributed project and business process management and monitoring capabilities may pave a way to on-demand business collaboration, which has a more structured but flexible collaboration adaptability.

In the case of Product Life Cycle Management (PLM), the parties interact in a dynamically established virtual team and enterprise setting, during the concept, design, build, or servicing of a product to create a more innovative, profitable, higher quality product brought to market sooner than the parties would have accomplished on their own. By bringing PLM into the ideal eBC setting, we move from ad hoc and transactional interactions to constructing, activating, tracking and, monitoring collaborative development and design processes of a product involving multiple companies or organizations inside one company. An example eBC deployment scenario is illustrated in Fig. 1.

A product company may engage an ASIC supplier, an EMS, or some other service providers at various stages of product design and development [4]. These “partners” may in-turn engage other partners down their value chain. The timing and the infrastructure supporting these interactions vary in terms of frequency, automation, and the individuals who participate in the process. Some partners are connected through B2B

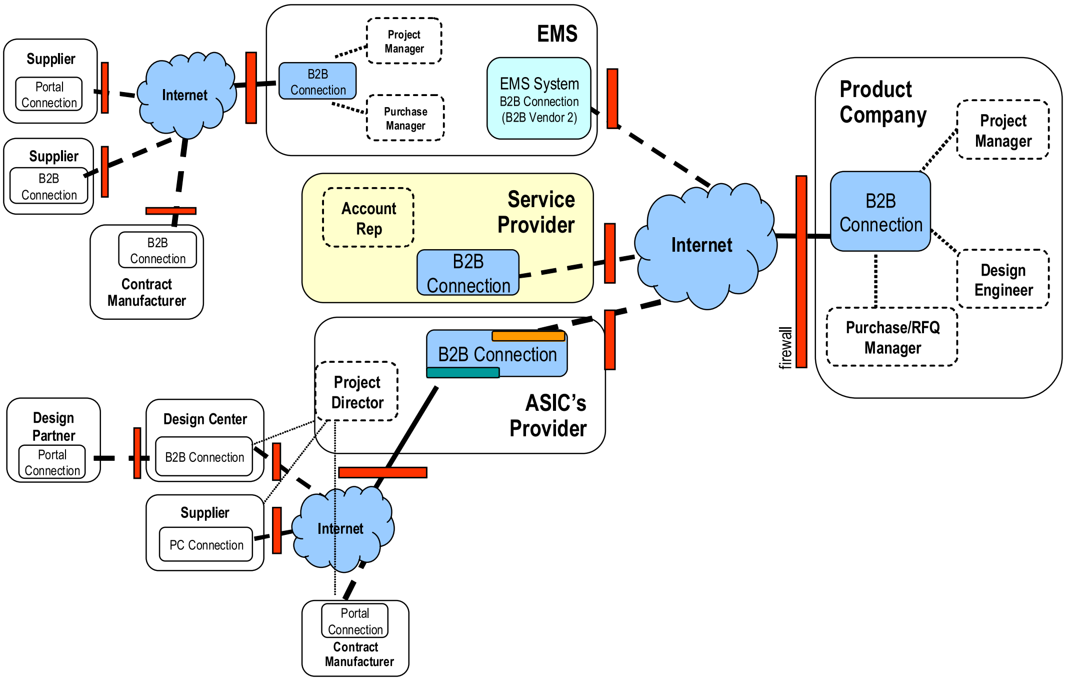


Fig. 1. Extended business collaboration deployment example for PLM.

software, to support transactional exchanges, while others may connect occasionally through a portal interface. The patterns of active connectivity and their durations and where information is in this design and supply *hyper-chain* are neither fixed nor static. This is a not a traditionally well-defined supply chain where all participants could be known in advance. Each partner knows and interacts "on-demand" with its immediate partners and information gets propagated up and down this hyper-chain. In this setting, an eBC solution should support the observation and control of related supply chain management and collaboration activities during the design and development of a product. Meanwhile, the desired eBC solution should manage the dynamics characterizing this environment including the formation and dissolution of collaborating teams and the related effects. From the solution management point of view, the eBC solution needs to monitor the design process status, design data status at any granularities, across all design and development partners, and across the individual participants. In case of business exceptions or when quick decisions are required, the desired eBC solution should provide the mechanisms to enable efficient escalation activation and timely problem resolution of issues that impact the design and development processes and schedules of a product and related business activities.

Although Web services infrastructure provides a good foundation to build such a flexible eBC infrastructure, there remains a variety of challenges to achieve these goals. One of the key challenges involves the type and nature of information exchanges. As Fig. 1 illustrates, partners in the design and supply hyper-chain do not have homogenous environments whether for internal or external operation and connectivity. Furthermore, it maybe impossible in this setting to determine a priori all levels and type of exchanges. Hence, it is imperative to provide sufficient *annotations* during an interaction process to permit the parties at the interacting edges to exchange and interpret what is required and where to access information or data. Areas requiring special attentions that drove our research activities included:

- defining a flexible annotation information representation format,
- delivering the information in a standardized or standardize-able fashion,
- interpreting, processing, and directing information exchanges based on needs,
- controlling information exchanges and flow dynamically,
- monitoring the status of process flows and documents exchanges,
- insuring and managing the security of all communication channels and information access.

Therefore, in the next section, we presents an XML, Web Services and other standards based Annotated Business HyperChain (ABH) technology for creating and managing eBC infrastructure and solutions.

3. Annotated Business HyperChain technology for eBC enablement

In this paper, we propose a new technology—Annotated Business HyperChain—that addresses the areas of semantic representation, collaborative exchange protocols, and on-demand information exchange model to enable extended business collaboration. The Annotated Business HyperChain technology consists of three major components, namely, the *eBC Ontology*, the *Collaborative Exchange Protocol* (CxP), and the *HyperChain Manager*. Annotated Business HyperChain technology also extends the concept of hyperlinks of an object in HTML files to the solution components and the resources involved in a business collaboration chain.

The eBC ontology, defining the commonly shared knowledge regarding the business semantics of the information that get exchanged during business collaboration, provides the foundation for understanding and interpreting the information involved in the process. The eBC ontology enables a flexible and uniform annotation representation for information exchanges of various non-structured, and ad hoc data without requiring pre-defined schemas. Based on the eBC ontology, CxP defines the set of elemental and composite messages that may be exchanged between two or more parties engaged in collaborative business activities. CxP is a business goal-oriented protocol supporting a wide variety of business constructs and a versatile message composition that accommodates the needed variations in the life cycle of a collaborative business process.

A basic operation principle relating to the proposed information exchange model in this paper is to communicate schema-less HyperChain annotation data. Recipients would then follow the HyperChains to access or fetch required and detailed information, such as design files, design specifications, Bill of Materials (BOM) files, based on the roles of the recipient or their position in the business chain. These on-demand file and data transfer modes are enabled through self-retrieving or agent-based file transfer services. In the meantime, the model supports tractable information associated with design files, design processes and BOM.

In general, an on-demand information exchange model which enables implementation of the above-described operations is designed to achieve the following goals: (i) provide a flexible and uniform annotation representation for information exchange of various non-structured data without requiring predefined schemas; (ii) automate the annotation data generation process; and (iii) capture and automate business collaboration interaction patterns for information exchange based on the annotation data. Moreover, delivery policies are provided to control how the on-demand contents are to be delivered. There are at least four types of delivery models:

Scheduled content delivery—On a predetermined, periodical schedule, the information content can be delivered to the intended recipients.

On-demand content delivery—Ad hoc, based on user's request, i.e., following HyperChain specified links that provide with the annotation data and download the data. In most cases in a design collaboration scenario, the design file is very large and a server-to-server file transfer mechanism may be needed for assisting on-demand content delivery.

Access control-based content delivery—Delivering contents depending upon the role and authorization of a recipient and user credentials. As business collaboration generally involves multiple enterprises, the regular single-sign-on security mechanism has to be enhanced to incorporate annotated access control policy in the business annotation data based on entitlements. For example, who can see a document or modify it, and when it should be sent back or forwarded to other participants.

Push-based content delivery—Sending annotation data along with attachments. In general, this model is suitable for small size file transfer.

To effectively annotate the data for business collaboration, we propose an extensible data structure based on the proposed eBC ontology for data annotation, describing design collaboration processes, design activities such as design requirements, references, specifications, and design tools. The eBC ontology is built on the Resource Description Framework (RDF) [13] that provides the flexibility and versatility to support various data format required in collaboration message flow and document exchanges. A detailed description on the eBC ontology will be given later.

HyperChain Manager is the core processing engine that is responsible for creating, sending, receiving, and processing annotation messages. Additionally, it implements the on demand information exchange model and escalation process launch. It also provides an enabling platform to dynamically configure business constructs that guide the follow-on interactions between design partners. It serves as a platform to provide an extendable data aggregation mechanism to integrate information from multiple partners' data sources for effective monitoring and visibility control. As shown in Fig. 2, a diagram illustrates a HyperChain manager, which serves as a CxP engine.

The deployment architecture comprises an extended business collaboration portal/dashboard, an extended business collaboration (eBC) manager, a HyperChain manager, a B2B Gateway such as WebSphere Business Integration Connect (WBI-C) [16] layer that servers as a gateway between the HyperChain manager and the lower-level web application server such as a WebSphere [17] layer that is a typical Java-based web application server for hosting business applications. The portal or dashboard includes applications that can access the HyperChain Manager via the application-programming interface (API) layer provided by eBC manager.

HyperChain manager comprises a collaborative directory (with manager and directory), an annotation manager, a message sender, a message receiver

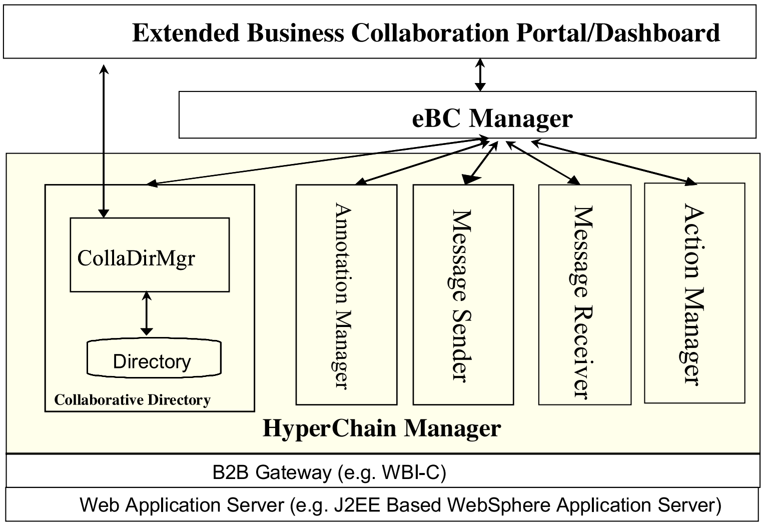


Fig. 2. HyperChain manager.

and an action manager. The collaboration directory manager component manages the resources tracked by the HyperChain Manager, such as organizations (partners), users, projects, tasks, etc., and the resources are RDF-based. The CxP messages are sent and received by the message sender and receiver modules and they are Simple Object Access Protocol (SOAP) [15] messages or other protocol messages such as Message Queuing (MQ) messages. The Message sender and Message Receiver shown in Fig. 2 have Web services interfaces. In fact, from the deployment point of view, few existing collaboration technologies or frameworks can be deployed on different application platforms (Windows, Linux) and environments (tool-specific environments). That is, platform-independent collaboration technologies have not been widely adopted in the loosely coupled business process integration domain. In this paper, we leverage the emerging Web services technology to create platform-independent interfaces to support flexible information exchange across multiple enterprises.

The annotation manager processes the meta data or annotations created for the documents and information exchanged via CxP messages. Examples of annotations are file name, file type, version, author name, etc. In addition, annotations can also be used to specify "actions" to be performed on the documents. Examples of such actions may be "review" document, perform "RFTP" (reliable file transfer) and send actions to legacy applications like Enterprise Resource Planning (ERP) and Product Data Management (PDM), etc.

The annotations in the received messages are forwarded to the action manager, which is an integration layer to back-end legacy applications as well as components like RFTP. The action manager invokes the proper actions on the documents.

As shown in Fig. 3, a "collaborator (1, 2,... *M*, *N*,...)" is a business entity that participates in a business collaboration process with one or more external business entities. "App" (l, 2, 3,...) denote the backend applications that the HyperChain manager integrates with the existing business process through an action manager component. The "collaborative directory" stores the resources of the business collaboration, such as projects, tasks, users, organizations, documents, as well as annotations/meta data that are managed by the HyperChain manager. The "HyperChain dashboard" in Fig. 3, is a graphical user interface (GUI) providing management and monitoring functions through which people interact with the collaboration resources stored in the collaborative directory.

The sections that follow describe in detail main components of Annotated Business HyperChain architecture and the underlying technology components.

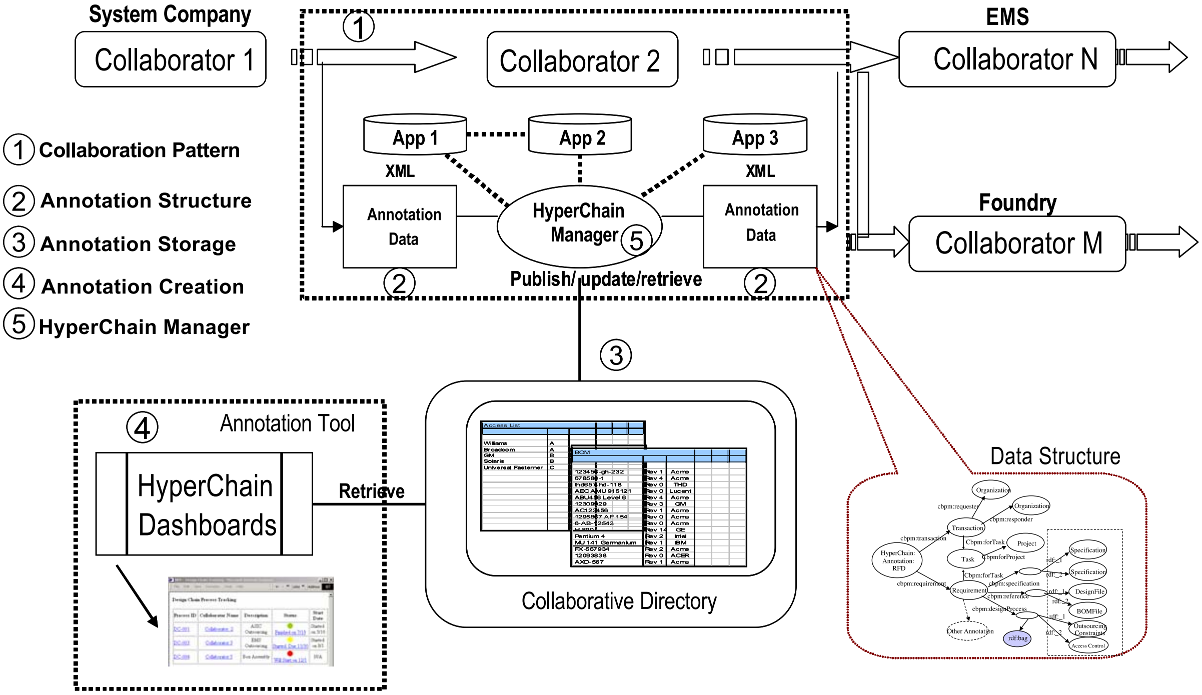


Fig. 3. HyperChain information propagation architecture.

3.1. Semantic annotation model for eBC

An ontology or commonly shared knowledge defines the business semantics to annotate information to be exchanged. An ontology is required for providing a foundation for business collaboration. Without such shared common knowledge, participants will not be able to decipher the exchanged information. Existing collaboration solutions are usually based on fixed knowledge pre-configured at each collaborator side reducing flexibility and functional scalability of collaborative activities.

The semantic annotation model for eBC, referred as eBC Ontology, which provides a flexible and uniform annotation representation for information exchanges of various non-structured, ad hoc data without requiring pre-defined schemas. We use Resource Definition Framework (RDF) to capture all three types of semantic representations, namely, Organizational Behaviors, Data Entities for Business Collaboration, and Activity Ontology for Extensible Application Integration. eBC Ontology is an instance of RDF schema [3] for extended business collaboration. It consists of all three semantic representations listed above to address the organization behaviors, external activities as well as individual resources in the context of eBC. An example Request for Design annotation RDF graph is shown in Fig. 4.

As shown in Fig. 4, all collaborators use the basic ontology to exchange business information. The business collaboration ontology uses a RDF model for specification purposes. Annotation is one part of the ontology. For example, we define RDF resources such as “Site,” “Organization,” “Project,” “Task,” “Requirement,” “Transaction,” “Documents,” “Annotation,” etc. in an RDF schema. The RDF schema serves as the basic ontology definition that all collaborators need to understand.

An RDF-based business collaboration ontology treats all newly added entities as resources. Thus, the same mechanism used to handle existing resources may be used to handle newly added resources. The extensive and flexible features of the business collaboration ontology allow the definition of any annotations without being restrained by the schema of the annotation data. Thus, as mentioned above, Hyper-

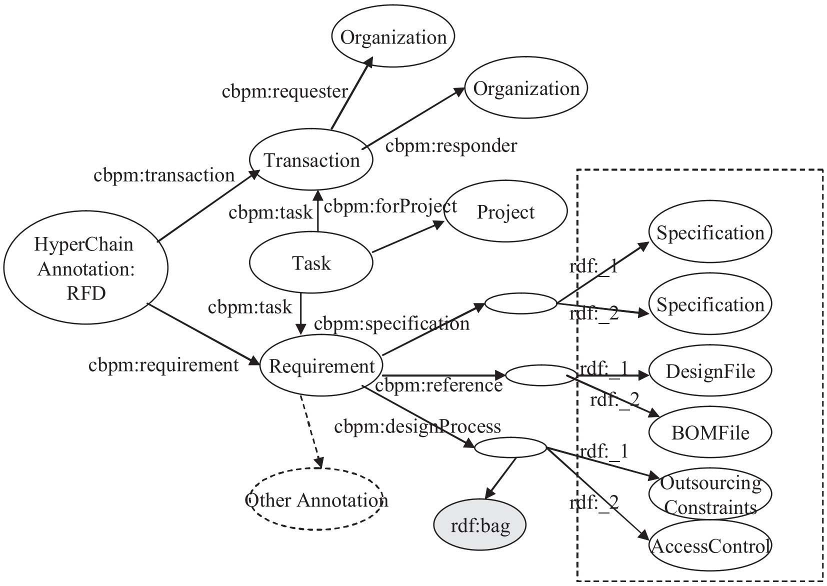


Fig. 4. RDF graph for HyperChain annotation data.

Chain annotation data that conforms to the business collaboration ontology is effectively schema-less.

In our approach, the ontology definitions are the basis that the collaborators need to understand and agree upon before exchanging information. Some example resources of the ontology currently provided by eBC are: Site, Product, Part, DesignComponent, Service, Party, Organization, Individual, Project, Task, OutsourcingTask, HomeTask, Message, Document, DesignFile, Requirement, Specification, Reference, Status, etc.

In addition, each collaborator can define their own ontology (Extended Ontology) and add additional annotations into the basic ontology for their own special needs. For instances, we can use fileName, fileSize, and format to annotate a specific design file. Again, these extended ontology or annotation definitions also need to be propagated to business partners prior to business exchanges take place. eBC Ontology is the collection of the basic and extended ontology, which is used by HyperChain Manager, which will be described in detail later, to create resources and model.

As shown in Fig. 5, a diagram illustrates a hierarchical annotation structure of entity classes defined for a HyperChain manager.

In Fig. 5, root class is the Site, which can be associated with zero or more Organization classes, representing business entities. Each Organization class can be associated with zero or more Project classes, each of which in turn can be associated with zero or more Task classes, as well as zero or more PeopleCollab utilities. Each Task class can be associated with zero or more Transaction classes, which in turn can be associated with zero or more (CxP) Message classes. In addition, each Task class can be associated with zero or more Requirement classes, representing requirements to be sent to the partners. Each Requirement class can be associated with zero or more Annotation (which may include meta data to describe the requirement), Specification, and Reference classes.

Annotation Property is the Java class that the actual annotations are created from, e.g., filename, authorname, price, etc. Hence, the relationship indicates “use”. “PeopleCollab” refers to the agent or broker that conducts a human collaboration process, which is part of the extended business collaboration process. The example human collaboration process may be launching a chat program, creating a discussion thread in a discussion forum, and so forth. “0...*” in Fig. 5 means that the association relationship is 0 or more,

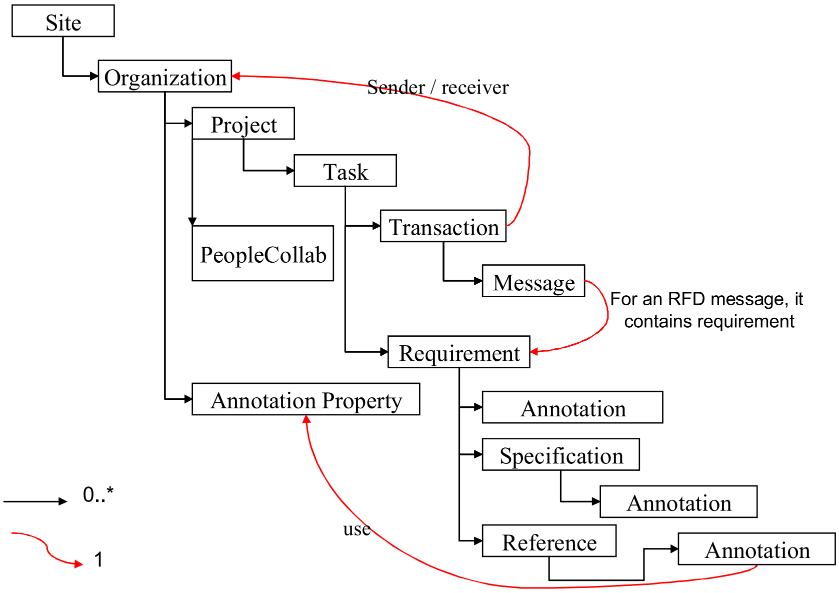


Fig. 5. Class diagram of the hierarchical annotation structure of eBC.

i.e., source class can be associated with zero or more instances of the target type where the straight arrow (→) is pointing to. "1" refers to the association relationship being one to one.

An example annotation data instance is shown in Table 1. It shows a sample business process annotation data instance for an RFD message of the entire design chain. The "transaction" resource defines the exchange context of the messages, RFDTransaction, the requester, PDT, and the responder, MyComputerCorp. The "Task", T61MotherBoardDesign, and "Project", T61BoardDesign, resources define which project and task the messages are bound to. Several containers are defined to group the following annotations or metadata: Design specification annotation, T61Specification.pdf, Design file annotation, T21MotherBoard DesignFile.cat, BOM file annotation, T21BOMFile.bom, Design process annotation, outsourcingConstraint, and other related annotations, such as access control. Note that "cbpm" stands for "Collaborative Business Process Model" we defined in this paper. "rdf" stands for "Resource Definition Framework".

The design activity annotations mainly focus on constraints in the design collaboration process. There are several types of activity annotations, e.g., design requirements, design configurations/specifications, the design files, BOMs, design processes, etc., and each one is for a different purpose. Thus, each contains different annotation data with a different format. However, they should all follow the same design rule. The following example in Table 2 shows a design activity annotation where several constraints are specified as well as the access control using OASIS eXtensible Access Control Markup Language (XACML) [11] to express the constraints.

3.1.1. Annotation storage: collaborative directories

We continue to use design collaboration to illustrate eBC concepts. The design collaboration ontology is defined in RDF schema format, and is stored in RDF format. Annotation is one part of the ontology. There are diverse requirements for annotation in design collaboration, and new requirements emerge endlessly. In addition to the pre-defined annotations for electronic business collaboration, users can define custom annotations. Table 3 is a sample of storage of an annotation definition.

These annotation definitions are applied to various elements during the information exchanges in design collaboration processes. An example RDF representation request for design (RFD) message during the RFD primitive for a design project will be illustrated in details later.

Table 1
RFD message

```
<rdf:RDF
<cbpm:RFD rdf:ID=”007”>
          <cbpm:transaction>
          <cbpm:Transaction  rdf:about=
”http://www.pdt.com/dc/directory/transaction#007001”_>
          <rdf:type>&cbpm;RFDTransaction</rdf:type>
          <cbpm:requester
rdf:resource=”http://www.pdt.com/dc/directory/Organization#PDT”/>
          <cbpm:responder
rdf:resource=”http://www.pdt.com/dc/directory/Organization#MyComputerCorp”/>
          <!--anything about the Transaction-->
          </cbpm:Transaction>
          </cbpm:transaction>
          <cbpm:task>s
          <cbpm:Task
rdf:about=”http://www.pdt.com/dc/directory/task#T61MotherBoardDesign”>
          <!--anything about the task-->
          <cbpm:forProject>
          <cbpm:Project
rdf:about=”http://www.pdt.com/dc/directory/project#T61BoardDesign”>
          <!--anything about the project-->
          </cbpm:Project>
          </cbpm:forProject>
          </cbpm:Task>
          </cbpm:task>
          <cbpm:requirement>
          <cbpm:Requirement
rdf:about=”http://www.pdt.com/dc/directory/requirement#00d034334”>
          <!--anything about the requirement-->
          <cbpm:specification>
          <rdf:Bag>
          <rdf:li>
          <cbpm:Specification  rdf:about=”http://www.pdt.com/pdf/T61Specification.pdf”/>
          </rdf:li>
          </rdf:Bag>
          </cbpm:specification> <cbpm:reference>
          <rdf:Bag>
          <rdf:li>
          <cbpm:DesignFile
rdf:about=”http://www.pdt.com/pdf/T21MotherBoardDesignFile.cat”/>
          </rdf:li>
          <rdf:li>
          <cbpm:BOMFile  rdf:about=”http://www.pdt.com/pdf/T21BOMFile.bom”/>
          </rdf:li>
          </rdf:Bag>
          </cbpm:reference>
          <cbpm:designProcess>
          <rdf:Bag>
          <rdf:li>
          <cbpm:OutsourcingConstaints
```

Table 1 (*continued*)

```
rdf:about=”http://www.pdt.com/outsourcingConstraint”/>
        </rdf:li>
        <rdf:li>
        <cbpm:AccessControl rdf:about=”http://www.pdt.com/accessControl”/>
        </rdf:li>
        </rdf:Bag>
        </cbpm:designProcess>
        </cbpm:Requirement>
        </cbpm:requirement>
</rdf:RDF>
```

The annotation can be stored in a collaborative directory, which can be deployed on each enterprise site. Collaborative Directory stores all the HyperChain annotation data and partner’s profiles as well as the links to different other data sources accessed by participants in a design chain. The collaborative directory consists of Web services utilities and a relational database or plain XML file for storing the collaborative data. At the same time, the collaborative directory provides Web services utilities that used to populate services for updating/publishing data; monitor the status of the services at different levels and for eBC dashboard. It acts as a File Transfer Agent to invoke the file transfer service on B2B collaboration environment; as well as to connect with HyperChain Manager.

Since the data with embedded status information (e.g., about a project, tasks, exchanged documents, etc.) are stored in multiple collaborative directories, the information from these distributed collaborative directories can be aggregated based on an access control policy carried in the annotation data. Another deployment of the collaborative directory is to act as a hub where the hub manages collaborative resources of multiple organizations that use the hub as a central repository in support of collaboration activities.

3.1.2. Annotation creation and portal integration

Annotation creation is a major function of the eBC enabling platform. It may be performed with the assistance of annotation tools. As mentioned above, all the annotation data of various resources used in business collaboration are stored in annotation storage, collaborative directories, such as plain text files or relational databases. The annotation creation process may operate on the storage to create annotations. In general, creation of annotation includes the following steps:

(1) Collect information by use of extended business collaboration portal or other GUI interfaces. The information includes the description of various resources such as partners, projects, tasks, specification annotations, reference design file annotations, and other related annotations.
(2) Store all the collected information into the annotation storage.
(3) Extract required data from the storage to organize the annotation message to be exchanged.

Let us take the RFD message creation as an example to illustrate the process. First, the user creates a new task as an outsourcing task or internal

Table 2
Design activity annotation

```
<cbpm:design-activity-annotation>
<cbpm:desc>
<cbpm:checkpointConstraints>. . . </cbpm:checkpoint constraints>
<cbpm:outsourcingConstraints>... (XACML) </cbpm:outsourcingConstraints>
<cbpm:acl>     ... (XACML)    </cbpm:acl>
<cbpm:documentFormatConstraints>http://temporg.com/tempuri/documentFormatContraints.htm
</cbpm:documentFormatConstraints>
</cbpm:desc>
</cbpm: design-activity-annotation>
```

Table 3
Sample RDF schema for HyperChain annotation

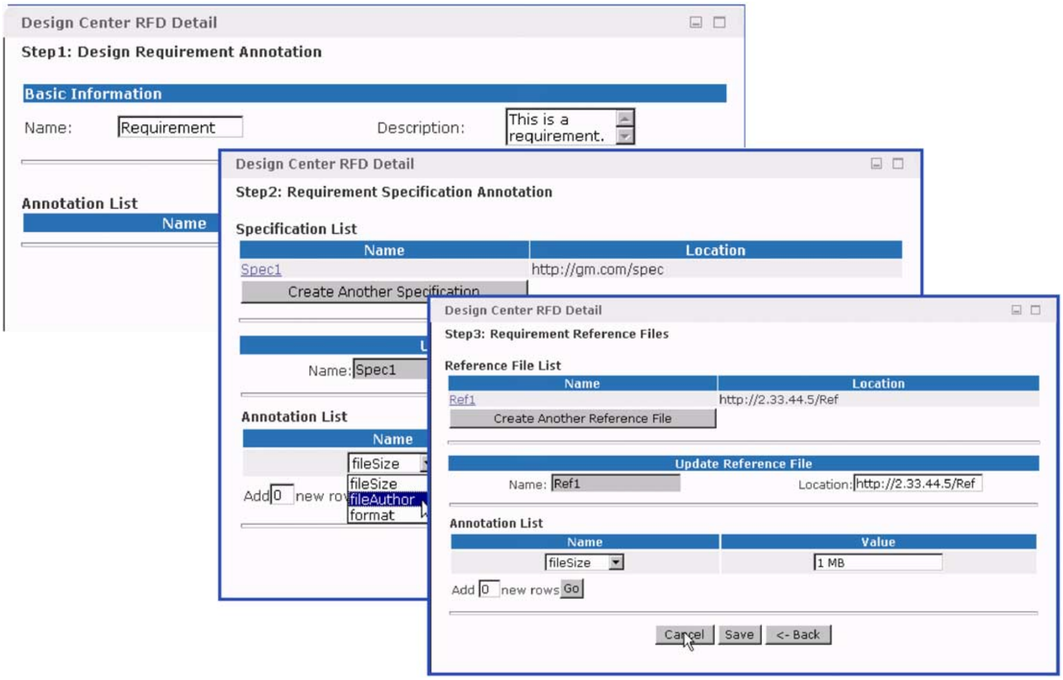
```
<rdf:RDF
  xmlns:daml='http://www.daml.org/2001/03/daml+oil#'
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  xmlns:rdfs='http://www.w3.org/2000/01/rdf-schema#'>
  <daml:DatatypeProperty rdf:about='http://www.ibm.com/ibm/pdt#fileName'
    rdfs:label='fileName'>
    <rdfs:comment>The name of a file, huh?</rdfs:comment>
    <rdfs:domain rdf:resource='http://www.w3.org/2000/01/rdf-schema#Resource'/>
    <rdfs:range rdf:resource='http://www.w3.org/2000/10/XMLSchema#string'/>
    <rdfs:isDefinedBy rdf:resource='urn:Organization:PDT@PDT'/>
  </daml:DatatypeProperty>
  <daml:DatatypeProperty rdf:about='http://www.ibm.com/ibm/pdt#fileSize'
    rdfs:label='fileSize'>
    <rdfs:comment>The size of a file</rdfs:comment>
    <rdfs:domain rdf:resource='http://www.ibm.com/cbpm#Document'/>
    <rdfs:range rdf:resource='http://www.w3.org/2000/10/XMLSchema#string'/>
    <rdfs:isDefinedBy rdf:resource='urn:Organization:PDT@PDT'/>
  </daml:DatatypeProperty>
  <daml:DatatypeProperty rdf:about='http://www.ibm.com/ibm/pdt#format'
    rdfs:label='format'>
    <rdfs:comment>The format of a file, huh?</rdfs:comment>
    <rdfs:domain rdf:resource='http://www.ibm.com/cbpm#Document'/>
    <rdfs:range rdf:resource='http://www.w3.org/2000/10/XMLSchema#string'/>
    <rdfs:isDefinedBy rdf:resource='urn:Organization:PDT@PDT'/>
  </daml:DatatypeProperty>
</rdf:RDF>
```

task. Then, the user may specify design requirements for the design task. The requirements may include specifications, reference design files, design process constraints, etc. Fig. 6 illustrates the creation of a Request for Design (RFD) message, including design requirement annotations, specification annotations, and annotations about the related reference documents.

After the information is collected and stored in the annotation storage, the annotation creation process starts. The annotation creation module extracts required data from the storage and forms the RFD annotation message based on the eBC ontology. The generated RFD message will be sent to design partners. After receiving the RFD message, partners can view the annotation and merge it with their own annotation storage. If a partner desires to learn more about one of the annotated resources, the partner can get the annotation link (such as design file annotation link) and request more information. The sender will generate an annotation for the design file and send it back to the partner. The partner can determine whether or not to retrieve the actual design file based on the annotation. Thus, on-demand information exchange is performed.

3.2. Collaborative exchange protocols (CxP)

Based on the eBC ontology, CxP uses RDF to annotate business collaboration processes by defining industry specific ontology, allowing peer-to-peer interaction between collaborative processes. CxP is a typical collaboration pattern in Fig. 3. CxP comprises of the messages to be exchanged between two parties or among multiple parties, some predefined message exchange sequences, and a set of business goal-oriented protocols composed by some predefined message exchange sequences. CxP builds on top of a set of standard protocols and adds the features needed for extended business collaboration processes. CxP are used to transmit the semantic representation and control the information exchange flow as well as monitor the on-going activities in a dynamic fashion.



Design Center RFD Detail

Step1: Design Requirement Annotation

Basic Information

Name: Requirement   Description: This is a requirement.

Annotation List

Name

Design Center RFD Detail

Step2: Requirement Specification Annotation

Specification List

Name	Location
Spec1	http://gm.com/spec

Name: Spec1

Annotation List

Name
fileSize

Add 0 new ro

Design Center RFD Detail

Step3: Requirement Reference Files

Reference File List

Name	Location
Ref1	http://2.33.44.5/Ref

Update Reference File

Name: Ref1   Location: http://2.33.44.5/Ref

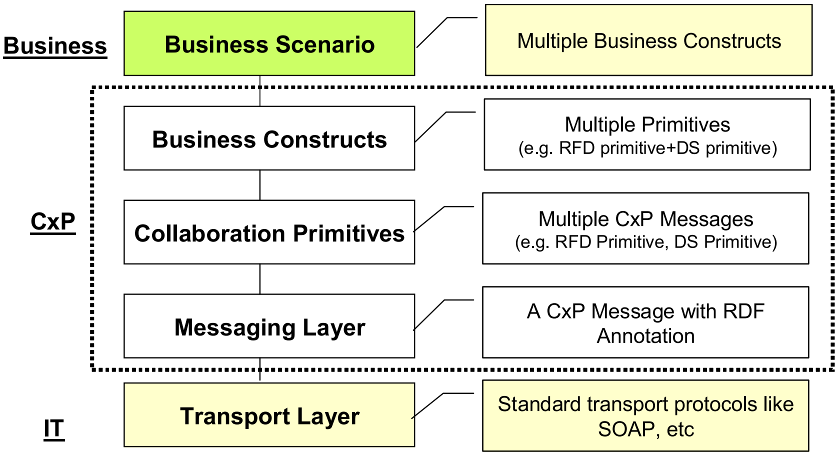
Annotation List

Name	Value
fileSize	1 MB

Add 0 new rows

Fig. 6. Flexible CxP message creation using web portal.

As shown in Fig. 7, the collaborative exchange protocol stack of eBC supports elements of various granularities, which are configurable building blocks for creating adaptive solutions to achieve a business goal. As shown, in the protocol architecture, the following elements are defined: business scenario, business constructs, collaboration primitives, messaging layer and transport layer. The corresponding descriptions on each layer are multiple business constructs, multiple primitives (e.g., request for design (RFD) primitive and design submission (DS) primitive), multiple CxP messages (e.g., RFD primitive, DS primitive), CxP message with HyperChain annotation, and standard transport protocols. In the



Business
Business Scenario
Multiple Business Constructs
CxP
Business Constructs
Multiple Primitives (e.g. RFD primitive+DS primitive)
Collaboration Primitives
Multiple CxP Messages (e.g. RFD Primitive, DS Primitive)
Messaging Layer
A CxP Message with RDF Annotation
IT
Transport Layer
Standard transport protocols like SOAP, etc

Fig. 7. Collaborative exchange protocol (CxP) stack.

messaging layer, RDF is used to represent business collaboration annotation. On top of the messaging layer, a set of primitives is defined as collaboration primitives for communication and collaboration between the parties.

A business construct is a basic unit of message exchange sequences which serve a single business goal. For example, an RFD business construct is used when a request for design is initialized, e.g., a design center, Product Design Team (PDT), shown in Fig. 8, can send RFDs to its design partners to do motherboard designs or to do mechanical and electrical designs. Following that, an Accept or Reject primitive may be received from the design partners. A business scenario serves a more complex business goal-like design-outsourcing scenario. Each business scenario may comprise several business constructs depending on the corresponding business context.

Collaboration primitive, business construct and business scenario concepts are described in detail below.

In CxP, an atomic message is defined as a rudimentary exchange of information between collaboration partners, e.g., an RFD message. A set of choreographed messages forms a primitive. For example, RFD primitive may comprise two messages, e.g., RFDMessage and AckMessage. Furthermore, one or more primitives form a business construct. For example, RFD business construct may comprise two primitives, e.g., RFD primitive and Acceptance/Rejection primitive. Scenarios are sequences of business constructs that represent a complex interaction among business partners, such as design initialization, engineering change management, and opportunity launch. In addition, CxP primitives and business constructs are targeted for specific collaboration goals and, even though configurable, they are relatively fixed. While business scenarios can be composed in several ways and thus are quite flexible.

As we introduced earlier, a design collaboration primitive is a group of message exchanges for a specific and micro-design collaboration goal. Several core design collaboration primitives are defined for CxP: Request For Design (RFD), Accept or Reject a request (Accept/Reject), Design Submission (DS), Request For Information (RFI), Information Submission (IS), Request For Update (RFU), Update Submission (US), and so forth.

Let us take RFD as an example; each collaborator uses the RFD primitive to request a partner to

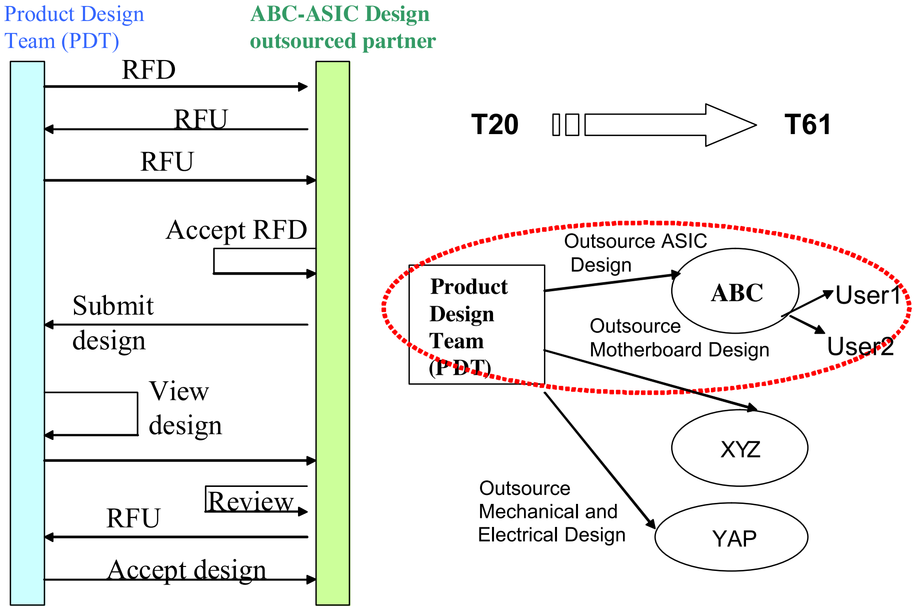


Fig. 8. Example ThinkPad design process templates.

perform a design task. An RFD primitive comprises three messages: RFD, RFD_Receipt_Ack, and RFD_Acceptance_Ack messages. This is illustrated in Fig. 9.

> *RFD Message:* sent by the originator, e.g., a design center, to a recipient, e.g., design partner. Contains a requirement comprising specifications, references, and annotations.
> *RFD_Receipt_Ack Message:* sent by the recipient; a response to RFD message, indicating the RFD message has been received by the recipient.
> *RFD_Acceptance_Ack Message:* sent by the recipient, containing a flag indicating whether the recipient accepted or rejected the RFD.

Table 4 is an example of RFD Message.

Each design partner may accept or reject the request after the partner received either an RFD or RFU. One example of an Accept primitive to an RFD is illustrated in Table 5.

3.2.1. Business construct

A business construct comprises a group of collaboration primitives, which can be selectively configured for a business construct. Once configured, a business construct is organized in a relatively fixed fashion to achieve a single design collaboration goal.

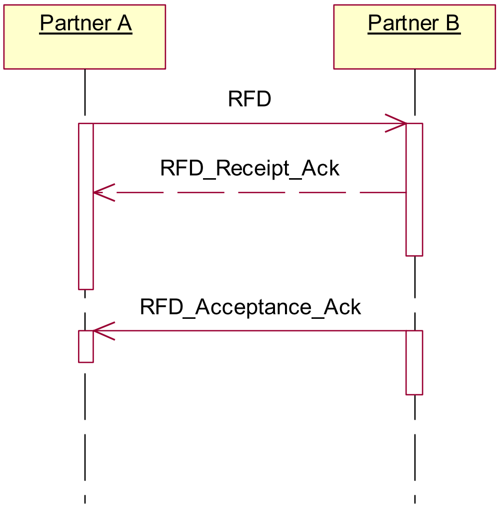


Fig. 9. Message sequence diagram of RFD primitive.

The following business constructs are based on the primitives previously discussed:

- RFD business construct (RFD primitive + Accept/Reject primitive + DS primitive)
- RFU business construct (RFU primitive + US primitive)
- RFI business construct (RFI primitive + IS primitive)
- US business construct (US primitive)
- IS business construct (IS primitive)

Based on these business constructs, collaborators can define any complex business scenario if they so desire. A standard representation for a business process modeling language, such as Business Process Execution Language (BPEL4WS) [8], can be used to represent CxP business constructs. Once represented by BPEL4WS, multiple business constructs can form a business scenario, which can be dynamically composed by using dynamic composition technology for Web services flow such as Web Services Outsourcing Manager (WSOM) [19,24].

3.2.2. Example: RFD business construct

A RFD business construct may contain one RFD primitive, one Accept/Reject primitive, and one DS primitive. The RFD micro-flow can be represented using BPEL4WS in Table 6. “process” in Table 6 stands for “business process”, which is a standard way to describe business process in BPEL4WS.

In fact, the business collaboration or design collaboration patterns can be very complicated as they often involve multiple interactive messages based on the primitive protocols. Take design outsourcing for example. In Fig. 8, the right-hand side shows a design center, Product design team, which out-sources parts of the ThinkPad design to different design partners, e.g., to ABC for ASIC chip design, to XYZ for motherboard design, and to YAP for mechanical and electrical design. The left-hand side of Fig. 8 shows the design collaboration patterns between Product design team and ABC. The various messages flow between the two partners, starting from a request for design (RFD), followed by requests for updates (RFUs) by the acceptance of the RFD, by the submission of design, by the viewing of the design, by further RFUs and inter-

Table 4
Sample RFD message in CxP

```
<RDFNsId2:RFD
rdf:about='urn:RFD:SoundCard_test1@SoundCard_test1@SoundCard@Workstation@PDT@PDT'
RDFNsId2:description=''
RDFNsId2:identifier='SoundCard_test1'
RDFNsId2:status='Accepted'>
<RDFNsId2:requirement>
<RDFNsId2:DesignRequirement
rdf:about='urn:DesignRequirement:test1@SoundCard@Workstation@PDT@PDT'
RDFNsId1:cpuFrequency='500'
RDFNsId2:identifier='test1'
RDFNsId2:name='test1'
RDFNsId2:description='test1'>
<RDFNsId2:specification>
<rdf:Bag>
<rdf:li>
<RDFNsId2:Specification rdf:about='urn:Document:test@test1@SoundCard@Workstation@PDT@PDT'
RDFNsId2:identifier='test'
RDFNsId2:name='test'
RDFNsId2:description='d:\there'/>
</rdf:li>
<rdf:li>
<RDFNsId2:Specification rdf:about='urn:Document:@test1@SoundCard@Workstation@PDT@PDT'
RDFNsId2:description=''
RDFNsId2:name=''
RDFNsId2:identifier=''/>
</rdf:li>
</rdf:Bag>
</RDFNsId2:specification>
<RDFNsId2:reference
rdf:type='http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag'/>
<RDFNsId2:forTask rdf:resource='urn:Task:SoundCard@Workstation@PDT@PDT'/>
</RDFNsId2:DesignRequirement>
</RDFNsId2:requirement>
<RDFNsId2:transaction
rdf:resource='urn:Transaction:SoundCard_test1@SoundCard@Workstation@PDT@PDT'/>
<RDFNsId2:creationTime>Jan 16, 2003 5:10:17 PM</RDFNsId2:creationTime>
</RDFNsId2:RFD>
```

mediate reviews, finally concluded with the acceptance of design.

The sample design collaboration pattern shown in Fig. 8 demonstrates the flexibility and versatility of the RDF to support various data format required in collaboration message flow and document exchanges.

4. Related work

In today's Web services infrastructure [20], there is a lack of a uniform semantic representation for individual solution components. For example, WSDL concentrated on describing the basic information about a Web service; Some information about WSDL are published in Web services registries, namely, UDDI registry or WSIL documents, which are two different type of Web services registries. However, there is no place to describe capability information about a Web service, method signature mapping, and the like in current Web services related specifications. Moreover, there remains a set of open issues for incorporating WS-Security mechanisms within a project context, or business flow (e.g., BPEL4WS) context, and other solution components such as UDDI registry, SOAP invocation engine, and even WSDL documents. This is one of the rationales for forming WS-I Forum [18] in order to address the interopera-

Table 5
Example message for accept primitive of RFD

```
<RDFNsId2:AcceptanceAck
rdf:about='urn:AcceptanceAck:SoundCard_test1AcceptanceAck@
    SoundCard_test1@SoundCard@Workstation@PDT@PDT'
RDFNsId2:identifier='SoundCard_test1AcceptanceAck'
RDFNsId2:ack='Accept'
RDFNsId2:status='Sent'>
<RDFNsId2:transaction
rdf:resource='urn:Transaction:SoundCard_test1@SoundCard@Workstation@PDT@PDT'/>
<RDFNsId2:creationTime>Jan 16, 2003 5:38:38 PM</RDFNsId2:creationTime>
</RDFNsId2:AcceptanceAck>
```

bility issue among multiple standards specifications. Additional languages include Web Service Choreography Interface and others jointly defined by major e-business companies [1].

Most researchers in the field of Web services are realizing that semantic information are needed for effective Web services discovery [6,23], dynamic Web services composition as well as collaboration at run-time. Some semantic representation approaches have been proposed to address this issue. For example, (1) DAML-S is being proposed to describe more information about individual Web services for discovery and composition [5]; (2) Regular XML-based annotation languages have been defined to capture different types of semantic information. One example XML annotation, Web services relationship language (WSRL) [22], is proposed to capture the Web services relationships at different granularities, which we think will be an important facilitator in selecting and composing the right set of services that meets the customer's requirements. Additionally, a business requirement language, Business Process Outsourcing Language (BPOL) is proposed to capture the business requirements such as conceptual flow, preferences, business rules, relationship bindings, and event-action mappings for automating the Web services discovery and flow composition that matches customers' requirements [19,24]. (3) Organizational behaviors associated with an e-business solution refer to the semantic representations about the organizations, the on-going projects in an organization, tasks in a project, requirements and transactions in a task, additional annotation about any other related resources such as value-added services involved in an e-business solution.

For the first two, namely (1) and (2), as we mentioned earlier, we can find some example solutions to address the semantic representations for different aspects about individual Web services. For example, in traditional e-commerce environment, a workflow-based document routing language was proposed to address the inter-organizational collaboration [10]. It could be a very useful foundation for helping create intelligent documents. However, it is a regular XML-based description language which lacks the extensibility and schema-less feature. Moreover, there was no configurable business protocol creation framework to enable the business process based collaboration across multiple enterprises. The third is more systemic view of a semantic representation for building and managing a Web services based e-business solution. We think Resource Definition Framework (RDF) [13] could provide a uniform and efficient way to capture all three types of semantic representations. In this paper, the realization of Web services collaboration [20] is to create an extended business collaboration ontology, which is built on top of RDF [13], DAML-S [5], and other XML-based semantic representations, to effectively create and manage the Web services based extended business collaboration solutions.

Other existing human-based collaboration activities such as chat, session-sharing, white boarding and document shared-access [2] do not provide links to the B2B processes as they are centered only on human communication. In the solution framework outlined in the paper, we provide the means, technology and implementation for a deployment and operation of the human-assisted collaboration integrated in the context of managed business processes. These aspects alone cover a white space in the interaction and collaboration of enterprises in the new era of global commerce [21], outsourcing and cross-enterprise busi-

Table 6
RFD business construct in BPEL4WS

```
<process name="RFDmicroflow"
        targetNamespace="urn:samples:BusinessConstructs"
        xmlns:tns="urn:samples:BusinessConstructs"
        xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/">
  <partners>
        <partner name="RFDoriginator"
                 serviceLinkType="tns:RFDoriginatingSLT"
                 myRole="RFDoriginating"/>
        <partner name="RFDreceiver"
                 serviceLinkType="tns:RFDreceivingSLT"
                 myRole="RFDreceiving"/>
        <partner name="buyer"
        serviceLinkType="tns:buyingSLT"
        MyRole="buying"/>
  </partners>
  <variables>
        <variable name="RFDinvoke" messageType="tns:RFDinvoke"/>
    <variable name="RFDmsg" messageType="tns:RFDmsg"/>
        <variable name="RFD_Receipt_Ack" messageType="tns:RFD_Receipt_Ack"/>
        <variable name="Accept" messageType="tns:Accept"/>
        <variable name="DSinvoke" messageType="tns:DSinvoke"/>
        <variable name="DSmsg" messageType="tns:DSmsg"/>
        <variable name="DS_Receipt_Ack" messageType="tns:DS_Receipt_Ack"/>
  </variables>
   <correlationSets>
        <correlationSet name="POIdentifier" properties="POIdentifier"/>
        <correlationSet name="RFDIdentifier" properties="RFDIdentifier"/>
   </correlationSets>
  <sequence>
    <receive partner="buyer" portType="tns:buyerPT"
                        operation="purchase" variable="RFDinvoke"
                        createInstance="yes" name="ReceivePurchase">
                        <correlations>
                        <correlation set="POIdentifier" initiate="yes"/>
              </correlations>
        </receive>
        <invoke name="invokeRFDoriginator"
               partner="RFDoriginator" portType="tns:RFDoriginatorPT"
               operation="sendRFD" inputVariable="RFDinvoke" outputVariable="RFDmsg">
        </invoke>
        <invoke name="invokeRFDreceiver"
               partner="RFDreceiver" portType="tns:RFDreceiverPT"
               operation="receiveRFD" inputVariable="RFDmsg"
outputVariable="RFD_Receipt_Ack">
        </invoke>
        <invoke name="invokeRFD_Accept_Ack"
               partner="RFDreceiver" portType="tns:RFDreceiver"
               operation="sendRFDAccept"inputVariable="RFDmsg"
outputVariable="RFD_Accept_Ack">
        </invoke>
     <invoke name="invokeRFD_Accept_receive"
               partner="RFDoriginator" portType="tns:RFDoriginator"
               operation="receive_Accept" inputVariable="Accept">
```

Table 6 (*continued*)

```
        </invoke>
        <invoke name="invokeDS"
                        partner="RFDreceiver" portType="tns:RFDreceiver"
                        operation="submitDS" inputVariable="DSinvoke" outputVariable="DSmsg">
        </invoke>
    <invoke name="invokeDS_Receipt_Ack"
                        partner="RFDoriginator" portType="tns:RFDoriginator"
                        operation="receiveDS" inputVariable="DSmsg" outputVariable="DS_Receipt_Ack">
        </invoke>
    </sequence>
</process>
```

ness activities. The enablement of these extended interactions, collaborations and services, by leveraging web services technology, is an emerging area with the promise of reducing the overhead of implementation and by providing an ''impedance-match'' to allow companies with different systems, legacy applications, and various means of implementing business processes to participate in their respective business value nets with practical levels of investments in software and at competitive integration cost. These observations are based on our customer engagement experience in support of industry solutions.

The environment of interest in this paper encompasses business process flows that are deployed on different enterprise sites. Hence, the research focus of this paper is to find an efficient and effective approach to capture the business collaboration context (e.g., organizational structure, projects, tasks, requirements and the relationships among them) and configure customized business protocols to facilitate the information exchange among loosely coupled business processes.

The relationship among CxP, RosettaNet and BPEL4WS can be summarized as follows. CxP identifies the primitives for the collaborative ''Partner Profile Processes (PIP)'' (in the RosettaNet sense) [14] as well as the extendable hyperlinked data descriptions. CxP is immediately compatible at a high level with the RosettaNet PIP model. As we have illustrated in this paper, CxP can be restructured to be BPEL4WS compatible. The CxP Message data is extensible to support hyperlinked document types with RDF graph. They are used to compose collaborative business primitives such as the different variety of Request for Information (RFI) and Request for Updates (RFU).

One of the disadvantages of the HyperChain Manager presented in this paper is that it cannot directly process the platform or channel specific information such as CAD files, RosettaNet protocols, ebXML protocols, etc. However, the HyperChain Manager can route these activities to the right applications based on the business-annotated data carried in the CxP messages.

5. Conclusions and future work

In this paper, we have presented an on-demand business collaboration solution approach that supports a new model of integration of collaboration workplace with B2B collaborative process flow. A new breed of ontology-based information-exchange protocols, human–machine process primitives for design collaboration, middleware and complementary tool to support rapid B2B collaboration process design were also created to support the evolving domain of extended business collaboration paradigm.

The Collaborative exchange Protocols (CxP) stack of eBC supports elements of various granularities, which are configurable building blocks for creating adaptive solutions to achieve a business goal. What we have learned while developing the CxP enabling infrastructure is that modularizing and composing the reusable components in a flexible and extendable way was one of the major challenges. The traditional approach of using code template did not address the flexible configuration of new business protocols in the service oriented computing environment. However, the proposed eBC model and enabling framework brings human assistance aspect into different levels of a business process, i.e., at the CxP primitive level, business construct level, and solution scenario level, for exception handling, escalation, and decision making. Web portal server is used to integrate people's activities into collaborative business processes.

Finally, we foresee the following issues that can be treated as future research topics in the field of extended business collaboration.

- Multiple variety of business scenarios can be simulated by supporting the coordination of multiple BPEL4WS sequencing fragments through an extension mechanism such as a BPEL++.
- Using eBC infrastructure and configurable business protocol framework to extend RosettaNet specification and enabling infrastructure to support flexible payload formats as well as to compose PIPs in a manageable way.
- Defining more industry specific ontology extensions for eBC ontology to create a rich business collaboration foundation.
- Creating a pluggable business scenario creation framework to reuse or extend the existing solution components to satisfy the changing customers' requirements.
- More work is required to define an adaptive integration action manager that can seamlessly integrate a new application into eBC environment by minimizing the code changes of the existing components in eBC infrastructure.
- Investigating the convergence of UML based model driven approach [12] and RDF based semantic approach for creating eBC solutions.
- Enabling eBC solution infrastructure in Autonomic [9] and Grid computing environment [7] for business process outsourcing, integration and collaboration [25].

Acknowledgements

We would like to thank Henry Chang and Jen-Yao Chung for their support of this research project. We also would like to thank Cristiane Hilkner and Pierre Darmon for their leadership in delivering an eBC prototype. The evolution of the on-demand business collaboration solutions underlying this work was not possible without the key contributions in design and implementation of dedicated outstanding scientists and engineers. They include Tian Chao, Shunxiang Yang, Jingming Xu, Yingnan Zuo, David Flaxer, Santosh Kumuran, Yiming Ye, Shang Guo, Yu Long, and numerous others. We also like to extend our thanks to Michael Silfen for his valuable business input in the preparation of this paper.

References

[1] S. Aissi, P. Malu, K. Srinivasan, E-business process modeling: the next big step, IEEE Computer 35 (5) (2002 May) 55–62.

[2] G. Bafoutsou, G. Mentzas, A comparative analysis of web-based collaborative systems, 12th International Workshop on Database and Expert Systems Applications, 2001, vol. 3–7, IEEE Computer Society, USA, 2001 (Sept.), pp. 496–500.

[3] D. Brickley, R.V. Guha, RDF Vocabulary Description Language 1.0: RDF Schema, http://www.w3.org/TR/rdf-schema/ (2003 Sept.).

[4] Cadence White Paper on Design Chains, http://www.cadence.com/.

[5] DAML-S: DAML Services, http://www.daml.org/services/.

[6] ETTK, Emerging Technologies Toolkit, IBM alphaWorks, http://www.alphaworks.ibm.com/tech/ettk.

[7] I. Foster, C. Kesselman, J.M. Nick, S. Tuecke, Grid services for distributed system integration, Computer 35 (6) (2002) 37–46.

[8] IBM, Microsoft, SAP, and Siebel Systems, Business Process Execution Language (BPEL4WS), http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.

[9] J.O. Kephart, D.M. Chess, The vision of autonomic computing, IEEE Computer 36 (1) (2003) 41–50.

[10] A. Kumar, J.L. Zhao, Workflow support for electronic commerce applications, Decision Support Systems 32 (2002) 265–278.

[11] OASIS eXtensible Access Control Markup Language, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml (2003).

[12] OMG Model Driven Architecture, http://www.omg.org/mda/.

[13] RDF: Resource Description Framework, http://www.w3.org/RDF/.

[14] RosettaNet, http://www.rosettanet.org.

[15] R. Schmelzer, et al., XML and Web Services Unleashed, SAMS Publishing, Indiana, USA, 2002.

[16] WebSphere Business Integration Connection, http://www-3.ibm.com/software/info1/websphere/index.jsp?tab=products/businessint.

[17] WebSphere software platform, http://www-3.ibm.com/software/info1/websphere/index.jsp.

[18] WS-I, Web Services Interoperability Organization, http://www.ws-i.org.

[19] WSOM, Web Services Outsourcing Manager, http://www.alphaworks.ibm.com/tech/wsom/, IBM developerWorks USA (2002).

[20] L.-J. Zhang, M. Jeckle, The next big thing: web services

collaboration, Proceedings of the 2003 International Conference on Web Services – Europe (ICWS-Europe'03), LNCS 2853, Spring-Bergle, Sept. 23–24, 2003, Erfurt, Germany.

[21] L.-J. Zhang, H. Chang, T. Chao, J.-Y. Chung, Z. Tian, J. Xu, Y. Zuo, S. Yang, Q. Ao, Web services hub framework for e-sourcing, in: IEEE Conference on System, Man, and Cybernetics (SMC'02) vol. 6, IEEE, USA, 2002, pp. 163–168.

[22] L.-J. Zhang, H. Chang, T. Chao, Web services relationships binding for dynamic e-business integration, International Conference on Internet Computing (IC'02), Las Vegas, CSREA Press, USA, 2002, pp. 561–567.

[23] L.-J. Zhang, T. Chao, H. Chang, J.-Y. Chung, XML-based Advanced UDDI Search Mechanism for B2B Integration, Electronic Commerce Research Journal, (2003) 25–42.

[24] L.-J. Zhang, B. Li, T. Chao, H. Chang, On Demand Web Services-Based Business Process Composition, IEEE International Conference on System, Man, and Cybernetics (SMC'03), pp. 4057–4064.

[25] L.-J. Zhang, Q. Zhou, J.-Y. Chung, Developing Grid Computing Applications, IBM DeveloperWorks Journal, 2003 (May), pp. 10–15.

Dr. John Sayah is currently the Program Director of Emerging Business Opportunities Strategy for the Industry Solutions in IBM Software group. His main focus is on emerging business opportunity initiatives and linkages of research and industry teams for driving next generation solution offerings. Dr. Sayah has held several management and technical lead assignments at IBM covering a wide spectrum of application and industry areas including business-to-business and business integration, internet service provider management, and computer-aided-design applications. John Sayah received a PhD in Computer Engineering from the University of Wisconsin-Madison, an MSc in Electrical and Electronics Engineering from the University of London-UK, and a Licence D'enseignement in Physics from the Lebanese University in Beirut-Lebanon.

Dr. Liang-Jie (LJ) Zhang joined IBM China Research Lab in 1996 and is currently a Research Staff Member at IBM T.J. Watson Research Center. He is part of the e-Business solutions research team with a focus on collaborative business process integration and management innovations. He is actively creating novel business process integration solutions by leveraging and enhancing Web Services and Grid Computing technologies. Before this position, LJ was the lead inventor of IBM HotVideo technology and a key architecture board member of IBM HotMedia Product. Dr. Zhang has 5 issued patents, 25 filed patent applications, and about 70 published papers. He is an IEEE Senior Member and the Chair of the IEEE Computer Society Technical Community for Services Computing (TCSC). He is the General Chair of the 2004 IEEE International Conference on Web Services (ICWS 2004) and the General Co-chair of the 2004 IEEE Conference on E-Commerce Technology (CEC 2004). Currently, he is the Editor-in-Chief of the International Journal of Web Services Research (JWSR). Liang-Jie received a BS in EE at Xidian University in 1990 and then at Xi'an Jiaotong University an MS in EE in 1992 and a PhD in Pattern Recognition and Intelligent Control in 1996 at Tsinghua University.