Collaboration-Protocol Profile and Collaboration-Protocol Agreement in B2B transactions

Mihaela Vrhoci[1], Dubravka Đelmiš[2], Jarmila Maksimović[1]

[1]University of Zagreb
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, CROATIA
Phone: +385 1 6129-626 Fax: +385 1 6129-616 E-mail: mihaela.vrhoci@fer.hr

[2]Agrokor d.d.
Trg Dražena Petrovića 3, Zagreb, Croatia
Phone: +385 1 4894 711 E-mail: dubravka.djelmis@agrokor.hr

Abstract - Collaboration-Protocol Profile (CPP) and Collaboration-Protocol Agreement (CPA) are part of ebXML, the emerging standard intended to facilitate efficient and flexible B2B collaboration. CPP is a document that describes specific capabilities of one trading partner to engage in electronic message exchange with other partners. CPA is a document used to specify the interaction between two trading partners. CPA is derived from the intersection of two or more CPP's.

This paper describes and discusses benefits of CPP and CPA documents and their use in business transactions. Since structure of these documents is specified and they are written in XML format, in some cases it is possible to create applications that can derive CPA from two or more CPP documents automatically. Then, that document can be a suggestion for final agreement between two or more engaged parties. Example of application that can partially create CPA is given here. Application that can help creating CPA, if automatic creation is not possible, is also described.

I. INTRODUCTION

Companies must collaborate with each other extensively if they want to complete their mission. Although environment today is very turbulent there are some trends in world economy that can not be avoided. Countries and regions are being more and more specialized which brings lots of benefits to each region/country and to world economy as well. Traffic of goods is faster and cheaper and companies that are very distant from each other often collaborate. Although there are great advancements in telecommunications and information flow it is not easy to find the best business partner. Problem is searching at right places. Even if we presume that the companies know facts about each other and that they want to conduct business, unexpected problems often occure. Companies can have different application software and run-time support software from different vendors. Exchanging data and various types of documents between them can be extremely difficult if they want to integrate these capabilities in existing software.

Electronic Data Interchange (EDI) has given companies the prospect of eliminating paper documents, reducing costs and improving efficiency by exchanging business information in electronic form. But idea of ad hoc conducting business between companies without prior agreement of any kind was not possible.

In the last few years, the Extensible Markup Language (XML) has rapidly become the first choice for defining data interchange formats in new eBusiness applications on the Internet. XML enables more open, more flexible business transactions than EDI and might enable more flexible and innovative "eMarketplace" business models than EDI, as can be found in [11] and [12].

Globally accepted standard that will enable collaboration between companies is still missing. The standard should enable companies to find most suitable business partner and to establish agreement between parties in a short time and if possible automatically. Such standard should enable companies to collaborate even if they have different business applications.

ebXML is one of the emerging standards, still in the last phases of development but it conforms to all of the requirements previously specified. Many large companies and enterprises are involved in developing and evaluating this standard, which makes it very attractive for the others to embrace it.

Collaboration-Protocol Profile (CPP) and Collaboration-Protocol Agreement (CPA) are part of the ebXML standard and their main purpose is to help find the most suitable business partner and make ad hoc agreements on the electronic document communication as specified in [1] and [8]. A CPP defines the party's capabilities to engage in electronic business with other parties. Capabilities include both business processes that the party supports as well as specific technical details of supported means of message exchange. CPA is a document that represents specific terms of message exchange that is agreed between the two parties. It represents a formal handshake between collaborating parties.

II. CPP AND CPA IN COLLABORATION SCENARIO

As it was said before, ebXML is an emerging standard intended to facilitate efficient and flexible B2B collaboration. Companies of all industry branches are able to use it and it is not restricted to only large companies.

Best way of introducing ebXML and CPP and CPA is to present their usage through general scenario of collaboration between two interested parties as in [3]. To analyse usage of CPP and CPA and their possible flaws we imagined one specific scenario of collaboration between two companies in Croatia. This scenario will be described later but it is important to say that it can be generalized (like most relationships in practice) to scenario presented on Fig.1. An emphasis was given on CPP and CPA usage. There are three main entities that will take part in business collaboration: two companies (Company A, Company B) and another company or facility that is responsible for maintaining and adjusting repository and registry (Company X). That is necessary because the standard is still developing and the number of companies using ebXML is expected to encrease. Repository is specified by ebXML as large data storage, as described in [10]. Beside specific processes and profiles of organizations that decide to use ebXML benefits, there are specifications of ebXML and core components that companies can use for generation of their specific business processes and profiles. The idea is to store applications that could help companies to build their profiles and also specify supported business processes in short time. This paper contains descriptions of such applications dealing with CPP and CPA.

Registry is an index of data stored in repository. The idea is that there would not be one repository but, for example, every country could have its own repository. Those repositories are going to be connected as presented on Fig.1. That will enable companies to establish connection with possibly distant business partners.

First step that company has to do, if it wants to take advantage of ebXML, is to build its CPP. Since CPP is referencing specific business processes that are supported, those documents must be built as well. Company then registers its CPP within the registry along with documents specifying business processes. It can also store there information of products, services, quality, quantity, prices and other information which can be found by possible interested parties. In our example Company B searched registry, which enables discovery down to the atomic data level, and retrieved information about possible business partners (step two). Presuming that Company B is interested in collaboration with Company A, it needs Company's A CPP to see how to communicate with it. CPP must be retrieved from the repository (step three). Since Company B has both CPP documents it can generate CPA that satisfy other party's requirements (step four). This paper (chapter 4) describes applications that can help build CPA in short time. Then Company B suggests CPA to Company A, which can agree on it or it can propose some changes. That interaction can proceed until both sides are satisfied with CPA (step five and six). It is important to notice that companies communicate through interfaces and that they can have totally different internal business applications and be unaware of it. After final agreement has been made, which actually represents formal handshake between involved parties, both sides must adjust their business service interfaces to be compliant with agreed parameters.

Company can not be restricted to only one business collaboration at the time. Agreed means of message communications can be different for communicating with different parties. That is why interfaces must recognize and distinguish messages from different parties and manipulate them as agreed in CPA. Identifier placed on the beginning of each message solves that problem.

Business service interface must adjust data in received business documents so internal business applications could interpret it.

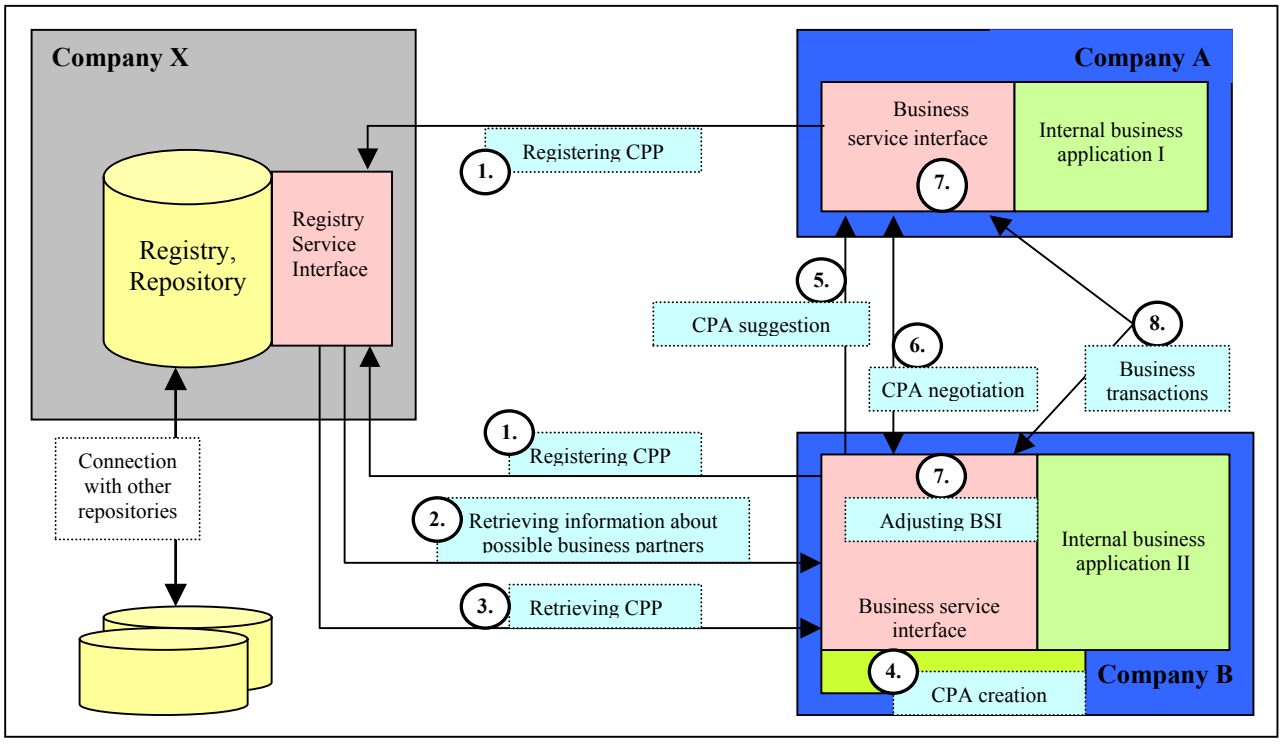

Fig. 1. General scenario

All documents that are exchanged between business partners (like purchase orders, acknowledgements of purchase order, invoices, notices about credits, etc.) have their electronic XML version developed by Open Application Group. They can be found in [6]. Since they are developed and evaluated by many companies, from different industry branches, ebXML embraced their solutions.

Most of internal applications, actually Enterprise Resource Planning (ERP) applications, are able to interpret documents in XML format, so development of business service interface is even easier.

Our scenario included companies Agrokor and Podravka. They exchanged various documents, needed to buy or sell goods, for instance: purchase order, acknowledge purchase order, ship notice, invoice, etc. In order to agree their communication we have built a CPP for each company according to the ebXML specification. Detail structure of general CPP and example for one of the companies will be given in the next chapter.

III. CPP AND CPA

CPP and CPA are XML documents with elements specified in Collaboration-Protocol and Agreement Specification, Version 1.0. Specification was studied and CPP documents for two companies, Agrokor and Podravka, involved in our scenario of starting up a communication and doing business, were made [5]. Idea was that one company acts like a buyer and another like a seller of some goods. Elements of a general CPP will be explained and examples for some of them, that are part of specific CPP for Agrokor, will be presented.

CPP contains basic information about a business partner. It includes contact information, industry classification, business processes, interface requests and messaging service requests. CPP and CPA describe details about transport, message exchange, security restrictions and connection with business process specification.

CPP is an XML document containing elements that describe the processing of a business unit. Required elements within CPP structure are:

- PartyInfo,
- Packaging

The company might want to distinguish its departments considering their roles in possible business collaborations. Location of its sections might also be a reason for different representation within CPP document. CPP can have multiple PartyInfo elements and each party is then represented by a separate PartyInfo element. CPP may be digitally signed, in which case CPP contains ds:Signature element. The XML Digital Signature Specification defines the content of this element and its subelements.

PartyInfo element is the main element of a CPP document. It describes supported business collaborations, role in the business collaboration and the technology details about message exchange.

If CPP contains more PartyInfo elements we can differentiate them by their PartyId. PartyRef element provides a link to additional information about company, for example, the company's web site. Layered structure of the PartyInfo element is presented in Fig.2.

ProcessSpecification layer defines the services (business transactions) that business partners can request of each other and transition rules that determine the order of requests.

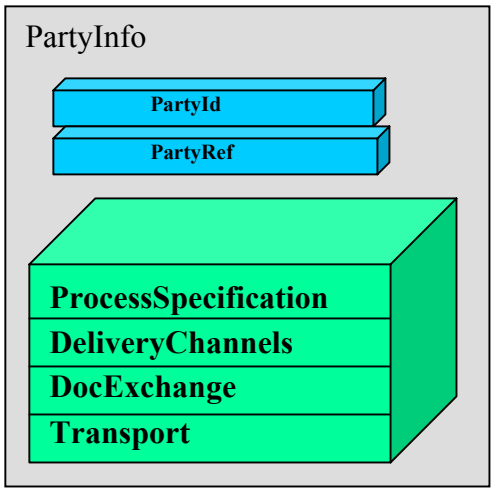

Fig. 2. Layered structure of CPP

Process specification document is a separate document that defines interaction between partners. CPP and CPA include references to that document. Generally, the process specification defines role that each company plays. For example, company could be a buyer, seller, deliverer of goods, insurance company etc. Process specification also specifies business collaborations and their parts, business transactions. Business transaction represents business document and response to it, as specified in [7]. In example presented in Fig.3., company Agrokor acts as a buyer.

```
<tp:ProcessSpecification tp:version="1.0"
      tp:name="buySell"
      xlink:type="simple"
      xlink:href="http://www.agrokor.hr
        /buySell.xml"/>
<tp:Role tp:name="buyer"
      xlink:type="simple"
      xlink:href="http://www.agrokor.hr
       /buySell.xml#buyer"/>
```

Fig. 3. Reference to supported business process and role that company plays in it

DeliveryChannels layer describes the characteristics of Message-receiving. CPP can have several delivery channels. It consists of one document-exchange definition and one transport definition. The subelements of DeliveryChannel element describe the security characteristics and other attributes of the delivery channel.

DocExchange layer accepts the business document of one company from ProcessSpecification layer, encrypts and/or adds a digital signature (based on specification), and sends it to transport layer for transmission to another partner, which is described in [4]. Reverse steps are realised with received document. DocExchange element includes the message-exchange properties. If reliable message delivery is required, this layer is added and it controls the order of message receiving.

Transport layer is responsible for message delivery using the selected transport protocol, which is specified by [9]. Transport element defines communication protocols, encoding and transport security information. In the following example, (Fig.4.), we can see that the company's protocol for sending messages is FTP. In our application, described later, another company does not support FTP but HTTP. If company supports HTTP and wants to conduct business with another company that supports only HTTP, SendingProtocol element in CPA document should be changed to HTTP.

```
<tp:Transport tp:transportId="N02">
        <tp:SendingProtocol
           tp:version="1.1">
            FTP
        </tp:SendingProtocol>
          <tp:ReceivingProtocol
             tp:version="1.1">
              HTTP
          </tp:ReceivingProtocol>
        <tp:Endpoint
           tp:uri="http://www.agrokor.hr"
           tp:type="allPurpose"/>
        <tp:TransportSecurity>
          <tp:Protocol tp:version="3.0">
            SSL
          </tp:Protocol>
        </tp:TransportSecurity>
</tp:Transport>
```

Fig. 4. Example of transport layer within CPP

Packaging element provides information about message header packaging for transmission over the transport layer. It is consisted of elements that describe MIME type, supported namespaces, grouping of message parts, and encapsulation.

CPA describes terms of interaction between the business partners and the way these interactions are handled. It is result of intersection of two CPPs and contains common, or compatible, elements between partners. The structure of CPA consists of:

- general information about this document,
- two PartyInfo elements, one for each business partner,
- Packaging element

CPA may also be digitally signed like CPP, in which case it includes ds:Signature element. If partners do not agree about some values in CPA, they can negotiate over it. Negotiation is a process of exchanging CPA document untill both business partners agree on the content.

General information of CPA describe the status of process creation, date and time when CPA starts to be valid, date and time when CPA must be renegotiated by business partners, and limits of conversations. Fig.5. illustrates these information.

```
<tp:Status tp:value="proposed"/>
<tp:Start>2001-12-21T09:00:00Z</tp:Start>
<tp:End>2002-12-21T09:00:00Z</tp:End>
<tp:ConversationConstraints
  tp:invocationLimit="100"
  tp:concurrentConversations="100"/>
```

Fig. 5. Example of general information of CPA

PartyInfo elements have the same structure like corresponding PartyInfo element contained in CPP, but if there was negotiating involved, values of some elements could be changed.

Packaging element must be the same as corresponding element in CPP. There is only one Packaging element in CPA and it has to match at least one element in each of basic CPP documents.

When CPA is agreed, both companies use identical copies of CPA to configure their run-time systems, actually to adjust previously explained business service interfaces.

IV. APPLICATIONS USED FOR MANIPULATING CPP AND CPA

Description of applications that we built for manipulating CPP and CPA documents is given here. Considering wide range of companies that are expected to use ebXML in their business activities and time that they should spent for building CPP and CPA, applications that would speed up the process would be very useful. Two applications for this purpose have been built.

A. Presenting information contained in CPP and CPA

For simple review of business profile and agreement, an application for displaying CPP and CPA elements was made. For the regular user, XML document is not easy to read. With this application, it is very easy to find some information or elements of CPP or CPA.

Graphic user interface (GUI) for CPP and CPA was made first. Swing components and Swing containers were used for the purpose of reviewing specific elements. Textfields were used for presenting values of elements.

Our idea was to enable the user, who is creating CPA, to have a parallel view of picked elements of CPP documents belonging to the companies interested in business collaboration. Application should make it easy for the user to compare elements. Fig.6. presents the interface for reviewing CPP. A button called 'Vanjska informacija' provides direct connection to the company's web site. Clicking the button, Internet browser opens and the company's web site appears.

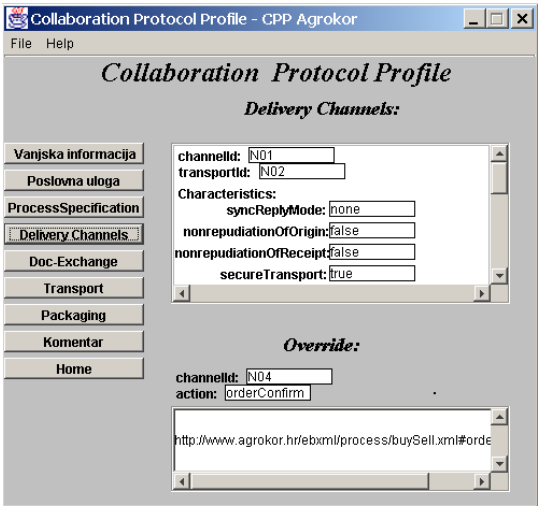

Fig. 6. Graphic user interface for CPP reviewing

By clicking any button on the left side of the application interface, specific information is presented. Fig.6. illustrates the information contained in the DeliveryChannel element of CPP specified for the company Agrokor.

To connect GUI and XML document, which is retrieved from Oracle database, XML document must be parsed. For that, Dom parser was used as instructed in [2]. Result of parsed document is a series of nodes from where interested elements or attributes could be reached and placed in Textfield inside GUI.

The method for reaching values of attributes is presented in Fig.7. Inputs of the method are the names of element and its attribute whose values we are interested in.

```
public String getAttributeValue(
  String elementName,
  String attributeName
)
{
  NodeList nl =
   doc.getElementsByTagName(elementName);
  NamedNodeMap atr_list =
   nl.item(0).getAttributes();
  return
     atr_list.getNamedItem(attributeName).
     getNodeValue();
}
```

Fig. 7. Code for reaching values of specified attribute

CPA interface was built similar to the CPP. Searching through CPA and seeing all agreed information for exchanging business documents is enabled. It can be seen that some elements are not the same like in CPP. For example, previous mentioned SendingProtocol element in one CPP had FTP value, but in CPA it is changed because another business partner does not support FTP, only HTTP. The change is presented and easily found in interface illustrated in Fig.8.

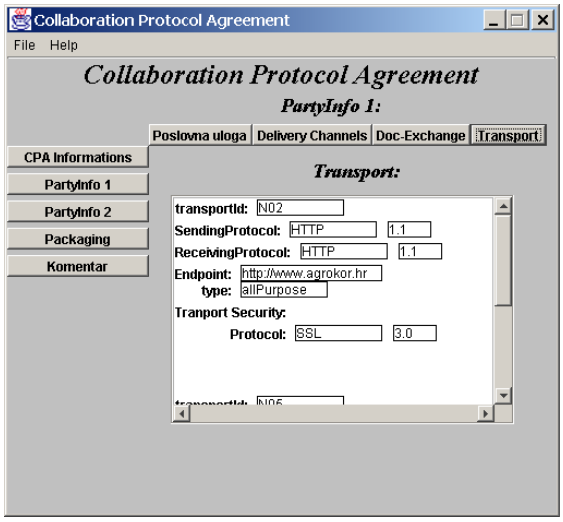

Fig. 8. Graphic user interface of CPA

B. CPA generation

Detailed procedures for CPA formation from CPP documents are currently left for implementers and no normative specification is provided in that area. There are few reasons for that.

One is that company could be capable of using one mean of communication but for some reasons (security, scalability etc.) does not want to advertise it in its CPP. However, if it does not want to refuse very attractive offer, it could agree on using that mean of communication that is not specified in its CPP.

In some cases optimal solution that satisfies every level of requirement and every other conditional constraint does not exist. Compromises may be made on security, reliable messaging, levels of signals, acknowledgements and other matters in order to find some acceptable means of doing business. Then the user (human) must decide what is acceptable and change some values directly in CPA. That can not be done automatically.

There are two main approaches of CPA forming. One involves creating CPA from a CPA template. That can be used when the capabilities of one party are limited and already known. Then the CPA would already contain information about one side and interaction. There would also be placeholders for other party's information.

Other approach is to derive CPA from two or more known CPP documents. In our scenario we used that way of building CPA. Since specification of CPP and CPA is still under revising and some elements within them could change, we have built an application that generally works with any documents in XML format. Interface of that application is presented in Fig.9. Location of two CPPs, in this example CPPAgrokor and CPPPodravka, must be specified first. Then one or more elements that XML documents contain can be entered. Application will find all matching elements. Fig.9. illustrates finding the same Packaging element within documents CPPAgrokor and CPPPodravka.

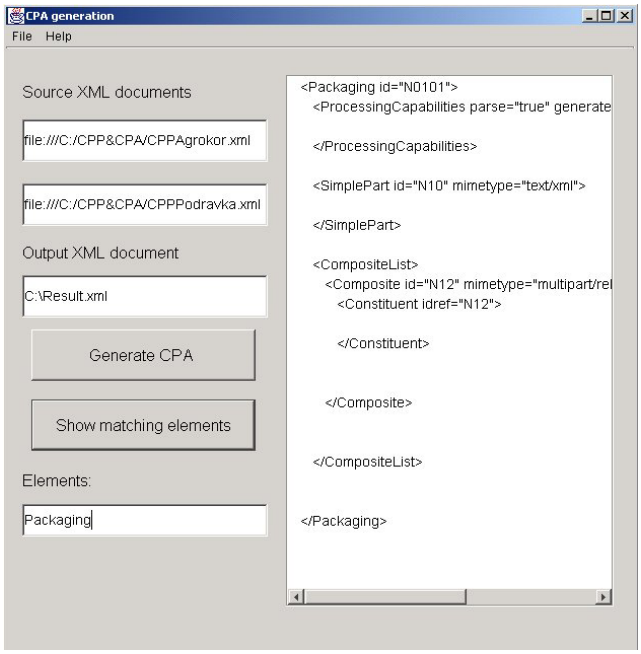

Fig.9. Automatic creation of CPA elements

It is possible to choose appearance of the result. Result can be displayed for user to see it, or it can be written in specified XML document.

Automatic finding matching elements could speed up the process of forming CPA so it could be very useful. User would not be compelled to go through CPP and read one element at the time and compare it with elements in another CPP. After finding the same corresponding elements, the user can easily insert it at the right place inside CPA.

Application can also be used for creating main structure of CPA. After that, elements could be easily inserted into it. Created root element of CPA is presented in Fig.10.

JBuilder 5 was used for building application. Sax parser was used for XML document parsing. Graphic user interface was made by Swing components.

```
xml version="1.0"?>
<tp:CollaborationProtocolProfile
xmlns:tp="http://www.ebxml.org/namespaces/tradePa
rtner"
xmlns:xsi="http://www.w3.org/2000/10/XMLSchema
-instance"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xsi:schemaLocation="http://www.ebxml.org/namespa
ces/tradePartner
http://ebxml.org/project_teams/trade_partner/cpp-cpa-
v1_0.xsd"
tp:version="1.1">
</tp:CollaborationProtocolProfile>
```

Fig. 10. Root element of CPA

V. CONCLUSION

Companies often depend on each other and are compelled to collaborate. Finding the best partner at specific moment could be quite a challenge, since prices of products change, quality changes, new companies with new technologies appear on the market etc. After finding the possible partner, it takes significant time to agree on the terms of the electronic dialog.

CPP and CPA documents specified within the ebXML standard, which is developed and tested by many companies from all over the world, are used to make ad hoc agreements on the electronic document communication between interested parties.

We investigated CPP and CPA documents and their use in one scenario of business collaboration.

CPP and CPA documents are written in XML format. Elements within them form a layered structure analogous to real communication requirements. XML format is a good choice. It enables easy understanding and interpretation of elements and attributes. It is flexible enough to accept different changes and it is easy to manipulate with. We proved it by creating applications for easy reviewing the information contained in CPP and CPA documents. Application that can compare two CPPs and build XML document with CPA structure was also presented.

Since CPP and CPA documents improve communication between interested parties and are easy to manipulate with, we expect that the idea of such documents will be embraced and used by many companies in very near future.

REFERENCES

[1] *ebXML Collaboration-Protocol and Agreement Specification*, http://www.ebxml.org.

[2] B. McLaughlin, *Java and XML*, 1st Edn., O'Reilly & Associates, Sebastopol CA, 2000.

[3] I. Matasić, and M. Šimunić, "EbXML standard for B2B transactions", unpublished, 2002.

[4] I. Magdalenić, I. Pejaković, and D. Pranić, "Business documents presentation and interchange using SOAP", unpublished, 2002.

[5] M. Topolnik, D. Pintar, and M. Sokić, "Experimental Implementation of Emerging e-Business Technologies: ebXML and PKI", unpublished, 2002.

[6] http://www.openapplications.org/

[7] ebXML Business Process Specification Schema, http://www.ebxml.org.

[8] ebXML Glossary, http://www.ebxml.org.

[9] ebXML Message Service Specification, http://www.ebxml.org.

[10] ebXML Registry Services Specification, http://www.ebxml.org.

[11] ebXML Technical Architecture Specification, http://www.ebxml.org., 2001.

[12] Extensible Markup Language (XML), World Wide Web Consortium, http://www.w3.org.