RC22608 (W0210-172) October 28, 2002
Computer Science

IBM Research Report

A Conceptual Framework for Electronic Contract Automation

Heiko Ludwig
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

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

A Conceptual Framework for Electronic Contract Automation

Heiko Ludwig
IBM T. J. Watson Research Center
Yorktown Heights, NY, USA
hludwig@us.ibm.com
http://www.research.ibm.com/people/h/hludwig

October 29, 2002

Abstract

The contracting process is the mechanism by which organizations enter into and manage business relationships. Establishing business relationships in a context of electronic services requires an electronic and, in particular, partially automated implementation of this process. Network-enablement and automation require many issues being addressed such as representation,signature, legal validity, decision-making and more. This report introduces a conceptual framework of the contracting process to enable the systematic analysis of its automation issues.

1 Introduction

Today's networked economy allows customers to obtain information about products and services from providers around the globe. Customers also expect to obtain these goods or services instantaneously, in particular if those services that are delivered electronically, e.g., application hosting services, or whose delivery can be managed electronically, e.g., logistics services such as parcel delivery. This entails that customer and provider of a service can negotiate their relationship over a network in an efficient manner and that they can set up their internal systems to deliver and consume the service that has been promissed in their agreement. Today, this process is automated only for highly commoditized goods and services. Negotiation and fulfillment of flexible, case-by-case business relationships is predominatly established and fulfilled with significant human involvement. In this technical report, a framework for the design of electronic contracting processes that deal with the trade of between flexiblity and automation is proposed.

1.1 Electronic Contracts in an Electronic Commerce Environment

When independent parties - organizations or consumers - enter a business relationship, a *contract* between these parties is being established. The contract defines the mutual rights and obligations of the parties in this relationship, e.g., which services to perform, which goods to deliver, and how much to pay. In many legislations, there are no particular formats prescribed in which a contract has to be expressed, for most matters, anyway. What establishes a legally binding contract as such is the *process* in which the contracting parties express their will to enter into a concordant agreement [11]. This contract establishment process involves at least the steps of offer and acceptance, where the offer contains the content of the proposed contract.

Contracts can be established verbally, in writing, or in an electronic format. If a contract has an electronic representation we call it an *electronic contract*. This term applies to contracts containing just natural language, formal, i.e., machine-interpretable language, or both. In many cases, contracts are between two parties, which we call *provider* and *customer*. In some domains multiple parties enter a contract or are

referred to in it. This is the case, for example, in the domain of digital rights, where a rights owner licenses playing rights for some media to a customer and a named set of parties are granted the right to sub-license from the primary licensee. However, we will focus on bilateral contracts for most of this report.

The (full) *contracting process* is the process in which contracts are negotiated, established, and the fulfillment is managed. This contracting process includes, for example, the advertisement of a contract for goods or services by a provider, the search for suitable providers by a customer, the negotiation, the signing procedure, the planning of the fulfillment process by the provider, a service tracking process by a customer, a payment process, and an analysis of a contract post execution. Figure 1 illustrates the contracting process on a high level.

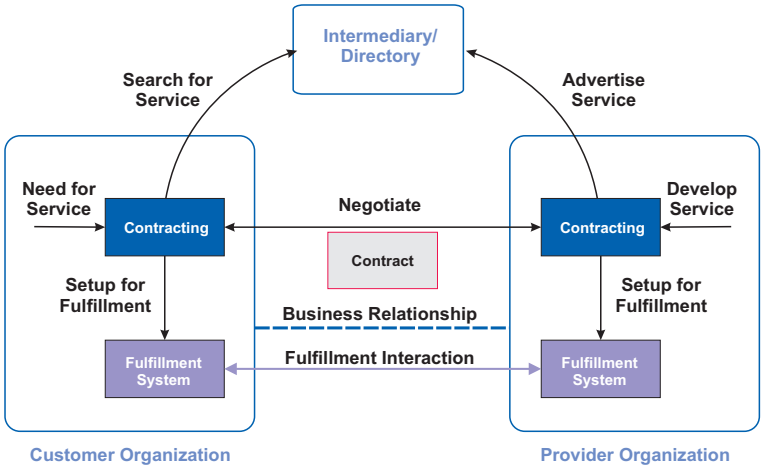

Figure 1: Overview of the contracting process.

The contracting process in figure 1 involves two parties, a customer and a provider. Prior to contracting process, a customer discovers a need for a particular service or a provider develops a service. They learn about a potential for entering a business relationship through an intermediary, e.g., yellow pages or an online directory service. Then, the parties negotiate the terms and conditions of their relationship by exchanging offers, which finally result in a contract. The contract establishes a business relationship between the parties in terms of a set of rights and obligations of those parties. This contract governs their further interaction to actually deliver and consume what is promissed in the contract at fulfillment time. Having signed the contract, the two parties have to prepare their fulfillment system accordingly. Depending on the type of service, this might be very different from case to case. If the service consists of sending mail ordered books to a consumer, it has to be initiated that the book is taken from the shelf, packaged and sent. The customer has to prepare to receive the book, e.g., make sure that somebody is at home when it is delivered. When all obligations have been fulfilled, the business relationship is dissolved.

Conceptually, we distiniguish a *contracting function*, which implements the contracting process, from the *fulfillment function* that implements the "core" service-implementing and service consuming functions of an organization. In the sequel of this report, we will further discuss the structure of the contracting functions of providers and customers, their interaction among each other, and the interaction with the fulfillment system and the rest of the environment, to work out under which circumstances parts of the contracting process can be automated. Apparently, the automation of the contracting function and the concept of electronic contracts is particularly interesting for services whose fullfilment system is automated or can be monitored and managed through a network.

If a contracting process is fully or - more likely - partially executed using a networks and computers we call it an *automated contracting process* or an *electronic contracting process*, since it will be based on an electronic contract. Apparently, the property of being automated or manual is not discrete but there is a

gradual transition between a fully manual and a fully automated process.

There are a number of benefits of automating the contracting process:

- Automating the contracting process results in increased *speed* of negotiation and setup for fulfillment.
- Another important aspect is the reduction of *costs* associated with employees throughout the contracting process. Key to the (partial) automation of the contracting bases is an electronic contract expressed in an machine-readable way to serve as input for automated parts of the contracting process.
- Once the contracting process is automated, service customers are able to buy services on much *shorter notice*, thus enjoying the benefit of deferring the buying decision to the point in time when they actually know exactly their current requirements.
- Benefiting from those reduced contracting costs and times, providers and customers can buy services at *finer granularity* or services can be offered individually that are only available in bundles in an environment of expensive contracting processes.
- Finally, the availability of fine-grained services provides *opportunities for service intermediaries* to re-bundle services from different providers into new aggregate offerings.

1.2 Example cases

We use two example cases of electronic contracting to illustrate the discussion in this report. The first case is an application hosting service that features a completely automated fulfillment system, the hosting platform. The second example is a logistics service that warehouses goods and delivers goods from the warehouse to customers. This service can be managed in an electronic way. However, since it involves the delivery of physical goods, a part of the service performed by employees.

Application hosting: A service provider organization hosts electronic commerce applications on its server cluster. Customer organizations can either run preconfigured applications such as a Web server, a shopping cart application or a payment application, or they can bring their own. The applications are accessed by users from the public Internet or from closed user groups. The service provider targets different types of customers: Small and medium-sized organizations typically buy pre-configured application services at chosen response time and availability levels. The pricing depends on the disk space used and the number of application transactions per hour up to which the service levels are guaranteed. Large customers bring their own applications in addition to the standard ones. The pricing is negotiated individually. The service provider advertises the standard packages on an electronic marketplace for application services and registers its business with the public UDDI directory.

Logistics service: A logistic service offers warehousing goods for customers and delivering these goods to final destinations, which could be either again clients of the customers or sites of customers that need supplies. An example of such a service is described in a paper on the CrossFlow project [6]. Customers can negotiate warehouse capacities, restocking procedures, and the extent to which they can intervene in the delivery process. For example, they can monitor where in the delivery process a package currently is, parcels can be sent back to a warehouse, and the delivery address can be changed while a parcel is on its way. The logistics service has an electronically managed warehouse and an automated system for monitoring delivery progress and receiving management operations through a Web interface or through SOAP over HTTP. Also, recipients can make delivery appointments through the Web if delivery failed the first time. Our logistics provider has some standard offerings that it offers to particular industries, for example, to mobile phone service companies. Mobile phones are warehoused for the mobile phone companies. If a new phone service customer is subscribed, a phone is delivered to this customer. Most parameters of such a service contract are negotiable due to the large size of the contracts.

These examples will help illustrating the discussion and analysis of the contracting process.

1.3 Objective and Structure

The objective of this report is to provide a conceptual model of the contracting process to enable the analysis of a contracting process in particular cases, analyze the potential to automate parts of the contracting function, and design automated functions of the contracting process.

We proceed as follows: In the next section we introduce our conceptual model of the contracting process, decomposing the function and interactions on the service provider and customer side. Subsequently, we address in more depth the information associated with interactions with respect to their contents and format. This includes in particular also the contract document itself. Based on this understanding of the contracting process, we analyze issues of automating the contracting functions and propose some strategies. The summary and conclusion finishes the report.

2 Conceptual Model of the Contracting Process

To understand the potential for automation of parts of the contracting process, we need to discuss it in more detail and have a good model of this process, the functions involved, and the interactions that take place.

2.1 Contracting process phases

The contracting process goes through a number of phases. There are multiple models of phases of business relationships, such as the one used in the context of the SeCo project [10], [11]. For the purposes of this text, we structure the contracting process as outlined in figure 2.

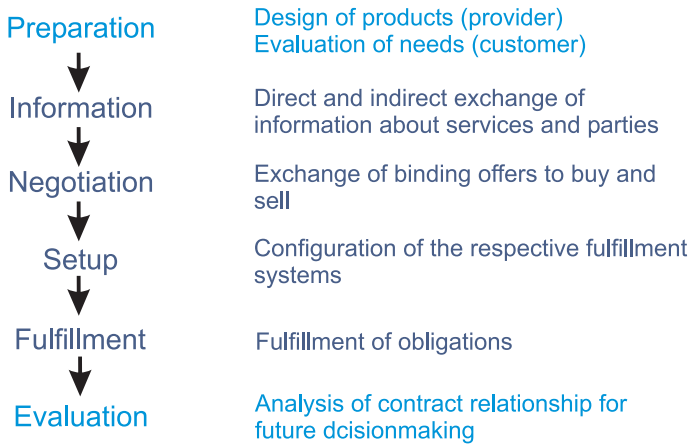

Figure 2: Phases of a Contract Life-Cycle.

The term contracting process relates to the dealing with a single contract. The *preparation phase* precedes this contracting process (hence kept in light blue, while the contracting process steps are dark). In this phase, a provider designs the service it wants to offer and takes all preparatory steps to actually provide it to customers. On the customer's side, this step contains in particular the evaluation of the needs that lead to entering the contracting process.

In the *information phase*, both parties can identify potential business partners, gather information about their counterparts and the services in question. This phase can be either conducted by directly requesting

and sending information between the parties or by using an intermediary, e.g., yellow pages or a public Web site.

During the *negotiation phase* parties exchange offers, counter-offers, acceptance, and rejection messages. This can be either conducted in a direct interaction or mediated by an intermediary. Depending on the negotiation protocol in use, this phase can lead to different interaction patterns. The negotiation phase is completed once both parties have accepted an offer.

Once a contract is signed, both parties have to prepare for the fulfillment of the contract in the *setup phase*. This comprises potentially the technical setup of the fulfillment-system, e.g., in the case of the application hosting service the system must be installed and applications started. In the context of network and application services, this step is often referred to as *provisioning*. In the case of the logistics system, warehouse space must be assigned or even be built. On the administrative side, current and future financial flows have to be accounted.

In the *fulfillment phase*, both parties manage their obligations and monitor the compliance of the other party. In general, the management of a party's obligations means providing service or a payment to fulfill the contractual obligations. However, in some cases, a party may decide to not live up to its obligations and cause a dispute. Hence, dispute resolutions are also part of this phase. When the service is completed and there is agreement that all contractual obligations have been fulfilled by all parties, the fulfillment system for this particular contract can be dismantled. In the case of the application hosting service, this may comprise disconnecting the application-hosting servers from the network such that they cannot be accessed anymore by the customer and reassigning them to the pool of servers for new contracts.

After the fulfillment phase, the contracting process is completed. Still, parties may want to analyze the performance of their contractual relationships. This is done in an*evaluation phase*, which conceptually is performed after the contracting process. Since the evaluation is not necessarily done each time a contracting process is completed but may be done decoupled from it for a whole set of contracts at a time, it is represented in a lighter color in the figure.

2.2 Contracting function of a service provider

With the understanding of the phases of the contracting process, we now analyze further the role of the contracting function. Providers and customers have different requirements and will be discussed separately. A decomposition of the contracting function for a provider, the flows between its components and external partners, as well as the information associated with these flows are shown in figure 3.

We decompose the contracting function of a service provider into five elements:

- The role of the *advertising function* is to deal with the information phase of contract life-cycle. It decides how to advertise a service and to whom. It also answers requests from potential customers. These advertisements are based on a description of the service to be advertised, which has been developed prior to the contracting process in the preparation phase and are input to this function. The contracting function segments the market for a particular type of service and creates advertisements that correspond to different types of customers. In the context of the application hosting example, the same application hosting service would be advertised as a service with pre-configured applications to small and medium sized businesses and as a bring-your-own-application service to enterprise customers. The advertising function sends the advertisement directly to potential customers and to intermediaries such as directories or online marketplaces.

- *The negotiation function* negotiates contracts with potential customers, hence it corresponds to the negotiation phase of the contract life-cycle. It receives the set of current advertisements as input. The negotiation function comprises two parts: (1) *Interaction*; The negotiation is conducted by exchanging offers and counter-offers with potential customers. If an offer is accepted a contract is established. The negotiation process can be initiated either by a potential customer reacting to an advertisement or by the provider responding to a call for bids. (2) *Decision-making*; Offers are made or evaluated based on the resource situation at the time the customer wants the service, the status of the customer, e.g., the

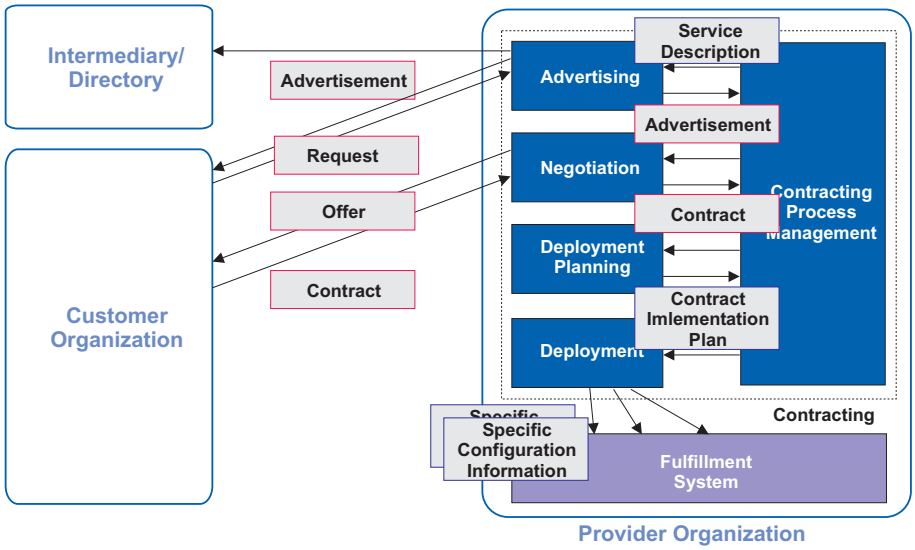

Figure 3: Provider functions and flows.

customer is very loyal and does much business with the provider, and the general market situation. It may be that because of generally low demand the provider is prepared to accept a lower offer and vice versa. Negotiations can be conducted with multiple potential customers at the same time. This function can be very complex, if the parties are very flexible with respect to what can be negotiated, or relatively simple, if the advertisements, which form the basis of the negotiation, are very specific already, e.g., a specific services bundle at a fixed price.

The decision-making problem in negotiation has been analyzed in many disciplines, e.g., operations research, economics, business administration, and psychology. A good overview can be found in Bichler [2] and Ströbel [29].

- Once a contract has been agreed upon, or signed, a provider organization must plan which resources to assign to fulfill the obligations arising from the contract, which is the *deployment planning* function. In the case of the application hosting service, this comprises the assignment of server machines, the choice of install images and the choice of network service providers. In the case of the logistics service, warehouse space must be assigned, the delivery workflow must be defined, and the provisioning of the Web interface has to be prepared. The result of this planning function is the *contract implementation plan*. In a classic production environment, this function corresponds to a new entry to the master production schedule.

- The *deployment* function executes the contract implementation plan. It creates specific configuration information for the involved automated components of the fulfillment system and executes the respective provisioning processes. For example, if the contract implementation plan foresees to provide an application service on a Linux server, it would install a Linux image and perform Linux-specific configuration steps, which are different from, e.g., the provisioning process of a Windows 2000 server.

  If the service involves elements performed by employees, the service process is started and the employees are informed about their task assignments.

- The *contracting process management* function manages the execution of the contracting functions. The contracting functions are not necessarily executed in a strict order. Advertisement is done in many cases far prior to negotiation. Also, deployment planning and the deployment itself can be separated

if services are purchased in advance. In addition, the contracting process management maintains the state of the contracting processes and stores the corresponding documents.

The contracting function deals with a number of documents (conceptually, not necessarily in the form of document files). Advertisements, requests, offers, and contracts are external documents that are exchanged with customers and intermediaries (red outline in figure 3). The contract implementation plan and the specific configuration information is internal to the service provider.

2.3 Contracting function of a service customer

The contracting function of a service customer can be decomposed in a way similar to the service provider's but the individual sub-functions are partially different. Functions and interactions are outlined in figure 4.

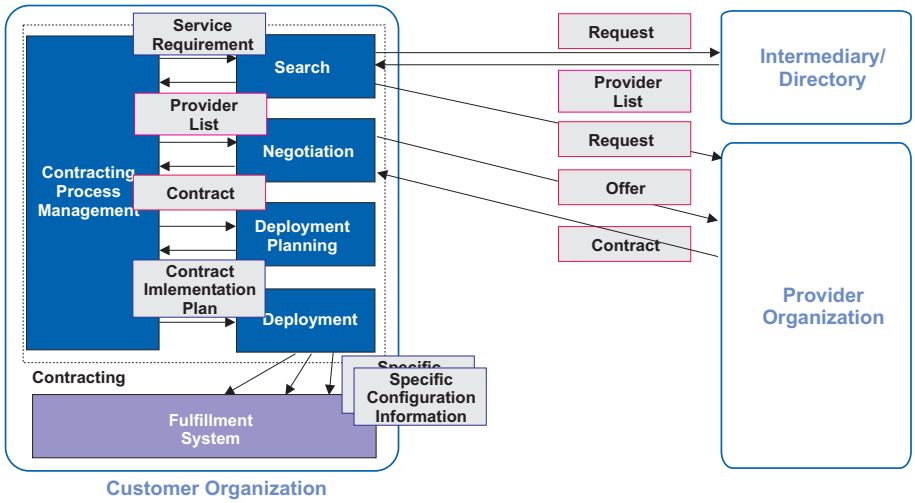

Figure 4: Provider functions and flows.

- In the information phase, the *search function* identifies provider organizations that can potentially address a customer organization's *service requirements*. It sends *requests* to intermediaries, e.g., directory services, which yields a list of service providers that meet the criteria specified in the request. In addition, as discussed in the previous sub-section, a request can be sent to a service provider directly to receive an advertisement. The result of the search is a list of suitable providers for a particular service requirement (*provider list*). This function can be designed in different ways. It can rely entirely on a particular single directory service for finding its partners or inquire many different sources. For each particular service request an external search can be performed or search results and provider advertisements can be stored locally and particular service requirements can be linked to providers from local information.

- The *negotiation function* receives a list of suitable providers and the service requirements. Its objective is to negotiate a contract with a provider and hence it corresponds to the negotiation phase of the contract life-cycle and organizationally to the purchasing department within an organization. The customer's negotiation function is the counterpart of the provider's and interacts with it through offers that establish a contract on acceptance.

  Like the provider's negotiation function, the customer's comprises an interaction element and a decision-making element. The decision-making of the customer takes into account the service requirements

and the constraints of providers to fulfill these requirements. Requirements and constraints relate to different attributes such as service time, quantity (application service accommodating 500 transactions per hour), quality (average response time less than 2 seconds), and price. All of these attributes have to be taken into account when evaluating an offer and submitting a counter-offer.

- The customer's *deployment planning function* is different from the provider's in that it deals with the setup of the service consumption fulfillment system, rather than the service-providing one. In the case of an application service, configuration information for service clients has to be prepared and roll-out procedures have to be designed. In the case of the logistics service, a warehouse customer has to list the suppliers that need to be informed to deliver to the service supplier's warehouse. In addition, e.g., in case of a customer being a mobile phone service company, internal business processes must be changed such that delivery requests are routed to the service provider. For this purpose, e.g., the mobile phone company's CRM application may have to be reconfigured to trigger a service invocation to a provider. On the administrative side, the customer has to plan for the payments due in the course of the service. The result of the planning is the contract implementation plan.
- Like in the case of a service provider, a customer's *deployment function* executes the contract implmentation plan. For this purpose, it executes a process that configures the elements of the fulfillment system with specific configuration information. It also encompasses the administrative measured foreseen in the contract implementation plan, e.g., opening an accounts payable position in the creditor accounting system.
- The *contracting process management* function facilitates the interaction among these functions and implements the contract life-cycle by involving them. Like in the provider's case, the contracting process management functions maintains the state of contract life-cycles.

The documents involved in the customer's contracting process are similar to the provider's. To be added are the internal service requirements document and the external request document.

3 Document Structures and Formats

Having outlined a model of the contracting functions of a service provider and the service customer, we can discuss in more detail the artifacts involved in the contracting function, in particular the contract itself.

The internal information, service description, service requirement, contract implementation plan, and the specific configuration information for the fulfillment system carry the information needed to fulfill their purpose and, since they are internal to an organization, do not need to adhere to an agreed format. In most cases, these internal information items are not literally one document but are available from many different sources. However, looking towards the automation of contracting functions, formats that can be shared among applications implementing (parts of) contracting functions are beneficial. The formats of the external documents, however, must be well understood by all involved parties, including intermediaries.

3.1 Information phase documents

The *service description* is the internal representation of what type of service can be offered to customers and allows the advertising function to combine it with market knowledge and create advertisements. This encompasses the information that would be shared with customers, e.g., in the case of application hosting service, the type of application, the network connections, etc., but also internal details of the service, e.g., that Linux is used as server OS, which types of computers are deployed, and the costs associated with resources used in a service.

On the customer side, the *service requirement* information must contain the information about the type of service needed, when, how much, in which price range, and other criteria.

The *advertisement* is the document type by which a provider publishes information about a service to potential customers, either directly or through an intermediary. The items published in the service are those that a service provider deems relevant for the decision of a potential customer to start negotiating. This may include properties of the provider organization and its service as well as constraints on the customer, e.g., the type of customers for which this particular service is suited [9]. The difference between the content of an advertisement and an offer is on two aspects: (1) Not all details of an offer may be in the advertisement if they are not considered decision-relevant for negotiation, e.g., the "small-print" of an offer. (2) A provider may include information that is not part of an offer but may be relevant for the decision to start negotiating. Examples are credit ratings of a provider or reference customers. While offers are legally binding, advertisements are "invitations to treat" and therefore not directly binding [11]. They can entail some, weaker liability in some legislations.

Since the advertisement is the first point of contact between a provider and a customer, it must be in a format that can be interpreted by the customer without further agreement. What makes a format suitable depends largely on the context in which it is used. If advertisements are scanned by people, HTML Web pages, potentially indexed by search engines, may be good. If the matching process should be automated, a more formal representation of the advertisement content is necessary. For example, if providers of Web services advertise to UDDI [17], they describe the "businessEntity", which is the entry describing a provider, and the "businessService", which is the service itself, with a number of parameters. To interpret these parameters, their semantics, the ontology of this domain, must be generally known, either imposed by an intermediary such as marketplace provider or by a domain-specific standards body. The virtual marketplace system ViMP is an example for a technology for intermediary-managed ontologies [9]. Standards for classifying goods and services are, for example, the ECCMA Universal Standard Products and Services Classification scheme (UNSPSC) [8].

Like the advertisement depends on the target of advertising, the content and format of a *request* depends on the system that is requested. It can be as simple as a keyword search to Google and an email to a sales representative of a provider or as complex as a query to a virtual marketplace or a directory service. The request process could extend multiple phases, e.g., for step-wise refinement of search criteria and revelation of information.

The result of a search for service providers is a *provider list*. The provider list could be ordered, e.g., by the extent to which provider advertisements fulfill the search criteria.

Advertising and search functions of providers and customers may want to support a multitude of formats to be able to use multiple intermediaries and potential business partners. This requires not only to observe different syntaxes but also dealing with different ontologies.

3.2 Negotiation phase documents and the contract

The documents of the negotiation phase are *offers* and *contracts*. An offer is a binding proposal for a contract. Its content corresponds to the content of a contract. However, a part of its content can still be kept flexible, to be chosen, or filled in, by the other party within the limits specified. Also, offers usually have an expiration date. Acceptance of an offer, which may entail deciding on the options of the offer, establishes a contract. The acceptance notice is not shown in the figures of the previous section because it usually corresponds to the contract. Although, as mentioned in the introduction, the intent to enter into a contract can be expressed in many ways, e.g., verbally, it is common practice on many occasions to sign written contracts, which is easier to use in court in the case of dispute. Since electronic signatures are recognized in the legal process of many countries today, e.g., in the United States and the members of the European Union, there is little obstacle for expressing offers, and thus the contract, in an electronic format. An interesting example of a system to exchange offers and sign contracts electronically is the Secure Contract Container (SeCo), which was developed in a project involving the Universities of St. Gallen and Zurich and the Zurich Chamber of Commerce [24], [12]. A similar approach was chosen by the COSMOS system [22].

Given the absence of formal requirements for the actual *contract content and format*, the content can contain the required information in a format suitable for those interpreting the contract. Since the contract

is used to set up the fulfillment system, it must be understood by the deployment planning function and, possibly later, the part of the fulfillment system dealing with a dispute.

The content of the contract defines the rights and obligations of the contracting parties [10]. Conceptually, on an abstract level, the contract comprises three types of information elements:

1. *Description of the parties.* This comprises all relevant properties of parties, e.g., name and address, but also technical properties such as interfaces, if applicable. This comprises primarily the parties signing the contract but can also extend to "third" parties that are just mentioned, e.g., a public directory service to be used.

2. *Description of the rights and obligations.* This is core of the contract. It describes which party must perform an action (deliver a parcel), achieve a particular state (average response time is less than 5 seconds), or allow another party to do something (use intellectual property, e.g., play a piece of music), which is a right of the benefiting party.

3. *Definitions to establish a shared ontology.* It is beneficial if all parties to the contract interpret the rights and obligations in the same way. This reduces dispute handling effort at fulfillment time. Specifying the rights and obligations unambiguously requires a common ontology between the contracting parties. In many cases, this common ontology cannot be assumed and thus has to be defined within the scope of the contract. In our application hosting example, the term average response time of a request over a network may be ambiguous. Is it the time that the service provider's application server needs to process the request? Is it measured from the customer's infrastructure point-of-view, including the network delays? These issues must be clarified. The definitions section can comprise a large section of the contract.

If the deployment planning and deployment functions are fully performed by people, natural language is apparently the most convenient way of expressing the contact content. However, as we are looking in our example as, at least partially, automated services, we also want to automate deployment planning and deployment to some extent. What are the options of formalizing contact content?

- The simplest approach to formalization is to apply *structural markup* to a natural language document. One can sub-divide the contract into section, clauses and so on. This markup can be enriched to carry some specific semantics in labels such as "preamble" and "termination clause".

- *Name-value tuples* are a simple and convenient way to express formal content. In the application hosting scenario, quality of service guarantees could be expressed as

  ```
  average_response_time = 1.5
  availability = 0.98
  ```

  but also could carry more complex content such as

  ```
  customer_address = "235 W 102nd Street, New York, NY 10025, USA"
  ```

  as is potentially required in the logistics example. The contract-interpreting functions must understand the semantics of the field names and be able to interpret the content. As such, the name-value tuple approach is an extension of the structural markup by detailed semantics of fine-grained fields, as opposed to rough structuring of content. Name-value tuples can be used for every aspect of a contract, party description, ontology, and obligations.

- Beyond name-value tuples, contractual content can be defined using a more *complex format or formal language*. We can separate the issues of defining the obligations and defining the common ontology, the shared definitions of the parties.

In the *definitorial part* of a contract, we have a wealth of formal languages to support the definition of those terms to be used in specification of the rights and obligations. A very general approach to defining terms are models and languages to describe ontologies, e.g., DAML and OIL [4], which are based on the Resource Description Framework [32] and underlie the World Wide Web Consortium's approach for semantic markup of resources. Using these approaches, one can define classes, properties and sub-class and instance relationships. Beyond this general approach, we can use *specialized description languages*, which exist already in many cases. For example, to describe an interface of a Web service, e.g., in the application hosting scenario, we can use WSDL [17] or CORBA IDL. For common process descriptions, the Business Process Execution Language for Web Services (BPEL) can be used [5]. An early approach to inter-organizational process descriptions, based on Petri-nets, was developed by Lee [18], another one by Dan et al. [7]. Some *formal agreement languages* for particular domains have means to describe contract ontology: The Web Services Agreement Level (WSLA) language contains a model and syntax to define how quality of service parameters should be measured or computed from low-level metrics [21]. The Open Digital Rights Language (ODRL) has a model to describe assets, e.g., a music file, to which rights expressions refer [15]. The CrossFlow contract language provides syntax to describe an outsourced business process and its transactional behavior [16].

There are a number of models and *formal languages for rights and obligations*. Several *general purpose* models and languages of rights and obligations are based on deontic logic. So propose Weigand and Xu to represent contractual obligations in Dynamic Deontic Logic [31]. In the Open Distributed Processing (ODP) environment, work from multiple groups, e.g., Cole et al. [3], led to ODP Enterprise Language, standardized as ITU-T Recommendation X.911 [14]. Despite its name, X.911 defines a model of "enterprise policies" for a "community" that can be of type obligation, permission, prohibition, and authorization, explicitly aligned with the deontic logic paradigm. Steen and Derrick propose a syntax for X.911 [25]. Obligations can also be expressed in a rich way in some some *domain-specific or subject-specific* agreement languages. This is particularly the case in the aforementioned ODRL, which knows the concept of a "right" to an "asset", and in the CrossFlow contract language, which has rich means of expressing the rights of an outsourcing customer to intervene into a business process. WSLA also has a model and language to define service level objectives, which define the assertion of a performance level for a service, and action guarantees, which are promises to perform an action if a particular condition holds, e.g., to send a notification if a service level objective is violated.

Apparently, it is difficult to formalize the entire contract content. However, this is not necessary. There will always be aspects of a contract that are to be interpreted and assessed by people. Hence, the electronic contracts that we see emerging will be *mixed documents* containing both natural language parts and formalized elements on all levels of sophisticated formal expression. Likewise, it appears difficult to envision a single formal language that suits all formalization needs. Coexisting, and possibly interrelated special-purpose languages seems to be a good approach.

3.3 Setup phase documents

The *contract implementation plan* contains three types of information : (1) Which components of the fulfillment system will be involved to either fulfill contractual obligations, to consume contractual rights, or to supervise the activities of the other contract party? (2) How to configure these components? (3) How are elements of the contract mapped onto configuration parameters of the fulfillment system?

The contract implementation plan can be defined in many different ways, depending on the particular components involved and the implementation of the deployment function. If the fulfillment system is non-technical and the deployment function is performed by an employee, natural language appears to be the most convenient way of representation. Since we are interested in electronically accessible services, we assume that a subset of the components of the fulfillment system is automated and thus can benefit from an automated deployment system, requiring a formal description of (a part of) the contract implementation plan. There are very few representations today that were explicitly designed for containing the information

of a contract implementation plan. An example is the "internal enactment specification" of the CrossFlow system, which describes the set of components to be instantiated and how to map contract elements into their configuration information [20], [13]. This language is not too complex because all components share a way to be instantiated and configured [19].

More generic approaches to describe how to set up or provision a system are scripts, workflow specifications, and policies. In scripts, configuration operations are defined as invocations of configuration programs. A workflow specification could contain a configuration activity for each component, which performs the individual configuration invocations within this activity. Thus, workflows provide a higher level of abstraction. Policies describe behavior, e.g., to prioritize network traffic. An information model for policies is defined in the IETF RFC 3060 [23]. In many cases, policies are put in a central repository where can be retrieved by components behaving according to policies. As opposed to the script and workflow case where configuration information is pushed to components, components pull their configuration information in this case.

The *specific configuration information* is the aspect of the contract implementation plan that contains the information specifically needed for a component in a format that this component understands. If the contract implementation plan already contains the configuration information in a suitable format, the specific configuration information is just a part of the contract implementation plan. Otherwise, we need a translation step as part of the deployment function.

4 Issues and Approaches in Automating the Contracting Function

In the previous sections, we got some insight in the details of the contracting process and saw its complexity. As discussed in the introduction, speed and cost are important parameters of the process. To improve those parameters is the primary target of automating the contracting function. Other parameters, which are dealt with well in many manual contracting processes, are *flexibility*, i.e., the ability to deal with a wide spectrum of contracts, and the *quality* of decision-making and execution in each function. Improving cost and speed while maintaining flexibility and quality is not easy for a number of reasons:

- Maybe surprisingly, an important issue in discussing the automation of contracting functions is the *complexity of the fulfillment system*. Sometimes, it is very difficult for service providers to understand what their fulfillment system is capable of doing. In our application service provider example, we can assume that the provider has a number of servers and networking components available to fulfill customer contracts. It is not trivial to decide how many servers of a particular type and which network bandwidth is needed to achieve a particular quality of service, for example, sub-second response time up to 1000 invocations per minute. The performance of storage, network, memory, and CPU are interdependent. This is particularly true if customers bring their own application. This is of course a general problem, not only occurring when automating the advertising and negotiation function. However, a person can try it out.

- We discussed in the introduction that advertising or search as well as negotiation functions potentially interact with a large number of different partners, intermediaries or potential business partners. People are usually good at dealing with a *heterogeneity of interaction formats*, in particular because they can establish common understanding - shared ontology - in the course of a conversation. This is requires complex behavior.

- The most difficult issue of automating contracting functions is the inherent *complexity of the decision-making*. When advertising a service, people take many inputs, including cultural knowledge and understanding of market behavior, to design advertisements and choose the right channels. While negotiating, people draw information from many sources and sometimes decide intuitively, which is difficult to formalize. However, there are also shortcomings in human ways of decision-making, in particular to process vast amounts of data. The discipline of decision support systems addresses this issue.

Considering the above issues, which approaches can we take to automate a subset of contracting functions:

- The first approach is to keep people assigned to functions involving decision-making and focus on *improving interaction* between functions and between organizations. To connect people executing contracting functions within an organization by integrating them in a business process and letting them share information is low-hanging fruit and can be done in a standard business process reengineering and automation project. For the above-mentioned reasons of establishing common ontology and using a common format, automating inter-organizational interaction is more difficult. However, interaction for which common ontology exists, maybe because it is not domain-specific or standardized beforehand, and all parties share the same format can be automated. To advertise and search services, organizations can use the known format of UDDI [30] and use, e.g., the standard UNSPSC to categorize their service. In the negotiation phase, parties can use the aforementioned SeCo or COSMOS systems. They only rely on very few, domain-independent concepts such as offer, accept, signature and thus do not pose an ontology problem. Ströbel goes beyond this static simple semantics and proposes in the SilkRoad project a negotiation protocol whose first phase is the design of "negotiation media" or offers, assuming a name-value-tuple level of formalization of the negotiated parts of the offer [28]. This includes the clarification of the semantics of the fields to be negotiated. Subsequently, the actual negotiation starts. This protocol is facilitated by a negotiation intermediary.

- The second approach aims at *improving people's decisions-making* quality. As we mentioned above, people have problems making objective decision in a complex product space in which each product has many interrelated attributes. Besides general decision-support systems, there have been a number of approaches to support buyers of services to evaluate and compare multiple offers. For example, Stolze proposes a system enabling "soft navigation" in product catalogs by assigning scores to the relevance of attributes [26]. Another approach facilitates an interviewing process that aims at helping a customer clarifying preferences and making a decision [27].

- A third approach is based on *early decision-making and later execution based on templates.* In some scenarios, in particular where either a mass market demands relatively homogeneous services or a fulfillment system can only create a limited number of different services, decisions can be "stored" in templates that can be used later by simpler, automated functions. To do so, advertisements are associated to a contract template. The contract template of a service provider contains a number of open fields to be negotiated, e.g., using the SilkRoad system introduced above, or simply to be filled in on a Web site. These *contract templates* can be associated with constraints that limit the number of different contracts that customers can create. For these contract templates, providers also define a *contract implementation plan template* (CIPT) that lists components of fulfillment systems to be instantiated. More flexibly, the deployment planning function implements a decision of component choice based on the customer's selection. In addition, the CIPT describes how to map elements of the contract to configuration parameters of the components. This can include both negotiated and pre-set parts of the contract. If the contract implementation plan created from the CIPT already corresponds to the specific configuration formats and no manual translation is required, the deployment function can also be fully automated. With this approach, the whole contracting function for a particular contract instance can be automated by investing effort in the template design. However, this comes at the price of reduced flexibility. This approach has been implemented in the CrossFlow project in the context of business process outsourcing and is described in [13]. The template approach can not only be used by service providers but also by service customers if they procure similar services very frequently.

Based on the decision criteria discussed above, a designer of a contracting process can modify and combine these approaches to suit the needs of the particular situation.

5 Summary and Conclusion

In this report, we introduced a conceptual model of the contracting process to facilitate the discussion of automating parts of the functions implementing the process. The discussion is motivated by the demand of business to buy and sell services through a network fast and at low cost, thus requiring a - partially - automated contracting process. Since many legislations do not require a specific form or format for contracts on many subjects, we can include formal specifications in a contract. The formal elements can comprise known specification languages such as WSDL or specifically made languages such as ODRL and the WSLA language.

The conceptual model decomposes the contracting function in sub-functions and interaction between them. Beyond advertisement, search, offer and contract, the contract implementation plan is information item in the contracting process that is particularly important to automate the setup phase of the contract life-cycle.

The report identifies the complexity of the fulfillment system, heterogeneity of interaction formats, and the complexity of decision-making functions as major issues that must be addressed when automating parts of the contracting function. Three approaches that help automating the contracting process are proposed: process automation, improved decision-making, and the use of templates.

Based on the conceptual model and the proposed approaches, contracting processes for specific scenarios can be designed.

For many issues that are addressed we can already find partial or domain-specific solutions, be it representing formal contract content or supporting functions such as advertising, search, and negotiation. However, many issues are still open, in particular the representation of contracts across specific domains, representation and processing of contract implementation plans, and the interoperability of different automated functions in the contracting process.

References

[1] S. Angelov and P. Grefen. B2B eContract Handling - A Survey of Projects, Papers and Standards. CTIT Technical Report 01-21, Universiteit Twente, The Netherlands, 2001.

[2] M. Bichler. *The Future of e-Markets - Multidimentional Market Mechanisms.* Cambridge University Press, Cambridge, United Kingdom, 2001.

[3] J. Cole, J. Derrick, Z. Milosevic, and K. Raymond. Policies in an Enterprise Specification. In *Proceedings of the International Workshop on Policies for Distributed Systems and Networks (POLICY 2001), Lecture Notes on Computer Science 1995*, Bristol, UK, 2001. Springer-Verlag.

[4] D. Connolly, F. van Harmelen, I. Harrocks, D. McGuinness, P. F. Patel-Schneider, and L. A. Stein. *Annotated DAML+OIL Ontology Markup.* World Wide Web Consortium, 2000.

[5] F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, and S. Weerawarana. *Business Process Execution Language for Web Services, Version 1.0.* BEA Systems, International Business Machines Corporation, Microsoft Corporation, Inc., 2002.

[6] Z. Damen, W. Derks, M. Duitshof, and H. Ensing. Business-to-business E-Commerce in a Logistics Domain. In H. Ludwig, Y. Hoffner, C. Bussler, and M. Bichler, editors, *Proceedings of the CAISE*00 Workshop on Infrastructures for Dynamic Business-to-Business Outsourcing*, Stockholm, Sweden, June 2000.

[7] A. Dan, D. Dias, R. Kearney, T. Lau, T. Nguyen, F. Parr, M. Sachs, and H. Shaikh. Business-to-Business Integration with tpaML and a B2B Protocol Framework. *IBM Systems Journal*, 40(1), February 2001.

[8] Electronic Commerce Code Management Association. *UNSPSC - Universal Standard Products and Services Classification.* Bethlehem, PA, 1999.

[9] S. Field, C. Facciorusso, Y. Hoffner, A. Schade, and M. Stolze. Design Criteria for a Virtual Marketplace (ViMP). In C. Nikolaou and C. Stephandis, editors, *Research and Advanced Technology for Digital Libraries*, Berlin, 1998. Springer-Verlag.

[10] M. Gisler. *Vertragsrechtliche Aspekte Elektronischer Märkte - nach Schweizerischem Obligationenrecht.* PhD thesis, Universität St. Gallen, St. Gallen, Switzerland, 1999.

[11] M. Gisler, K. Stanoevska-Slabeva, and M. Greunz. Legal Aspects of Electronic Contracts. In H. Ludwig, Y. Hoffner, C. Bussler, and M. Bichler, editors, *Proceedings of the CAISE*00 Workshop on Infrastructures for Dynamic Business-to-Business Outsourcing*, Stockholm, Sweden, June 2000.

[12] M. Greunz, B. Schopp, and K. Stanoevska-Slabeva. Supporting Market Transactions through XML Contracting Container. In *Proceeding of the 6th Americas Conference on Information Systems (AMCIS 2000)*, Long Beach, CA, 2000.

[13] Y. Hoffner, S. Field, P. Grefen, and H. Ludwig. Contract-Driven Creation and Operation of Virtual Enterprises. *Computer Networks*, 37:111–136, 2001.

[14] International Telecommunications Union. *ITUT Recommendation X.911 - Information Technology - Open Distributed Processing - Reference Model - Enterprise Language*, 2001.

[15] IPR Systems Pty Ltd. *Open Digital Rights Language (ODRL), Version 1.1*, 2002.

[16] M. Koetsier, P. Grefen, and J. Vonk. Contracts for Cross-Organizational Workflow Management. In *Proceedings of the 1st International Conference on Electronic Commerce and Web Technologies (EC-WEB)*, London, UK, 2000. Springer-Verlag.

[17] H. Kreger. *Web Services Conceptual Architecture 1.0.* IBM Software Group, May 2001.

[18] R. M. Lee. Distributed Electronic Trade Scenarios: Representation, Design, Prototyping. RP 1998.09.01, Erasmus University Research Institute for Decision and Information Systems (EURIDIS), 2000.

[19] H. Ludwig and Y. Hoffner. CRAFT: A Framework for Integration Facilitation in Cross-Organisational Distributed Systems. In K. Klöckner, editor, *Proceedings of the 9th EUROMICRO Workshop on Parallel and Distributed Processing (PDP'01)*, pages 317–326, Mantova, Italy, 2001. IEEE Computer Society.

[20] H. Ludwig and Y. Hoffner. The Role of Contract and Component Semantics in Dynamic E-Contract Enactment Configuration. In *Proceedings of the 9th IFIP Workshop on Data Semantics (DS9)*, pages 26–40, Hong Kong, 2001.

[21] H. Ludwig, A. Keller, A. Dan, and R. King. A Service Level Agreement Language for Dynamic Electronic Services. In *Proceedings of the 4th IEEE International Workshop on Advanced Issus of E-Commerce and Web-Based Information Systems (WECWIS)*, Newport Beach, CA, 2002. IEEE Computer Society.

[22] M. Merz, F. Griffel, T. Tu, S. Müller-Wilken, H. Weinreich, M. Boger, and W. Lamersdorf. Supporting Electronic Commerce Transactions with Contracting Services. *International Journal of Cooperative Information Systems*, 7(4):249–274, 1998.

[23] B. Moore, E. Ellesson, J. Strassner, and A. Westerinen. Policy Core Information Model - Version 1 Specification. RFC 3060, IETF, February 2001.

[24] A. Runge, B. Schopp, and K. Stanoevska-Slabeva. The Management of Business Transactions through Electronic Contracts. In A. Camelli, A. Min Tjoa, and R.R. Wagner, editors, *Proceeding of the 10th International Conference on Database and Expert Systems Applications (DEXA 99)*, pages 824–831, Florence, Italy, 1999.

[25] M.W.A. Steen and J. Derrick. ODP Enterprise Viewpoint Specification. *Computer Standards and Interfaces*, 22:165–189, September 2000.

[26] M. Stolze. Soft Navigation in Product Catalogs. In *Proceedings Second European Conference on Research and Advanced Technology for Digital Libraries*, pages 385 – 396, Heraklion, Greece, 1998. Springer-Verlag.

[27] M. Stolze and M. Ströbel. Utility-based Decision Tree Optimization: A Framework for Adaptive Interviewing . In *Proceedings of the 8th International Conference on User Modeling*, pages 105 – 116, Sonthofen, Germany, 2001.

[28] M. Ströbel. Communication Design for Electronic Negotiations. *Computer Networks*, 39:661–680, 2002.

[29] M. Ströbel. *A Design and Implementation Framework for Multi-Attribute Negotiation Intermediation in Electronic Markets*. PhD thesis, Universität St. Gallen, St. Gallen, Switzerland, 2002.

[30] UDDI Version 2.0 API Specification. Universal Description, Discovery and Integration, uddi.org, June 2001.

[31] H. Weigand and L. Xu. Contracts in E-Commerce. In *Proceedings of the 9th IFIP Workshop on Data Semantics (DS9)*, Hong Kong, 2001.

[32] World Wide Web Consortium. *Resource Description Framework - RDF Model and Syntax Specification*, 1999.