

Contracts for Cross-Organizational Workflow Management

M. Koetsier, P. Grefen, J. Vonk

Abstract*

Nowadays, many organizations form dynamic partnerships to deal effectively with market requirements. As companies use automated workflow systems to control their processes, a way of linking workflow processes in different organizations is useful in turning the co-operating companies into a seamless operating virtual enterprise. The CrossFlow Esprit project aims at developing information technology for advanced process support in dynamic virtual organizations with contract based service trading. Contracts are necessary for flexible service outsourcing. This report presents contracts as a way of finding suitable partners, connect WFMSs of different kinds, control outsourced workflow, and share an abstraction of the workflow specification between the partners. The contract defines the data, process, and conditions relevant to the co-operation and the outsourced workflow on an abstract level. This information can be fed through an interface to the WFMSs on both sides of the outsourcing in order to automate fully the co-operation between the partners.

* The work presented in this report is supported by the European Commission as ESPRIT project No. 28635.

1. Introduction

In today's businesses, the application of workflow management systems (WFMSs) is widespread. The use of WFMSs ensures a well-structured and standardized handling of these processes within the organization. Traditionally, the emphasis of workflow management has been on processes within the boundary of a single organization. In the context of close cooperation between companies, where companies combine their efforts and become virtual enterprises, processes crossing organizational boundaries have to be supported. This implies extending the functionality of workflow support so that workflow management systems in different organizations can be linked to manage integrated cross-organizational processes. This extended workflow support should be able to deal effectively with heterogeneous workflow environments, well-specified levels of autonomy of partners in a virtual enterprise, and dynamic formation of new and dismantling of existing collaborations.

A typical pattern of collaboration in virtual enterprises is that of service outsourcing. In this pattern, an organization, called the service consumer, outsources part of its process to another organization, called the service provider, to prosecute a result collaboratively. The resulting cooperation is called a virtual enterprise. In specific business sectors (vertical markets), standard types of services exist. This opens the possibility of dynamic outsourcing. Based on a common understanding of standard services, services to be outsourced can be exchanged on electronic markets through service trading facilities. Parameterization of standard services is necessary to cater for required flexibility in service enactment.

To enable flexible service trading, detailed contracts are required for both service trading and service execution. These contracts provide the structure for specification and parameterization of complex workflow-supported services. To link workflow management systems properly in the enactment of outsourced services, these contracts should not only specify static parameters of the service being traded, but also service process structures and enactment clauses. Process structures allow a mapping of service execution to workflow management systems on both service consumer and service provider sides. High-level enactment clauses allow the specification of additional execution characteristics, like monitoring of quality of service parameters or explicitly controlling the progress of outsourced services by the service consumer party.

To allow high-level, flexible enactment of dynamically specified services, an infrastructure is required on top of traditional workflow management functionality that can handle both contract making and contract enactment. To effectively match the dynamic nature of contracts, this infrastructure should be dynamic as well. Its configuration should be determined by the contents of the contracts. For this purpose, a well-structured contract specification language is required. We chose XML as the basis for this language, as it allows the specification of complex structures.

In the CrossFlow ESPRIT project [Gref99a], contract-based support for cross-organizational workflow is developed for dynamic virtual enterprise settings. Along the lines presented in this report, a conceptual contract model and contract specification language have been developed to model and specify flexible outsourcing. On top of a commercial workflow management system, a contract making and contract enactment infrastructure has been designed to advertise contracts via a trader, search for contracts matching outsourcing requirements, make contracts, and enact services specified in contracts.

This report focuses on the contract model and language developed in the CrossFlow project. The application of the language in the project context is discussed to illustrate the use of contracts. The contribution of this report to the state of the art is threefold: it shows that contracts are required for flexible workflow based outsourcing, it explains what form contracts

should have by describing a contract model and specification language, and it shows the application of the approach in a real work context.

1.1 Structure of this report

Below, we start with a discussion of related work, in which we position our work with respect to other work in the fields of cross-organizational workflow management and structured information exchange. In Section 3, we describe the context in which we propose the use of contracts as the basis for cross-organizational workflow management. The conceptual structure of contracts is discussed in Section 4. Here we see how requirements from the context are mapped into a conceptual contract structure. In Section 5, we show how this conceptual structure can be operationalized into an XML-based contract specification language. Next, the application of contracts in the CrossFlow project context is presented as a concrete application of our ideas. We end the report with conclusions and a description of future work.

2. Related work

In the past decade, workflow management technology has emerged as a basis for structured process management [Geor95]. The emphasis has mainly been on workflow management in homogeneous environments within the boundaries of an organization. With the advent of virtual enterprises and electronic commerce, however, cross-organizational workflow management is attracting much attention these days [Ludw99]. Basic interoperability between workflow management systems has been addressed by the Workflow Management Coalition [WFM98], but realistic virtual enterprise settings require more than this. The FlowJet project at Hewlett-Packard aims at coupling various types of workflow systems in E-business contexts [Shan99]. Dynamic resource brokering is within the scope of the project, but explicit contracts for detailed service specification are not considered. The WISE project is comparable to FlowJet as it uses cross-organizational workflow management technology for business-to-business E-commerce scenarios [Alon99]. Electronic trading of workflows is also considered in other research efforts [Ditt99]. Our work distinguishes itself from these other approaches by the use of contracts as fine-grain specifications of the workflows (or rather services) to be traded.

In the past, a substantial amount of effort has been dedicated to electronic description and communication of structured information. Most notable are developments in the context of EDI [Cann93] and EDIFACT [EDIF]. This work generally concentrates on electronic exchange of product specifications, whereas our work focuses on trading, exchange and enactment of service specifications, including execution characteristics. The work in the XML community aims at structured information specification [XML, Grah99]. We see this work as input to our work, as our contract specification language is based on XML.

3. Context

In this section, we discuss the context in which contracts for cross-organizational workflow are used. First, we describe the concept of contracts. Next, we show the cross-organizational environment in which contracts are applied. From concept and environment, the requirements for contracts follow. These requirements are the basis for the conceptual contract structure presented in Chapter 4.

3.1 The concept of contracts

Contracts are needed in many types of business transactions. In the first place, the contract should specify exactly the product or service to be exchanged in such a way that buyer and seller know what they can expect and what is expected of them. In the second place, a contract establishes the rules of the engagement. In case disagreements occur, the contract should ultimately contain the information to judge who is right. Contract making has been done and studied for centuries, so in itself this is nothing new [Beal90].

In a vertical market, where the same products or services are traded on a regular basis, standard form contracts can be used [NJL98]. These are contracts with standard wording, leaving fields empty for specifying the individual characteristics of each transaction (such as the date and the buyer's name). By using standard form contracts, making a contract is less complicated than when the phrasing of the entire contract has to be formulated with each agreement.

In this report, we discuss contracts for dynamic outsourcing of workflow. We present contracts that are an electronic variation of standard form contracts for trading standardized workflow processes. This electronic form offers more flexibility in specifying contracts than traditional standard form contracts, as they can be dynamically composed from contract elements.

3.2 An environment for contracts

We focus here on an environment where transactions take place between automated systems. Contracts are used to define these transactions. The goal is to have a fully automated cooperation between two systems that will be set up and managed without human intervention. The consequence of this is that the contracts have to be established automatically and have to be interpreted by automated systems.

The product that is traded is a service implemented by a workflow process. The automated systems on both sides are Workflow Management Systems (WFMSs), extended with contract handling facility such that they are able to deal with the outsourcing interaction. On one side there is a consumer WFMS that wants to outsource a process, on the other side a provider WFMS that can execute this process for the consumer.

The outsourcing starts when the consumer WFMS notices that it needs to outsource part of its process. Reasons for this may be that the process is not supported within the company at all, or that all the resources needed for this process within the workflow context are already in use. It may also be the case that the provider can perform the service more efficiently or effectively than the consumer. The consumer WFMS will then, through its outsourcing infrastructure, automatically send out a request to a service trader that results in a contract with a provider. Both the provider and consumer WFMS must be able to read from the contract what is expected of them and they will start executing the process without human intervention.

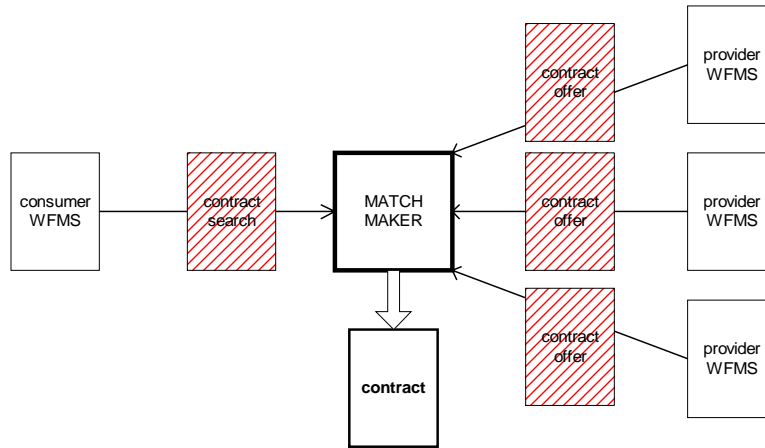


Figure 1: Making contracts

The processes are traded by the use of an automatic matchmaker. As shown in Figure 1, providers can advertise service offers via the matchmaker. The matchmaker will find the provider offers closest to the consumer's wishes and present the resulting set of offers to the consumer. The consumer can then finalize the deal between himself and the provider. Since the contracts are standardized with only certain fields that can be used to define variations it is not too difficult to accomplish the matchmaking. The matchmaking process itself is described in [Hoff99], we will focus here on the contract itself and its use in cross-organizational workflow. The consumer WFMS has now bought a process that the provider will execute for it. It is, however, not a black box situation where only the process name is known to the consumer. Through the contract, the consumer knows the internal specification of the process. The consumer WFMS will, through the outsourcing infrastructure, monitor the progress of the purchased process and influence the process, even though it is executed elsewhere. The provider WFMS is obliged to keep the consumer WFMS up to date on the progress and to obey a certain set of commands from the consumer. The contract defines all the properties of this close co-operation.

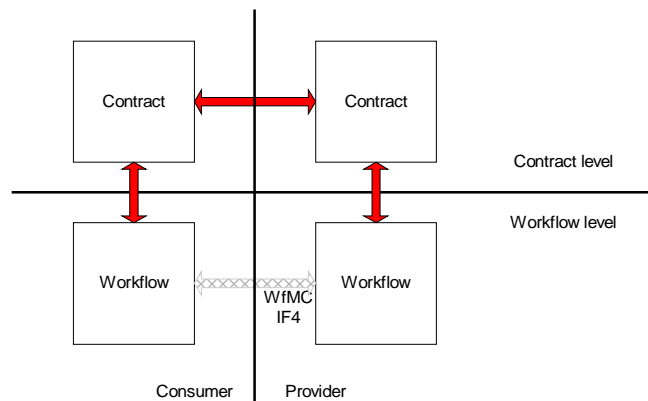


Figure 2: Contract and workflow levels

There is not a direct co-operation on workflow level between the WFMSs, in the sense that two workflows are connected directly through, for example, a WfMC IF4 interface. The WfMC interface defines the requests and responses that two workflow engines must exchange when interoperating across a network [WfMC]. In our case, both WFMSs will have a contract enactment infrastructure, which can translate the relevant contract information into concepts that can be interpreted by the WFMS and will manage issues that are outside of standard WFMSs itself. The level of co-operation is elevated to the level of the contract (Figure 2).

This is a flexible situation, which is required for dynamic outsourcing. If workflow systems are connected directly, a static situation is created. By elevating the co-operation to the contract level, new links can be made at any moment, without having to reorganize the basic workflow infrastructure. Such flexibility means, for example, that different providers can be used each time a specific service is needed, depending on the requirements and offers at that moment. The implications of such an environment for contracts will be discussed in the next section.

3.3 Requirements to contracts

In order to have contracts that can be used as a basis for automatic co-operation, these have to comply with several requirements:

Structured and complete contents. The contract must be searchable and interpretable by electronic systems. This means the contract must have a clear structure and use unambiguous naming conventions. It also means that one cannot leave out things, relying on the (human) common sense of the contract parties, every interaction issue that can possibly occur must be described in the contract.

Flexibility. The contract must be usable in each kind of outsourcing situation. It should be flexible in respect to usage and reusability. For example it may be practical to reuse the same contract for multiple uses of a service. The contract should also be flexible with respect to the enactment characteristics, allowing adaptation of contracts to organizations and circumstances.

Heterogeneity. The contract should be interpretable by each WFMS that is on the market. The WFMSs on both sides read, through the outsourcing infrastructure, the specifics of the transaction from the contract. The contract should therefore be stated in terms that have meaning to each type of WFMS. Each kind of WFMS must be able to map the contract view on the process to its own process specification language.

Encapsulation. The contract contains a specification of the process. The provider is the expert on the outsourced process, it has well worked out procedures, a set of internal rules and details of how to execute this process successfully. The provider's WFMS will know exactly how to use the provider's resources in order to have the fastest and most cost effective way of executing this process. The provider may not be willing to disclose all its process information to the consumer. Also, to the consumer not all details about the provider are relevant. The consumer wants to know that a certain set of actions is executed and in which order, so it will know when to expect a result or when to inquire about service execution status. The consumer WFMS does not need so many details that it can actually run this process. Therefore, the contract should hold an encapsulated view on the process, with enough information to be meaningful to the consumer without containing irrelevant detail or private information of the provider.

Legality. The contract should be a legally binding document that defines the co-operation between the companies. Even though it is an automated affair, we are still dealing with a business transaction between two organizations. In case of conflict the contract should contain the information for settlement.

From these requirements, the essential elements of the contract can be derived. These elements and the way these are structured in the contract are described in the next section.

4. Contract structure

From the requirements in the previous section, five basic contract elements can be identified, which together describe all aspects of automatic workflow outsourcing:

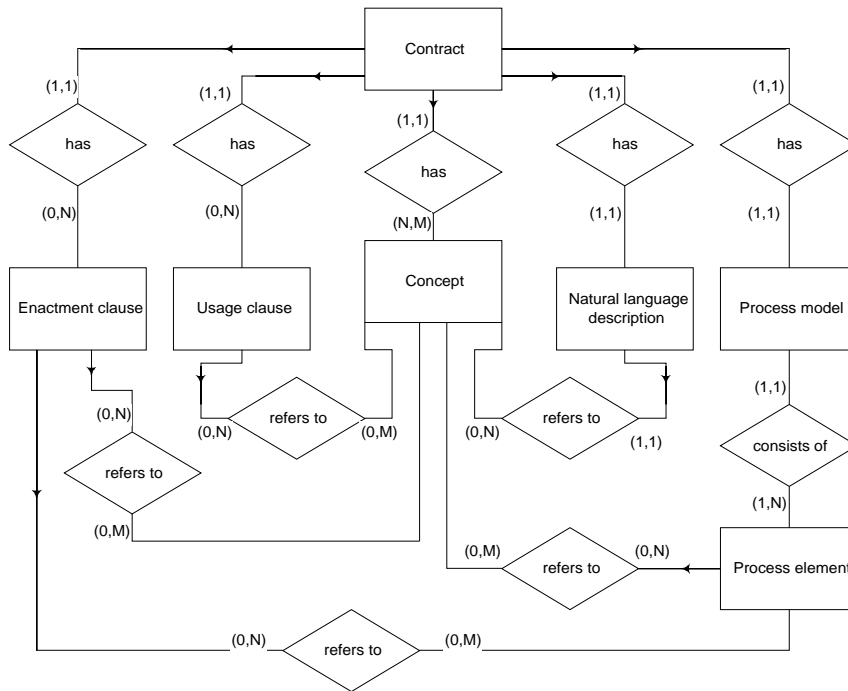


Figure 3: Conceptual contract structure in EER notation [Elm94]

Concept model. All concepts that are used in the contract must be defined clearly, creating a concept space in which the other contract issues can be specified. This is not dissimilar to the terminology statements in the first section of a regular contract.

Process model. The process model describes the internal structure of the workflow process implementing the service. The process is composed of process elements, e.g. the individual activities and transitions. The process needs to be specified in a way that allows the provider to map it to its actual process and allows the consumer to understand the sequence of events and make decisions based on this knowledge.

Enactment model. The enactment details must be described. This part of the contract specifies the co-operative support that is offered during the service. For example remuneration, authentication, control options for the consumer, etc.

Usage model. The usage model defines manners in which the contract can be used. The simplest case is where one contract is made to start one instance of the service immediately. Other possibilities are contracts made to start multiple executions of the service, or contracts made to reserve the resources of the provider for a service execution at a later moment. The usage model describes the different usage possibilities of the contract and their conditions.

Natural Language Description. The natural language description is a piece of text that is not meant for electronic interpretation, but for human reading. This piece of text can be used to

describe the service in an easily understandable way and to refer to the legal context of the transaction.

The structure of a contract is depicted in Figure 3. This picture shows the above mentioned elements that the contracts consist of and their relation to each other. As can be seen in this figure, the elements of the contract are not independent, many of the elements can refer to other elements. For example, clauses in the *enactment model* can specify that the consumer is allowed to change the value of a *concept* during a particular *activity in the process*, thereby referring to both the concept model and the process model in the enactment model. Concept, process, and enactment model are the most important and most interesting parts of the contract model. In order to be concise, only these are described in more detail below, the complete contract model and language are described in [Koe99a] and [Koe99b].

4.1 Concept model

In a traditional contract, the terminology of the contract will be established in the first section. For example, a phrase in the first part of the contract stating: Mr. X. who will be referenced as CUSTOMER from now on. This has the advantage that the basic text of the contract can be reused without changes, only the specifications of the references in the first section have to be altered for each new contract.

A similar approach is used in the electronic contract for cross-organizational workflow. In the contract, all the objects that can be parameterized are defined and given a value. The specific properties of the service are established in this part of the contract as parameters. Since the contracts are interpreted automatically, these parameters should be defined in a manner that automated systems can process. This means that for each parameter the following attributes should be defined:

- A unique name, that identifies the parameter.
- The type, whether it is a string, integer, etc. This makes it easier to manipulate the parameters electronically if necessary.
- The value, if required. The value does not in each case have to be specified in the contract. If the parameter is used for communicating the results of the process, the parameter will only receive its value during the execution of the process.
- An optional explanation, this is not essential for electronic processing, but it makes the contract more readable for humans.
- Additional attributes, defining whether it is mandatory for this parameter to have a value, or who should provide the value of the parameter (consumer or provider). These additional attributes enable validation of contracts before starting the execution of the service.

Several types of parameters can be identified, these types determine whether and when a value will be specified for the parameter and in what manner the parameter must be treated by the infrastructure:

- Parameters that define the co-operation, such as the name and address of the consumer, the name and address of the provider, the date of the contract, etc.
- Parameters that have to serve as input for the outsourced process, this means that the consumer should provide its value.
- Parameters that will supply the results of the process, these parameters will not have a value in the contract but they will be defined so that the consumer knows what to expect.

Just like the terminology in regular contracts, these established parameters can be referred to in other parts of the contract. For example, it may be mentioned somewhere in the contract that the

provider will update the value of a certain parameters after each activity. Only the name of the parameter will have to be mentioned, the structure and the meaning of it can be looked up in the concept model.

4.2 Process model

The contract contains a workflow specification of the process that is being traded. This workflow specification ensures that both parties have a shared view on the outsourced process. To comply with the heterogeneity requirement, the workflow specification must be set in a language that allows a mapping to any workflow specification language of a particular WFMS. The WfMC workflow description language is used for this [WFM98].

The encapsulation requirement states that the specification of the process should only contain the process information that is functional for the co-operation. This is the process information that is to be shared between consumer and provider. It is not necessary to actually implement a workflow from the contract, the workflow is already fully implemented at the provider's side. The process that is in the contract is therefore an abstracted view on this provider's process. This is reflected in the contract specification in two ways:

The full WfMC workflow specification language contains activities, transitions, workflow relevant data, roles and application definitions. In the contract we only use activities and transitions to specify the process. This way all that is specified is which activities will take place and in which order. This is sufficient information for the consumer WFMS to track the process. The data shared between both parties is defined in the parameter section. The roles and applications are only of interest to the provider WFMS in order to be able to run the process, these do not have to be shared between the contract parties.

Not all the activities of the provider have to be specified in the contract. A detailed set of activities may be grouped into one aggregated activity in the contract. The contract view on the workflow will then be a higher level view on the process, which is more understandable and meaningful to a consumer. The provider however should pay close attention to the mapping from the contract to its real process, it can mean he has to perform a whole set of activities before he can report one single contract activity as 'finished' to the consumer. The structure of the workflow specification as it is used in the contracts is depicted in Figure 4.

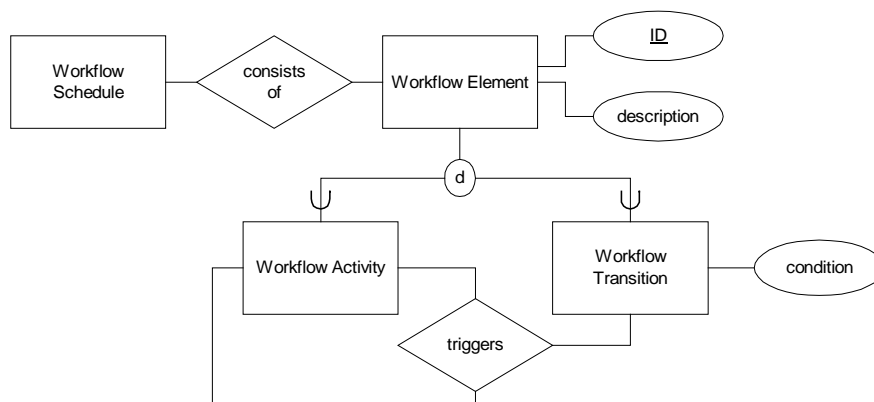


Figure 4: Structure of the contract model in EER notation [Elm94]

4.3 Enactment model

The enactment clauses in the contract define the additional functionality that is provided during the execution of the process. These options go beyond a plain connection of two WFMSs and need support from the connection infrastructure. This can include:

- monitoring, allowing the consumer to look into data of the provider's workflow process during the execution.
- control operations, allowing the consumer to exercise control on the running provider's workflow process.
- control of service flexibility, giving the consumer possibilities to adapt the provider's workflow.
- workflow transaction mechanisms, offering the possibility of cross-organizational workflow transactions.
- remuneration, offering mechanisms for automatic handling of service payments.
- authentication, offering security mechanisms for the co-operation.

In the enactment part of the contract, the provider specifies the additional features that it is willing to provide during the execution of the service. The particular enactment features in the contract are specific for a given context. When a very short and simple service is outsourced, no enactment features may be necessary at all. For more complex services, monitoring and control options are useful to the consumer. Authentication support is important if confidential information is transferred between consumer and provider during the execution. The particular enactment features that are implemented can be adapted to the requirements of the service.

5. Contract specification language

Basically, the contract is a highly structured electronic document that is transferred between different kinds of information systems. The obvious language solution for such a situation is XML, since XML is particularly designed to deliver structured content over a network. In the next two sections we will explain how XML was used as the basis for the contract language.

5.1 XML as the basis for contract specification

XML is a markup language for documents containing structured information [XML, Grah99]. XML is not unlike HTML, however in XML one can define one's own tags. A Document Type Definition (DTD) is used to define for a set of documents the tags that can be used, the order of these tags and the attributes that can accompany those tags. A complete XML document can be mapped to a tree structure, with elements that consist of other elements.

The contract model has been mapped to a contract DTD, establishing the content and structure for all contracts. The root of the contract DTD looks like this:

```
<!ELEMENT Contract (DataSection, NaturalLanguageDescr?, WorkFlow,
    Enactment, UsageClauses? )>
```

This line establishes that the contract consists of a data section, a natural language description (optional), a workflow specification, an enactment specification, and usage clauses (optional). Each element that makes up the root of the contract is specified into detail in turn, defining a complete structure that all contracts must conform to.

However, a DTD alone cannot offer adequate constraints for specifying a complete contract language. Additional constraints have been identified, for example constraints that relate the attribute values of an element to its content. XML in itself cannot impose constraints on text content within elements. These additional constraints form together with the DTD the complete contract language.

5.2 Mapping the conceptual contract structure to XML

In the mapping of the contract model to the contract language explicit design decisions have been made to make the contract operationally more suitable. Some features that are defined as attributes in the contract model have been implemented as individual elements in the XML structure, because for the co-operation infrastructure it is more straightforward to look up explicit tags in the XML specification than to search through the tree structure for a specific value.

Below the DTD specification of the workflow model is given. This structure is obtained by mapping the conceptual model as depicted in Figure 4 into XML.

```
<!ELEMENT WorkFlow (Activity | Transition )+ >
<!ELEMENT Activity (Name, Description?) >
<!ATTLIST Activity ActID ID #REQUIRED>
<!ELEMENT Name (#PCDATA) >
<!ELEMENT Description (#PCDATA) >
<!ELEMENT Transition (Name, Description?, To, From, Condition?)>
<!ATTLIST Transition
    TransID ID #REQUIRED>
<!ELEMENT To (ActRef) >
<!ELEMENT From (ActRef) >
```

```

<!ELEMENT Condition (#PCDATA | ParamRef | ActRef)*>
<!ATTLIST Condition
    type      (parameter | workflow.state | workflow.event | workflow.time
| CDATA)      #REQUIRED>
<!ELEMENT ParamRef EMPTY>
<!ATTLIST ParamRef ParamID IDREF          #REQUIRED>
<!ELEMENT ActRef EMPTY >
<!ATTLIST ActRef ActID          IDREF          #REQUIRED>

```

Because of the encapsulation principle (see Section 3.3), this is a relatively simple piece of DTD. The activities have an identifier attribute, to enable referring to them in the rest of the document, just like the parameters, which are defined in another part of the DTD. Conditions are attributed with a type in order to configure the proper evaluators in the infrastructure. The conditions can refer to both activities and parameters, enabling statements that refer to a particular state of an activity or a parameter value. A condition language is one of the additional constraints to the contract language, prescribing the statements that can be used in this mixed content element.

6. Contracts in CrossFlow

In this chapter we will explain how the contract model and language are developed and used in the CrossFlow ESPRIT project.

6.1 The CrossFlow project

Seven European institutes form the consortium of the CrossFlow ESPRIT project. The project researches dynamic outsourcing of workflow processes. An infrastructure is developed to connect workflow systems dynamically, based on contracts. High level support is obtained by abstracting services and offering advanced co-operation support. CrossFlow developments are driven by requirements from real-world scenarios.

6.2 Contract model and language

The language described in this document is used for the contracts in the CrossFlow project, with the addition that particular details of the enactment model, which are context specific (see Section 4.3) are specified. In the CrossFlow project three types of service support have been chosen for implementation and detailed contract language specifications for these have been developed. The implemented enactment features are Level of Control, Flexible Change Control and Quality of Service.

Level of Control provides means for high level cross-organizational transaction management and consumer process control over outsourced services. The LoC enactment model in the contract contains a part specifying *what* control the consumer can carry out, under what *conditions* it is allowed to carry this out, *how* the consumer can carry this out and what *effect* using this control has on the agreement. The transaction part of the contract specifies *compensation activities*, which are used to support cross-organizational rollbacks [Gref99b, Von99].

Flexible Change Control allows for dynamic changes to execution paths of services during their execution. The FCC model in the contract specifies *alternative* or *unordered* paths in the service specification, and what the *effects* are of choosing these paths on service parameters [Klin99].

Quality of Service gives the consumer the possibility to monitor the current execution and compare it to a set of preconceived quality standards. The QoS enactment model specifies the quality *standards* that are guaranteed for this service, and ways for the consumer to monitor these standards. For each monitoring option is specified *what* (which service parameter) the consumer can monitor, under which *conditions* this can be monitored, *how* the consumer can monitor this, and whether the monitoring can be done actively by the consumer (*monitoring operation*) or whether the value is pushed to the consumer by the provider (*notification*).

6.3 Contract support architecture

The CrossFlow architecture supports both contract making and contract enactment. The architecture uses commercial workflow management system technology as a basis, shielded by an interface layer from the CrossFlow technology. In the project, IBM's MQSeries Workflow is used [IBM]. The CrossFlow architecture is shown in simplified form in Figure 5.

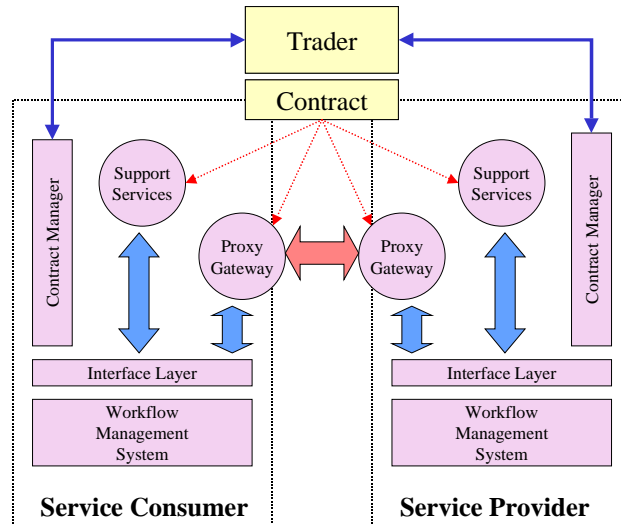


Figure 5: The CrossFlow architecture

The CrossFlow architecture uses CSS (Co-operative Support Service) modules that each manage a specific kind of co-operation support. The contract is used for the selection and configuration of these CSS modules. This created a flexible architecture, depending on the situation, a number of CSS modules are plugged into a software bus [Gref98], making it possible to adapt the type of implemented support services to each kind of process.

6.4 Cross-organizational workflow scenarios

The CrossFlow project uses on two real-world scenarios as input: a logistics scenario and an insurance scenario. The scenarios are both used for the elicitation of requirements to contract-based cross-organizational workflow support and for the deployment and evaluation of the prototype system produced by the project. The scenarios are:

The insurance scenario: AGFIL, an insurance company in Ireland, outsources the handling of the individual car insurance claims to a specialized organization. Since this is a very close co-operation AGFIL is interested in making a direct link between the workflows of both companies, thereby gaining more monitoring capabilities of the outsourced activities. Because of the regular use that AGFIL makes of this service, a contract can be made that can be used for multiple instances of the provider service, granting AGFIL extensive service support as an appreciated customer.

The logistics scenario: Logistics company TNT wants to give its customers the opportunity to link the package transport process directly to their own process. TNT, as service provider, can in this manner increase the variety of the service portfolio without an increase in costs, so customers perceive tailor-made solutions complying with their individual needs. Integration with the systems of customers is an important challenge, e.g. a direct connection to their enterprise resource planning systems. Consumers can automatically make contracts each time the need for transporting a package arises or, if they have to transport packages on a regular basis, make a multiple service contract with TNT. In the Crossflow scenario, TNT acts as a service provider for KPN Telecom in the distribution of mobile telephones. This requires high level coupling of TNT's and KPN's workflow support, offering powerful monitoring and control functionality.

7. Conclusions

By the use of contracts, highly flexible dynamic outsourcing in virtual enterprises is possible. Contracts enable flexibility in the outsourcing by lifting the level of co-operation to a higher level. By providing an abstracted view on the interaction on a level above the basic workflow level, process details are shared between the co-operation partners, offering more transparency than with a black box approach. This abstraction approach results in the means to deal with heterogeneity, making this type of co-operation available to workflow management systems in general, and encapsulation, shielding private provider details from the consumer and protecting the consumer from irrelevant information. The contract also serves as a legal basis for the transaction.

A contract language has been developed that can be used to define contracts that are suitable for supporting fully automated outsourcing transactions. With this language, all aspects of the interaction can be defined in a structured manner, enabling the co-operation infrastructure to set up and manage the outsourcing environment. A prototype of this infrastructure is under development in the CrossFlow project.

In future, the model and language will be refined and extended, based on the experience with the CrossFlow user scenarios. The CrossFlow prototype will be implemented as proof of concept. Contract enactment mechanisms will be developed further. Tools will be developed to edit and create contracts, validate them, and present contracts in a readable form.

Acknowledgments

All members of the CrossFlow consortium are acknowledged for their contribution to the contract framework presented in this report. Special thanks go to Yigal Hoffner and Heiko Ludwig of IBM Research for their feedback on contract model respectively contract language. Furthermore, we would like to thank Roel Wieringa for his feedback on this article.

References

- [Alon99] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, N. Weiler, 'WISE: Business to Business E-Commerce', *Proceedings International Workshop on Research Issues in Data Engineering*, Sydney, Australia, 1999, pp. 132-139.
- [Beal90] Beale, H.G., Bishop W.D., Furnston, M.P, *Contracts, Cases and Materials*, Butterworths, London, 1990.
- [Cann93] E.Cannon, '*EDI guide : a step by step approach*', Van Nostrand Reinhold, New York, 1993.
- [Ditt99] K. Dittrich, D. Tombros; 'Workflow Management for the Virtual Enterprise', *Proceedings International Process Technology Workshop*, Villard de Lans, France, 1999.
- [EDIF] *The UN/EDIFACT directories*, <http://www.unece.org/trade/untdid/welcom1.htm>
- [Elm94] Elmasri, Navathe, '*Fundamentals of database systems*', The Benjamin/Cummings Publishing Company, Inc., 1994.
- [Geor95] D. Georgakopoulos, M. Hornick, A. Sheth, 'An Overview of Workflow Management: From process Modeling to Workflow Automation Infrastructure', *Journal of Distributed and Parallel Databases*, Vol. 3, No. 2, 1995.
- [Grah99] I. Graham, L. Quin, '*XML Specification Guide*', Wiley Computer Publ, New York, 1999.
- [Gref98] P. Grefen, R. Remmerts de Vries, A Reference Architecture for Workflow Management Systems, *Journal of Data & Knowledge Engineering*, Vol. 27, No. 1; North-Holland - Elsevier, 1998; pp. 31-57.
- [Gref99a] P. Grefen, Y. Hoffner, 'CrossFlow: Cross-Organizational Workflow Support for Virtual Organizations', *Proceedings 9th IEEE International Workshop on Research Issues in Data Engineering*, Sydney, Australia, 1999, pp. 90-91.
- [Gref99b] P. Grefen, B. Pernici, G. Sanchez, '*Database support for workflow management, The WIDE project*', Kluwer Academic Publishers, 1999.
- [Hoff99] Y. Hoffner, 'Supporting Contract Match-Making', *Proceedings 9th IEEE International Workshop on Research Issues in Data Engineering*, Sydney, Australia, 1999, pp. 64-71.
- [IBM] *IBM MQseries Workflow*, <http://www-4.ibm.com/software/ts/mqseries/workflow/>
- [Klin99] J. Klingemann, J. Wäsch, K. Aberer, 'Adaptive Outsourcing in Cross-Organizational Workflows', *Proceedings 11th International Conference CAiSE'99*, Heidelberg, Germany, June 1999, pp. 417-421.
- [Koe99a] M. Koetsier, P. Grefen, J. Vonk, *Contract Model*, CrossFlow Deliverable D4b, University of Twente, 1999 (on request available via www.crossflow.org).
- [Koe99b] M. Koetsier, P. Grefen, J. Vonk, *Contract Language*, CrossFlow Deliverable D4c, University of Twente, 1999 (on request available via www.crossflow.org).
- [Ludw99] H. Ludwig, C. Bussler, M. Shan, P. Grefen, 'Cross-Organisational Workflow Management and Co-ordination – WACC '99 Workshop Report', *ACM SIGGROUP Bulletin*, Vol. 20, No. 1, 1999, pp. 59-62.

- [NJL98] New Jersey Law Revision Commission, *Final Report Relating to Standard Form Contracts*, 1998 (available via www.lewrev.state.nj.us).
- [Shan99] M. Shan, 'FlowJet: Internet-Based E-Service Process Management', *Proceedings International Process Technology Workshop*, Villard de Lans, France, 1999.
- [Von99] J. Vonk, P. Grefen, E. Boertjes, P. Apers, Distributed Global Transaction Support for Workflow Management Applications, '*Proceedings 10th International Conference, DEXA '99*', Florence, Italy, August 30- September 3, 1999, pp. 942-951.
- [WFM98] WorkGroup1, '*Workflow Management Coalition, Workflow Standard-Interface 1: Process Definition Interchange Process Model*', Document number WfMC TC-1016-P, November 12, 1998.
- [WFMC] *The Workflow Management Coalition*, <http://www.aiim.org/wfmc/mainframe.htm>
- [XML] *XML.com website*, <http://www.xml.com>