Cooperative Information Systems in Virtual Districts: the VISPO Approach

Enzo Colombo, Chiara Francalanci, Barbara Pernici, Pierluigi Plebani
Politecnico di Milano
Massimo Mecella
Università di Roma "La Sapienza"
Valeria De Antonellis, Michele Melchiori
Università di Brescia

Abstract

The VISPO project studies cooperative information systems in virtual districts to derive possible cooperation patterns and to provide an architectural framework for cooperation. Different organizations cooperate through e-*Services, which are available through a Service Oriented Architecture.*

1 Introduction

The last few years have witnessed several efforts for the development of service oriented environments, with the purpose of supporting the retrieval, composition, and execution of *e*-Services over the Internet [1, 2, 3].

Within the Italian VISPO (Virtual-district Internet-based Service PlatfOrm) project we aim to create a flexible environment to support the design of service-based cooperative processes and the dynamic composition of *e*-Services, by considering not only the design of complex *e*-Services starting from simple ones, but also the dynamic substitution of failed or modified *e*-Services.

A district is a consortium of independent member enterprises which operate in an integrated and organic way, often on the basis of personal and informal relationships, to exploit business opportunities. The aim of the project is to consider the application of information technology to such realities, in order to (semi-)automate the creation of business relationships (not necessarily based on geographical aggregation) and thus obtaining virtual districts.

In our model, cooperating organizations follow predefined process specifications when interacting. The structure of the process and the modalities of interaction are designed according to the nature of interactions in the virtual district, ranging from completely hierarchical structures to open markets [4].

Cooperative processes are defined as a composition of abstract services. Only at execution time, given an abstract service specification, a compatible concrete *e*-Service is retrieved and invoked. The VISPO architecture allows the execution of dynamically evolving cooperative processes based on these

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

specifications. In particular, mechanisms for run-time substitution of e-Services with compatible ones and for adapting e-Services are being developed.

This paper is organized as follows: Section 2 introduces the VISPO cooperation model. Section 3 presents the VISPO architecture for the composition and execution of cooperative processes based on e-Services, whereas Section 4 outlines the development environment.

2 VISPO Cooperation Model

Cooperative processes in VISPO are defined and enacted on the basis of cooperating e-Services. The following models are provided:

- an e-*Service Model*: the characteristics of e-Services are specified in terms of abstract services;
- an *Orchestration Model*: the interactions between e-Services and their coordination and enactment are specified;
- a *Business Transaction Model*: the process is designed on the basis of pre-defined business transaction patterns, which specify the characteristics of interactions between organizations and provided e-Services .

On the basis of such models, a cooperative process is designed on the basis of the business transaction model, which specifies how to assemble together different organizations in terms of hierarchical structures; the effective enactment of the cooperative process, to be carried out as coordination of different e-Services provided by the identified organizations, is specified through the orchestration model; finally the e-Services are designed and provided according to the e-Service model.

2.1 *e*-Service Model

Each e-Service is defined as a set of abstract services, each defined in terms of: *(i)* its interface, specified as the set of incoming and outgoing messages, and *(ii)* the admissible sequences of messages, in terms of state diagrams, specifying the possible sequence of pairs of input and output messages.

Each e-Service is specified in WSDL using port-types to define abstract services, and with an XML based description of its state diagram (WSTL) [5].

Compatibility and substitutability relationships among e-Services can be derived by analyzing their interfaces and behaviors [6]. A descriptor provides the condensed information needed for intelligent retrieval of compatible e-Services, and can be derived from the interface provided by the e-Service, in terms of incoming and outgoing messages and their parameters [7].

e-Services are registered in a registry, based on UDDI [8]; in addition, for each e-Service, the descriptor is provided.

2.2 Orchestration Model

The orchestration model specifies *(i)* the global evolution of a cooperative process involving two or more e-Services in terms of received and sent messages, and *(ii)* the association of global control task on the cooperative process. The orchestration model is based on Coloured Petri Nets, and distinguishes between control places, which are used for controlling the execution order of activities in an e-Service involved in the process, input/output places used to synchronize and compose messages, and orchestration places used to indicate transfer of responsibility on the overall process as the process evolves [9].

Cooperative processes need to be controlled and monitored by some organizations, but such a global control task cannot be assigned to a single organization for all the process duration, conversely it should be moved all along the process enactment. Therefore the model allows specifying the shifting of the global control task among organizations participating in the process in a controlled way, assuming that only one organization can be responsible for controlling an instance of the process at a given time, while the task can move between organizations as the process evolves.

2.3 Business Transaction Model

The business transaction model is used to help in the definition of coordination and control of *e*-Services in a process. A set of business transaction patterns have been defined, according to a basic classification of possible interactions as market-oriented or hierarchy-oriented. In the first case, organizations (and provided services) are involved assuming a market context: services may be selected among the ones available, negotiation may be needed, post-settlement activities performed after service execution. Conversely in a hierarchical organization of the district, we assume that one organization coordinates all services, and negotiation is not performed when a service is used, but rather the contract conditions are established in advance; on the other hand, the coordinating organization can control the service execution while it is performed.

In order to support the different business transaction patterns, we assume that each *e*-Service, in addition to the abstract services it can provide, may be able to interact with other *e*-Services to negotiate its parameters and conditions for activation, and that it may provide quality control data on the services it is executing.

According to business transaction paradigms, patterns for messages used for coordination and control have been designed, and these messages must be provided by *e*-Services invocated in a given context. The basic transaction structures are used to design the orchestration for a cooperative process, including not only service execution, but also control and coordination message exchanges [10, 11].

3 VISPO Architecture

The VISPO architecture is illustrated in Figure 1. The environment for supporting cooperative process execution is composed of the following modules:

- *Registry*: the registry contains *e*-Services descriptions according to UDDI specifications, augmented with their descriptors extracted from the service interfaces for supporting *e*-Service retrieval. In addition, the registry stores the orchestration schemas for the cooperative processes in the virtual district, providing the characteristics of the service to be invocated to execute a process instance.

- *Orchestration Engine*: the orchestration engine controls the process execution following the orchestration specifications for the process obtained from the registry. The orchestration engine is composed by a set of local orchestration engines, one for each participating organization. Each local orchestration engine is responsible for local workflow enactment, and, when the organization is responsible for the whole process, the orchestration engine is also responsible for the global execution of the process and for composing output messages into input message for other *e*-Services according to the orchestration specifications.

- *Compatible Service Provider*: *e*-Services to be instantiated are chosen according to the specifications of services in the orchestration schema. The compatible service provider provides functionality to select services that, in terms of interface and state diagrams, are compatible with the

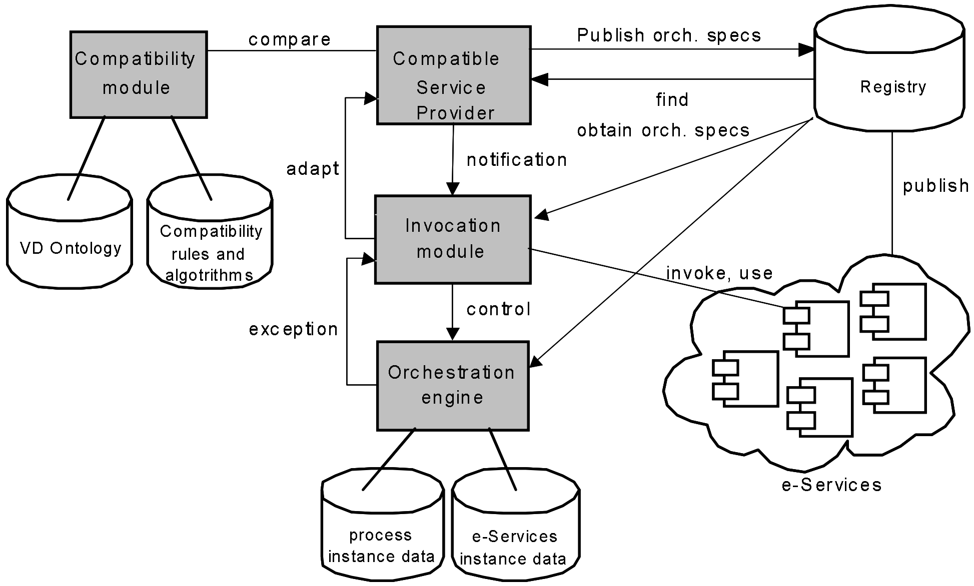

Figure 1: VISPO architecture

description in the specification. The compatible service provider ranks compatible services on the basis of a domain ontology, which allows the tool to recognize similarities between services. A compatibility module evaluates similarities using descriptors.

- *Invocator*: it selects the e-Service to be invocated in a process, selecting from the compatible service list notified by the compatible service provider, and provides wrapping functionalities. Each e-Service is invocated in terms of concrete services it provides, and the mapping to actual parameters is performed by the invocator. A first implementation provides only wrapping facilities based on mapping of names of message or parameters, but more advanced functionalities are being studied.

4 Implementation

VISPO is being developed in a service-oriented implementation framework. Each component is a service which provides functionalities to other modules in the architecture. Currently, the IBM DB2 implementation of UDDI has been extended with API to support descriptor-based service retrieval, thus obtaining the Registry. The Compatible Service Provider interacts with the ARTEMIS tool [12] for ontology management with request-response interactions based on HTTP. The Invocator and Orchestrator have been implemented as a single module using IBM MQ Series Workflow [13] as a basis for the workflow engine. The Invocator interacts with the Compatible Service Provider over HTTP, receiving a list of compatible services for each e-Service to be invocated in the process.

5 Concluding Remarks

Future work in VISPO will concentrate in dealing with exception during process execution, to handle both failures local to organizations, and infrastructure failures on the basis of a peer-to-peer approach for interaction between architectural components. In addition, the invocation module will be extended with features to adapt available e-Services to given process specifications.

The work described in this paper will evolve in the framework of a new project, entitled MAIS (Multi-channel Adaptive Information Systems). The main focus in MAIS will be on providing *e*-Services on a multi-channel platform, and in studying mechanisms for transaction flexibility according to global parameters, such as network connections availability and bandwidth, middleware services, local computing and storage facilities, quality of service parameters.

Acknowledgments

This work has been partially supported by Italian MIUR through the "Fondo Strategico 2000" Project *VISPO (Virtual-district Internet-based Service PlatfOrm)* (`http://cube-si.elet.polimi.it/vispo/index.htm`) and the "FIRB 2001" Project *MAIS (Multi-channel Adaptive Information Systems).*

References

[1] J. Yang, W. Heuvel, and M. Papazoglou, "Tackling the Challenges of Service Composition in *e*-Marketplaces," in *Proceedings of the 12th International Workshop on Research Issues on Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE-2EC 2002)*, San Jose, CA, USA, 2002.

[2] C. Bussler, R. Hull, S. McIlraith, B. Pernici, M. Orlowska, and J. Yang, Eds., *Proceedings of the CAISE 2002 Workshop on Web Services,* e-*Business, and the Semantic Web: Foundations, Models, Architecture, Engineering and Applications (WES 2002).* Springer-Verlag LNCS, in press.

[3] F. Casati and M. Shan, "Dynamic and Adaptive Composition of *e*-Services," *Information Systems*, vol. 6, no. 3, 2001.

[4] O. Williamson, *Mechanisms of Governance.* Oxford University Press, Oxford, 1996.

[5] M. Mecella and B. Pernici, "Bulding Flexible and Cooperative Applications based on *e*-Services," submitted for publication, November 2002.

[6] M. Mecella, B. Pernici, and P. Craca, "Compatibility of *e*-Services in a Cooperative Multi-Platform Environment," in *Proceedings of the 2nd VLDB International Workshop on Technologies for* e-*Services (VLDB-TES 2001)*, Rome, Italy, 2001.

[7] V. D. Antonellis, M. Melchiori, and P. Plebani, "An approach to Web Service compatibility in cooperative process," in *Proceedings of Workshop on Service Oriented Computing (SOC)*, Orlando, 2003.

[8] UDDI.org, "UDDI Technical White Paper," http://www.uddi.org/pubs/lru_UDDI_Technical_Paper.pdf, 2001.

[9] M. Mecella, F. Parisi Presicce, and B. Pernici, "Modeling *e*-Service Orchestration Through Petri Nets," in *Proceedings of the 3rd VLDB International Workshop on Technologies for* e-*Services (VLDB-TES 2002)*, Hong Kong, Hong Kong SAR, China, 2002.

[10] E. Colombo, C. Francalanci, and B. Pernici, "Modeling coordination and control in cross-organizational workflows," in *Proceedings of the 10th International Conference on Cooperative Information Systems (CoopIS 2002)*, Irvine, CA, USA, 2002.

[11] E. Colombo and C. Francalanci, "Formalizing governance in virtual districts," in *Proceedings of Workshop on Service Oriented Computing (SOC)*, Orlando, FL, USA, Jan. 2003.

[12] S. Castano and V. D. Antonellis, "A schema analysis and reconciliation tool environment for heterogeneous databases," in *Proceedings of IDEAS'99 Int. Database Engineering and Application Symposium*, Montreal, Canada, August 1999.

[13] IBM, "MQSeries Workflow," http://www-3.ibm.com/software/ts/mqseries/workflow/, link checked November 2002.