A VOC Based Approach to Support Virtual Organizations

Hua Zhou[1], Min Liu[2],Cheng Wu[3]

[1] Department of Automation, Tsinghua University, Beijing 100084, China
Zhouhua02@ tsinghua.org.cn
[2]Department of Automation, Tsinghua University, Beijing 100084, China
lium@mail.tsinghua.edu.cn
[3]Department of Automation, Tsinghua University, Beijing 100084, China
wuc@tsinghua.edu.cn

ABSTRACT

Employing IT as the key enable technology, virtual organization (VO) is primarily characterized as being a network of independent, geographically dispersed organizations (member organization) providing electronic services via Internet. To align these services effectively, one of the main challenges is to model cooperation in VO and provide correspondent management tools. Here we present a VOC (VO Structure-Organization Resource-Character) approach to model and run VO. VOC consists of three models. VO Structure model describes how VO functions in terms of VO Role (VR), Protocol; Organization Resource model describes potential service providers (potential member organization) capable participate VO. Character Model describes dynamic binding relationship between VO role and member organization. Following the VOC model，a platform supporting the design, administration, and running of VOC is given.

Keywords: Virtual Organization, VOC model, VO Structure Model, Organization Resource Model, Character Model

1. INTRODUCTION

A virtual organization (VO) is primarily characterized as being a network of independent, geographically dispersed organizations with a partial mission overlap[1,2]. Each VO is built to carry out some processes and achieve certain goals, and as the products and services provided by a virtual organization are dependent on innovation and are strongly customer-based, the organization structure must be adaptively dynamic. Within the network, all partners provide their own core competencies as deliverable services, and in a networked computational environment, some of these services are delivered in electronic form, for example, web services. To make a VO work flexibly and smoothly, a proper VO model and correspondent supportive running environment that can align these e-services are definitely necessary.

In this paper, we present a VOC (VO Structure-Organization Resource-Character) based approach to model and run VO. The VOC approach consists of a VOC model and correspondent supportive platform.

In section 2, we illustrate VOC model and some core processes, in section 3 a VOC-based VO management platform prototype is introduced and several correspondent core processes is then illustrated. Then we present a collaborative manufacturing example in section 5.Finally we conclude some characteristics of the VOC approach.

2. VOC MODEL

In a VO, each member organization (MO) has to assume certain duties, deliver prescriptive services, join and leave VO dynamically. In VOC approach, we model the whole VO in several different models with each one focusing on different aspect of VO.

VOC model consists of three sub models:

- VO Structure Model (VSM)
- Organization Resource Model (ORM)
- Character Model (CM)

The VOC meta model is described in Figure 1.

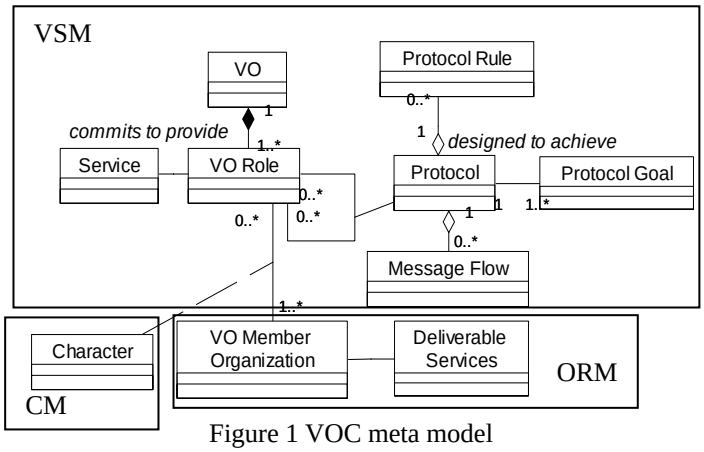

Figure 1 VOC meta model

2.1 VSM

The VO Structure model (VSM) describes VO structure in terms of VO Roles and Protocols. VO Role (VR) acts as a duty placeholder and commits to realize certain services in VO. VR will be dynamically assigned to MO

at VO design time and runtime (MO being the undertaker of VR). VR undertaker must realize the services VR commits to provide. VR cooperates with each other to achieve VO goals. The cooperation relationship between VR is modeled as Protocol. Protocol includes protocol rule and Message Flow. The former represents constraints that VR has to comply with. The latter illustrates message template and message sequence exchanged between protocol participants (VR). Protocol is instantiated as conversation which maintains the concrete context and threads participants will refer to frequently during cooperation.

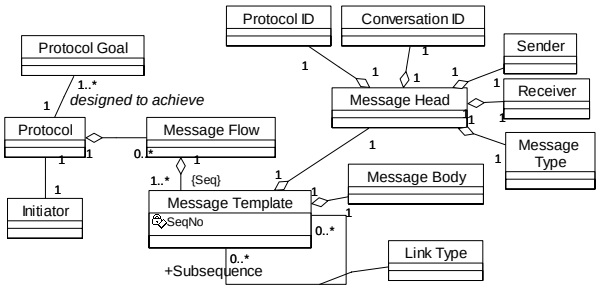

Figure 2 Message Flow

VR models static aspect of VO while Protocol models dynamic aspect. Each Protocol is designed to satisfy certain cooperation goals. For example, we can design a protocol based on Contract Net with its goal being task delegation. During runtime, VR undertaker can only choose from prescriptive protocols to cooperate with other VR, and follow exactly the Message Flow (Figure 2). Each protocol has an Initiator who is responsible for creating a protocol-based conversation. Each Message Template has a head and body. The Head includes message related protocol ID, conversation ID, sender VR, receiver VR and message type. The Body contains message content. Conversation participant must send message in according with the sequence and template specified in Message Flow to help each other indicate, understand, respond message quickly and properly.

2.2 ORM

Organization Resource Model (ORM) describes MO's capability and deliverable services. ORM of VO contains all potential MOs of a VO. Organization in ORM can be deputed VR dynamically by VO Administrator.

2.3 CM

Character Model (CM) is responsible for binding ORM and VSM. To make a VO functions as will, each VR has to be assigned to certain MO. We name this kind of 'Play' relationship between VR and MO as character. A MO can assume several characters at the same time.

3.VOC-BASED PLATFORM

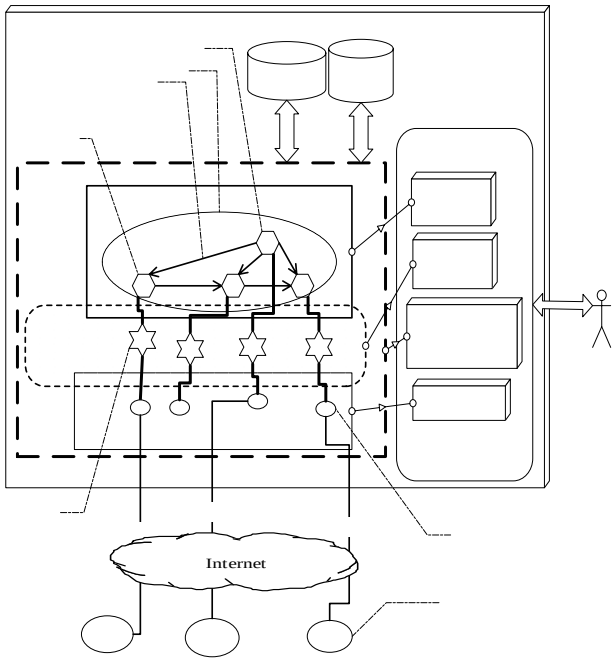

Figure 3 VOC based Platform structure

Based on VOC model, we developed a VOC based platform prototype (Figure 3). The core VOC model element is realized as Java object running in classified pools: VSM Pool; Character Object Pool; MO Agent Pool. Each kind of pool is managed by a correspondent software component.

By using the platform, we can design and run VO. The platform includes a VOC Model DB, a VO Instance DB, and VO Tools.

- VOC Model DB

VOC Model DB stores all VOC model data, including VSM, ORM and CM. VOC model works just like a class in Object –Oriented technology. A running VO is an instance of a special VOC model and consists of different types of software objects (or components) living in correspondent object pools.

- VO Instance DB

VO Instance Db stores all VO instance related data such as VO start time, VO original VOC model, VO MO etc.. By providing VO Instance DB, we can adjust VO in a case-based approach and thus providing a flexible way of managing VO.

- VSM Mgr

VSM Mgr is responsible for VSM management, including VSM creating, saving and dynamic adjustment, VSM Object initiation and life cycle management. VSM Mgr is the manager of VSM pool. There are four types of object in VSM Pool:

  - ✓ VO Object

A VO object is the controller of a running VO. When a VO is initiation the system will create a VO object and this VO object will be responsible for managing the whole life cycle of VO and create, manage, destroy other VSM object in this VO as needed.

✓ VR Object

Each VR in VSM will create a VR Object at VO runtime.

✓ Protocol Object & Conversation Object

Each protocol in VSM will create a Protocol Object at VO runtime. When a protocol-based interaction is initiated, a Conversation Object is created to manage it. Conversation Object acts as a communication channel between VO Object which can only send messages through the interface Conversation Object provided and in the protocol specified form and sequence. Conversation Object can get message and rule related information from Protocol Object at runtime. For there may be tens of thousands conversation being running at the same time, saving protocol related information in Protocol Object is wise choice to avoid frequent DB access.

- ORM Mgr

ORM Mgr manages ORM, and can instantiate MO Agent Object (MAO) according to the registered MO in ORM. Any organization that wants to participate in the VO must be registered in ORM. There are two ways of registering MO: active way and passive way. In active way, organizations must register themselves in ORM by providing their own capacities and deliverable services information. In passive way, VO Administrator has to search above information and identify appropriate organizations, contracts with them and registers them manually in ORM. Each registered organization will get a reference (though Web Service) to a MAO which is created by ORM Mgr in MO Agent Pool. When a registered organization exits from ORM, the correspondent MAO will terminated.

✓ MO Agent Pool

A MAO can only lives in MO Agent Pool. Each MAO is just a proxy of registered MO in VOC Platform and is exposed to related MO as an accessible web services. MAO provides an effective way of communicating with MO and is independent from the internal realization structure of original MO. We also provide an interface for MAO so that it can interact with Character Object smoothly.

- Character Binder

Character Binder manages the binding between MO and VR. A VO can not be instantiated until each VR has been assigned to a concrete MO. A MO-VR pair is represented as a character and instantiated as a Character Object (CO). Thus, within a VO, MO can cooperate with each other in this way: MO→MAO→CO→VR Object→Conversation Object→MAO→MO. This indirect way of interaction may bring some efficiency problems but illustrate a flexible way of reconfiguration. By making all objects active in pool (RAM), we can alleviate the efficiency problem to some extent.

- VO Admin Center

VO Admin Center is the center controller of the whole VOC-based platform. VO Administrator uses it to manage all VO running in the VOC-based Platform. VO Admin Center calls VSM Mgr, Character Binder, ORM Mgr to create, destroy, and adjust VOC model and instantiated VO as well.

3. VO CORE PROCESS

To build VO based on the platform in an effective way, we identified some core processes in constructing VOC model and running VO. These are processes the platform supports and each MO has to comply with. We illustrate several processes below:

- MO Registering Process

MO can participate in special VO by first registering itself in VOC-based Platform and then get a MAO reference.

- MO Terminating Process

MO can only terminate from running VO after issuing MO-TERMINATION message to all other VRs and complete works on hand. Thus other VRs won't send it any more messages after their completing to-be-terminated MO related cooperation work. The platform will then destroy related CO, MAO.

- MO Entering VO Process

After a special MO is selected to assume a VR, it has to realize all services specified in VR specification. First, the platform will send the target MO a VR specification identifying all services and protocols MO has to realize in detail. Then MO must design and realize it, and then answer the platform with a SERVICE-READY message. Only after this process, the platform can confirm the VR-INSTANTIATION-READY status of VR.

- VSM Instantiating and Terminating Process

The platform first checks all its VR to confirm all of them being in VR-INSTANTIATION_READY status, and then creates the whole VO instance in this sequence: VO Object→VR Object→Character Object.

To terminate a exist VO instance, the platform first check all VR to confirm all MO has issuing MO-TERMINATION message and then destroy the whole VO instance in this sequence: Character Object→VR Object→VO Object.

4. SAMPLE APPLICATIONS

By referring a production process in an aeronautic assembly factory, we construct our sample application

to verify the effectiveness of the platform (See Figure 4). The product experience 2 stages before assembled. Due to the cost-reduction problem, the first two stages is outsourcing to two different factories. But the contractual manufacturer is not fixed to allow more flexibility to purchase lower cost and higher quality. And production stages will change from time to time due to technology development. That is to say, the production routine and contractor list is not static. Thus the factory has to issue production task and cooperate with contractual manufacturers in an unstable production net. A production-net-adaptive platform supporting collaborative production task allocation is helpful.

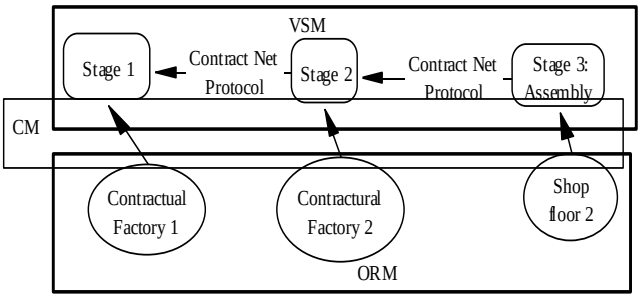

Figure 4 Collaborative Manufacturing Example

By applying VOC approach in this example, we can build Stage as VSM VR, and manufacturing contractor as MO, and the contractual manufacturing relationship as Character, thus we can adjust the collaborative relationship flexibly. Meantime, by designing proper VR Protocol suitable for production task allocation, we can change task allocation procedure as well.

5. CONCLUSIONS

The VOC based approach of managing VO has the characteristics as follows:

- ✓ Reusable

We divide the whole VO into three models with each one focusing on different aspects of VO. VSM models the static structure and cooperation relationship between MO. ORM works as a resource pool while VO Administrator can choose from. Character Model connects VSM and ORM to form a concrete VO. Thus we can conclude the reusable issue into these three levels. In VSM level, a historical VO structure model can be a reusable unit. For example, we can elicit a proved effective VSM into enterprise knowledge base and turn it into the original reference model of an upcoming VO (like the case of section 5). In ORM level, a MO is shared between different VO and can be assigned different VR from different VO at the same time. A MO's Character history can identify what kind of VR (protocols, services) it is most suitable for.

- ✓ Reconfigurable

As MO cooperate with each other in an indirect way and the existence of VO Instance DB, VO Administrator can reconfigure the VO structure by adjusting VSM, ORM or CM. For example, you can change the cooperation constraints by applying different protocol between VR.

- ✓ Model driven

Each VO is initiated and run based on the VOC model. You can influence the VO behavior by just adjusting VO case model stored in VO Instance DB.

- ✓ Scalable

ORM works as a resource pool where all potential VO participants can enter and exit flexibly.

Future research will focus on dynamic VR delegation and dynamic VR selection.

ACKNOWLEDGEMENT

ICEB2004 is supported by the National Natural Science Foundation of China.

REFERENCES

[1] Yates, J., Orlikowski, W.J., Woerner, S.L., "Virtual organizing: using threads to coordinate distributed work", *Proceedings of the 36th Annual Hawaii International Conference*, pp271-280, Hawaii,6-9 Jan. 2003

[2] Troy J. Strader, Fu-Ren Lin b, Michael J. Shaw C, "Information infrastructure for electronic virtual organization management", *Decision Support Systems*, Vol.23, pp75–94,1998

[3] IGLESIAS C. A, GARIJO M, and GONZALEZ J. C., "A survey of agent-oriented methodologies", *Proceedings of the 5th International Workshop on Intelligent Agents*, pp317-330, Heidelberg, Germany: Springer-Verlag, 1999.

[4] WOOLDRIGE M et al., "The Gaia Methodology for Agent-Oriented Analysis and Design", *Journal of Autonomous Agents and Multi-Agent Systems*, Vol.3, No. 3, pp 285-312, 2000

[5] FERBER J, GUTKNECHT O.,"A Meta-Model for the Analysis and Design of Organizations in Multi-Agent Systems", *www.madkit.org/publication*