Institution-Governed and Contract-Ensured Hierarchical Self-Organization of Service Cooperation and VOs

Ji Gao and Shiping Ye

***Abstract*—This paper proposes a multi-agent technology-based framework ICHO for supporting the self-organization of service cooperation and virtual organizations. Based on the model of IGTASC, ICHO can remove the so-called "trust" crisis which occurs due to the inherent non-controllability of business services across different management domains and be integrated closely with real-life application software by developing three mechanisms: Institution-governed, Contract-ensured, and Hierarchical self-Organization.**

***Index Terms*—contract-ensured; institution-governed; self-organization; service cooperation**

I. INTRODUCTION

Along with the development of Service-Oriented Computing (SOC) and Service-Oriented Architecture (SOA), constructing Virtual Organizations (VOs) by creating service cooperation (i.e. service-oriented cooperation) has become the mainstream approach for reforming the development of application software systems in Internet computing environments [1], [2]. Evidently, without the self-organization of service cooperation, it is difficult to realize the large-scale deployment of VOs. However, the inherent non-controllability of business services across different management domains has brought on the so-called "trust" crisis that the success and benefit of cooperation cannot be ensured. And it is this crisis that cumbers the self-organization of cooperation, makes the organization of cooperation have to depend on a great deal of manual intervention.

Endowing services with autonomy and intelligence by extending descriptive structures of web services can support the self-organization of service cooperation in a certain extent (as semantic grid community has advocated [3]). But, the granularity for descriptive structures of web services is too small to accommodate the mechanism for removing this "trust" crisis.

We think that providing high-performance autonomy and intelligence is the prominent advantage of Agent and Multi-Agent System (AaMAS) technology, and the agent cooperation-based VOs have been researched for a long time. So it seems good to achieve the self-organization of service cooperation depending on this technology. However, such self-organization also suffers the same "trust" crisis when agents participating dynamically in cooperation distribute in different management domains. Another hindrance is that this technology is disjoined with real-life application software systems. Due to the complexity and heterogeneity of traditional development environments and infrastructures for those systems, integrating seamlessly the technology into them is very difficult. Even though there are some successful cases, the integration methods are special, and therefore lack the value for generalization.

This paper proposes a framework for the self-organization of service cooperation, called ICHO (Institution-governed and Contract-ensured Hierarchical self-Organization), which can support the institution-governed self-organization of service cooperation and VOs effectively and remove the two hindrances depending on the model of IGTASC [4] we have established (see section Ⅱ). While contract-ensured self-organization ensures, by signing role-enacting contracts and service contracts (simply, service contracts), that service cooperation-based VOs achieve appointed objectives and benefits, hierarchical cooperation can simplify self-organization and self-management of complex VOs by creating multi-level architecture (nested VOs) based on the fractal structure of agent local business activities.

This paper is organized as follows. The next section introduces the foundation for achieving ICHO. Then, section Ⅲ expatiates upon the framework of ICHO, including the logical model of ICHO and the mechanisms for implementing institution-governed and contract-ensured hierarchical self-organization of service cooperation and VOs. After the implementation description and application analysis in section Ⅳ, the discussion and comparison of relative work (in section Ⅴ) and the conclusions (in section Ⅵ) are given.

II. REALIZATION FOUNDATION

In order to support the self-organization of service cooperation and VOs, We have established the model of IGTASC (Institution-Governed Trusted and Autonomic Service Cooperation) to eliminate the two hindrances mentioned in section Ⅰ. It is this model that creates the foundation for achieving ICHO.

A. Towards the Resolution of "Trust" Crisis and "Disjoined" Hindrance

Although current techniques of network and information security and reputation management can enhance the creditability for service provision, they are only infrastructure-level techniques for solving "trust" crisis, and unable to remove the origin of this crisis: the inherent non-controllability of business services across different management domains.

As a recent research hot-point of behavior theory of

Manuscript received March 18, 2010. This work was supported in part by the National Science Foundation of China (Grant 60775029), the National High-Technology Research and Development Program (863) of China (Grant 2007AA01Z187), the Priority Theme Emphases Project of Zhejiang Province, China (Grant 2008C13074), and the Natural Science Finds of Zhejiang Province, China (Grant Y107446).

J. Gao is with the College of Information Science & Technology, Zhejiang Shureng University, and also the College of Computer Science & technology, Zhejiang University, Hangzhou, CO 310015 China (phone: +86-571-8799-6934; e-mail: gaoji1@zju.edu.cn).

S. Ye is with the College of Information Science & Technology, Zhejiang Shureng University, Hangzhou, CO 310015 China (e-mail: zjsruysp@163.com).

multi-agent systems, institution-governed cooperation (the core mechanism for implementing normative multi-agent systems) has facilitated the resolution of the "trust" crisis greatly. This mechanism aims at establishing sound protocol systems called e-institutions as the regulations for constraining, in macro level, outside-visible social and cooperation behaviors between members (agents) dynamically participating in systems so that the behaviors and their effects of members can be predicted and controlled precisely as long as all of them conform to those regulations [5]-[8].

However, how to ensure that all of cooperation behaviors conform to those regulations is confronted with a real challenge due to the non-controllability problem mentioned above. In order to overcome this challenge, we have established the model of IGTASC to reform the mechanism of institution-governed cooperation by making it couple closely with other two mechanisms: policy-driven self-management and cooperation facilitation management (see section Ⅱ.*B*).

Another reformation made by IGTASC is adopting the "service-oriented" concept as a main line to transform the mechanism of institution-governed cooperation and design the mechanisms of policy-driven self-management and cooperation facilitation management in order to remove the "disjoined" hindrance mentioned above and to enable those mechanisms and the AaMAS technology as their basis to support seamlessly service cooperation-based VOs.

B. Brief Introduction of IGTASC

In order to support effectively the autonomic construction, running, and evolution of VOs, IGTASC proposes a three-level Virtual Society (VS) as the environment where VOs live and work (Fig. 1). VS is defined as a 3-tuple:

VS = (AC, TV, RA)

• AC: the Agent Community for VOs and agents to live.

• TV: the set of TAVOs (Trusted and Autonomic VOs).

• RA: the set of rational agents registering in the community.

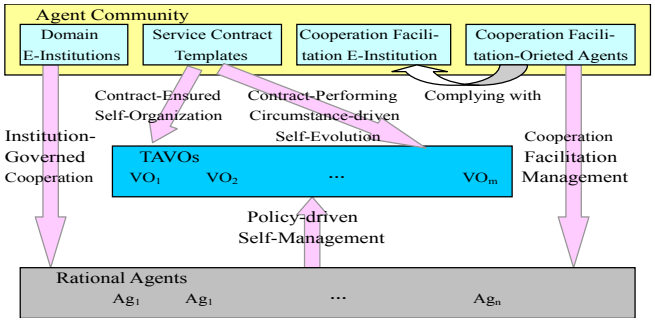

Fig. 1 Three-level virtual society supported by three technologies constituting model IGTASC

IGTASC depends on three technologies to make service cooperation both trusted and autonomic: institution-governed cooperation, policy-driven self-management, and cooperation facilitation management. The former formulates domain e-institutions (electronic institutions) as social regulations to govern service cooperation in macro level so that it can be trusted that service cooperation created dynamically will achieve required objectives as long as cooperation behaviors all conform to relevant regulations respectively. The middle aims at using management policies to drive agents to make their own micro-level behaviors comply rationally with relevant regulations while the latter deploys the agents enacting cooperation facilitation-oriented roles formulated in the agent community in order to facilitate and force cooperation behaviors' conformity to regulations.

A domain e-institution is composed of two parts: social structure standards and coupling cooperation behavior norms. The former, as the hard constraints cooperation participants can not violate, aims at formulating business services provided or consumed, business operation-oriented roles which agents can enact in service cooperation, and the distributed business process for multiple agents to cooperate. In contrast, the latter is the soft constraints which the business operation-oriented roles and distributed business process should comply with, including obligations, forbiddances, and rights (promises).

III. Self-Organization Framework ICHO

ICHO restricts the organization of a TAVO to the most familiar cooperation form in human society: an alliance based on service providing-requiring relations, which is sponsored and created by some physical organization to satisfy a business requirement dynamically occurring (such as making new products, solving complex problems, searching for knowledge, purchasing merchandise, etc.). Of course, every member of a VO should set up an agent as its broker.

Such an alliance often concerns multiple binary collaborations which are managed by the sponsor centrally, but there are no interactions between other members. Although there may be the requirement for direct interactions between the providers of different services, these interactions can be removed by partitioning business activities reasonably and arranging the appropriate messages sent by the alliance manager. Also, although there are other cooperation forms in human society, alliance form is still the mainstream.

A. Logical Model of Self-Organization

According to the proposed organizational form of a TAVO, the agent that requires reaching some local business goal by cooperation becomes the sponsor organizing such a TAVO while the local business process for reaching this goal is used as the basis of organization. Thus, the logical model of ICHO is defined as a multi-tuple:

ICHO = (*m*, *g*, *lbp*, bs-set, pvm-set, vm-set, matchmaker, ie-set, contract-set, joint-intention, service-obeying, recommending, nego-selecting, contract-signing), where

• *m*: the sponsor of a TAVO which organizes and manages the VO centrally;

• *g*: *m*' local business goal required to reach by cooperation;

• *lbp*: the local business process for *m* to conform to in reaching *g*; *lbp* is formulated by the owner of *m* and used as a constituent representing desires in *m*' mental model BDI;

• bs-set: the set of business services requiring to be acquired from the outer, which are provided by partner agents and used to reach *g* according to *lbp*; here, different services may be specified in different domain e-institutions;

• pvm-set: the set of potential members relevant to the VO; each *pvm* (∈pvm-set) is a business operation-oriented agent which is able to provide *bs* (∈bs-set);

• vm-set: the set of members in the VO; here, vm-set = {*m*} ∪ { *pvm* | *bs* ∈bs-set and *pvm* = nego-selecting (*bs*) }, *m* and *pvm*s, which are selected by *m* to provide the business

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

services in bs-set, constitute the VO dynamically;

• matchmaker: the agent aiming at recommending *m* with proper providers of business services by performing the cooperation facilitation service of partner-recommendation;

• ie-set: the set of domain e-institutions relevant to bs-set;

• contract-set: the set of service contracts; here, contract-set = {*sc* | *bs*∈bs-set, *vm*∈vm-set\{*m*}, and *sc*= contract-signing (*bs*, *m*, *vm*) };

• joint-intention, which is a joint commitment for VO members to achieve service cooperation according to ie-set and contract-set;

• service-obeying: bs-set → ie-set; here, for each *bs* (∈bs-set), both its provider and consumer should conform to the service providing-requiring standards and cooperation behavior norms formulated in the *ie* (∈ie-set) relevant to *bs*;

• recommending: bs-set → $2^{\text{pvm-set}}$; here, matchmaker recommends *m pvm*s which can provide the business services in bs-set and possess the capability matching applicability requirement;

• nego-selecting: bs-set → pvm-set; here, for each business service *bs* (∈bs-set), *m* determines, by means of negotiation and selection, the provider of *bs* from *pvms* recommended by matchmaker;

• contract-signing: bs-set × {*m*} × pvm-set ↛ contract-set; here, for each business service *bs* (∈bs-set), *m* signs a service contract with *pvm* (= nego-selecting (*bs*)).

Based on this logical model, three mechanisms for achieving the self-organization of service cooperation and VOs: institution-govern, Contract-ensured, and Hierarchical self-Organization, are proposed in next subsections.

B. Institution-Governed Self-Organization

This mechanisms supports service cooperation self-organization from three aspects:

1) The social structure standards formulated in domain e-institutions enable agents to participate in cooperation dynamically and freely as long as they register relevant roles to enact in agent community and are configured with the skills for providing / consuming those services.

2) The detailed partition of application domains enables domain ontologies of e-institutions to support the precise specification of service capability and applicability.

3) The service contract templates defined in domain ontologies and the uniform 'negotiation' service defined in the social facilitation e-institution oriented to agent community support autonomic rational negotiation and contract creation effectively.

Next, two methods are created to implement institution-govern self-organization of service cooperation and VOs: ACDP-based method for finding applicable service providers and policy-driven 4-phase method for VO self-organization.

a. Finding service providers based on ACDPs

The classification systems for domain services and Applicability Circumstance Description Patterns (ACDPs) form so-called service applicability circumstance ontology, which is used as the uniform semantic foundation to describe service applicability exactly, comprehensively, and facilely.

Definition 1 (Applicability Circumstance Description Pattern ACDP): define the ACDP of business service *bs* (∈bs-set), denoted with ACDP^{bs}, as the set of feature slots fs_i^{bs} such that $\text{ACDP}^{bs} = \{fs_1^{bs}, fs_2^{bs}, \ldots, fs_n^{bs}\}$, $fs_i^{bs} = <fsn_i^{bs}, tcs_i^{bs}>$; fsn_i^{bs} indicates the name of feature slot which must be single in ACDP^{bs} while tcs_i^{bs} indicates the set of selectable terms which can be filled as a slot value: $tcs_i^{bs} = \{term_{i1}^{bs}, term_{i2}^{bs}, \ldots, term_{im}^{bs}\}$.

Definition 2 (service Applicability Circumstance AC): let AC(*k*) indicate the description of applicability circumstance for providing (requiring) service *k* (as an instance of *bs*), then $AC(k) = \{fs_1^k, fs_2^k, \ldots, fs_n^k\}$, $fs_i^k = < fsn_i^{bs}, tcs_i^k >$, $tcs_i^k \subseteq tcs_i^{bs}$.

Thus, let *u* indicate a service provided by some business operation-oriented agent, *v* a service required by another agent, and *u* and *v* are all the instances of *bs*. As long as let applicability circumstance AC (*u*) as the index of *u*' advertisement and AC (*v*) as the description of *v*, the matchmaker can determine the applicability of *u* for *v* by compatible match (the compatibility of slots and slot values).

As an example we have created, the e-institution of DropShop defines the classification system of RetailBroker service, which is expressed as a classification tree of for-sale goods provided, and creates ACDPs for each bottom class of goods. Two ACDPs for class "GoodsA": GoodsAProperties and GoodsAUseCondition are as following:

```
ACDP   GoodsAProperties
       Material: ("bowlder", "woodiness", "metal");
       Colour: ("red", "blue", "green", "yellow", "black");
       ColourMethod: ("natural", "dope", "paint");
ACDP   GoodsAUseCondition
       Humidity: ("dryness", "wetness", "none");
       Temperature: ("low", "room", "none");
```

They enable the applicability circumstances specified respectively by the provider and consumer of RetailBroker service to comply with the same semantics. Suppose service provider "pro" specifies the ACs for Properties and UseCondition of GoodsA respectively as follows:

Material: ("woodiness", "metal"); Colour: ("red", "blue", "green"); ColourMethod: ("natural", "dope", "paint");
Humidity: (("dryness", "wetness"); Temperature: ("room", "none").

And service consumer "req" specifies them as follows:

Material: ("bowlder", "woodiness"); Colour: ("blue", "green", "yellow"); ColourMethod: ("natural");
Humidity: ("dryness"); Temperature: ("none").

Then *pro* just is the service provider satisfying the applicability circumstance of *req* due to the intersection of all relevant feature slots.

By the detailed partition of application domains, we can design the service applicability circumstance ontology according exactly with domain features from the angles of service performance, quality assurance, application condition, service maintenance, etc. therefore make the method for finding applicable service providers possess excellent performance.

b. Policy-driven 4-phase VO self-organization

The dynamical creation of a VO can be viewed as the process for *m* and potential VO members to create joint intention. This process is partitioned into four phases: Determine service cooperation requirement, Request recommending cooperation partners, Select service cooperation partners, Form contract-based joint intention.

In these phases, all of local behaviors of *m* and *pvm*s are policy-driven. Every policy for managing local behaviors is represented as a 6-tuple:

policy = (name, type, processing, target, trigger, update),

which aims at specifying the name, type (Authorization or Obligation), processing activities, target (work module for performing this policy), trigger (policy activating condition),

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

and update (last update date) of a policy. Therein, processing activities are expressed as the sequence of operations, rules and rule groups. A policy for transferring outer messages received by *m* to its work modules is as following:

```
Policy
    Name:   "OuterMessageTransfer";
    PolicyType:   "Obligation";            //"Obligation" type policy
    Processing:   (ruleGroup   MessageTransferProcessing);
    Target:   (@Service   "MainControl"   "GS");
        // "MainControl" is a work module obligated to perform this policy
    Trigger:   (@Message Type:?s   PartyID:?x   Service:?y   Action:?z
            ConversationId:?w   CooperationRole:?v   MessagePayload:?u);
        //This policy is activated by a message current agent receives
    Update:   2009-3-21;    //Last update date of this policy
End   Policy
ruleGroup   MessageTransferProcessing
    mode:           p;          //Production rules
    select:         first;      //Use the first activated rule
    ruleList:
    (→   ($= ?s "WebRequest")   ($SendMessage "MainControl" ?u) );
      //If the message is from Web browser of user, send the binding value
        of ?u to work module "MainControl"
    (→   ( ($= ?s "ServiceReturn")   ($= ?y "PurchaseNegoutiation") )
          ($SendMessage   "Negotiation"   ($CreateConceptInstance
            "CooperationProposal" ($TakePartyID) ?x ?y ?w "provider"
            "opponent" ?u *) )
     //If the message is the purchase proposal from service provider, send
    the binding value of ?u (i.e. the proposal) to work module "Negotiation"
    …
      End   ebXMLMessageForwardProcessing
```

All work modules of an agent, including MainControl, the modules for supporting VO self-organization (Service Scheduling, Cooperation, and Negotiation), and the modules for VO running, are configured with a number of behavior management policies respectively. It is those policies that drive *m* and potential VO members to make their own behaviors in VO self-organization conform to cooperation behavior norms formulated in e-institutions and the business instructions sent over by agents' owners. Also, the mapping functions included in ICHO multi-tuple (see section Ⅲ.*A*) are all implemented by activating one or more policies.

(1) Determine service cooperation requirement

Whenever a local business goal (*g*) requiring to be reached by cooperation occurs, *m* determines the business services requiring to be provided from the outer according to the local business process (*lbp*) for reaching *g* (Fig. 2). Here, *lbp* describes the parameterized business process formulated by *m*' owner and represents the business activity for reaching *g* (g-activity) by composting the next-level business activities (n-activities). Some of those activities will be completed by invoking the business services provided by outer agents, and the standards (hard constraints) and norms for providing / consuming these services are formulated in relevant e-institutions. The BNF definition of *lbp* is defined as follows:

```
<g-activity> := {<Steps> | ( loop <Steps>)}+
<Steps> := { (← {return | <n-activity> | <activitySet>} [<condition>]) |
            (or { (← {<n-activity> | <activitySet>} [<condition>])  }+ ) }+
<n-activity> := l-activity | o-activity          //"l" and "o" indicates the business
        // activities executed by invoking the services provided from the local
          or the outer respectively; "return" indicates exiting from loop or lbp.
<activitySet> := ({sequence | concurrency} {(← <n-activity> [<condition>])}+)
```

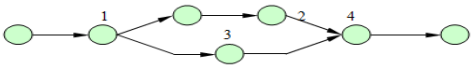

Fig. 2 A possible lbp of m, where some of n-activities (node 1-4) need to be completed by invoking services from the outer.

(2) Request recommending cooperation partners

Once determining the services requiring to be provided from the outer, *m* requests the matchmaker in Agent Community to recommend applicable partners for providing these services. Before making this request, *m* asks its owner to formulate the strategies for finding applicable providers of each service.

Every strategy is represented as a 7-tuple:

ServiceFindingStrategy = (StrategyID, ServiceName, SearchMethod, Constraints, MatchNumber, NegotiationAttitude, MaxNegotiationTimes),

where SearchMethod denotes the method for searching applicable service providers (e.g. Invite, Negotiation, Auction, etc.), Constraints the applicability conditions for service providers, MatchNumber the number of desired candidate service providers, NegotiationAttitude the attitude for *m* to adopt in negotiation of service cooperation (e.g. Anxious, Smooth, Cool, Resistant, etc.).

"Contraints" itself is expressed as a 3-tuple:

Constraints = (NecessaryConstraint, AttachedConstraints, ConstraintItems),

where NecessaryConstraint and AttachedConstraints denote the necessary and attached conditions for service providers to satisfy respectively. If there is more than one provider satisfying NecessaryConstraint, the providers satisfying more Attached Constraints are better. ConstraintItems are used to specify the constraints of negotiable items, including the initial, desired, and affordable extreme values and weights.

Note that each service required to create a providing-requiring contract should be configured with a contract-signing operator, and the parameters of this operator are defined as some concepts in domain ontology. It is those concepts and their slots that become the basis for formulating "Constraints" of service provider applicability.

Return to the RetailBroker service mentioned in section Ⅲ.*B.a*, the strategy for finding applicable providers of this service is formulated by *m*'owner as follows:

```
(@ServiceFindingStrategy   PolicyID: "001"   ServiceName: "RetailBroker"
SearchMethod: "Negotiation"   Constraint: (@ServiceConstraintType
NecessaryConstraint: (@PurchaseOrder   OrderHeader: <…>   Items: (
(@ServiceItem   Classification: <…>   Classificationsystem: <URL of classification
system>)   Item: (@GoodsA   UnitCost: ?x1   Number: 1000   Deposit: ?y1
Currency: "$"   GoodsAProperties: <…>   GoodsAUseCondition: <…>)   )
(@ServiceItem   Classification: <…>   Classificationsystem: <URL of classification
system>)   Item: (@GoodsB   UnitCost: ?x2   Number: 1000   Deposit: ?y2
Currency: "$"   GoodsBProperties: <…>   GoodsBUseCondition: <…>)   )   )
OrderSummary: <…>)   ($<   ?x1   120)   ($<   ?y1   12000)   ($<   ?x2   200)
($<   20000)   ConstraintItems: (
(@ConstraintItem   ItemName: ?x1   BottomSlotPath: <…>   InitialValueInNegotiation:
50   esiredValue: 70   AffordableExtremeValue: 100   ItemWeight: 10)
(@ConstraintItem   ItemName: ?y1   BottomSlotPath: <…>   InitialValueInNegotiation:
5000   DesiredValue: 7000   AffordableExtremeValue: 10000   ItemWeight: 2)
(@ConstraintItem   …)   (@ConstraintItem   …)   )   )
MatcherNumber: 5   NegotiationAttitude: "Smooth"   MaxNegotiationTimes: 5)
```

Here, concept "PurchaseOrder" is the single DBI (Definition of Basic Information) parameter of contract-signing operator for RetailBroker service (but defining more than one parameter is promised), prefix "@" denotes a concept instance pattern, and prefix "$" a relation expression or truth function expression. It is using such parameters as basis that enables general methods for formulating service finding / providing strategies, recommending applicable service providers, and supporting service cooperation negotiation to adapt to a variety of specific application domains.

This strategy indicates: two kinds of goods are required: GoodsA and GoodsB; purchase condition of GoodsA is

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

UnitCost (?x1) < $120, Deposit (?y1) < $12000, and GoodsAProperties and GoodsAUseCondition must be satisfied according to ACs (Applicability Circumstance) specified in the service consumer "req" mentioned in section Ⅲ.*B.a*; the initial, desired, and affordable extreme values of GoodsA' UnitCost (?x1) are $50, $70, $100 respectively; its ItemWeight is 10; etc. Besides, this strategy also indicates: SearchMethod, MatchNumber, NegotiationAttitude, and MaxNegotiationTimes are "Negotiation", 5, "Smooth", and 5 respectively.

Framework ICHO also requires the service providers to send the matchmaker their advertisements and specify the service provision strategies in similar mode (but the direction of constraints may be opposite). Since service finding and provision strategies are formulated on the same basis, the matchmaker can accurately find and recommend the service providers satisfying constraints.

(3) Select service cooperation partners

The mode for selecting service cooperation partners depends on the SearchMethod indicated in ServiceFinding Strategy. Here, we only consider the negotiation mode.

ICHO normalizes the negotiation process into the one for business service consumers to continually invoke cooperation facilitation service 'negotiation' provided by candidate business service providers and acquire returned responses. Although 'negotiation' service is defined in E-Institution^F (the social facilitation-oriented e-institution), this service itself must be configured into agents providing business services, and specialized into adapting to special business services by embedding the instantiations of service contract templates (see section Ⅲ.*C*) into input / output parameters of 'negotiation' service.

The process negotiating with a service provider, denoted as ServiceNegotiation, is sponsored by *m*, and expressed as a 7-tuple:

ServiceNegotiation = (*bs*, *m*, *can-pro*, Proposal, Neg-Status Neg-Policy, neg-service)

• *bs*: the business service requiring to create a service contract by negotiation

• *m*: the consumer of *bs*, which sponsors the negotiation process for creating this contract.

• *can-pro*: the current candidate provider of bs, *m*' negotiation opponent.

• Proposal: the proposal set of this contract, including an initial proposal and a number of reverse proposals. Each proposal is created by instantiating input parameters of contract-signing operator of *bs*, and included in the instantiation of input / output parameter of negotiation service (exactly, its negotiation operator).

• Neg-Status: the possible negotiation statuses; a status is represented with the current negotiation circumstance (e.g. the number of candidate providers for *bs*, the number of done negotiation turns, the score of current proposal received, etc.) and the negotiation strategy specified in ServiceFinding Strategy (e.g. ConstraintItems, NegotiationAttitude, Max NegotiationTimes, etc.).

• Neg-Policies: the policy set for driving the negotiation process and guiding the creation of proposals.

• neg-service: Proposal × Neg-Status → Proposal; here, *m* or *can-pro* receives a proposal, and then creating a reverse proposal based on the current negotiation status and the negotiation policies set up for them. Note, the approval or rejection of received proposals is regarded as an especial kind of reverse proposals.

m sponsors the negotiation process by invoking neg-service (the negotiation service defined in E-Institution^F and configured into *can-pro*) to transmit an initial proposal, whereas *can-pro* transmits a reverse proposal by returning the output parameter of the negotiation service. By transmitting reverse proposals, the cooperation contract is revised continually, and thereby the negotiation process is pushed forwards until the contract is accepted by both *m* and *can-pro* or rejected by one of them.

ICHO creates three negotiation policies for *m* and *can-pro* respectively: NegotiationEvaluation, NegotiationStrategy, and NegotiationTactic, which decide the content of proposals in three levels: evaluation, strategy, and tactic (referring to the method proposed by Jennings [9]). While ConstraintItems specified in ServiceFindingStrategy indicate negotiable items, NegotiationAttitude, and MaxNegotiationTimes affect the choice of negotiation strategies and tactics.

Based on such negotiation process, *m* can select, from several *can-pro*s, the provider of a service who approves the contract proposal corresponding to the highest score (in concurrent negotiation mode) or satisfying *m* (in sequential mode).

(4) Form contract-based joint intention

Once the cooperation partners providing business services from the outer are all selected and determined, *m* signs service contracts with these partners respectively by invoking the contract-signing operators provided in business services. Here, signing a contract means that both *m* and the provider validate this contract, including rights and obligations, and the relevant negotiation process finishes successfully. These signed contracts form the joint intention for them to reach cooperatively m' local business goal, and therefore result in the creation of a TAVO.

C. Contract-Ensured Self-Organization

Based on model IGTASC, framework ICHO creates two kinds of cooperation contracts: role-enacting, service providing-requiring, to ensure that the self-organization of service cooperation and VOs is trusted.

• Role-enacting contracts: ICHO requires every agent desiring to participate in service cooperation must register itself in the agent community. Then, once the registration for requesting to enact some business operation-oriented role in an e-institution is approved, a role-enacting contract is created and conserved in the community.

• Service contracts (service providing-requiring contracts): the cooperation contracts between the provider and consumer of a business service. Due to the proposed organization form of TAVOs mentioned in the beginning of section Ⅲ, only such two-party contracts are adopted.

The two kinds of contracts constrain agent cooperation behaviors in macro-level and micro-level respectively in order to implement contract-ensured self-organization of service cooperation and VOs. The former means that agents promise to make their own macro-level behaviors comply with the relevant soft constraints formulated in domain e-institutions, including the behavior norms for their roles and the ones for services provided / consumed by them. Norms can be domain-independent or domain-dependent.

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

For example, a domain-independent norm may stipulate when the provider of a service receives a proposal for requiring the service sent by a registered agent, this provider is obligated to return a reply (reverse proposal, agree, or rejection), whereas a domain-dependent norm may specify that providing some service must comply with a special environment protection law. It is such contracts that enable the creation and running process of service cooperation and VOs to be predicted exactly.

The latter constrains the micro-level behaviors for executing service cooperation, and it is those contracts that enable detailed service providing / consuming behaviors to be predicted exactly. Different from the behavior norms formulated statically in domain e-institutions, service contracts must be formulated dynamically. In order to avoid the difficulty for dynamically creating service contracts from scratch, ICHO allows for creating them by instantiating parameterized service-specific contract templates. Evidently, the templates created by human society in a variety of application domains (such as insurance and finance) can be used as samples.

A service contract template for business service *bs* in domain D, denoted by CT_D^{bs}, is expressed as a 3-tuple:

CT_D^{bs} = (DBI, QoSGT, CPPT)

• DBI: the Definition of Basic Information of service cooperation, which is used to specify the identity of both parties, the business transaction roles enacted by both parties, period of validity for this contract, service content (e.g. the operations or product items, price, number, and deadline), payment mode of requiring party, etc.

• QoSGT: the QoS (Quality of Service) Guarantee Template, which defines quality parameters and their measurement, and stipulates service level objectives (SLOs) based on those definition. Note, QoSGT includes some variables which need to be instantiated.

• CPPT: the Contract Performing Protocol, which is designed as a partial-order set composed of protocol entries represented as contract-performing norms. Also, CPPT includes some variables which need to be instantiated.

The instances of a SLO and a contract-performing norm for DataMining domain is given as following:

```
(eio:SLO
    SLOName:"SLOOfMining";     //Denote the SLO of operator "Mining"
    Operator:"Mining";   //This operator belonging to service "DataMining"
    SatifactionCondition:(@eio:ParameterValue   ParName:"ResponseTime"
        MeteringUnit:"Second"   Value:?x) ($<  ?x  ($+
        MaximumCommunicationTime   MaximumExecutionTime) )
    EvaluationPeriodUnit: "on-demand";     //evaluated once invoked
    EvaluationParty: "RespondingRole";     //evaluated by the provider   )
(eio:Norm                          //A simplified norm for ensuring operator quality
    NormNo: 21;                    //Norm 21 in contract performing protocol
    Performer:"RespondingRole" ; //The norm should be executed by provider
    Trigger: (@eio:OperationCall   Operator:"Mining"   CallTime:?x);
        //Triggered by an operator invacation event "(@eio:OperationCall
          Operator:"Mining " CallTime <...>)"
    Deadline: (@eio:dateTimePeriod   BeginTime:?x   Period:?nego_03);
      //The deadline completing the norm execution is ?nego_03 which begins
      //from ?x. Herer, the value of ?x come from unification examination
      //when this norm is triggered , and the value of ?nego_03 depends on the
      //nigotiation between providing and requring parties.
    Postcondition:(@eio:SLOSatisfactionStatus   SLOName:"SLOOfMining"
                Operator:"Mining"   Status:"True");   )
```

Here, the SLO requires that ResponseTime (?x) for invoking service operator "Mining" is less than the sum of stipulated values of parameters "MaximumCommunicationTime" and "MaximumExecutionTime", whereas the norm checks up whether this SLO is satisfied when the operator "Mining" is invoked.

However, such contract templates are domain-specific. In order to make them adapted to the general process of service cooperation self-organization, two definitions are given first:

Definition 3 (Contract Signing Operator CSO): define the CSO of business service *bs*, denoted by CSO^{bs}, as the operator for the providing and requiring parties to sign its input (denoted by CSO_i^{bs}) and output (denoted by CSO_o^{bs}) respectively.

Definition 4 (Signed Contract SC): define the SC of business service *bs*, denoted by SC^{bs}, as the $Sign_{req}(CSO_i^{bs}) \cup Sign_{pro}(CSO_o^{bs})$, where $Sign_{req}(CSO_i^{bs})$ indicates that the requiring party of *bs* signs on CSO_i^{bs}, which is the instantiation of service contract template CT_D^{bs}; and $Sign_{pro}(CSO_o^{bs})$ indicates that the providing party of *bs* signs on CSO_o^{bs}, which is a single parameter denoting the approval or rejection of CSO_i^{bs}.

Depending on the two definitions above, domain-specific contract templates enable the general process of contract-ensured self-organization to be specialized to suit a variety of application domains. Especially, such specialization can support last 3 phrases in the 4-phase method for VO self-organization efficiently.

• In phrase 2 (request recommending cooperation partners), CT_D^{bs} constitutes the common basis for both requiring and providing parties to formulating service finding / provision constraints. For instance, for RetailBroker service mentioned above, let $CT_D^{RetailBroker}$ = (PurchaseOrder, QoSGuarantee, ContractPerformingProtocol). The three components of $CT_D^{RetailBroker}$ not only constitute the input of $CSO_i^{RetailBroker}$, but also become the basis for formulating those constraints (see the example of ServiceFindingStrategy and its finding constraints for RetailBroker service in section Ⅲ.*B.b.(2)*).

• In phrase 3 (select service cooperation partners), the advance of negotiation process depends on contract proposal and reverse proposals. It is the instantiation of CSO_i^{bs} that constitutes those proposals. More important is ConstraintItems defined in ServiceFindingStrategy and ServiceProvisionStrategy. Those items specify the constraints of negotiable items in proposals, and therefore become the basis for negotiation policies to decide negotiation behaviors and proposal content. ConstraintItems for requiring party of RetailBroker service has also given in section Ⅲ.*B.b.(2)*.

• In phase 4 (Form contract-based joint intention), *bs*' requiring party invokes CSO^{bs} and sends $Sign_{req}(CSO_i^{bs})$; and then *bs*' provider return $Sign_{pro}(CSO_o^{bs})$ (signing CSO_i^{bs} and CSO_o^{bs} is completed by the communication modules of relevant parties). When $Sign_{pro}(CSO_o^{bs})$ is created by *bs*' provider and received by *bs*' requiring party, both parties send SC^{bs} (i.e. $Sign_{req}(CSO_i^{bs}) \cup Sign_{pro}(CSO_o^{bs})$) to the agent community to make a notarization. It is the notarization that makes SC^{bs} inure and enables the performance of SC^{bs} to accept the government of community regulating mechanism for forcing the conformity to cooperation contracts.

D. Hierarchical Self-Organization

The business activity for reaching *m*' local business goal *g* may have a fractal structure: the top-level business activity is completed by performing the local business process (*lbp*) composed of next-level business activities, and some next-level business activities are also completed by

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

performing the *lbp*s composed of next-next-level business activities; thus, the structure extends down until bottom-level business activities which can be performed by invoking basic service operators. It is the fractal structure that supports the implementation of hierarchical cooperation self-organization and the dynamical creation of nested VOs (Fig. 3).

Since VOs in different levels are controlled centrally by relevant (often different) agents, the nested VOs result in distributed central controls created dynamically, and thereby integrate the advantages of central and distributed control and avoid their deficiencies.

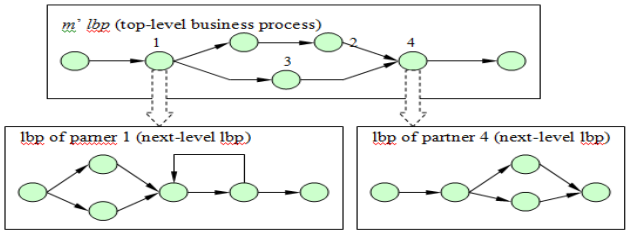

Fig. 3 The fractal structure of business activities results in the creation of a nested VO (Note: node 1-4 denote the activities requiring to be performed by partner 1-4)

IV. Implementation and Application Analysis

A. Implementation of ICHO

Depending on IGTASC-based TAVO development framework DFTAVO [4], we have implemented the framework for supporting service cooperation self-organization ICHO, including the three mechanisms for achieving the self-organization of service cooperation and VOs: institution-govern, Contract-ensured, and Hierarchical self-Organization. The key techniques for achieving those mechanisms are summed up as following:

- formulating the management policies for driving VO self-organization process, including policies oriented to main control, cooperation, lbp scheduling, and negotiation behaviors of an agent;
- defining the method for describing ACDPs (Applicability Circumstance Description Patterns), and the ACDP-based compatible match algorithm for finding applicable service providers;
- defining the method for specifying, based on service contract templates, ServiceFindingStrategy and Service ProvisionStrategy and the algorithm for examining matching and matching-degree between the two strategies;
- creating the negotiation service general in agent community and configured into business operation-oriented agents, and providing the method for being specialized into adapting to a variety of domain-specific business services;
- creating negotiation policies and the method for adopting those policies to drive negotiation process according to ConstraintItems, NegotiationAttitude, and MaxNegotiation Times specified in ServiceFindingStrategy;
- providing the method for creating Role-enacting contracts in macro level and Service contracts in micro level to realize contract-ensured self-organization of service cooperation and VOs
- defining the method for formulating service contract templates and the method for making those domain-specific templates adapt to the general process for service cooperation self-organization (especially, the last 3 phrases in the 4-phase method for VO self-organization);
- creating the method for dynamically establishing nested VOs by designing the fractal structure of business activities and *lbp*s in order to achieve dynamically-formed distributed central controls of service cooperation and VOs.

B. Application Analysis

We have used ICHO to establish several experimental TAVOs, such as small meeting arrangement, knowledge provision, data mining, multi-part device cooperation production, and multi-department crisis cooperation transaction. The experimental results indicate that ICHO can support the creation of service cooperation and VOs effectively in very different application domains.

Next, we adopt a supposed application of data mining to analyze the VO self-organization process based on ICHO. In order to complete a data mining task delegated by some enterprise, *m*, as a broker of the company accepting this task, will firstly choice a suitable *lbp* based on the analysis of task situation (e.g. the *lbp* displayed in the upside of Fig. 3), determine the next-level business activities which should be executed by invoking outside services (e.g. activity 1-4), and accept the instructions of ServiceFindingStrategy, including service applicability constraints, from the company (and the enterprise).

Because e-institution DataMining has formulated the social structure standards for Data Mining (DM) domain, including B-Service^{DM} (Business Service set), B-O-Role^{DM} (Business Operation Role set), DBP^{DM} (Distributed Business Process), and suppose there are a certain number of companies for providing each of those services and the agents as the brokers of those companies have registered for enacting relevant *bor*s ($\in$B-O-Role^{DM}) formulated in this e-institution, so *m* can requests the matchmaker agent (i.e. the agent enacting cooperation facilitation role "Service-matchmaker") to recommend the outside service providers.

Since the applicability constraints from *m* and service providers are both proposed based on the same syntax and semantics (see section Ⅲ.*B.b.(2)*), plus ACDP-based compatible match examination (see section Ⅲ.*B.a*), the matchmaker can accurately find the service providers satisfying applicability constraints.

Note, *m* is also the organizer of service cooperation and VO, which invokes the negotiation services to carry through the policy-driven negotiation for acquiring the outer services completing activity 1-4, and signs two-party service contracts with chosen service providers by invoking contract-signing operators belonging to those services. Thus, the TAVO completes its self-organization and *m* becomes the manager of this TAVO.

Suppose the providers of services completing activity 1 and 4 need to complete the provision of those services by creating TAVOs in the next level (as illustrated in Fig. 3), therefore the hierarchical cooperation in form of nested TAVOs can be formed. It is this form that enables even more complex service cooperation to be created conveniently and smartly and controlled efficiently.

Because of the creation of role-enacting contracts and service contracts, plus the policy-driven activities for conforming to service norms (see section Ⅲ.*B.b*) and the community regulating mechanism for forcing the conformity to cooperation contracts, *m* trusts the partners selected for providing outside services and believes their cooperation behaviors are both predictable and controllable.

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. Relative Work

The research of cooperation self-organization can be traces up to the study of distributed artificial intelligence in the eighties of last century. The growing-up of AaMAS technology in nineties impelled this research greatly, facilitating the deployment of AaMAS in VOs (virtual organizations), supply chain management and inter-enterprise interoperability provisioning [10], [11]. However, Due to suffering the two hindrances mentioned in section Ⅰ, there is no any commercially successful application to be reported in this area [12].

Recent research work of normative multi-agent systems, especially institution-governed cooperation, has facilitated the resolution of the "trust" crisis greatly, but is still confronted with two challenging problems: how to ensure that all of cooperation behaviors conform to the regulations formulated in e-institutions, and how to couple with real-life application software systems. The model of IGTASC we have established overcomes the two challenging problems, and hence creates the basis for developing the framework ICHO (see section Ⅱ).

The communities for researching semantic web [13], semantic grid[14], [15], autonomic computing [16] have also made some research work for achieving the self-organization of service cooperation and VOs, but the attention of most research projects has only focused on parts of self-organization process, no real-life sound frameworks for supporting whole process have been reported. We think that the root cause resulting in such a status is just the "trust" crisis brought on due to the inherent non-controllability of business services across different management domains.

Along with the hindrances and challenging problems were removed, based on IGTASC, ICHO has been constructed as the feasible framework for supporting the whole self-organization process. By developing the mechanisms of institution-govern, Contract-ensured, and Hierarchical self-Organization and the key techniques for achieving those mechanisms, ICHO can support effectively each phrase in the 4-phrase process of self-organization of service providing-requiring cooperation and VOs, find accurately applicable service providers by executing the ACDP-based method, achieving contract-ensured self-organization by creating role-enacting and service contracts, and reduce the complexity of large-scale VO creation by implementing hierarchical cooperation self-organization. In the same time, by executing policy-driven self-management, agents can make their own behaviors in VO self-organization always conform to cooperation behavior norms formulated in e-institutions and the business instructions sent over by agents' owners.

In contrast, it is the above two challenging problems that make the traditional research of institution-governed cooperation focused only on the abstract level disjoining with real-life application software systems, in terms of objective, sub-objective, scene, scene transition, landmark (sub-objective in a cooperation process), etc[5]-[8]. Thereby, there are only few successful real-life application cases to be report so far, let alone the cases for supporting service-based ones.

VI. Conclusions

Constructing Virtual Organizations (VOs) by creating service cooperation (i.e. service-oriented cooperation) has become the mainstream approach for reforming the development of application software systems in Internet computing environments. However, the inherent non-controllability of business services across different management domains has brought on the so-called "trust" crisis, which cumbers the self-organization of cooperation, and makes the organization of cooperation have to depend on a great deal of manual intervention.

This paper proposes a multi-agent technology-based framework ICHO, which, based on the model of IGTASC, can support the self-organization of service cooperation effectively by developing three mechanisms: Institution-governed, Contract-ensured, and Hierarchical self-Organization, and remove the "trust" crisis and be integrated closely with real-life application software systems.

The future work will be the formalization of the three mechanisms and the development of real-life application systems based on ICHO.

References

[1] M. P. Papazoglou, P. Traverso, S. Dustdar and F. Leymann. 2007. Service-oriented computing: state of the art and research challenges. IEEE Computer, 40(11): 64-71.

[2] M. Stal. 2006. Using architectural patterns and blueprints for service-oriented architecture. IEEE Softw, 23(2): 54-61.

[3] D. De Roure, N. R. Jennings and N. R. Shadbolt. 2005. The semantic grid: past, present and future, Proceedings of the IEEE, 93(3): 669-681.

[4] J. Gao, H. Lü, H. Guo, F. Zhang, Y. Cheng, C. Fu and C. Wang. 2009. Trusted autonomic service cooperation model and application development framework. Science in China Series F: Information Sciences, 52 (9): 1550-1577.

[5] G. Boella, L. van der Torre and H. Verhagen. 2008. Introduction to the special issue on normative multiagent systems. Auton Agent Multi-Agent Syst, 17:1-10.

[6] A. Garcia-Camino, J. A. Rodriguez-Aguilar, C. Sierra and W. Vasconcelos. 2006. A distributed architecture for norm-aware agent societies. Proceedings of the third International Workshop on Declarative Agent Languages & Technologies (DALT 2005), LNAI 3904, 89-105.

[7] H. Aldewereld, F. Dignum and J.-J. Ch. Meyer. 2007. Designing protocols for agent institutions. Proceedings of AAMAS'07, Honolulu, Hawaii, US, May, 138-140.

[8] V. Dignum, J. Vázquez-Salceda and F. Dignum. 2005. OMNI: Introducing social structure, norms and ontologies into agent organizations. Proceedings of the Second International Workshop on Programming Multi-Agent Systems (ProMAS 2004), Selected Revised and Invited Papers, LNAI 3346, 181-198.

[9] N. R. Jennings, T. J. Norman, P. Faratin, P. O'Brien and B. Odgers. 2000. Autonomous agents for business process management. Journal of Applied Artificial Intelligence, 14 (2): 145—189

[10] [10] P. Giorgini, J. P. Müller and J. Odell. 2003. Agent-oriented software engineering Ⅳ. In LNCS No. 2935, Heidelberg: Springer.

[11] [11] G. D. M. Serugendo, M. P. Gleizes, A. Karageorgos. 2005. Self-organization in multi-agent systems, The Knowledge Engineering Review, 20 (2), 165-189.

[12] M. Pěchouček, V. Mařík. 2008. Industrial deployment of multi-agent technologies: review and selected case studies. Auton Agent Multi-Agent Syst (2008) 17:397–431.

[13] S. Ferndriger, A. Bernstein, J. S. Dong, Y. Feng, Y Li and J. Hunter. 2009. Enhancing Semantic Web Services with Inheritance. Lecture Notes in Computer Science, 5318, 62-177.

[14] O. Corcho, P. Alper, I. Kotsiopoulos, P. Missier, S. Bechhofer, C. Goble. 2006. An overview of S-OGSA: A reference semantic grid architecture, Web Semantics: Science, Services and Agents on the World Wide Web 4(2): 102-115.

[15] M. Salvadores, P. Herrero, J. Bosque and M. S. Peréz. 2010. A semantic collaborative awareness model to deal with resource sharing in grids. Future Generation Computer Systems, 26 (2): 276-280.

[16] Xu J, Zhao M and Fortes J. 2009. Cooperative autonomic management in dynamic distributed systems. Lecture Notes in Computer Science 5873, Springer Berlin / Heidelberg, 756-770.

ISBN: 978-988-17012-9-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)