Agent-Based Virtual Organisations for the Grid

Jigar Patel[1], W. T. Luke Teacy[1], Nicholas R. Jennings[1], Michael Luck[1], Stuart Chalmers[2], Nir Oren[2], Timothy J. Norman[2], Alun Preece[2], Peter M. D. Gray[2], Gareth Shercliff[3], Patrick J. Stockreisser[3], Jianhua Shao[3], W. Alex Gray[3], Nick J. Fiddian[3] and Simon Thompson[4]

[1] School of Electronics and Computer Science, University of Southampton, United Kingdom
[2] Department of Computing Science, University of Aberdeen, United Kingdom
[3] School of Computer Science, Cardiff University, UK
[4] BT R&D, Adastral Park, United Kingdom

Abstract. The ability to create reliable and scalable virtual organisations (VOs) on demand in a dynamic, open and competitive environment is one of the challenges that underlie Grid computing. In response, in the CONOISE-G project, we are developing an infrastructure to support robust and resilient virtual organisation formation and operation. Specifically, CONOISE-G provides mechanisms to assure effective operation of agent-based VOs in the face of disruptive and potentially malicious entities in dynamic, open and competitive environments. In this paper, we describe the CONOISE-G system, outline its use in the context of VO formation and perturbation, and review current efforts to progress the work to deal with unreliable information sources. [5]

1 Introduction

The engineering of systems using approaches that establish a fixed organisational structure is not sufficient to handle many of the issues inherent in open multi-agent systems (in particular, heterogeneity of agents, trust and accountability, failure handling and recovery, and societal change [1, 2]). Such issues are becoming increasingly important in the context of Grid computing, which aims to enable resource sharing and coordinated problem-solving in dynamic, multi-institutional virtual organisations (VOs) [1].

VOs are composed of a number of autonomous entities (representing different individuals, departments and organisations), each of which has a range of problem-solving capabilities and resources at its disposal. While such entities are typically self-interested, there are sometimes potential benefits to be obtained from pooling resources: either with a competitor (to form a coalition) or with an entity with complementary expertise (to offer a new type of service). The recognition of this potential can be the cue for the formation of a VO in which distinct, autonomous entities come together to exploit a perceived niche. When this is successful, the collection of independent entities acts as

[5] Primary contact: Michael Luck, School of Electronics and Computer Science, University of Southampton, SO17 1BJ, United Kingdom
Email: mml@ecs.soton.ac.uk
Telephone: +44 23 8059 6657
Fax: +44 23 8059 3313

a single conceptual unit in the context of the proposed service, requiring that the participants cooperate and coordinate their activities in delivering the services of this newly formed organisation. Part of this demands that the participants have the ability to manage the VO effectively. In dynamic environments, however, the context may change at any time, so that the VO may no longer be viable. It must then either disband or re-arrange itself into a new organisation that better fits the circumstances. This paper describes technologies developed to address both these phases.

VOs thus provide a way of abstracting the complexity of open systems to make them amenable to application development. The organisational structure, participant responsibilities, synchronisation concerns and economic mechanics of the VO are hidden from the VO user. This has two benefits: first, agents can be used to bridge between requester and providers to organise the VO and to provide a layer of flexibility between requesting applications and the underlying service infrastructure; second, the VO fulfils the role of information hiding in that the internal mechanics are abstracted away from the requesting application, and the VO formation and management system either supports a request or fails at well-defined points.

While the notion of VOs underpins the vision of Grid computing, the conditions under which a new VO should be formed, and the procedures for its formation, operation and dissolution, are still not well-defined. This automated formation and ongoing management of VOs in open environments thus constitutes a major research challenge, a key objective of which is to ensure that they are both agile (can adapt to changing circumstances) and resilient (can achieve their aims in a dynamic and uncertain environment). In addition to traditional constraints that relate to issues such as resource management and bidding strategies, we must also consider softer constraints relating to contract management, trust between VO participants and policing of contracts.

The CONOISE-G project (Grid-enabled Constraint-Oriented Negotiation in an Open Information Services Environment, *http://www.conoise.org*) is directed at addressing just these issues. It seeks to support robust and resilient VO formation and operation, and aims to provide mechanisms to assure effective operation of agent-based VOs in the face of disruptive and potentially malicious entities in dynamic, open and competitive environments. In this paper, we describe the CONOISE-G system, in which VO formation is grounded on three key technologies [3]: the agent decision-making, auctions for allocation of contracts, and service discovery incorporating quality of service (QoS) assessment.

In addition, however, to operate an effective VO in open, dynamic and competitive environments, it is essential that we also consider how to encourage good interactions, and cope effectively with bad ones. In our view, this requires that QoS levels are monitored, that uncertainty in participant behaviour, possibly arising from participant self-interest and strategic lying and collusion, is minimised, and that mechanisms for recognising and addressing contract violations once they have occured are established. Addressing these concerns is integral to the wide-scale acceptance of the Grid and agent-based VOs.

The contribution of this paper lies in the construction of an implemented prototype for dynamic re-formation of VOs through the integration of several different techniques. The paper begins with a motivating example that introduces the need for VO formation

Table 1. Potential Service Providers

SP	Ent	News	Text	Games	Tkts
SP1	30	20			5
SP2		10	50		
SP3			100	30	5
SP4	30	10		60	
SP5			50	45	10

and operation. It then describes the system architecture, elaborating the different aspects identified above in support of robust and resilient operation. The paper ends with a description of the implemented prototype that underlies the core of the current work in the CONOISE-G project to achieve effective VO formation and operation within a Grid environment.

2 A Motivating Scenario

Lucy visits London in 2012 for the Olympic Games, using her PDA to access various multimedia services (news, clips from the Games, tickets for events, text messaging, and *ad-hoc* entertainment opportunities, such as streaming video). Many service providers offer such services, so Lucy must determine potential providers, select an optimal package, and then track the changing market for better deals.

In such situations, creating a VO on demand can greatly simplify the problem, allowing users merely to specify their service requirements, with VOs providing the required services. However, forming and operating a VO is complex. By way of example, suppose there are five service providers (`SP1`, ..., `SP5`), as in Table 1, each offering relevant multimedia services. These services form three groups: *video content* (Entertainment and Game Clips services), *HTML content* (News and Ticketing services) and *text messaging* (Text service); and they can be requested individually or taken as a package, with the constraint that the two services offered by `SP2` must be taken together.

We assume that these providers may demand different prices for the same service, depending on the number of units requested. For example, `SP1` may offer 20 news updates per day at £30 per month, and 10 updates at £25 per month. Also, the quality of services may not be stable: `SP4` may offer Games clips with a frame rate of no less than 24 frames per second, but actually provide a rate that drops below that level. Finally, not all service providers are trustworthy, and what they claim may not be what a requester will get: `SP5` may advertise sought-after tickets that it does not possess, and orders for tickets through `SP5` may not always be honoured.

Now, suppose that Lucy wishes to purchase the service package of Table 2. It should be clear from Table 1 that many different solutions are possible. For example, for 50 minutes of entertainment, both `SP1` and `SP4` must be used, but different compositions of the two services are possible, with different price, quality and degree of trust. To find a good solution for a given service request, therefore, several issues must be addressed.

Service Required	Units Required
Entertainment	50 mins per month
News	10 updates per day
Text messages	100 per month
Game Clips	60 mins per day
Ticketing	10 alerts per day

Table 2. Example service package request

During VO formation, multiple service providers may offer broadly similar services, each described by multiple attributes including, for example, price, quality, reputation and delivery time. We therefore need to determine how the relevant services for a given service request may be discovered, and how an optimal package may be selected, based on the above attributes. During VO operation, however, the services available may change over time: new services may become available, or providers may alter the way in which existing services are offered. Quality of service and provider reputation may also change over time. There is thus a need to monitor the performance of the members of a VO in terms of their trustworthiness, quality of service and conformance to contract, and to restructure the VO when necessary so that the integrity and usefulness of the VO are maintained. Thus, a poorly performing service may be replaced, a contract-breaking service may be dropped, and a new user requirement may be accommodated.

Creating and then effectively managing a VO in a dynamic environment thus poses significant research challenges. In seeking to address them, we have developed a system for dynamic formation and operation of VOs. In the following sections, we outline the system architecture and describe its key components.

3 The CONOISE-G Architecture

In essence, the CONOISE-G architecture comprises several different agents, including *system agents* and *service providers* (SPs), as shown in Figure 1. The former are those needed to achieve core system functionality for VO formation and operation, while the latter are those involved in the VO itself. For simplicity, we omit some specific components that perform basic functions, such as a Yellow Pages (YP) agent, since they add little to the elaboration of the issues to be discussed here.

Assuming that service providers have already advertised their services to a YP, the VO formation process starts with a particular SP acting on behalf of a user, the Requester Agent (RA), which analyses the requester's service requirements, locates the relevant providers through the YP, and then invites the identified providers to bid for the requested services. The quality and trustworthiness of the received bids are assessed by the Quality Agent (QA) and the trust component, respectively, and the outcome is combined with the price structure by a Clearing Agent (CA) [3] to determine which combination of the services/providers will form an optimal VO (in terms of price, quality and trust) for the requester. At this point, the VO is formed and the RA takes on

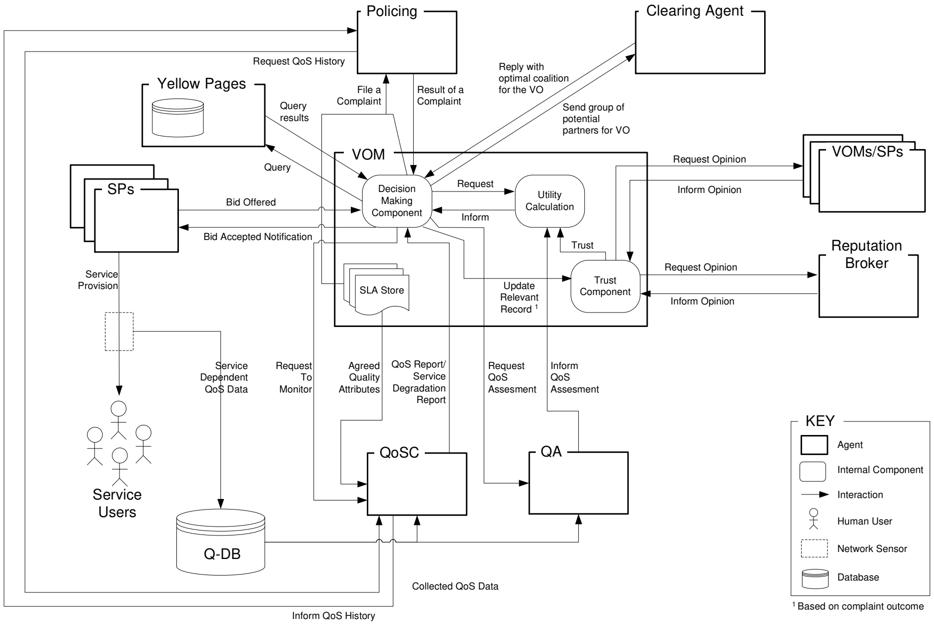

Fig. 1. The CONOISE-G system architecture

the role of VO Manager (VOM), responsible for ensuring that each member of the VO provides its service according to contract.

During the operational phase of the VO, the VOM may request the QoS Consultant (QoSC) to monitor any services provided by any members of the VO, and any member of the VO may invoke the Policing agent to investigate any potential dispute regarding service provision. Ultimately, our aim is for monitoring to take place to inform the user when the actual service level diverges from the agreed service level. At present, however, this is achieved by configuring the levels of QoS for each service that will cause the QoSC to alert the VOM, using predetermined service provision and quality level simulations. When the QoS provision of a service (say the *news* service in the scenario) in the VO falls below an acceptable level of service, or some breach of contract is observed, the QoSC alerts the VOM, which initiates a VO re-formation process; relevant information is fed into the trust component to ensure that the provider concerned is penalised to an appropriate level by updating its record of trust.

In this re-formation process, the VOM issues another message to the YP requesting a list of SPs that can provide the *news* service. As before, the YP identifies possible SPs, bids are received and evaluated, resulting in the CA determining the best SP to replace the failed provider. At this point, the VOM re-forms the VO with the new SP replacing the old one, and instructs the QoSC to stop monitoring the old SP and to monitor the new one instead. In the following sections, we discuss the core technical components of the architecture in more detail.

Service Discovery In open, dynamic environments, autonomous service providers may join and leave at any time. It is necessary, therefore, that participant behaviour is informed by such aspects as advertisements and ontologies, and their capabilities are discovered dynamically at the time when a service request is made. Various service description languages and matchmaking mechanisms have been proposed, e.g. [4, 5], but little support has been given to incorporating QoS assessment in service discovery.

In our model, we extend current approaches to service discovery by creating service and quality ontologies, allowing service providers and consumers to advertise their capabilities and request services using the two ontologies, and dynamically matchmaking between advertised and requested services based on functional as well as QoS requirements [6]. More specifically, we extend DAML-S [5] to include QoS specifications, and our matchmaking follows a two-stage process. First, a service request containing functional and/or QoS requirements is sent to the YP agent to search for providers who claim to offer the required service. Then, the QA is asked to assess how well each identified provider can actually provide the service at the quality level they claim, using a novel expectation-based quality calculation model [7]. The outcome of this assessment is used to weed out the providers whose QoS levels are too low and to provide another basis for negotiation.

Decision Making in VO Formation In a VO, a resource (or company providing the resource) is represented by an SP. When asked to contribute a bid for the provision of a service, this agent must check its current resource use (by prior commitments as a result of already successful bids), and decide if it can make an offer to provide the new service. It may also examine the collective resources available if it is in an existing VO, and make a new bid on the VO's behalf. Alternatively, it may decide that the provision of this new resource is more beneficial than its prior commitments and decide to break some or all of these to allow the successful creation of a bid for the new service provision. This reflects the self-interested nature of the component agents. It can also form a new VO to cater for the provision of the resources. These options are shown in Figure 2.

As should be clear, this decision-making process can be quite complex. In consequence, we use a cumulative scheduling algorithm based on Constraint Satisfaction Programming (CSP) techniques [8] to aid the agent in this process. The scheduling algorithm models the agent's available resources using two metrics: the duration for which the agent can provide the resource and the amount of resource available over that time. It then models the existing resource provision by constraining these metrics. The remaining free resources represent the units that can be used by the agent to construct its bid. To model the agent's ability to discard existing resources in favour of new ones, we use constraint reification [9]. This attaches a binary value to each existing commitments' time and resource constraints indicating whether that commitment has been satisfied. The CSP then attempts to satisfy the new resource provision by finding the maximal subset of reified values (and therefore the maximum number of existing commitments) that can be satisfied alongside the new commitment. The constraints with reified values not in this satisfied subset show the commitments that need to be broken in order for the adoption of the new commitment to be successful.

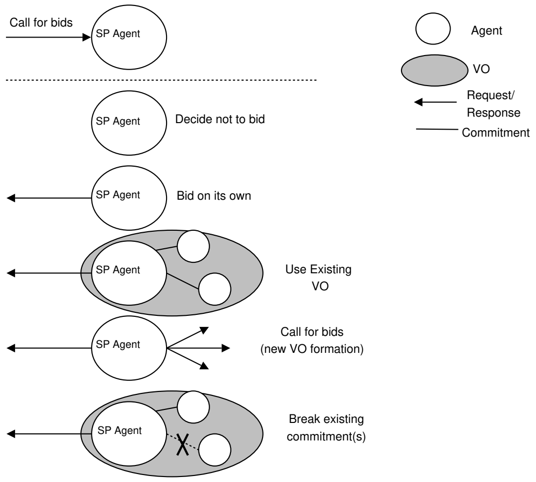

Fig. 2. Deciding whether and what to bid

When an SP decides that it is beneficial to bid for the provision of a certain task, it submits the bid to the RA that initiated the call for bids. Since multiple bids for the same request are possible, the bids received from the SPs must be *cleared*. That is, we must decide which ones to accept (or which partners to select) in the formation of the VO. Given the open nature of the environment and the lack of a pre-ordained structure, we believe this selection process is best achieved using some form of marketplace (auction). Two sets of clearing algorithms have been developed: one with polynomial complexity that has been shown to produce a solution within a finite bound of the optimal, and another that is not polynomial but is guaranteed to produce the optimal allocation [10].

Establishing Trust and Reputation Whenever interactions take place between different agents, the issues of trust and reputation become important. In particular, during the formation of a VO, we often have a choice of service providers to whom we may delegate tasks. In such cases, trust serves as an indicator of which of these possible partners are likely to carry out the task as specified, and its usefulness extends into the other stages of the VO lifecycle.

In CONOISE-G, we take trust to be a particular level of the subjective probability with which an agent assesses that another agent will perform a particular action, both before she can monitor such an action and in a context in which it affects her own action (adapted from [11]). This probabilistic view of trust allows us to determine the *subjective probability* by considering the outcomes of previous encounters (known as direct interaction-based trust). However, in an open community it is likely that an agent will interact with many unknown entities with which it may not share an interaction history. In the absence of this shared history, the CONOISE-G trust system uses *reputation* information to establish the level of trust to place in another. Reputation can be defined as *a commonly held set of opinions about an entity* [12], and it is the aggregation of these common opinions that forms a level of trust.

The trust and reputation system [13, 14] consists of two distinct parts. The first part is a trust component, which is internal to all agents that require a trust metric in their decision-making process. Its function is to provide its owner agent with a level of trust for a given service and service provider. The component is insulated from the external environment by the agent that embodies it. As the agent interacts with others in the community, the outcomes of these interactions are stored in this component, and are used to determine a trust value when required. Outcomes can either be successful or unsuccessful, where a successful interaction is defined as one for which the service provider has delivered the service specified by the contract. In addition to calculating trust, the trust component calculates a level of *confidence* to be placed in that trust value. Confidence represents the accuracy of the trust value, and is obtained by examining how much evidence was used to calculate it. It is used by the trust component to reason about whether an agent has adequate evidence or whether it needs to obtain further (reputation) information from other agents. When the confidence in its own calculation does not exceed a particular threshold, the trust component requests such reputation information from others. In our model, we do not assume that reputation is necessarily accurate, and allow for the possibility that an agent may intentionally mislead. In such cases, the trust component assesses the likelihood that a reputation provider supplies accurate information, based on accuracy of information supplied in the past.

The second part of the trust system is a *reputation brokering* agent, and several of these agents in the system may serve as a distributed store of reputation information. A reputation broker provides an aggregated store of trust information relating to specific service provider agents and each of their services. However, before any agent can query the broker, the broker must obtain the trust information that will form the query result. We achieve this using a *subscribe and publish* mechanism, by which the broker subscribes to agents in the community which then publish their internal information (the store of outcomes based on their individual direct experiences) to the broker. From a business perspective, we envisage that brokers will be arranged according to organisational hierarchies. For instance, each department within a company may have a broker which subscribes to the opinions of all agents belonging to that department. A company level broker may then subscribe to the opinions of departmental brokers, thus aggregating the opinions of all agents within that company. Agents in the community can obtain reputation information from these brokers by sending query messages, to which the brokers can reply with the relevant information or a failure message in the case where they do not have such information. When an agent does receive reputation information from a broker, it assesses the accuracy of this information, just as it would if the information was sourced from an individual reputation provider.

Policing within a VO While trust and reputation ratings are able to reduce the likelihood of poorly performing (or malicious) agents becoming part of a VO, they do not offer any mechanism for minimising the impact of undesirable behaviour, such as an agent contracting to provide services it does not deliver. The goal of the policing system is to determine whether a party is in breach of a contract, determine if any corrective action (as stipulated in the contract) should be taken, and inform the trust mechanism to allow sanctions to be imposed. Given the scalability concerns inherent in large, open distributed systems, the CONOISE-G system responds to reported exceptional circumstances, rather than monitoring operation.

The policing system initiates an investigation following the receipt of a complaint from a VO participant. The process begins by obtaining the relevant contract at the centre of the dispute, and gathering evidence to determine the actual state of affairs. This can take on a number of forms, including reports from agents in the system and other artifacts; it is recursive, in that one piece of evidence may have further evidence supporting or rebutting it. Furthermore, agents can submit evidence in support of or against a conclusion. The evidence gathered, therefore, constitutes a set of defeasible arguments in support of and in defence of the complaint. Our approach borrows ideas from computational models of legal reasoning and legal argumentation [15].

We thus view the policing system as consisting of a number of distinct components, contained in both the environment infrastructure and the individual agents: a component able to describe ideal system behaviour (requiring a contracting language and a set of contract instances); an interface to allow agents to provide arguments and evidence to the system, as well as a method to allow the system to request further information; a reasoning mechanism to determine the evidence to be gathered; and a technique for weighing up evidence, without which policing agents cannot combine arguments to reach a verdict.

In CONOISE-G the representation of contracts is based on the emerging Web Services standard for agreements, WS-Agreement [16]. We extend this language to represent concepts such as prohibited activities, transferable responsibilities and group actions that do not appear in the existing standard. We are also investigating methods for grounding the semantics of such contracts to bring these pragmatic approaches closer to formal contract specification languages such as those developed by Dignum *et al.* [17] and Pacheco and Carmo [18]. The evidence gathering mechanism employed is tightly coupled with the reasoning machinery; both activities are driven by sets of defeasible arguments. Agents involved in the contract may submit evidence to the policing agent, which can ask questions, obtain logs, etc., according to the rules of a dialogue game developed for the purpose of evidence gathering. Strategies for determining what evidence should be submitted or sought, as well as reasoning about how arguments and evidence interact and combine are being used to facilitate reasoning about contract failure, for which little related work exists, with some exceptions [19].

Monitoring QoS Levels During the operation of a VO, it is important that the QoS provision is monitored. The QoS data collected from this monitoring process is vital in supporting the creation of a resilient VO. First, it serves as "evidence" in a range of critical assessment. For example, the QoS data is used: by the Trust component to establish the level of trust that can be placed in a service and service provider; by the Policing agent to deal with complaints; and by the QA to assess QoS for services during future VO formations. Second, the QoS data helps monitor and predict any QoS degradation within a VO. Any detection or prediction of such degradation can result in a possible replacement of a VO member, or trigger re-formation of the VO, ensuring that the VO maintains an agreed level of QoS provision, limiting any damage to its reputation.

In the CONOISE-G system, the monitoring of QoS provision is carried out by the QoSC, which is designed to perform three main tasks. The first entails the recording and gathering of QoS data, a continuous activity that contributes to a QoS database. Here, data collection is performed through the use of network sensors and, for simplicity, we also assume that the QoS at any point on the link from the provider to the consumer is the same. The second task involves the monitoring of the QoS level. The current level of service provision is calculated from the data that is collected from the network sensors and compared to the QoS level stated in the service level agreement. Any service whose QoS has dropped below the level required is then reported to the VOM concerned. Since QoS data can be generated continuously at a very fast speed, and needs to be processed with respect to dynamic, ad-hoc monitoring requests from individual VOMs, we adopt a data stream [20] approach to QoS monitoring in constructing the QoSC agent. The third task to be performed by the QoSC is that of alerting the VOM to any anticipated drop in QoS. Taken together, these tasks provide a versatile, accurate and robust QoS monitoring mechanism within CONOISE-G.

4 The CONOISE-G Implementation

The CONOISE-G environment is FIPA[6] compliant and the implementation uses the JADE[7] agent platform. Agents communicate by exchanging FIPA ACL (agent communication language) messages, the content of which is defined using lightweight ontologies expressed in Semantic Web (SW) representations, following experience from previous work [21]. We chose these representations in preference to the more conventional use of FIPA-SL in the content of FIPA messages for a number of reasons. First, the SW representations are more widely used than FIPA-SL, so CONOISE-G is lent greater interoperability by aligning with W3C recommendations. Second, we can reuse existing schemas and ontologies; for example, we borrowed heavily from the DAML-S service ontology. Thus, we would be in a position to exploit any existing schemas or ontologies in a particular application domain. Third, particularly at the lower (RDF) layers of the SW formalism stack, the semantics of the data model are much simpler than FIPA-SL (while still adequate for operational use), so there is less of a learning curve for designers and implementors of CONOISE-G agents (and much well-tested software for processing RDF, unlike FIPA-SL).

In the current system, we have created a set of interrelated ontologies expressed in a relatively lightweight manner as RDF schemas. For now, RDFS is sufficiently expressive to capture usable structures, and has allowed us to rapidly develop the necessary message formats for inter-agent communication in our scenario. We envisage the definitions in the ontologies being refined with the addition of OWL (Web Ontology Language) statements once the formats have stabilised through further testing and refinement. Two sample RDF messages expressed using a number of the ontologies are shown in Figures 3 and 4. The first shows a sample call for bids, as issued to

[6] `http://www.fipa.org`
[7] `http://sharon.cselt.it/projects/jade`

```
<rdf:RDF
  xmlns:rdf=``http://www.w3.org/1999/02/22-rdf-syntax-ns#''
  xmlns:quality=``http://conoise.org/ontologies/quality#''
  xmlns:media=``http://conoise.org/ontologies/media#''
  xmlns:package=``http://conoise.org/ontologies/package#''>
  <package:Requirement rdf:about=``http://conoise.org/samples/request''>
    <quality:qualityPreference rdf:resource=
            ``http://conoise.org/ontologies/quality#minCost''/>
    <package:consistsOf
      rdf:type=``http://conoise.org/ontologies/media#PhoneCalls''
      media:numberOfMinutes=``25''/>
    <package:consistsOf>
      <media:MovieContent media:numberOfMovies=``72''>
        <media:subscriptionType rdf:resource=
            ``http://conoise.org/ontologies/media#monthly''/>
        <media:mediaStyle rdf:resource=
            ``http://conoise.org/ontologies/media#scienceFiction''/>
      </media:MovieContent>
    </package:consistsOf>
    <package:consistsOf>
      <media:HtmlContent media:updateFrequency=``24''>
        <media:mediaStyle rdf:resource=
          ``http://conoise.org/ontologies/media#news''/>
      </media:HtmlContent>
    </package:consistsOf>
    <package:consistsOf
      rdf:type=``http://conoise.org/ontologies/media#TextMessaging''
      media:numberOfMessages=``100''/>
  </package:Requirement>
</rdf:RDF>
```


Fig. 3. RDF call for bids sent to SPs

SPs. This consists of an instance of a user **Requirement** structure, stating a number of services that the user's requirement *consistsOf*, and also a *qualityPreference* property, indicating that the most important thing for this user is lowest cost. The descriptions of each required service are adorned with service-specific properties; for example, the **MovieContent** requirement specifies a number of movies (per month), a subscription preference, and a genre type. This illustrates the use of terms from three CONOISE-G ontologies:

– the `package` ontology describes service packages, defining terms such as the class **Requirement** and the property *consistsOf*;
– the `quality` ontology describes domain-independent quality-of-service terms such as the *qualityPreference* property, and its various settings such as "minCost";
– the `media` ontology defines all application domain-specific terms for the Olympics scenario, including the service classes **MovieContent**, **HtmlContent**, **PhoneCalls**, and **TextMessaging**, all of which the ontology defines to be (indirect) sub-classes of the generic CONOISE **ServiceProfile** class (closely based on DAML-S).

The second sample message, in Figure 4, shows a bid issued by one of the SPs in response to the call shown in Figure 3. The bid is for just one of the required services (the **HtmlContent** part); the **Bid** structure is similar to the **Requirement** structure in that it also employs the *consistsOf* property, but here there is also an identified instance of a

```
<rdf:RDF
  xmlns:rdf=``http://www.w3.org/1999/02/22-rdf-syntax-ns#''
  xmlns:media=``http://conoise.org/ontologies/media#''
  xmlns:profile=``http://conoise.org/ontologies/profile#''
  xmlns:package=``http://conoise.org/ontologies/package#''>
  <package:Bid rdf:about=``http://conoise.org/samples/pa2bid#bid2''>
    <package:providedBy>
      <package:Provider
        profile:fipaAddress=``pa2@conoise.org:15551/JADE''>
        <profile:name>Provider Agent 2</profile:name>
      </package:Provider>
    </package:providedBy>
    <package:consistsOf
      <media:HtmlContent rdf:about=
        ``http://conoise.org/samples/pa2bid#pa2news''
        media:updateFrequency=``72''>
        <media:mediaStyle rdf:resource=
          ``http://conoise.org/ontologies/media#news''/>
        <package:hasPriceStructure
          rdf:type=``http://conoise.org/ontologies/package#Price''
          package:min=``0'' package:max=``10'' package:unitPrice=``3''/>
        <package:hasPriceStructure
          rdf:type=``http://conoise.org/ontologies/package#Price''
          package:min=``10'' package:max=``50'' package:unitPrice=``2''/>
        <package:hasPriceStructure
          rdf:type=``http://conoise.org/ontologies/package#Price''
          package:min=``50'' package:max=``1000'' package:unitPrice=``1''/>
      </media:HtmlContent>
    </package:consistsOf>
  </package:Bid>
</rdf:RDF>
```

Fig. 4. RDF bid issued by SP

Provider, whose properties are defined using terms from the `profile` ontology (that also defines the **ServiceProfile** class mentioned above). This information allows the user to access the service if the bid is ultimately accepted as part of the winning package. Note also that the services offered in bids have **Price** structures attached, which are rich enough to identify different price *bands* depending on the volume the user might wish to consume.

These examples illustrate how the capability to create modular, interlocking ontologies using the SW formalisms allow us to build up quite elaborate information representations, all of which are easily serialisable in a portable, open XML syntax, and easily parsed and processed using tools such as Jena2[8].

In terms of the user interface, the GUI (in Figure 5) shows the agents registered in the VO as a column of dots for each agent, each dot representing a distinct service provided by that agent. When the VO is re-formed, the new SP is incorporated into the *traffic light* display. The monitored (simulated) quality of service providers is also represented. Currently, the CONOISE-G GUI shows the QoS being provided by each software agent in a VO as a dynamically expanding line graph. The QoS of a VO is a function of its members' QoS in two ways: first, if the agent responsible for a particular service is changed, then the QoS provided by the VO is a function of the new agent's

[8] `http://www.hpl.hp.com/semweb/jena2.htm`

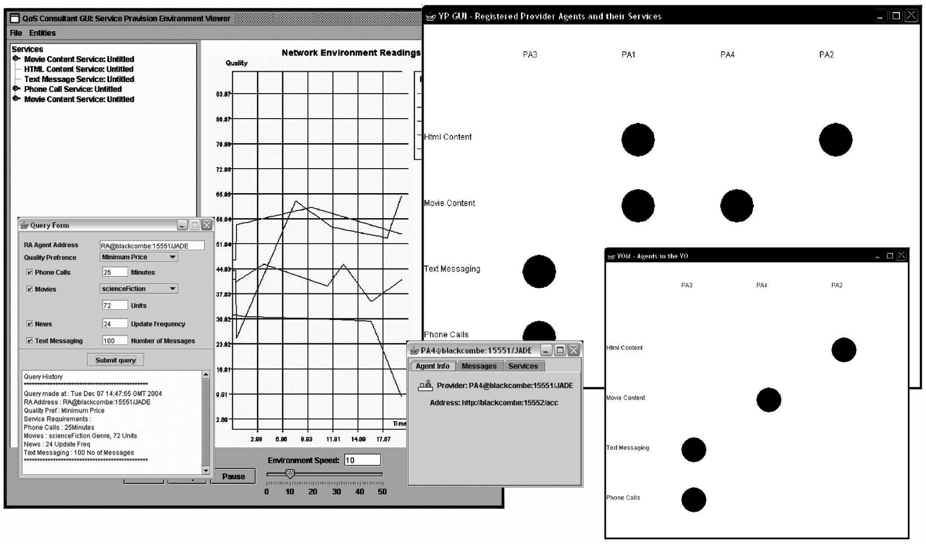

Fig. 5. The CONOISE-G user interface

performance, rather than the old agent's; second, the QoS of a VO may be a function not only of the performance of the agent responsible for that resource, but also of other agents in the VO that provide prerequisite resources.

5 Conclusions

The work described in this paper takes an approach in which issues relating to the formation and operation of robust VOs in the dynamic environments with unreliable agents are considered. In contrast to the "brawn" of the Grid, we have concentrated on the "brains" [1] — on the development of techniques for autonomous problem-solving in VOs. Thus, we have described an agent architecture for re-forming VOs in the face of unreliable information, through the use of a range of techniques that support robust and resilient VO formation and operation for application to realistic electronic commerce scenarios. We described our implemented prototype of the system, and elaborated the work being done on extending the system to incorporate more sophisticated application scenarios.

Acknowledgements

CONOISE-G is funded by the DTI and EPSRC through the Welsh e-Science Centre, in collaboration with the Office of the Chief Technologist of BT. The research in this paper is also funded in part by the EPSRC Mohican Project (Reference no: GR/R32697/01).

References

1. Foster, I., Jennings, N.R., Kesselman, C.: Brain meets brawn: Why Grid and agents need each other. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems. (2004) 8–15
2. Luck, M., McBurney, P., Preist, C.: A manifesto for agent technology: Towards next generation computing. Autonomous Agents and Multi-Agent Systems **9** (2004)
3. Norman, T.J., Preece, A., Chalmers, S., Jennings, N.R., Luck, M., Dang, V.D., Nguyen, T.D., Deora, V., Shao, J., Gray, W.A., Fiddian, N.J.: Agent-based formation of virtual organisations. Knowledge-Based Systems **17** (2004) 103–111
4. : Web services description language. http://www.w3.org/TR/wsdl (2003)
5. Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McIllraith, S., Narayanan, S., Paolucci, M., Payne, T., Sycara, K., Zeng, H.: DAML-S: Web service description for the semantic web. In: Proceedings of International Semantic Web Conference. (2002) 348–363
6. Deora, V., Shao, J., Shercliff, G., Stockreisser, P., Gray, W., Fiddian, N.: Incorporating QoS specifications in service discovery. In: Proceedings of Second International Web Services Quality Workshop (WQW 2004). (2004)
7. Deora, V., Shao, J., Gray, W.A., Fiddian, N.J.: A quality of service management framework based on user expectations. In: Proceedings of the First International Conference on Service Oriented Computing. (2003) 104–114
8. Caseau, Y., Laburthe, F.: Cumulative scheduling with task intervals. In: Logic Programing Proceedings of the 1996 Joint International Conference and Syposium on Logic Programming. (1996) 363–377
9. Chalmers, S., Preece, A., Norman, T.J., Gray, P.M.D.: Commitment management through constraint reification. In: Proc. 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems. (2004) 430–437
10. Dang, V.D., Jennings, N.R.: Polynomial algorithms for clearing multi-unit single item and multi-unit combinatorial reverse auctions. In: Proceedings of the Fifteenth European Conference on Artificial Intelligence. (2002) 23–27
11. Gambetta, D.: Can we trust trust? In: Trust: Making and Breaking Cooperative Relations. Basil Blackwell (1988) 213–237
12. Sabater, J., Sierra, C.: Regret: A reputation model for gregarious societies. In: Fourth Workshop on Deception Fraud and Trust in Agent Societies. (2001) 61–70
13. Teacy, W.T.L., Patel, J., Jennings, N.R., Luck, M.: Coping with inaccurate reputation sources: Experimental analysis of a probabilistic trust model. In: Proceedings of 4th International Joint Conference on Autonomous Agents and Multiagent Systems, Utrecht, the Netherlands (2005)
14. Patel, J., Teacy, W.T.L., Jennings, N.R., Luck, M.: A probabilistic trust model for handling inaccurate reputation sources. In Hermann, P., Issarny, V., Shiu, S., eds.: Proceedings of the Third International Conference on Trust Management (iTrust). Volume 3477 of LNCS., Rocquencourt, France, Springer-Verlag (2005) 193–209
15. Bench-Capon, T., Freeman, J.B., Hohmann, H., Prakken, H.: Computational models, argumentation theories and legal practice. In Reed, C.A., Norman, T.J., eds.: Argumentation Machines: New Frontiers in Argument and Computation. Kluwer (2003) 85–120
16. Czajkowski, K., Dan, A., Rofrano, J., Tuecke, S., Xu, M.: WS-Agreement: Agreement–based grid service management. In: Global Grid Forum. (2003)
17. Dignum, V., Meyer, J.J., Dignum, F., Weigand, H.: Formal specification of interaction in agent societies. In: Proceedings of the second Goddard workshop on formal approaches to agent based systems. (2002)

18. Pachheco, O., Carmo, J.: A role based model for the normative specification of organized collective agency and agents interaction. Autonomous Agents and Multi-Agent Systems **6** (2003) 145–184
19. Daskalopulu, A., Dimitrakos, T., Maibaum, T.: Evidence-based electronic contract performance monitoring. Group Decision and Negotiation **11** (2002) 469–485
20. Babcock, B., Babu, S., Datar, M., Motwani, R., Widom, J.: Models and issues in data stream systems. In: Proc. 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM Press (2002) 1–16
21. Grimnes, G., Chalmers, S., Edwards, P., Preece, A.: Granitenights – a multi-agent visit scheduler utilising semantic web technology. In: 7th International Workshop on Cooperative Information Agents. (2003) 137–151