Agent-Based Virtual Organisations for the Grid

Jigar Patel, W T Luke Teacy, Nicholas R Jennings, Michael Luck
School of Electronics and Computer Science, University of Southampton

Stuart Chalmers, Nir Oren, Timothy J Norman, Alun Preece, Peter M D Gray
Department of Computing Science, University of Aberdeen

Gareth Shercliff, Patrick J Stockreisser, Jianhua Shao, W Alex Gray, Nick J Fiddian
School of Computer Science, Cardiff University

Simon Thompson
BT, Adastral Park

Abstract

The ability to create reliable and scalable virtual organisations (VOs) on demand in a dynamic, open and competitive environment is one of the major 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. [1]

Keywords Virtual Organisations, Policing, Trust, Quality of Service Monitoring, Agents, Grid

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 large-scale open environments (in particular, the heterogeneity of the different actors, trust and accountability, failure handling and recovery, and societal change [11, 16]). Given this, such issues are increasingly coming to the fore in the context of Grid computing, which aims to enable resource sharing and coordinated problem-solving in dynamic, multi-institutional virtual

[1] 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

organisations (VOs) [11]; in this paper we report on our attempts to develop an agent-based system that can operate effectively in such contexts.

In more detail, 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 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. In turn, this requires 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 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.

Now, while the notion of VOs underpins the vision of Grid computing [12], the conditions under which a new VO should be formed, and the procedures for its formation, operation and dissolution, are still not well-defined. Thus, for example, in current Grid applications (like MyGrid [2], Astrogrid[3] and Combechem[4]) the research focus has primarily been on the integration of heterogeneous equipment and resource sharing. In these systems, therefore, VOs are statically defined by the users of the workflows, which means that they are incapable of handling dynamic situations and reconfiguring themselves in an automated manner. 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 constraints that relate to issues such as resource management and bidding strategies, we must also consider

[2] http://www.mygrid.org.uk
[3] http://www.astrogrid.org
[4] http://www.combechem.org

softer constraints relating to contract management, trust between VO participants and policing of contracts.

Against this background, 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. The CONOISE-G vision is a set of system agents, operating at the application layer of the Grid architecture, working together to support robust and resilient VO formation and operation. It 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, where entities compete for limited resources. (By contrast, in non-competitive environments in which entities collaborate, there is a much safer environment, so that the problems shift to those concerned with facilitating this collaboration.)

In addition, however, to operate an effective VO in open, dynamic and competitive environments, it is essential also to consider how to encourage good interactions, and cope effectively with bad ones. This requires that QoS levels are monitored (see Section 3), that uncertainty in participant behaviour, possibly arising from participant self-interest, strategic lying[5] and collusion[6], 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.

To that end, in this paper, we describe the CONOISE-G system, in which VO formation is grounded on three key technologies [18]: (i) the agent decision-making, (ii) auctions for the allocation of contracts, and (iii) service discovery incorporating quality of service (QoS) assessment.

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 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 project.

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 available providers, select an optimal package, and then track the changing market for better deals.

[5]Providing false information, based on knowledge of another's beliefs and modeling capabilities [6], which has a negative impact on the victim.

[6]Defined as competing agents working together for their mutual benefit. This behaviour typically has a negative impact on other non-colluding agents in the marketplace.

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). They can be requested individually or taken as a package, with the constraint that the two services offered by SP2 must be taken together.

Table 1: Potential Service Providers

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

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.

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

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.

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. After VO formation, when the services are being provided, the VO enters the operation stage. 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. In addition, the services provided may be subject to fluctuations in their quality and, in some cases, a service provider may simply (break the contract and) stop providing the service, leaving the VO short of one service. 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 in light of perturbations 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 this type of 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 the discussion of 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. The system agents that are shown in Figure 1 are discussed in more detail in the relevant subsections that follow.

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, known as 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) [18] 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 the role of VO Manager (VOM), responsible for ensuring that

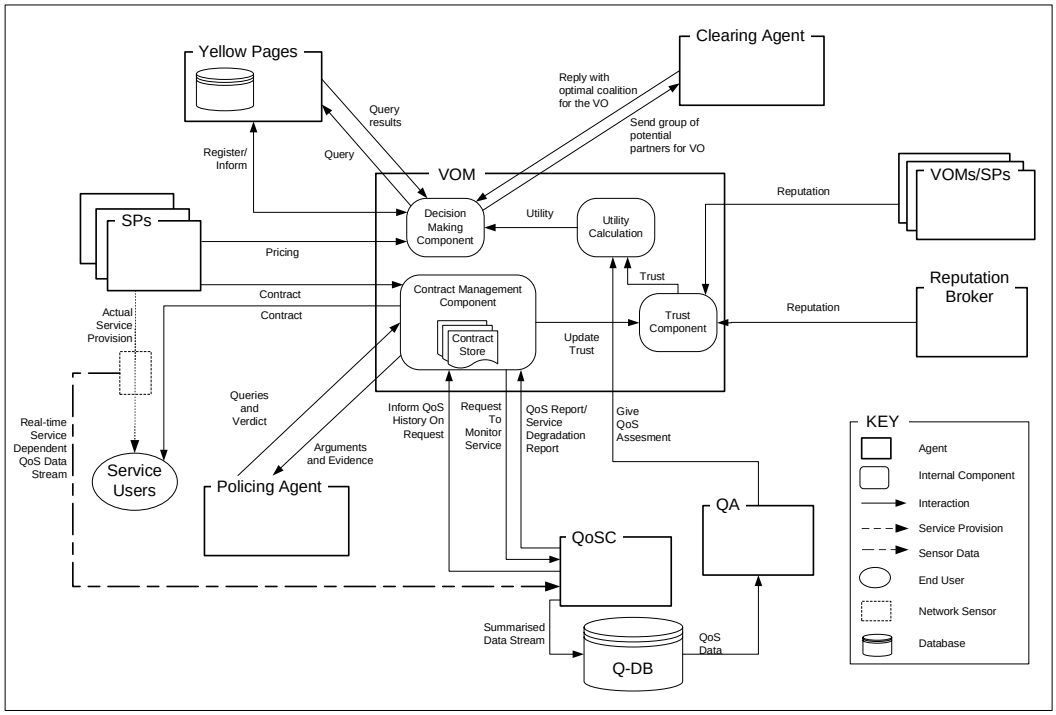

Figure 1: The CONOISE-G system architecture

each member of the VO provides its service according to contract.

During the operational phase of the VO, the VOM may request a QoS Consultant (QoSC) to monitor services provided by any members of the VO and any member of the VO may invoke a Policing agent to investigate a potential dispute regarding service provision. Ultimately, the 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, or some breach of contract is observed, the QoSC alerts the VOM, which initiates a VO re-formation process; the VOM passes relevant information (including service provider name and outcome of the contract held with that provider) to 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, and 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

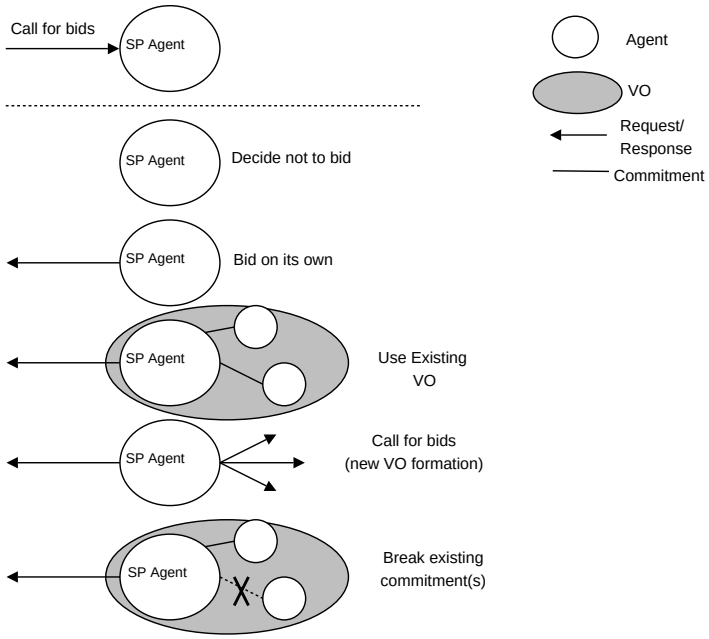

Figure 2: Deciding whether and what to bid

architecture in more detail.

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, the SP must check its current resource use (in terms of 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, a cumulative scheduling algorithm based on Constraint Satisfaction Programming (CSP) techniques [4] is used to aid the agent in this process. In more detail, 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, constraint reification [5] is used, which

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.

When a 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 (represented by the pricing interaction in Figure 1). Since multiple bids for the same request are possible, the bids received from the SPs must be *cleared*. That is, which ones to accept (or which partners to select) in the formation of the VO must be determined, a process which is handled by the Clearing Agent, as shown in Figure 1. 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). To this end, 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 [8].

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, there is often have a choice of service providers to whom tasks may be delegated. In such cases, trust serves as an indicator of which of these possible partners are likely to carry out the task as specified, but its usefulness also extends into the other stages of the VO lifecycle.

In CONOISE-G, trust is taken to be a particular level of the subjective probability with which an agent assesses that another agent will perform a particular action, both before it can monitor such an action and in a context in which it affects its own action (based on the definition from [13]). 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. Here reputation is defined as *a commonly held set of opinions about an entity* [23], and it is the aggregation of these common opinions that forms a level of trust.

In more detail, the trust and reputation system [26, 21] consists of two distinct parts. The first is a trust component which is internal to all agents that require a trust metric in their decision-making process, as

shown in Figure 1. Its function is to provide its owner agent with a level of trust for a given service and service provider, and it 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 subsequently used to determine a trust value when required. Outcomes can either be successful or unsuccessful, where a successful interaction is one for which the service provider has delivered the service specified by the contract. The sum of the successful and unsuccessful outcomes, with a particular agent, are used as α and β parameters for a beta distribution that represents the trust of this agent in another agent. More specifically, the expected value of the beta distribution obtained using the parameters is the actual trust value, and represents the trustworthiness of an agent as a number between 0 (completely untrustworthy) and 1 (completely trustworthy).

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, obtained by examining how much evidence was used to calculate it (it is directly related to the standard deviation of the beta distribution). It is used by the trust component to reason about whether an agent itself has adequate evidence or whether it needs to obtain further (reputation) information from others. When the confidence in its own calculation does not exceed a particular threshold, the trust component requests such reputation information from others. In this model, reputation is not necessarily assumed to be accurate, and the possibility that an agent may intentionally mislead is allowed. 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, several of which may serve as a distributed store of reputation information. These reputation brokers provide aggregated stores 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. This is achieved 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. Brokers are arranged according to organisational hierarchies; for instance, each department within a company may have a broker that 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. This is represented in Figure 1 by the reputation interaction between the trust component (of a VOM) and the Reputation Agent. 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. Currently, agents provide reputation information upon request, however these actions can be incorporated into a simple payment scheme, based on an information economy, to give the agents incentive to provide reputation information.

Policing within a VO

While trust and reputation ratings can 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. Given this, 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 all operations.

In more detail, 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. Given this, our approach borrows ideas from computational models of legal reasoning and legal argumentation [3].

Specifically the policing system is viewed 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) shown as the Contract Management Component of the VOM in Figure 1; an interface to allow agents to provide arguments and evidence to the system (in Figure 1 this is represented by the interactions between the VOM's Contract Management Component and the Policing Agent), and a Policing Agent that has the capabilities 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 arguments cannot be combined to reach a verdict.

In CONOISE-G, the representation of contracts is based on the emerging Web Services standard for agreements, WS-Agreement [7]. While a number of other contract representation schemes have been proposed (see for example [1, 15]), we base our work on WS-Agreement so as to align ourselves with the Grid world. However, due to the complexity of the environment, extensions to WS-Agreement are required,

while some parts of the specification, which are not useful in our domain are ignored, as described in [19], an RDF based contracting language has been created that allows for the concise representation of concepts such as repeated actions, contrary to duty obligations and sanctions. Methods for grounding the semantics of such contracts to bring these pragmatic approaches closer to formal contract specification languages, such as those developed in [10] and [20], are also being investigated. The evidence gathering mechanism employed is tightly coupled with the reasoning machinery; both activities are driven by sets of defeasible arguments. Thus 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. Currently the argumentation framework in which policing will take place is being developed. Thus at this time policing takes place by comparing the state of the world (obtained by observing the environment, messages from agents and notifications from the QoSC) to the state of the contract.

QoS Assessment and Monitoring

In a Grid environment, service providers will generally be inconsistent in the quality of service they provide to users both during a particular instance of service provision to a single user, and to the environment as a whole over a longer period of time. Furthermore, it is not possible to determine what distinguishes a good service from a bad one based purely on the level or pattern of service that they have provided (since users will have varying expectations in terms of what levels of quality they deem acceptable). The need for QoS assessment and monitoring arises from these two factors. In our model we consider assessment and monitoring to be part of a wider QoS lifecycle: the specification, assessment, monitoring and logging of QoS for service provision, supported by information flows between each stage and a QoS taxonomy for expressing user requirements.

CONOISE-G takes a user centric approach in fully supporting the QoS lifecycle [25]. Specifically, service providers are able to specify their QoS promises, and service users specify their QoS requirements using a DAML-S derived QoS taxonomy [9]. To allow for diverse services with potentially conflicting definitions of a particular QoS attribute, within this taxonomy QoS attribute specifications are separated from their measurement methods and allow mappings between the two to be established by individual services. That is, individual providers may define any QoS attributes in an expression they prefer for their services, but how such attributes will be monitored depends on which measurement methods the providers choose. This enhances the applicability and generality of the proposed framework.

In particular, three components are responsible for supporting the QoS lifecycle within the CONOISE-G architecture, as shown in Figure 1. First, during the operation of a VO the QoS Consultant (QoSC) monitors the performance of each of the delivered services against their promised QoS, so as to ensure that the VO as a whole is performing to agreed levels as well as each participating entity. QoS Monitoring performs three important tasks; providing data so resource brokers may locate appropriate components, streaming data to application steering operations and notifying system administration of possible performance anomalies. Supporting this range of monitoring requirements requires a flexible approach to the interaction between the QoSC and other components. To address this issue, the QoSC has been developed to allow registered components to obtain QoS information either periodically (at specified intervals) or when the QoS of a monitored service falls below a specified level. In addition, the model is designed to provide monitoring support independent of the underlying sensor mechanisms used. Since these sensors can typically be expected to generate a large amount of time-stamped data at a very high speed, we build our monitoring mechanism around a data stream processing engine [2]. This will allow the quality of delivered services to be monitored and dynamic, ad-hoc, continuous monitoring requests on the unbounded data sets to be handled with near real-time performance. Such an approach thus complements and contributes to the existing work on monitoring QoS in Grid environments [28][24].

Second, the streamed information that is generated from the QoS monitoring stage is summarised and stored within the QoS Database (QDB) component providing a permanent record of the performance of each service provider within the environment. This component can be queried to provide information that will serve as evidence in a number of decision making processes: by the policing component to substantiate allegations made against a service provider or consumer, by individual agents to determine whether to make an allegation against another agent and also as an aid in future instances of service discovery to determine whether a provider is able to meet a user's QoS requirements.

Third, the QoS Assessment (QoSA) component provides for the effective handling of a user's QoS requirements at the service discovery stage and uses the QDB to establish the likelihood that a particular service provider will be able to meet those requirements. In CONOISE-G current approaches to the incorporation of QoS Assessment in service discovery [17, 27] are extended to address two key issues in the area. Firstly, the effect of the relationships between QoS attributes of a service contrasts with the approaches taken in existing work which, in general, assume the attributes used to specify a user's requirements at service discovery are mutually independent [22]. By considering such relationships, the accuracy and validity of the assessed quality of a provider may be improved. Secondly, the information available when making a quality assessment may be substantial in volume, inaccurate and incomplete, and constraints will be imposed on the performance of the assessment algorithm both in terms of the length of time available to carry

out the assessment and the required accuracy of the result. This aspect is particularly relevant to the area of Grid computing, where dynamic, near instantaneous discovery, assessment and composition of resources is the norm.

4 The CONOISE-G Implementation

The CONOISE-G environment is FIPA[7] compliant and the implementation uses the JADE[8] 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 [14]). These representations were choosen 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, existing schemas and ontologies can be reused; for example, the representation borrows heavily from the DAML-S service ontology. Thus, any existing schemas or ontologies in a particular application domain can be exploited. 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, there is 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. The definitions in the ontologies can be 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 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*;

[7] `http://www.fipa.org`
[8] `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>
```

Figure 3: RDF call for bids sent to SPs

- 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 **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 serialis-

```
<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>
```

Figure 4: RDF bid issued by SP

able in a portable, open XML syntax, and easily parsed and processed using tools such as Jena2 [9].

In terms of the user interface, the GUI (in Figure 5) shows two large windows. The window split into four columns (in the background) shows the requirements made by a user for a composite service, the registered SPs and their services, the bids that the SPs make in response to a VOM's request and the last column shows the SPs that make up the VO. When the VO is re-formed, the new SP is incorporated into the *fourth column* display. The window in the foreground simulates a service provision episode of a movie service on a PDA. Here, the monitored (simulated) quality of service providers is also represented. Currently, the 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: (i) 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 performance, rather than the old agent's; (ii) 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 and Future Work

The work described in this paper takes an approach in which issues relating to the formation and operation of robust VOs in dynamic environments with unreliable agents are considered. In contrast to the "brawn" of

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

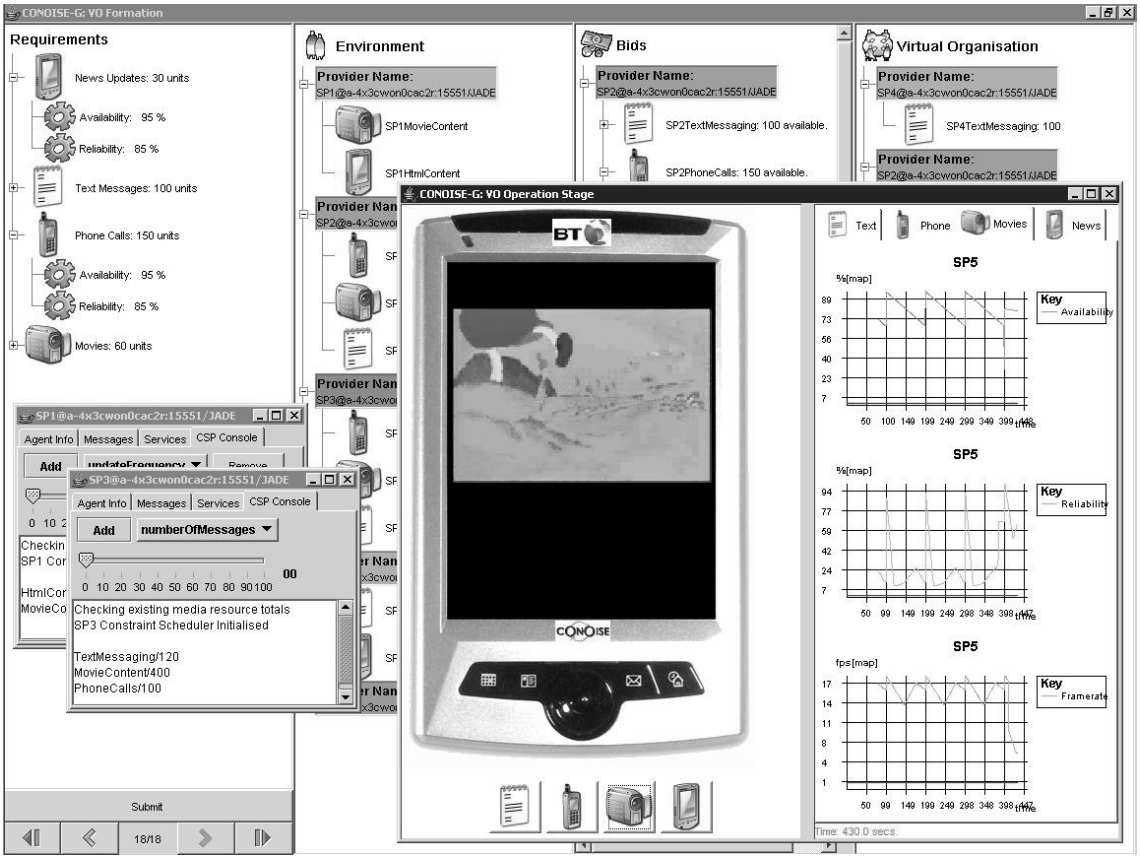

Figure 5: The CONOISE-G user interface

the Grid, we have concentrated on the "brains" [11] — 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 Grid scenarios. Specifically, the paper described our implemented prototype of the system, and highlighted the work being done on extending the system to incorporate more sophisticated application scenarios. The key problems encountered in the design phase of the system were those of standardising communication between agents and defining ontologies in a sufficiently detailed manner so as to allow agents to use them in their reasoning. These problems were overcome, wherever possible, by adopting standard technologies, such as RDF, FIPA and WS-Agreement. However, in other cases new technologies were needed (for example trust modelling and policing).

Future work will include a quantitative analysis of the performance of CONOISE-G VO structures. This will show how the mechanisms discussed in this paper will scale with the number of entities in the system. In addition, we will extend the trust modeling by incorporating information about social relationships (that may exist between members of a VO) into the trust value calculation, and enhance the reasoning capabilities

of the policing agent to allow more complex deliberation over evidence.

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] Abrahams, A. S.; Bacon, J. M.: *A software implementation of kimbrough's disquotation theory for representing and enforcing electronic commerce contracts;* In: Group Decision and Negotiation, Springer Science and Business Media; 11:6 (November 2002), pp. 487 – 524

[2] Babcock, B. et al.: *Models and issues in data stream systems;* In: Proccedings of 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM Press; 2002, pp. 1–16

[3] Bench-Capon, T. et al.: *Computational models, argumentation theories and legal practice*, In: Argumentation Machines: New Frontiers in Argument and Computation, Reed, C. A.; Norman, T. J., ed., Kluwer, 2003, pp. 85–120

[4] 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, pp. 363–377

[5] Chalmers, S. et al.: *Commitment management through constraint reification;* In: Proceedings of 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems; 2004, pp. 430–437

[6] Christian, D.; Young, R. M.: *Strategic deception in agents;* In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, IEEE Computer Society; 2004, pp. 218–226

[7] Czajkowski, K. et al.: *WS-Agreement: Agreement–based grid service management;* In: Global Grid Forum; 2003.

[8] 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, pp. 23–27

[9] Deora, V. et al.: *Incorporating QoS specifications in service discovery;* In: Proceedings of Second International Web Services Quality Workshop, Springer Verlag; 2004, pp. 252–263

[10] Dignum, V. et al.: *Formal specification of interaction in agent societies;* In: Proceedings of the second Goddard workshop on formal approaches to agent based systems, Springer Verlag; 2002, pp. 37–52

[11] Foster, I. et al.: *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, IEEE Computer Society; 2004, pp. 8–15

[12] Foster, I. et al.: *The anatomy of the Grid: Enabling scalable virtual organizations;* In: International Journal of Supercomputer Applications, MIT Press; 15:3 (2001) pp. 200–222

[13] Gambetta, D,: *Can we trust trust?;* In: Trust: Making and Breaking Cooperative Relations (Chapter 13), Basil Blackwell; 1988, pp. 213–237

[14] Grimnes, G. et al.: *Granitenights – a multi-agent visit scheduler utilising semantic web technology;* In: Seventh International Workshop on Cooperative Information Agents, 2003, pp. 137–151

[15] Kollingbaum, M. J.; Norman, T. J.: *Supervised interaction: creating a web of trust for contracting agents in electronic environments;* In: Proceedings of the first international joint conference on Autonomous agents and multiagent systems, ACM Press; 2002, pp. 272–279

[16] Luck, M.: *A manifesto for agent technology: Towards next generation computing;* In: Journal of Autonomous Agents and Multi-Agent Systems, Springer Science Business Media; 9:3 (2004) pp. 203–252

[17] Maximilien, E. M.; Singh, M. P.: *Toward autonomic web services trust and selection;* In: Proceedings of the 2nd international conference on Service oriented computing, ACM Press; 2004, pp. 212–221

[18] Norman, T. J. et al.: *Agent-based formation of virtual organisations;* In: Knowledge-Based Systems, Elsevier; 17 (2004) pp. 103–111

[19] Oren, N. et al.: *Service level agreements for semantic web agents;* In: Proceedings of the 2005 AAAI fall symposium on agents and the semantic web; 2005, To Appear

[20] Pachheco, O.; Carmo, J.: *A role based model for the normative specification of organized collective agency and agents interaction;* In: Journal of Autonomous Agents and Multi-Agent Systems, Springer Science Business Media; 6 (2003) pp. 145–184

[21] Patel, J. et al.: *A probabilistic trust model for handling inaccurate reputation sources;* In: Proceedings of the Third International Conference on Trust Management (iTrust), LNCS Volume 3477, Hermann, P. et al, ed., Springer-Verlag, 2005, pp. 193–209

[22] Ran, S.: *A model for web services discovery with qos;* In: SIGecom Exch; 4:1 (2003) pp. 1–10

[23] Sabater, J.; Sierra, C.: *Regret: A reputation model for gregarious societies;* In: Fourth Workshop on Deception Fraud and Trust in Agent Societies; 2001, pp. 61–70

[24] Schopf, J. M. et al.: *Monitoring and discovery in a web services framework: Functionality and performance of the globus toolkit's mds4;* Technical report, Argonne National Laboratory, 2005

[25] Shercliff, G. et al.: *Supporting qos assessment and monitoring in virtual organisations;* In: Proceedings IEEE International Conference on Services Computing; 2005, pp. 249–250

[26] Teacy, W. T. L. et al: *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, ACM Press; 2005, pp. 997–1004

[27] Tian, M. et al.: *Efficient selection and monitoring of qos-aware web services with the ws-qos framework;* In: Web Intelligence, IEEE Computer Society; 2004, pp. 152–158

[28] Tierney, B. et al.: *A grid monitoring service architecture;* Technical report, Global Grid Forum Performance Working Group, 2002

6 Biographical Notes

Jigar Patel is a PhD student in the Intelligence, Agents, Multimedia (IAM) group at the School of Electronics and Computer Science at the University of Southampton. His main interests include computational trust models and trust based decision making. He received his BEng in computer systems engineering from the University of Warwick.Contact him at IAM Group, Level 3, ECS Dept., Univ. of Southampton, Southampton, SO17 1BJ, UK. The remaining authors of this paper work together on the CONOISE-G project (details available from www.conoise.org)