Engineering Organisation-Oriented Software

M. Kollingbaum, T. Norman
Dept. Computing Science, University of Aberdeen, UK
{mkolling, tnorman}@csd.abdn.ac.uk

N. Mehandjiev
School of Informatics, University of Manchester, UK
mehandjie@acm.org

K. Brown
Cork Constraint Comp. Centre
Dept Computer Science
University College Cork, Ireland
k.brown@cs.ucc.ie

ABSTRACT

The conventional ways of building software are accepted to produce rigid systems that impede the processes of change typical for contemporary organisations. In this paper, we propose that software can be made more adaptable and tuned to the needs of changing organisations, if it is built using organisation-inspired principles and software structures such as Virtual Organisations, roles and norms. Agent-based software engineering is already using these principles, and we extend the state of the art in that domain by proposing an “open systems” approach, where agents can join and leave Virtual Organisations at will, taking on different roles as needed. Reasoning on organisational roles and norms is facilitated by formalised contract templates and automatic conflict resolution strategies. In terms of overall lifecycle, a system is initiated to satisfy a set of formalised requirements. Agents respond to bids for joining a Virtual Organisation, where each bid is for a contract-based coalition. In this paper, we describe our approach and outline a set of research challenges.

Categories and Subject Descriptors

C.2 [Computer-Communication Networks]: Distributed Systems; D.2 [Software]: Software Engineering

General Terms

Design, Management

Keywords

Virtual Organisations, multi-agent systems, distributed constraints

1. INTRODUCTION

The agility of contemporary organisations depends in no small part on the flexibility of their supporting systems. This includes the ability of companies to form short-term coalitions, which depends on the interoperability of their software systems, but also how software can support the activities involved in forming such coalitions of companies.

Today, these “virtual organisations”, binding a set of enterprises into a coalition, are mainly long-term supply networks, which is not surprising given that they are “handcrafted” and rather rigid after deployment. This is due to a need for an extensive human intervention to deal with customer-supplier relationships, resolving the terms and conditions of the flow of information, commodity and money and implementing – or, at least, configure – existing software systems to enable an interoperaton between companies. This rigidity in current software systems counteracts the need for enterprises to be responsive to changing circumstances and business opportunities in today’s markets.

We are specifically interested in improving the speed of response of different companies to emergent market opportunities, where they pool their skills and capabilities to fulfill a complex request issued by a customer. Fast response would require the reduction of direct human intervention and strengthening the adaptability and self-configurability of software components employed by the different companies.

In this paper we propose that software can be made more adaptable and self-configurable, if it is built using organisation - inspired constructs such as Virtual Organisations, roles and norms. The concept of a Virtual Organisation is discussed in the agent community and represents the undertaking to design multi-agent systems according to organisational principles. The resulting distributed software system is comprised of a set of agents that act and interact according to organisational principles. Special agents are needed for the formation of VO’s. Their design must allow them to reason about their role within an organisation – what are their duties and privileges (the norms established within an organisation), how they are related to other agents and how their role determines their actions [16, 15].

A specific emphasis in VO research is the automated formation of Virtual Organisations by a set of software agents. In terms of VO formation, agents will take on specific roles within a community of agents by undergoing a process of negotiation that results in a kind of agreement or contract. Such a role-based approach assigns specific duties and privileges to agents. These agents may represent organisational entities, parts of an organisation or different cooperating organisations. A clear benefit of using an organisational approach in the construction of software systems is the direct correspondence between organisational structures in the...
“real world” and those within the software system – the software system “reflects” the reality. This also helps to make software systems more responsive to organisational change – by employing self-motivated agents with the ability to reason about organisational structures, they can also detect and adapt to changes within organisations.

The dynamic formation of software systems according to organisational principles can be used to inform and guide the formation of a “virtual organisations” between different enterprises. Employing agent technology and concepts of Virtual Organisations, this formation of a coalition between enterprises is guided by the negotiation and self-configuration task of software agents representing these enterprises. It is expected that this removes the need for manual reconfiguration of software systems. It may also advise human decision-makers about feasible and non-conventional configurations. The end result is expected to be increased responsiveness and competitiveness of companies supported by such a software system.

In the following sections, the paper explores different organisational constructs for building distributed organisation-oriented software system, followed by details of an approach that is proposed to achieve adaptable software in support of cross-organisational collaborations.

2. ISSUES IN USING ORGANISATIONAL CONSTRUCTS TO BUILD SOFTWARE

Aspects of autonomy, adaptability and self-configuration have been discussed intensively by research efforts into multi-agent systems. In this context, the concept of a Virtual Organisation is a current focus of interest. Virtual Organisations are driven by the idea of introducing a social or organisational structure into agent societies. As usual, there are two options how to create software systems that correspond with such a social or organisational metaphor. Conventionally, the organisational structure is a part of the design of the agents themselves, as in some role-based agent system design approaches [14, 13], which means that change of the organisational structure is only possible via a redesign of the complete system and its re-deployment. Here we seek a more flexible solution, based on the “open system” approach, where a virtual organisation is formed dynamically with agents joining or leaving such a society at will.

To allow such a dynamic formation and disbanding, the agents need to have special qualities. They must be designed in a way that allows them to take on any role within the society (and not just a specific one). For this, we have to:

- describe organisations in terms of a set of roles that determine the social position of agents within such an organisation and their relationships to other members of the organisation; and we have to

- design the agents in a way that enables them to reason about the social position they gain by adopting a specific role, and what the correct behaviour in such a role would be.

If we regard a role as being characterised by a set of social norms – obligations, permissions and prohibitions that require the agent to act in a certain way – then agents joining an organisation dynamically must be able to reason about and act according to these normative standards. These normative standards can be understood as agreements between members of a society – if an agent joins a society, it will sign a contract. Such a contract will determine a set of norms or social constraints the agent commits to deploy and adhere to during its membership of a society. A Virtual Organisation, therefore, will emerge from at least two agents signing such a contract. We can also say that a Virtual Organisation is, in fact, described by such contract (or set of contracts). A way to design and describe an organisational structure is by designing contract templates [16]. These contract templates (held and issued by some normative authority) will then be used and instantiated to actually create a dynamically forming software system.

Initially, we pointed out that the software technology used influences the responsiveness of an enterprise. Agents operating in a concerted fashion in the form of Virtual Organisations and joining / leaving such software structures can actually support the needed short-term pooling of enterprises for specific business cases. It requires companies to use modern software infrastructures that support information exchange and collaboration between agents in open system environments, and a specific type of agents that are motivated in their actions by social or normative standards.

In the following sections, this vision will be outlined in more detail.

3. ORGANISATION-ORIENTED SOFTWARE ENGINEERING

In our vision of organisation-oriented software engineering, concepts of norm-governed agents are essential to a dynamic formation and evolution of software using virtual organisations. Of similar importance are mechanisms that allow agents to form coalitions in an emergent ad-hoc manner so that they can respond to requests for new services. During their operation within an organisation, agents have to assess, if their actions comply with existing social norms. For this purpose, agents will employ dynamic distributed constraint optimisation methods. These components will be described in detail after the outline of a scenario illustrating our approach.

3.1 Scenario of forming short-term coalitions

In Figure 1, a scenario is outlined that allows companies to form short-term coalitions. Software agents support this process and create the appropriate software infrastructure by forming software-based virtual organisations. This scenario uses logistics as an application domain. Accordingly, a Fourth-Party logistics provider (4PL) maintains and monitors an organisational structure. As outlined before, we use contract templates to describe such organisational structures and the 4PL provider acts as the manager that issues such descriptions. Advertisements for specific tasks (in this scenario, these are transport tasks) are issued and agents as representatives of companies respond to these advertisements. In our transport scenario, we assume that a set of agents take on roles in a specific logistics network.

These agents may represent shipping companies, transport and storage facilities (such as lorries, ships and warehouses) or information provider agents, customs clearing agents, etc. The 4PL provider takes on the role of an authority and issues a request for the formation of a specific logistics network. A subset of agents will respond to this request and form a virtual organisation (VO). Each agent will adopt a specific role
within this VO and, with that, takes on certain obligations and restrictions for its actions. For example, agent A takes on the obligation to deliver the first batch of parts to location X between 12 and 5, and the second batch between 2 and 5, agent B takes on the obligation to getting customs clearance by 6, etc. Within these contractual commitments, the agents have to agree on how to pursue their concerted transport activities - they have to agree on a set of global values that obey all the behavioural constraints expressed in their contracts. The agents will announce specific values for their actions. For example, agent A announces it will deliver the first batch at 2 and the second at 4. Now we can assume following scenario: agent B responds that it cannot start the customs process until it receives the second batch and that it will take 3 hours. Because of this, agent A will change the delivery sequence and agrees to ship the second batch at 2 and the first batch at 4. It can do that because this would still be in accordance with its contractual obligations.

3.2 Forming a Virtual Organisation

The previous example shows that in our approach, we assume (a) a certain form of social structure is mirrored in the software system, (b) pre-designed specifications are also formalised as software-based contract templates and (c) a protocol for establishing a Virtual Organisation. The core of a social structure is the role of a manager that represents a kind of “legal authority” (in the example, this role is adopted by the 4PL provider). This VO-manager is responsible for the formation and management of a virtual organisation – based on specific requirements of a customer request, it has to recruit a set of agents and take them under contract. The VO-manager holds a set of contract templates that are pre-specified and outline (a) a specific set of roles and (b) determine the rights, privileges and duties of agents recruited into such roles. A special VO-management protocol, called Supervised Interaction, was outlined in [16, 15]. Supervised Interaction proposes a specific management procedure that covers the selection of a specific contract template, the advertisement of roles and the recruitment of agents, the finalisation of a contract between these agents based on the contract template, and the execution of the Virtual Organisation. In detail, this management procedure comprises following steps:

- **Registration.** In the first phase, the VO-manager issues advertisements for the formation of a Virtual Organisation. The advertisement would specify a set of roles which would contribute to the goal of the VO being formed. A role can be taken by a single agent, or by a team of agents, which can form a coalition using a mechanism described below. Agents register their applications with the manager.

- **Negotiation.** A contract template is used, which is neutral to a specific business case and may lack information about, for example, a specific delivery date or the amount of goods to be delivered. This has to be negotiated with the recruited agents. The result is an instantiated contract signed by the contractees. If no contract can be established, the manager has to re-advertise and recruit new agents.

- **Execution.** The Virtual Organisation goes into execution - agents act according to their obligations specified in the signed contract.

Special agents are needed that are able to reason about norms within such a Virtual Organisation. The NoA agent architecture is an approach to implement such *norm-governed*...
agents. The NoA language was designed for the expression of contract templates. These concepts are described in [16, 15].

A contract specifies, what the obligations of single agents are, but does not determine how they actually go about to fulfill them. Certain contraints may restrict the operation of such a virtual organisation. As pointed out in the example above, there may be, on the hand, a certain bandwidth of activity the agent can choose within the agreed norms, but, on the other hand, certain constraints for its actions. The agents, therefore, have to carefully plan their actions in a distributed fashion. For this, distributed constraint satisfaction [2] should be used during the negotiation phase to make such a virtual organisation operational.

3.3 Emergent Composition of Coalitions

In the ideal situation we would have at least one agent capable of fulfilling each role specified by the VO-manager, and this agent is interested in applying to fulfill that role. This case is more likely to arise in mature “software markets”. In the opposite case, there may be a lack of agents capable of fulfilling a certain role, and this role would have to be decomposed down to a set of primitive roles. Any top-down decomposition, however, is likely to explode the number of alternative decompositions we need to search through if we are to find a match to those skills already existing on the marketplace.

A bottom-up coalition formation would be more beneficial in such circumstances. The details of this approach, originally developed for forming coalitions between SMEs to supply sub-systems in the automotive industry, are available elsewhere [4]. The essence is that an opportunity / goal such as a business order, or a role in our case, is placed on a shared blackboard area, which is monitored by a set of interested agents. The agents who are capable of contributing partial solutions would write these solutions onto the blackboard area. If a partial solution builds upon another partial solution already on the blackboard, the “newcomer” agent approaches the contributer of the latter to negotiate if the two can work together to provide a combined solution. When a partial solution “grows” to a full solution stage, it will be evaluated by the VO-manager against other candidate solutions. The coalition with the best bid would be appointed to the role.

3.4 Planning the Operation

In order to operationalise a contract, agents must plan their actions to meet the responsibilities assigned to them. They must also ensure that their actions are compatible with the other agents in the Virtual Organisation, and thus the agents must coordinate their decision making. For example, a logistics agent may be responsible for delivering multiple products to a warehouse from sites across a country; it may produce a plan that involves using one truck to do all deliveries, which includes a route plan with time windows for the truck. However, if that plan involves picking up a delivery from a factory at the start of the day, but the factory agent does not plan to produce the products until later, the global plan will fail, and thus the obligations of the Virtual Organisation will not be satisfied. The logistics agent and the factory agent must negotiate on the details of their plans, reassigning times to tasks so that the global plan can be completed.

To do this, we use distributed constraint satisfaction (or distributed constraint optimisation). During the formation of the Virtual Organisation, the common elements of the individual responsibilities must be identified, and these are represented as constrained variables - for example, the delivery time for a shipment is a variable, which may be constrained to lie within a time-window, and which affects the tasks of the logistic agent and the customs agent. Constraints are then imposed to link relevant variables - for example, the logistics agent’s delivery time must be before the commencement time for customs clearance. The individual agents must then propose values for their own variables, and receive reports from other agents as to whether the proposed values are acceptable; if not, new values must be found.

We assume that the individual agents are working towards a common goal, and are committed to achieving a plan, and thus the VO will arrive at a feasible plan if one exists. To ensure this, in the simplest case, the VO manager will determine a hierarchy among the agents, and the hierarchy will determine how conflicts between agents are to be resolved. To stop all the costs being loaded onto the lowest agents in the hierarchy, the original contract from the VO formation would specify a maximum cost that an agent is willing to incur to meet its obligations. Each agent is then limited in its ability to drive its own costs down at the expense of its subordinates. More complicated cases will require a more exhaustive specification of the rewards received by an agent for particular allocations to its constrained variables.

4. DISCUSSION

We believe that the proposed combination of organisation-inspired software techniques will create software which is flexible to react to organisational changes and able to support the pursuit of short-term business opportunities in goal-driven virtual organisations.

However, these techniques have been developed in isolation, and we expect significant conceptual challenges before reaching a stage of full integration and prototype testing, for example how can we automate the reasoning in cases where a coalition of agents has had their bid to fulfill a role accepted, and have signed the contract associated with that role.

Another set of problems will arise on the interface between software support and supported organisations. Changes in the latter would be reflected in software, but this may cause inconsistencies in the norms and contracts governing the operation of the software agents [17]. Thus dynamic changes in organisations have to be taken into account in the design of the agents recruited into such an organisational structure. For example, a common change within an organisation is a change in the responsibilities and privileges of an agent in a specific role. Specific actions the agent may have been able to deploy previously may suddenly be forbidden – the social or normative position of the agent changes. If in the course of such an organisational change new norms are introduced that create a conflict – the agent suddenly holds at the same time a prohibition and a permission for a specific act, then it is rendered incapable to determine if its options for action to fulfill its obligations are all consistent with its set of norms or not. Specific conflict resolution strategies must be made available to resolve such situations and keep a Virtual Organisation operational [17]. For example, an agent C operates in the role of a database manager that is obliged to
access a database for details about a specific shipment. A change in the VO can lead to a situation where agent C's permission to access this database is suddenly revoked and agent C is in a situation to be obliged to perform an action it is, at the same time, forbidden to perform.

Operational problems can also arise from a situation where contractual requirements and operational constraints lead to an unsatisfiable problem. For example, summing the minimum delivery times over all the agents in a delivery chain may show that it will take at least 12 hours to deliver, but the contract mentions a delivery deadline that is 10 hours away. Individually, the agents will not know there is a problem, but collectively they will need to discover it and take appropriate actions. This can be done on the normative level by renegotiating the terms of the contract – for example, re-negotiating an new delivery date with a customer.

5. RELATED WORK

Organisational modelling is reasonably well understood in terms of designing complex distributed information systems [3, 5]. State-of-the-art software technology to support distributed organisations are usually client-server solutions (using web service technology in their latest incarnation). These are instantiated from a given organisational model. Modern systems of this kind rely on accurate organisational models and exhibit no adaptability to changes within organisations.

Research into Multi-agent systems operates at the intersection between Distributed Systems and Artificial Intelligence. The goal is to introduce more flexibility into the operation of distributed software systems by constructing them as societies of autonomously acting agents. The problem of using organisation-oriented approaches in the design phase only is that the organisational structure is implicitly encoded into the agents - the agents behave according to the norms and regulations of the organisation because they are implemented that way. This leads to unacceptable inflexibilities during runtime - any change, for example, in the organisational structure requires re-implementations of parts of the system. Virtual Organisations [19, 7] and concepts of Electronic Institutions [8] introduce organisational structures into Multi-agent systems. Research efforts in this area are, for example, ADEPT [12] and CONOISE-G [19]. A recent development in this area is the use of normative concepts to specify a social layer for the coordination of agent behaviour [19].

Constraint programming [6] is a proven technology for decision support and optimisation. It has been successfully applied to many problems throughout organisational systems, including scheduling [1], resource allocation [11] and configuration [9]. However, it generally relies on the assumption that the problem is self-contained, static and well-defined. For application to complex enterprise systems, this assumption must be relaxed, and the technology must be extended to incorporate distributed models, in which solvers are autonomous and adaptable, reacting to changes in the environment, and coordinating solutions with other participants. These problems are addressed in the areas of robust constraint programming [10] and distributed constraint satisfaction [2, 18], where current research is still focused on relatively small academic problems.

6. CONCLUSION

The flexibility and responsiveness of enterprises depends on the degree to which their software systems can adapt to or even drive organisational changes such as the pursuit of short-term goal-driven collaborations. In concert with the Multi-Agent Systems (MAS) community [20, 14], we believe that using organisational structures and concepts (such as virtual organisations, coalition formation, contracts and norms) as first-class design constructs when building systems can produce software which is aligned with the real-world organisation supported by it, and is also easier to keep in tune compared with conventionally structured software. The approach proposed here, however, adds value to organisation-inspired designs encountered in MAS by employing mechanisms which allow the system to work in open and fluid environments, where organisations of software agents can be created and modified at run-time, including changing of agents’ responsibilities and resolving any inconsistencies this may generate.

Our approach builds on three existing pieces of research: (a) Norm-based Multi-agent systems and Virtual Organisations, (b) Distributed constraint solving and (c) Emergent Coalition Formation. We are currently developing a coherent integrated model of virtual organisation management to bridge these contributions within the context of software engineering for flexible support systems. This model will utilise both distributed constraint solving techniques and normative system specifications to enable the dynamic composition of flexible coalitions of autonomous agents. The autonomous components of systems developed using this model will be flexible and responsive to changing circumstances, and hence enable robust and dependable service provision that degrades gracefully.

7. REFERENCES


