Agents for Cross-Organisational Business Interoperabiltiy

Iain Duncan Stalker[1] and Nikolay Mehandjiev[2]

[1] School of Science and Technology, University of Teesside, Middlesbrough TS1 3BA, UK
i.stalker@tees.ac.uk,
[2] Manchester Business School, The University of Manchester, Manchester M15 6PB, UK

Abstract. Increasing complexity of products and services together with shrinking lead times means that demands are often best satisfied through dynamic networks of collaborating enterprises. Businesses seek agile partnerships where companies focus on core business activities and partners provide complementary expertise. Successful collaboration demands a tight yet flexible integration of business processes. In this paper we focus on issues of process interoperability and the use of agents to support *cross-organisational process interoperability.* We maintain that an informed decision on the place of agents within this context demands a systematic approach: such an approach forms the core contribution of this paper. We apply a modified version of the *function-behaviour-structure* (FBS) framework of Gero as an organising schema to develop a clarifying structure; and to help determine the role of software agents. In particular, we use the formal elements of the FBS framework to identify those areas where agents may be applied. We highlight the need to combine this with appropriate mechanisms to promote *semantic interoperability.*

Key words: process interoperability, agents, system design, semantic interoperability

1 Introduction

Increasing complexity of products and services together with shrinking lead times means that demands are often best satisfied through dynamic networks of collaborating enterprises. Businesses seek agile partnerships. Effective cross-organisational collaboration enables a company to focus on its core business within a dynamic network of partners which provide complementary expertise. This allows companies of all sizes to pursue immediate opportunities in a chosen market through participation in virtual enterprises (VEs) and dynamic supply chains. Moreover, the trend towards eBusiness places substantial demands on software infrastructures to ensure robust intersystem communication, meaningful information exchange and successful coordination of processes and activities. Interoperability at all levels is paramount and significant advances in enabling

software technologies are needed to support this. Compatibility of business processes and activities must be realised through *process interoperability.* As the information processing infrastructures of the participants are typically linked, a frictionless information exchange requires *interoperability at the systems level.* Activities arise within the context of compatible legal and organisational structures predicating *interoperability at the business level.* Moreover, any successful collaboration is informed and underpinned by a shared understanding and this reveals a fundamental need for *semantic interoperability.*

The nature of a collaborative business network is such that a seamless coordination of processes through cross-organisational workflows is fundamental to its effective operation and the fulfilment of the purpose for which it formed. This provides the focus for interoperability issues. We see interoperability as a systemic property of the set of collaborating entities, which emerges as a consequence of their interaction, rather than as an individual property of system components. Consider the following definition of the IEEE of interoperability as the *"... ability of two or more systems or components to exchange information and to use the information that has been exchanged"* (IEEE, 1990). This has the following consequences for the mechanisms which underpin our approach to interoperability:

Process interoperability is currently supported only in a limited manner by state-of-the-art approaches. Consider, the process interoperability support offered by current Workflow Management System (WfMS), is predicated upon standard cross-organisational reference processes which are separate from the internal workflow processes of individual organisations. The two are linked through (implementations of) *ad hoc* interfaces of individual suppliers: this creates inflexible and proprietary solutions. Instead, we make use of collaborating software agents to achieve a goal-driven dynamic formation of both the team and the workflow coordinating team activities. Intelligent agent technology can provide the capabilities for deliberative reasoning; dynamic goal-oriented behaviour; and explicit declaration of interaction intent and structure. Agents are therefore used to represent collaborating business entities, providing natural representations in software for business autonomy, information hiding, self-interested behaviour and coalition formation. Agents are also a natural choice to support an ontology-based approach to semantic interoperability: they employ sophisticated communication mechanisms, based on explicit declaration of intent and ontological commitment; and provide the negotiation and reasoning capabilities necessary to maintain a devolved ontology model (cf. Section 4).

Semantic interoperability is based on the use of ontologies; and negotiation mechanisms to find common ground and identify shared ontological commitments among interacting group members. This leads to a core ontology common to all group members and a number of peripheral ontologies, each of which extends this core and is specific to a sub-group or individual organisation. Novel negotiation protocols, informed by the theory of utility and supported by rigorous ontology mappings, enable the evolution of the core and the peripheral ontologies:

for example, to incorporate new capabilities and services within the task group. Automatic reasoning within this context is underpinned by the use of Formal Concept Analysis [4] and Lattice Theory [3]. The resultant model, e.g. [2], which we call Devolved Ontology, integrates centralised and distributed approaches to ontology engineering.

In this paper we focus primarily on issues of process interoperability and the use of agents to support *cross-organisational process interoperability.*/ Our approach to semantic interoperability is discussed elsewhere, see for example [2], though we do touch upon this in Section 4. The approach reported here differs from other efforts, for example, [11, 21, 14], in the use of a formal model of the cognitive aspects of our domain constructed *a priori*. This allows us to determine the place of agents in a systematic fashion rather than developing *ad hoc* solutions.

We treat workflow configuration as a design problem. Design problems are typically under-defined, that is, the information needed to properly define a particular design problem is not available from the problem statement [22, 23]. Some form of preliminary exploration is needed to clarify the structure of the problem (cf. decision or design) space and a significant part of any design effort is devoted to this learning. An *a priori* structuring of the cross-organisational aspects is an invaluable aid to preliminary exploration in business process coordination and workflow configuration. Moreover, an informed decision on the place of agents in decentralised workflow management systems demands a systematic approach and this, again, motivates some *a priori* structuring.

We suggest the *function-behaviour-structure* (FBS) framework [1] as an organising schema to conceptualise design of workflow structures and to help determine the role of software agents. FBS has been applied in a number of engineering domains such as civil engineering, process engineering and even software engineering, but not yet to constructing workflow software. We emphasise that here we are using FBS as an informing framework rather than proposing it as an alternative to existing approaches to the development of agent-based systems and management of cross-organisational workflows. In particular, we use its formal elements to develop a systematic approach to identify areas where agents may be applied, highlighting potential contributions, to the design and construction of decentralised workflows.

This paper is an elaboration of earlier work [5] and is structured as follows. In Section 2 we present the function-behaviour-structure framework of Gero [1]: we use a slightly modified version of this to develop a framework for cross-organisational team and workflow formations, clarifying the interactions between these. In Section 3 we use the framework to identify potential areas for the application of agents. In Section 4 we briefly discuss semantic interoperability as a necessary complement to process interoperability through agents, before reflecting on our contribution, discussing some issues we feel it appropriate to clarify and cosidering how the approach may be evaluated in Section 5. We close with some concluding remarks in Section 6.

2 FBS

The function-behaviour-structure (FBS) framework [1] underlies a knowledge-based approach to conceptual design and is motivated by the following presupposition:

> *... the metagoal of design is to transform function F ... into a design description D in such a way that the [design] artefact being described is capable of producing these functions.* [1]

The design description documents the structure of the (*design*) *artefact* which is the intended product of a conceptual design effort. In our case, the artefact is a process: a global workflow linking participants in a VE. As there is generally no function in structure nor structure in function, the transformation from function to description proceeds in stages; and the FBS framework, illustrated in Fig. 1, was devised to elaborate these.

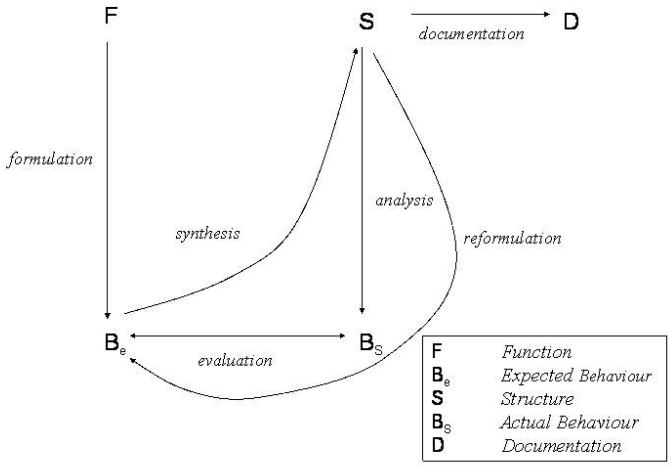

Fig. 1. The Function-Behaviour-Structure (FBS) Framework of [1]

The FBS framework identifies a number of categories of (domain) knowledge and separates this from the computational processes (transformations) which operate (act) upon it. With an identified function (or set of functions), F, we associate a number of expected behaviours, B_e, through which to realise this (*formulation*). We identify (a partial order of) components which exhibit these behaviours and assemble these into an appropriate structure, S (*synthesis*). We use appropriate techniques to predict the behaviours, B_S, of the resulting structure (*analysis*) and compare these with those expected (*evaluation*). If the two

sets of behaviours match, then we formalise the design (*documentation*); otherwise, we return to and possibly revise the set of expected behaviours and synthesise a new structure (*reformulation*): strictly speaking, reformulation is the full loop from S to S through B_S and B_e, which takes into account the information obtained through analysis and evaluation.

2.1 Modes of Reasoning and Enlarged FBS

The notions of *function, behaviour* and *structure* identify categories of knowledge which are useful in the design process without being prescriptive. In applying the FBS framework to team and workflow formation, we find it useful to insert an additional category, *capability,* between function and behaviour[1]. We consider formulation as associating function with capability and we use the term specification to denote the association of capability with behaviour. By decomposing function into subfunctions and mapping these onto necessary capabilities, we provide a first step in identifying potential team members, by matching desired capabilities with those possessed by a given resource. This will become clearer in the following discussion of applying the FBS to team and workflow formation. For ease of exposition, we consider a (cross-organisational or individual) (business) process to be formalised as a workflow.

2.2 Applying FBS to Team and Workflow Formation

We apply the FBS framework once we have established the desired outcome of the global workflow. The design of a workflow is intimately related to the formation of a core team of partners who will work together on a project, such as car assembly. However, this is not necessarily straightforward. We distinguish two modes of interaction between the team formation and workflow design: team formation precedes the workflow design; and team formation is interwoven with the workflow design. We could consider a third possibility: a core team assembles prior to workflow design, but is enlarged during workflow design as additional long term requirements are identified; this is a combination of the two previous modes.

Team formation precedes workflow formation We view this as an independent, preliminary application of the FBS framework. Here, the design artefact denotes the team to be assembled and the function is the capacity to realise, i.e. the ability to perform or produce, the global business process for which the team is assembled. This capacity translates into a set of capabilities (cf. formulation), such as car engine assembly, or parts manufacture such as the production of a car body. A potential participant in a VE is a *resource* possessing certain capabilities expected of a particular *resource class* (cf. [6]). Thus, subsets of the capabilities needed to realise the global business process can be used to identify

[1] Rather than revising Fig. 1 to reflect this insertion, we refer the reader to the central column of Fig. 3 to see where this is located.

relevant resource classes and thus potential team members. Since in team formation we do not need to go to the level of behaviours, we think of capability as abstract behaviour and do not distinguish between formulation and specification. Thus, we identify capability with behaviour here and use it to synthesise a team structure.

We examine the interdependencies of (the capabilities of) these resources and use the information to help to establish a team structure: for example, if one resource coordinates input from a number of resources, then this might suggest assigning management responsibilities to it, e.g. a partner responsible for assembling a car might manage those partners which supply the necessary components; this corresponds to synthesis, cf. Fig. 1. We analyse the team structure and the capabilities of the members to ensure that all phases and aspects of team interaction are supported by the core competencies (cf. actual behaviours in Fig. 1) of the team (cf. analysis in Fig. 1). We compare the competencies with the identified capabilities (cf. evaluation in Fig. 1). At this stage we might recognise that while the competencies subsume the necessary capabilities, our team lacks necessary competence to support an important aspect of the team interactions which was overlooked during the initial formulation: for example, by focusing on the physical artefact in car assembly, we may have neglected to ensure that our team has the necessary competence to support interorganisational accounting. Thus, we revise our initial set of capabilities to accommodate this and recycle through the FBS (cf. reformulation in Fig. 1).

Having established a core team of partners, we can apply the FBS to the different phases of the team operation to design appropriate workflow structures to support these. Where appropriate this would include the formation of a subteam, or task force, to address a particular phase: this is interwoven with the workflow design as suggested below, with the members chosen from our core team. For example, in the case of car assembly, we would cycle through the FBS to coordinate the supporting processes of parts manufacture and timely delivery to support a consistent throughput at the assembly location; in a wider sense we can use the distinctions of the FBS to design workflows to support the global business process, including bidding for contracts, or accounting and financial processes.

The application of FBS to team formation reveals an immediate potential for integration of model-driven approaches with agent-based approaches: an agent representing a partner within a given VE offers one or more services; potentially, these services can be described using fragments of languages which already exist in the domain of workflow management, for example, in the XML-based *eXchangeable Routing Language* (XRL) [7].

Team formation is interwoven with the workflow design In this case, we use the formation of a team within the FBS framework. Here, the design artefact is a global workflow and the function is the realisation of global business process. With the function we associate a number of capabilities. When team formation is interwoven with the workflow design, we distinguish formulation and specification. We identify the capabilities needed as in the case when

team formation precedes workflow formation, using team structure to inform the workflow structure.

Each capability is realised by a particular resource through a set of behaviours, or tasks, which we associate with it. The explicit statement of these expected behaviours denotes specification. We examine the interdependencies of the resulting behaviours: for example, some behaviours, or tasks, must follow or require information from others, others are independent, some may be alternatives and still others need to be carried out a number of times to achieve the desired end. These interdependencies can be classified as *sequential, parallel, selective* and *iterative* [6]. For example, in the car assembly, the manufacture of an axle must precede its attachment to the chassis (sequential); whereas, the manufacture of a dashboard is independent of manufacturer a disc brake (parallel). Identifying the interdependencies of tasks and behaviours allows us to partially order them into a global structure: this is referred to as *routing* [6]. This forms part of synthesis of Fig. 1. We examine the global workflow, using suitable tools and techniques, to confirm that it achieves the desired top level goal, by comparing this analysis with our formulated expected behaviours (cf. analysis and evaluation in Fig. 1).

Again, the application of FBS reveals a potential for integration of model-driven approaches with agent-based approaches: the analysis of the workflows can be undertaken using specialised Petri Nets, referred to as *workflow nets* in [6]. Ideally, the predicted behaviours should be a safe superset of the expected behaviours: that is, there should be no undesirable additional behaviours deriving from the proposed workflow structure, for example, *deadlocks,* where a process reaches an impasse, or *livelocks,* where a process falls into an infinite loop [6]. If the analysis and evaluation reveal inadequacies in our proposed workflow structure, we use this information to revise our set of expected behaviours (cf. reformulation of Fig. 1).

We can interpret the global workflow structure developed as a syntactic structure which has as its semantics the realisation of the global business process (cf. [1]). This reveals another potential integration of agent based approaches and model-driven approaches: not only can the description of services offered by particular agents, or resources, be supported through the use of XRL [7], so can the sharing of any semantic information (electronically) by agents during this stage.

2.3 Product-Process-Team

In many domains knowledge about the final product to be realised through the cross-organisational collaboration provides information which is invaluable when it comes to identifying the dependencies of the activities of the partnership members and these in turn influence the structure of the team. For example, the structure of a car and in particular its composition from identifiable components imposes a sequence on the tasks in the assembly process. Also, each component must have been manufactured, quality tested and delivered prior to assembly. Moreover, the need for a given component or the need for a given skill in the assembly process identifies necessary expertise which must be embodied by some

partner(s) in the network producing the car. Formally, the semantics of the product informs the semantics of the process and the semantics of the process informs the semantics of the team. As an analogy we can think of a client-server relationship existing between the product and process; and also between the process and team. We reflect this client-server relationship by a directional arrow in (the UML diagram of) Fig 2.

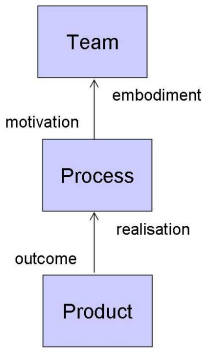

Fig. 2. Product-Process-Team Dependencies

Using this observation to inform the application of the FBS framework (discussed above) allows us to develop a model which clarifies the structures of teams and workflows and there interrelations. We illustrate this model in (the UML diagram of) Fig. 3.

3 Application of Agents

Using the model of the strategic level developed in the preceding section, we now examine (some of) the transformations among the categories distinguished above to identify potential opportunities for the application of agents. We are guided by the widely accepted definition of a (software) agent as a computer system capable of flexible, autonomous action which subsists within some environment. The capabilities of such a system were distilled into three expectations by Wooldridge and Jennings [8], namely:

- *Proactiveness* - an agent acts in a goal directed manner under its own initiative;
- *Reactivity* - an agent perceives its environment and responds appropriately, in a timely manner to changes to this; and
- *Social ability* - an agent communicates, as necessary, with other agents (including human agents) to achieve its goals.

In each case we consider the application type and the suitability of agents. We note that Wooldridge [9] presents the domain of workflow and business process

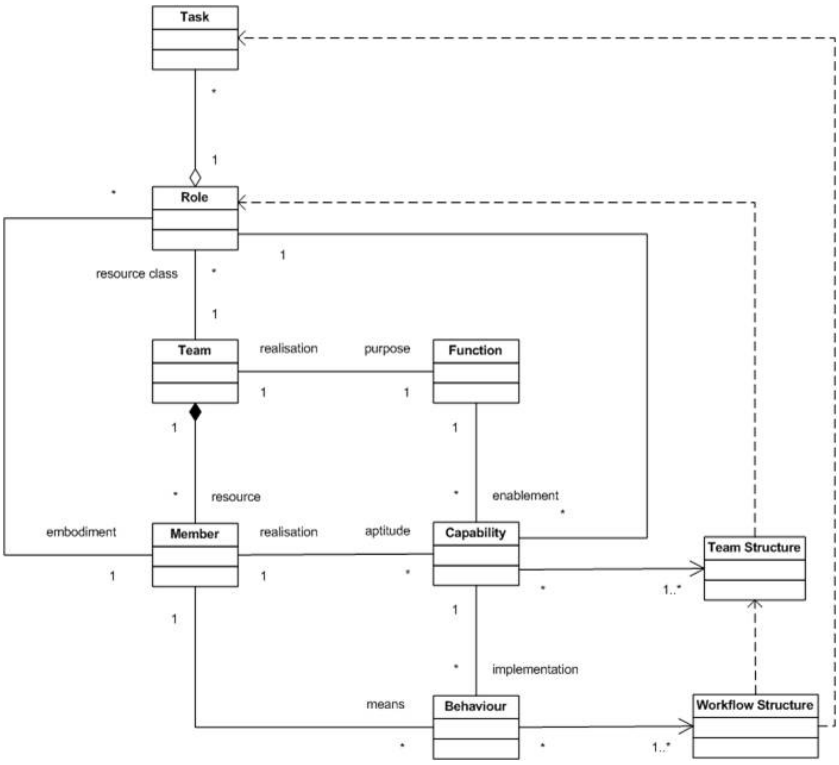

Fig. 3. Team and Workflow Structures

management as a notable application of multiagent technology (p. 245), including ADEPT [11, 10]. Of particular interest is the broad classification of applied agents into two groups by Wooldridge [9]:

– *Distributed systems* where agents become processing nodes in a network; and
– *Personal software assistants* where agents act as proactive, intelligent assistants to facilitate user tasks.

Representing each participant in a VE as an agent, or indeed, as a team of agents, corresponds to the notion of a distributed system above. In the each of the applications discussed below, we include a *coordinator.* This is a personal software assistant to facilitate the designer of the workflow.

3.1 Formulation and Specification

Formulation and specification are prime candidates for the introduction of personal software assistants, in particular, *a goal decomposition agent* to help the human designer. The goal decomposition agent may use one or more knowledge sources: for example, the appropriate agent has access to two knowledge sources,

one of which associates (sets of) function(s) and (sets of) capabilities, for example standard decompositions of global service(s) into component services, and one which associates capabilities with (sets of) behaviour(s), for example component services as a set of behaviours. Alternatively, it may be appropriate to consider more than one agent, each agent equipped with knowledge required to refine one level of abstraction, along the continuum from function to capability to behaviour, to a less abstract level, reflecting a recursive application of formulation. This decision depends on the application requirements and cannot be made in the general case.

In either case, a (possible) functional view of agents in formulation is illustrated in the use case diagram of Fig. 4.

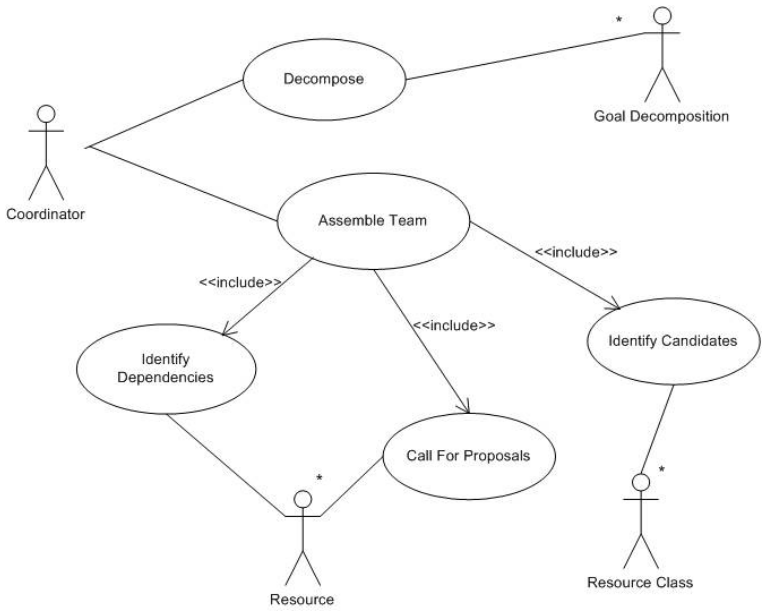

Fig. 4. Formulation in Workflow Design and Team Formation

The coordinator obtains from a (team of) goal decomposition agent(s) a set of services (capabilities) and tasks (behaviours) which are needed to fulfil the given top-level goal. If team structure is not predefined, then the coordinator can use the decomposition to assemble an appropriate team. This will involve making a call for proposals to relevant resources, identifying dependencies of tasks and resources and entering into negotiations. Depending upon the number of potential candidates, the coordinator might make use of one or more resource class agents to identify candidate team members before calling for proposals. A *resource class agent* would essentially wrap a library of resources, identifying

a number of resource classes, agents belonging to these and the appropriate addresses, for example, in the form of Universal Resource Locators.

In Table 1 we consider the three characteristics identified in the introductory remarks at the beginning of this section, cf. [8] for each actor in use case diagram of Fig 4. In particular, a tick in a cell indicates that the given characteristic (column) is definitely demonstrated by the appropriate actor (row); a cross indicates that the given characteristic is not shown; and where the cell is left blank, we are noncommittal. For example, for the goal decomposition agent, we are noncommittal about whether it requires reactivity or social ability to fulfil its given function. There might be a team of goal decomposition agents which work together and in such a case, reactivity and social ability are certainly exhibited. We consider the positive presence of at least one of these characteristics, as shown through a tick, to be sufficient motivation to explore the possibility of agents for the given function fulfilled by the actor. In the use cases presented, the resource class agent is an interesting actor. We have anticipated in the name that it will be an agent, but since in a general sense none of the characteristics is exhibited and we can clearly infer that no reactivity is needed, it is fair to propose that this agent be replaced by some other device, for example a look up table. In such a case, the device could be wrapped using by agent to endow it with communication abilities and knowledge of what service it provides; and thus allow it to participate in the agent based system, rather than functioning as an isolated set of methods.

Table 1. Actors in Formulation in Workflow Design and Team Formation

Actor	Use Case	Proactivity	Reactivity	Social Ability
Coordinator	Decompose Assemble team	✓	✓	✓
Goal Decomposition	Decompose	✓		
Resource	Identify Dependencies, Call for proposals			✓
Resource Class	Identify Candidates		×	

3.2 Synthesis

Synthesis is also an area where we could consider personal software assistants. For example, in constructing the global workflow, we might make use of a *routing agent.* The routing agent contributes by determining a partial ordering of the appropriate tasks, according to the dependencies established as a consequence of consultation and negotiation between resources and the coordinator; thus, removing the need for this to be done manually. Technically, synthesis involves identifying structural elements which exhibit one or more of the desired behaviours; discovering the dependencies among these; composing these structural

elements into a global structure which ideally retains the necessary behaviours. Naturally, determining the dependencies among structural elements is a non-trivial, knowledge-based task and we should note that these dependencies are not merely temporal. Synthesis can be informed by existing work, for example, process views [12] admit of safe abstractions of local workflows, removing detail irrelevant to the global perspective where synthesis occurs; again providing for integration of model-based and agent-based approaches. A (possible) functional view of synthesis is illustrated in Fig. 5. The use case *Establish Dependencies* parallels the use case *Identify Dependencies* of Fig. 4 However, the focus in the former is at the task level with a view to determining sequence, parallelisation, selection and iteration of these tasks, whereas in the latter, the focus is on which tasks and process requires inputs and interactions with other with a view to establishing a team structure.

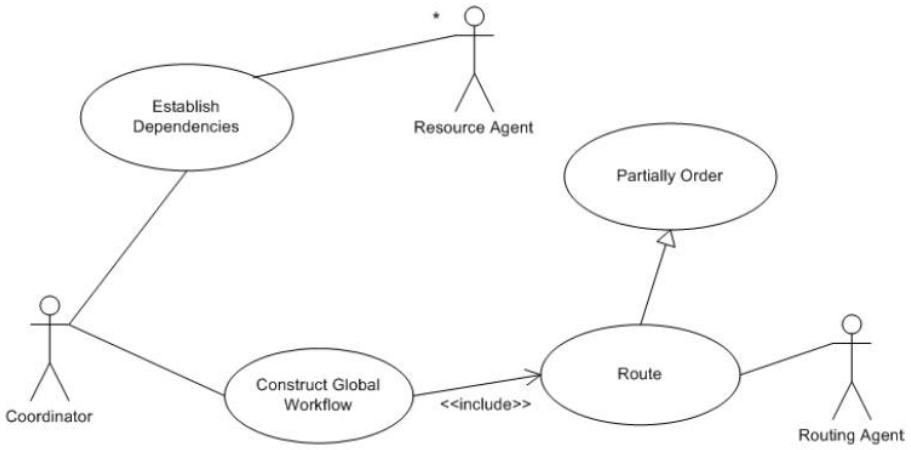

Fig. 5. Synthesis of Global Workflows

As before, in Table 2 we consider the three characteristics—proactiveness, reactivity and social ability—for each actor. Of particular interest here is the *routing agent.* Strictly speaking, to fulfil its function it does not require reactivity nor social ability, since we assume that the routing is undertaken once the dependencies among the resources are know. Thus, there is a case for having a "non-agent" to fulfill the routing function, for example, an appropriate automated tool from model-based approach to workflow construction. Again, the device could be wrapped using by agent to endow it with communication abilities and knowledge of what service it provides. Moreover, here the wrapping would provide a systematic way in which to integrate model-based tool into an agent-based system.

Table 2. Actors in Synthesis of Global Workflows

Actor	Use Case	Proactivity	Reactivity	Social Ability
Coordinator	Establish Dependencies, Construct Global Workflow	✓	✓	✓
Resource Agent	Establish Dependencies	✓		✓
Routing Agent	Route	✓	×	×

3.3 Analysis and Evaluation

There exist a number of tools for analysing workflows, e.g. [7]. An agent which is able to interpret the results of the analyses and use these in evaluation of workflows is a potential application for a personal software assistant, with a capacity to compare of the predicted behaviours with the expected behaviours. Where the comparison indicates inadequacies in the global workflow, the evaluator can communicate this information to those agents which participate in reformulation.

It might be convenient to have such a *workflow analyser* work in conjunction with a *well-formedness agent* which can ensure the validity of formal representations, such as XRL, which are used to represent the workflow. We illustrate such a scenario in the use cases of Fig. 6. Evaluation is often used to select among candidate proposals from project partners: the best chosen based upon the information obtained from analysis and comparison with appropriate ranking criteria.

Again, in Table 3 we consider proactiveness, for each actor in use case diagram of Fig. 6. As for the routing agent of the previous subsection, there is a case for having a non-agent to fulfill the analysis and evaluation functions; for example, appropriate automated tools from model-based approaches. These could be wrapped using by agents provide communication abilities and knowledge of what services provided, as a way in which to systematically integrate model-based techniques into an agent based system.

3.4 Reformulation

We treat reformulation as a revision of the formulation and synthesis stages, taking account of the additional information obtained through analysis and evaluation. For example, preferring a particular workflow from a number of alternatives based upon analysis and evaluation might lead to a reallocation of services or a re-ordering of tasks. The agents responsible for formulation and synthesis assume responsibility for reformulations.

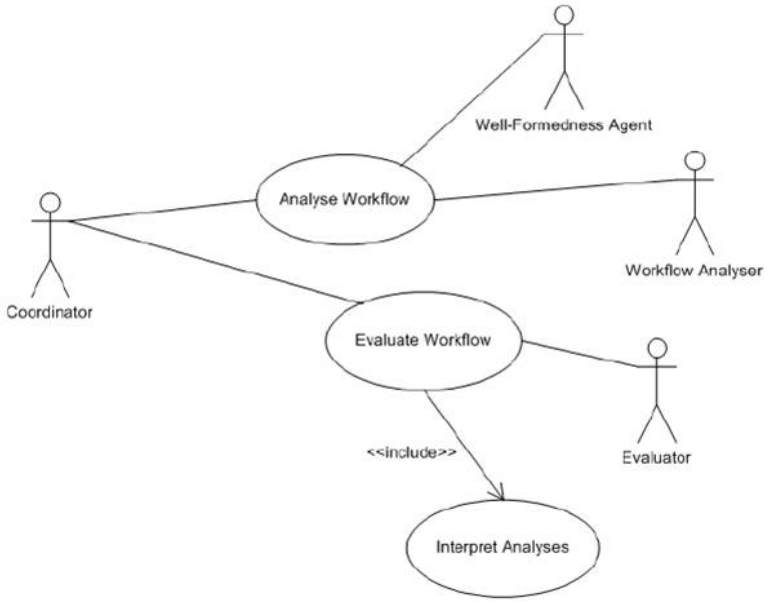

Fig. 6. Analysis and Evaluation of Global Workflows

Table 3. Actors in Analysis and Evaluation Global Workflows

Actor	Use Case	Proactivity	Reactivity	Social Ability
Coordinator	Analyse Workflow, Evaluate Workflow	✓	✓	✓
Well-formedness Agent	Analyse Workflow	✓		✓
Workflow Analyser	Analyse Workflow		×	×
Evaluator	Evaluate Workflow		×	×

4 Semantic Interoperability through Devolved Ontology

While multiagent systems offer much to foster the open nature of agile partnerships, for example, *ad hoc* interaction with new arrivals is supported through agent communication languages and standardised exchanges of messages for common conversation patterns (*interaction protocols*), there are limitations. To communicate effectively our agents must use a common vocabulary and agree on the semantics of the terms in it. Typically, communication in multiagent systems presupposes a common ontology[2] which is fixed in both content and semantics. Yet, the very nature of agile partnerships suggests neither a fixed ontology nor a unique semantics is appropriate. A fixed ontology precludes the enrichment of the partnership with a new range of services: the necessary concepts to describe this are typically not contained in the ontology. Such enrichments often arise within the partnership, when an existing partner introduces a new service; and are concomitant with the arrival of a new partner. The diversity of backgrounds means that partners often interpret the same term differently. Formally, each maintains a different view of the same concept and these may be inconsistent or even contradictory. As such the semantics of a given term is fluid within a partnership. More interestingly, a partner can learn the view of another, entertaining simultaneously a number of views of a single concept and choosing the most appropriate according to context.

We have devised a model of an evolvable, devolved ontology to promote *ad hoc* semantic interoperability in open environments and agile partnerships. We have developed a formal framework for devolved ontologies and use this to address novel concepts arising within an agile partnership. Technical details of this are not presented here; see, for example [2]. This is based on the use Formal Concept Analysis [4], lattice theory [3] and aspects of *Partially Shared Views* (PSV) [13].

Informally, a devolved ontology comprises of a core ontology and a number of peripheral ontologies. The core ontology provides a common ground for understanding among partners and is central to the partnership. The concepts included within this are agreed through negotiation of all partners. As such, the responsibility for the evolution and maintenance of the core is shared by the partners. Each peripheral ontology represents an extension of the core ontology into an application domain. The responsibility for the evolution and maintenance of each peripheral ontology *devolves* upon the appropriate partner or partners. This includes the responsibility for extending the core into the particular context and ensuring that the peripheral ontology remains consistent with the core. Moreover, we recognise the existence of interapplication domains or ontologies. For example, two partners may share a number of concepts which are not part of the core. The responsibility for the initial extension of the core into the interapplication ontology devolves upon two agents jointly; the responsibility for the further extension devolving onto the appropriate single agent. Thus, a devolved

[2] An *ontology* is a formal (partial) account of a domain of interest: this can be used to realise a controlled vocabulary to support communication.

ontology exhibits a recursive structure. Agents entering into an (independent) agreement prior to the formation of the partnership should strictly be treated as a federation. For the purposes of the negotiation of the partnership core the federation is essentially a single agent.

Since the participants in agile partnerships are fundamentally self-interested, we supplement our framework with appropriate mechanisms to capture this. We make use of the theory of utility to develop negotiation protocols [2].

5 Discussion

Beginning with the FBS framework of Gero [1], we have developed a model to clarify the strategic aspects of team and workflow formation in agile partnerships. Our aim here was to present a framework which can be used systematically to identify potential opportunities for the application of agent technology to foster cross-organisational (business or production) process interoperability.

We have shown how we can use this framework systematically to identify where agents might be applied. In particular, we have clarified the decision space for cross-organisational process and workflow design and construction, distinguishing a number of categories of knowledge and the transformations between these categories. We have used this separation of domain knowledge from the computational processes which operate upon it to highlight where agents may be applied and their contributions to the design and construction of decentralised workflows as means of promoting cross-organisational process interoperability. We have used this to propose elements of a generic framework which could be specialised to a particular application according to end-user requirements. Elements of the above model were used in CrossWork project [19] to inform the decision of where to apply agents in a set of concrete usage scenarios: for example, agents were used in team formation, see [20]. The approach also partly-informed software architectural design decisions in the ongoing European Project SUDDEN (`www.sudden.org.uk`).

We reiterate that here we are using FBS as an informing framework rather than proposing it as an alternative to existing approaches to the development of agent-based systems. We feel it important to clarify this as there exist a number of well-constructed approaches, which we use with our framework; in particular, *Aalaadin* [16] and *Gaia* [17]. Moreover, we are fully aware that work in this area continues, for example, recent developments include the model-based methodology of Jarraya and Guessoum [15]. Additionally, we are not proposing the approach as a way in which to manage, especially at an operational level, cross-organisational workflows. Again, we have used the framework with existing approaches, for example, in CrossWork we appealled to the three-tier model developed in the CrossFlow project [24].

We emphasise that the core contribution of the work expounded here is a systematic way in which to inform the decision of where and whether to use agents in coordination of decentralised, cross-organisational (business and to some extent) processes. We have proposed elements of a generic framework which could

be specialised to a particular application according to end-user requirements. Indeed, applying the proposed elements to as many concrete usage scenarios as possible, offers a means, perhaps, the only means of evaluating our work in this area. It is a systematic approach to inform decisions on the use of agents and as such must be evaluated on its utility in comparison with the typically *ad hoc* approaches that we have observed. Such an effort forms part of our ongoing work. Nonetheless, using the framework to *compare* the use of agents with existing approaches is an interesting line of research that will form part of our future work.

6 Conclusion

Approaches to business processes and the intimately related workflow systems have been traditionally based on two alternative approaches: model-driven, e.g. [6], in which a predefined (workflow) model, guides the execution of a global business process; and agent-based (e.g. [10]), in which a workflow emerges from the interaction of agents associated with different participants in the global business process. Each alternative has particular strengths and weaknesses; and while there have been (early) attempts to obtain the "best of both worlds" by combining the two, cf. [14], which typically, have followed *ad hoc* conceptualisations of the process of workflow design and management. An informed use of agents in cross-organisational (business) processes demands a more systematic approach. We propose function-behaviour-structure [1] as an organising schema to structure the process of workflow design and construction, which then helps us to identify the roles of software agents in the workflow design process. The FBS is particularly appealing as an organising schema because:

- *It is noncommittal about the expression of the categories of knowledge.* For example, in the case where the design artefact is actually a system, our initial function set might be expressed as a set of top-level use cases which we progressively refine; alternatively, our function might be a set of data flow diagrams. Behaviours could be expressed through state diagrams or activity diagrams, etc. In the case of workflows, our structure might simply be a Petri net.
- *It provides a starting point for identifying and classifying relevant domain knowledge, a useful prelude to developing an ontology for the domain(s) of interest.* For example, using an object-oriented approach, we identify categories of objects some of which are simply items which will be manipulated, rather than having any responsibilities, and typically, these can be associated with concepts of our domain of interest.
- *It does not prescribe the form of the operations on or transformations between the categories of knowledge.* This allows for the inclusion within it appropriate models of these transformations. For example, we might map from function to behaviours through a goal to task decomposition; or, if appropriate, we may appeal to a data-centric approach such as Jackson Structured Design; or, of course, agent based approaches.

The use of agents to support cross-organisational process interoperability reduces the burden on the designer, for example, by (partially) automating negotiation between and coordination of the participants in an agile partnership. However, an agent-based approach, like any automated approach, not only removes the onus from the user, it also removes a certain amount of control. Thus, we need to ensure that we promote an interactivity at each stage. For example, we could use the coordinator to feed back some of the design choices for confirmation by the designer and allowing the him to supervise the system, thus returning control without the burden.

References

1. Gero, JS, 'Design prototypes: A knowledge representation schema for design', *AI Magazine*, Winter: 26-36, 1990.
2. Stalker, ID and Mehandiev, ND, 'A Devolved Ontology Model for the Pragmatic Web', in M Schoop, A de Moor and J Dietz (Editors), *Proceedings of the First International Pragmatic Web Conference (PragWeb06)*, Lecture Notes in Informatics, 89, 2006.
3. Birkhoff, G, *Lattice Theory (3rd Ed.)*. AMS Colloquium Publ., Providence, RI (1967)
4. Ganter, B and Wille, R, *Formal Concept Analysis. Mathematical Foundations.* Springer-Verlag (1999).
5. Stalker, ID and Mehandjiev, ND, 'Agents for Cross-Organisational Business Interoperability'. In *Proceedings of ATOP 2005, Workshop of AAMAS 2005.* July 2005.
6. van der Aalst, W and van Hee, K, *Workflow Management: Models, Methods, and Systems*, Cooperative Information Systems, The MIT Press, 2002.
7. van der Aalst, WMP, Verbeek, HMW and Kumar, A, 'XRL/Woflan: Verification of an XML/Petri-net based language for inter-organizational workflows,' In K. Altinkemer and K. Chari, editors, *Proceedings of the 6th Informs Conference on Information Systems and Technology (CIST-2001)*, 30-45, Informs, Linthicum, MD, 2001.
8. Wooldridge, M, and Jennings, N R, 'Intelligent agents: theory and practice,' *The Knowledge Engineering Review.* **10**(2), 115-152, 1995.
9. Wooldridge, M, *An Introduction to Multiagent Systems*, Wiley & Sons Ltd, 2002.
10. Jennings, N R, Faratin, P, Norman, T J , O'Brien, P, Wiegand, M E, Voudouris, C, Alty, J L, Miah, T, and Mamdani, E H, 'ADEPT: Managing Business Processes using Intelligent Agents,' In *Proceedings of BCS Expert Systems 96 Conference (Intelligent Systems Integration Programme Track),* 5-23, Cambridge, UK, 1996.
11. Jennings, NR, Faratin, P, Norman, TJ, O'Brien, P, Odgers, B and Alty, JL, 'Implementing a Business Process Management System using ADEPT: A Real-World Case Study', in *Int. Journal of Applied Artificial Intelligence*, **14**(5), 421-465, 2000.
12. Liu, D-R and Shen, M, 'Workflow modeling for virtual processes: an order-preserving process-view approach.' *Information Systems.* **28**(1), 2003.
13. Lee, J and Malone, T W, 'Partially shared views: A scheme for communicating among groups that use different type hierarchies,' *ACM Transactions on Information Systems*, **8**(1), 1990.
14. Myers, KL and Berry, PM, *Workflow Management Systems: An AI Perspective.* Technical Report, Artificial Intelligence Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Jan 1999.

15. Jarraya, T and Guessoum, 'Towards a Model Driven Process for Multi-Agent System', *Multi-Agent Systems and Applications V,* Lecture Notes in Computer Science 4696, Springer, 2007.
16. Ferber, J and Gutknecht, O, 'Aalaadin: a meta-model for the analysis and design of organizations in multi-agent systems', in proceedings of *International Conference on Multi-Agent Systems (ICMAS 98)*, July 1998.
17. Zambonelli, F, Jennings, N R and Wooldridge, M, 'Developing Multiagent Systems: The Gaia Methodology,' In *ACM Transactions on Software Engineering Methodology,* **12**(3), 317-370, July 2003.
18. Kollingbaum, MJ and Norman, TJ, 'NoA - A Normative Agent Architecture'. *Proceedings of IJCAI 2003,*, 1465-1466, 2003.
19. Grefen, P, Eshuis, R, Mehandjiev, N, Kouvas, G, Weichhart, G, 'CrossWork: Internet-Based Support for Process-Oriented Instant Virtual Enterprises,' *IEEE Internet Computing*, 2009, to appear.
20. Stalker, ID, Carpenter, M and Mehandjiev, ND, 'Establishing Agile Partnerships in Open Environments: Extended Abstract', *OTM Conferences 2006*, Lecture Notes in Computer Science, LNCS 4277, Springer-Verlag, 15-16, 2006.
21. Huhns, MN and Singh, MP, 'Managing Heterogeneous Transaction Workflows with Co-operating Agents', *Agent Technology: Foundations, Applications and Markets*, N Jennings and M Wooldridge (Eds), Springer, 1998.
22. Navinchandra, D, *Exploration and Innovation in Design: Towards a Computational Model*, Springer-Verlag, 1991.
23. Simon, HA, 'The structure of ill-structured problems', *Artificial Intelligence*, **4**, 181-201, 1973.
24. Grefen, P, Aberer, K, Hoffner, Y and Ludwig, H; CrossFlow: Cross-Organizational Workflow Management in Dynamic Virtual Enterprises, *Computer Systems Science & Engineering*, **15**(5), CRL Publishing, 277-290, 2000.