Business Contract Driven Application Development and Control

Jean Bacon and Ken Moody
University of Cambridge Computer Laboratory, October 2003

Part 1: Previous research and track record

Background to this proposal
This is a resubmission of our proposal of 2002 which was then for a postdoc for Alan Abrahams for October 2002 onwards. Two reviewers were highly positive and the third commended the proposal but was of the opinion that the risk was too high because computer scientists have insufficient knowledge of business law to contribute to this area; the proposal was not funded. Alan has a first class honours degree in business science with a first class minor in business law. He has also spent two periods totalling six months at the University of Pennsylvania’s Wharton Business School, a leading business school in the US, on research visits with Professor Steven Kimbrough, a recognized expert in formal specification of business contracts.

Alan has continued his thesis work here in the area of the proposal after obtaining his PhD in October 2002, funded by Microsoft Research. He has now accepted a faculty post at Wharton UPenn from January 2004 and will continue this research there. During his recent visit he spent time with Professor Insup Lee of the High Assurance, Systems, Tools, and Environments (HASTEN) group at Penn’s Department of Computer and Information Science (CIS). This led to a bloodbank database application for Alan’s work, with US Federal Regulations as the contractual specification. This will be used as one case study, with an e-commerce example as another, as described in Part 2. Professor Kimbrough is strongly in favour of our continued collaboration, please see attached letter.

While a PhD student and postdoc here Alan produced first EDEE then CamPACE (Policy Analysis, Checking, and Execution Environment), available: http://www.cl.cam.ac.uk/users/asa28/CamPACE Install. CamPACE is an extension of the initial EDEE prototype described in his thesis which provides an enhanced user interface and faster consistency checking engine. This has been achieved through his work with David Eyers, now a final year graduate student, leading to several publications.

David Eyers completed his double degree (BE/BSc) in computer engineering and pure maths with first class honours at the University of New South Wales in Sydney, Australia. He taught and researched within the Database Group there on multimedia and XML database technology before obtaining a Cambridge Australia Trust Scholarship to support his PhD.

This proposal is for a postdoctoral Research Associate post for David Eyers to work in the Computer Laboratory in collaboration with Alan Abrahams at the Wharton Business School. Wharton’s eminence in business law and the Computer Laboratory’s in system design together create a unique environment for this research and a great opportunity for substantial progress to be made in this area.

Previous research: Recent research grants in this area
Jean Bacon and Ken Moody lead the Opera research group at the University of Cambridge Computer Laboratory which focuses on the design and deployment of open, large-scale, widely distributed systems. Two major thrusts are in the areas of asynchronous middleware (the Cambridge Event Architecture (CEA) and Hermes) and an open access control architecture for secure interworking services (OASIS RBAC: role-based access control). We have extended these areas to include access control policy expression, enforcement and management and trust and risk in global computing.

Recent grants that have motivated this proposal are:
(2) OASIS Access Control, PhD studentship, EPSRC July 1999 - June 2002
(3) Travel and subsistence support to advise Glasgow Caledonian and Strathclyde on using OASIS for electronic courseware delivery, EPSRC Jan 2002 - Dec 2003.

The Opera group were early to incorporate event handling into standard middleware. The application areas for this work included multimedia presentation management and tracking mobile entities in sensor-rich environments, including its augmented, virtual representation. Under (1) we added event storage to the architecture, used ODL for event typing and OQL for querying event stores. The software is available from our web pages, http://www.cl.cam.ac.uk/Research/SRG/opera/projects/index.html. We federated heterogeneous event systems (e.g. CORBA and Java-based) using gateways and contracts. Here, automatically generated XML was used as the inter-domain transfer syntax. CEA is appropriate for tightly-coupled sources and sinks with static dependencies such as in an active building. In Hermes we have explored loosely
coupled event-based communication, appropriate for large-scale, widely distributed systems. Hermes is a publish/subscribe system built above a peer-to-peer routing network. Filtering is distributed across a network of event brokers to minimise the communication overhead of event propagation; advertisements and subscriptions meet at distributed rendezvous nodes. Early ideas on how to approach event composition were presented at Middleware03 and achieved the best paper award. In this proposal we are using the event-driven paradigm in the form of an active database which embodies the contractual control of an application.

The starting point for OASIS RBAC was interworking between independently developed services in different administrative domains. Our Addenbrookes colleagues in the Clinical and Biomedical Computing Unit (CBCU), through their company CBCL, have used OASIS access control in their pilot software for electronic health record (EHR) management within the UK National Health Service. As part of his thesis research, David Eyers has worked with CBCL to reimplement OASIS using Enterprise JavaBeans technology. He has built a demonstration infrastructure using a web portal and SSL for initial authentication and encryption. In (3) he has successfully exported this technology to Strathclyde and evaluation is about to take place. In (4) a graduate student has worked on access control policy management for OASIS, building a policy store as an active database over Postgres. We shall build on this experience to incorporate existing distributed database best practice for contract-driven application control.

Our access control policy takes the form of Horn clause logic expressed in XML. It is therefore easy and efficient to enforce automatically. More complex policy must be expressed in order to capture contractual control of applications. Our aim is to minimise the need for human intervention and to exploit active database technology for maximum efficiency.

**Track record: Research supervision and dissemination**

In addition to grant-focussed research, we have supervised graduate students in distributed systems, leading to 28 PhD awards. We have examined 23 other internal and 20 external (including international) PhDs in related topics. Our recent graduate students have employment which includes: Citrix Research UK, AT&T Research UK, University Lecturers in Cambridge, Israel, Singapore and Egypt, Bellcore US, Cisco US, Fore UK, FutureTV Cambridge, Microsoft Research Cambridge, Cornell University.

**Track record: Journals, programme committees, panels**

Jean Bacon was Editor in Chief of IEEE Concurrency during 1999 and 2000. From mid 2000 she has been EIC of the IEEE Computer Society’s first online magazine, Distributed Systems Online, http://computer.org/dsonline/. She was elected to the IEEE-CS Board of Governors for 2002-2004. Our recent conference participation and panels include:

- ACM SIGOPS European Workshop 1996, Programme Chair 1998 (Dr Bacon)
- EDBT and BNCOD (many times, up to and including 2003) (Dr Moody)
- IEEE Services in Distributed Networked Environments 1995,6 (Dr Bacon)
- IEEE International Conference on Distributed Computing Systems (Dr Bacon) vicechair Middleware 2004
- IFIP 8.1 Information Systems in the WWW Environment 1998 (Dr Moody)
- ACM Symposium on Operating Systems Principles, 1999 (Dr Bacon)
- ACM SACMAT Symposium on Access Control Models and Technologies, 2002,3,4 (Dr Bacon)
- IEEE Data Engineering 2001 (Dr Bacon)
- Proposer and organiser of ICDCS workshop on event-based systems, Vienna, July 2002 (Dr Bacon)
- 16th IFIP WG11.3 Conference on Data and Application Security. Cambridge, July, 2002 (Dr Moody)
- VLDB-TES, Technologies for E-Services, PC 2002, 2003 (Dr Bacon)
- Percom (Pervasive Computing) PC 2003, 2004 (Dr Bacon)
- Distributed Objects and Applications (DOA 2003) (Dr Bacon)
- ESORICS 2003, 8th European Symposium on Research in Computer Security (Dr Moody)
- EU project CORTEX (sentient computing) 2001–4 reviewer (Dr Bacon)
- Distinguished dissertation panel (Dr Moody) and reviewers

**Track record: Publications**

Because of space constraints, please see our publications on:
http://www.cl.cam.ac.uk/Research/SRG/opera/publications/index.html

The page includes summaries by topic of our publications on business contracts, access control, event-based middleware and policy, as well as details of publications by year and PhD theses.
Part 2: The proposed research

A Motivation and Background

Contracts are central to the operation of commercial organisations in that they constrain and direct the behaviour of a company and its agents. We propose to make contracts central to the development and control of e-commerce applications. To this end, we shall represent contractual provisions in a form suitable for machine interpretation. Contracts may then be queried, executed and monitored automatically; that is, we shall provide contract-driven enforcement. Our contracts are not static but are subject to change and this change must be managed. We propose to develop applications by ensuring that participants adhere to the provisions of the contracts that bind them. This is a novel approach which will integrate the specification of an e-business application with the implementation of its code base.

Managing contracts effectively requires a powerful semantic model and a generic storage framework. If contracts are to be enforced automatically then the representation must capture the relevant semantics in full. Which contract provisions are applicable must be determined in rapidly changing circumstances and in the light of frequent alterations and additions to the contracts themselves. Conflict resolution facilities must be capable of highlighting mutually exclusive provisions and deciding between them. Schedulers must enact obligations. Monitors must flag violations. Change management facilities must maintain a history of the nature and status of the organisation’s past contracts in order to resolve any disputes that may arise.

Background and Related Work

At present, business policies are typically recorded in multiple hard-copy contracts with customers, suppliers, employees and partner organisations. Internal contracts are captured in procedure manuals and user requirement specifications. In e-business the ideal is that these policies are captured precisely by the implementation code-base of the company. The difficulty is that the semantic form of business contracts is lost during the translation to conventional software. Instead, the progress of business should be governed by monitoring what takes place against a precise representation of a continuously changing set of contractual provisions. In object-oriented and procedural software, methods must be invoked in a hard-coded sequence. The introduction of new contractual provisions requires a tedious process of search and inject, in which appropriate method-invocations must be manually inserted into all relevant invocation sequences. We propose instead to develop an event-driven execution model. Intuitively, we regard contractual provisions as patterns with associations, such as obligations; whenever an occurrence takes place we determine which patterns match. We represent provisions as queries which are recorded within the generic storage framework, and are themselves subject to interrogation.

The semantics of obligations, powers and authority are lost in sequential invocation paradigms. It is therefore not possible automatically to assess the mutual obligations of participants who are subject to contract. Conventional Interface Definition Languages (IDLs) define available method names, and their argument and return types, but do not provide a way to specify pre- and post-conditions for method invocation. This absence of semantics makes it impossible to relate program state to the business process. We shall design a contract definition language to express the fundamental legal concepts that arise in corporate environments. This will make it possible to represent contracts directly, so allowing real and hypothetical business activities to be modelled and monitored.

Our recent work has addressed access control policy management and event monitoring as described in Part 1 above. We intend to pursue extensions to these paradigms in order to monitor and control interaction in commercial settings. As scenarios evolve participants are bound by a changing set of legal relations, in accordance with specified contractual terms and conditions.

The relationship of our work to other state-of-the-art research in event monitoring, workflow, policy management, contract evaluation and contract architectures is as follows:

1. Traditional event monitors [2, 1, 20, 15] match against a rapid stream of transient events. Event information is discarded according to one of four consumption policies which choose which of a time-limited set of buffered events to use when matching a given subscription; typically the choice of policy is hard-coded in each application. Once an event has been matched it is consumed and cannot be used in further matching. This is ill-suited to commercial applications in which event consumption is determined on the fly by the business context, so that the record of any occurrence must be persistent.

2. In conventional implementations of workflow the notion of contracts is ignored altogether. Constructs generally follow those provided by the Process Specification Language (PSL) [21] and the Workflow Management Coalition’s (WfMC) [23] reference model (which is, for instance, adopted by the OMG’s Workflow
Management Facility [22]): workflow specification is based on process synchronisation using \texttt{and/or -split}
and \texttt{-join} primitives [13], without regard for legal constraints such as rights and powers of parties. Several
workflow management systems, such as SAP R/3 and BaanERP base their modeling language on Petri Nets [24]. Here
synchronisation is by the use of tokens that move between locations when transition conditions are satisfied. Lee [18]
proposes to extend Petri Nets with deontic (legal) states of affairs brought about at each location, but targets the approach
towards modeling document and communication flows and their legal effects, rather than executing contracts. Further, the approach suffers from the drawback that Petri Nets are stateless by default, and so maintain no occurrence history to be queried; this makes it impossible to assess the legal implications of past actions retrospectively.

3. Work in the field of \textit{policy management}, such as Imperial College’s Ponder [9], Bell Lab’s Policy
Description Language (PDL)\textsuperscript{1} [14, 4], and Koch’s Policy Definition Language (PDL) [11], incorporates event
monitoring and policy triggering mechanisms broadly similar to our work. However, these policy management
systems are targeted at low-level network management and are ill-suited to e-commerce applications. Stream
semantics, where events are short-lived and rapidly consumed, is assumed by the event monitors in both Ponder and Koch’s PDL\textsuperscript{2}. These engines are therefore inappropriate for e-commerce applications which require persistent event histories. In papers describing extensions to Bell’s PDL [14, 4] the authors argue for history-based policies in which epochs of events are maintained, but no implementation is described. We have successfully prototyped such functionality in \textit{EDEE} and \textit{CamPACE}.

Koch’s PDL and Ponder are targeted at specifying low-level policies only and cannot represent high-level
goals: \textit{actions} can be obliged, but not \textit{states of affairs}. As Cole et al. point out [3], this is inadequate when the
states of affairs to be brought about or avoided are known, but the actions which would bring about such
states of affairs are unclear. \textit{EDEE}’s approach is based on \textit{occurrences}, which provide a uniform treatment of

4. Investigation into \textit{contract evaluation} has been undertaken by Daskalopulu et al. and Peyton Jones et
al., and work in \textit{contract architectures} includes COSMOS, Milosevic et al.’s BCA, and a conceptual model
from Hewlett Packard Laboratories.

Recent theoretical work by Daskalopulu et al. attempts to assess contract status, and implications of
eventualities, using a finite state machine (FSM) approach [7, 8]. Such an approach can only be appropriate
if the reduction to a particular FSM will remain valid for all time, which is unrealistic in the volatile world
of business contracts. In order to assess contract status properly it is essential to maintain a complete and
persistent occurrence history.

Peyton Jones et al. [19] use the functional language, Haskell, to represent a small class of contracts -
financial option contracts where performance is measured by cash flows only. There is no representation of
obligations, and the representation is used for option description and valuation, rather than contract
execution.

The COSMOS Workflow Engine [12] invokes functions in accordance with temporal constraints extracted
from contracts. COSMOS assumes conflict-free specifications and can reason neither about conflicting obliga-
tions, nor about the powers of parties. It has been criticized for ignoring the possibility of deviation from expected behaviour and its inability to reason about the consequences of violation (such as secondary
obligations coming into force) [8].

Milosevic et al.’s Business Contract Architecture (BCA) [16] assumes that contracts are provided a
priori; additional terms cannot be introduced subsequently and the architecture is therefore unable to cater
for dynamically changing business and regulatory environments. SeCo [10] bases its architecture on the
BCA, and suffers similar drawbacks. SeCo’s monitoring service monitors and logs negotiation-phase events
only, not settlement-phase events. Negotiation-phase events are logged to the XML container which stores the
parties, product descriptions, and payment and delivery conditions for the contract. SeCo does not implement an enforcement service, nor interoperation with payment and logistics services. In order to attain
those goals SeCo would need to encode product description and payment and delivery condition semantics
more formally.

Hewlett Packard Laboratories [17] present work-in-progress towards a high-level architecture for regulat-
ing electronic marketplaces using contracts embodied in XML. The authors propose a contract repository,
validator, monitor and evidence store, which is similar in structure to the BCA. The conceptual model
remains to be implemented.

We have already built a prototype implementation \textit{EDEE} based on persistent occurrence histories and

\textsuperscript{1}Unlike Ponder and Koch’s PDL, Bell Lab’s PDL does not provide for prohibition policies.

\textsuperscript{2}Ponder’s monitor evolved from their Generalized Event Monitor (GEM) [15]. Koch’s monitor was also inspired by GEM.
incremental continuous query evaluation. Our model addresses all of the issues identified above, and we can reason about the applicability and fulfillment of obligations in a world of evolving contracts. Further, we have already shown that the primitive elements of a formal contract definition can be exposed by a human analyst from an English language requirements specification using a set of analytic guidelines. This integration between the analysis and implementation phases of e-business development has not been addressed by other contract architectures, which focus on technical facilities without regard for requirements analysis and formalisation.

B Programme and methodology
The various phases of the research programme will implement real contracts. We will start with a blood-bank example taken from Alan Abrahams’ internships at Wharton, and a FTSE 100 e-commerce company, described in section C.

B.1 The EDEE contract enforcement prototype
EDEE was the prototype system for representing and enforcing business contracts developed by Alan Abrahams during his work towards a PhD. The expressive power is that of a semantic network. Networks are stored by reducing a general graph structure to a set of triples (participant, named-and-typed-association, role). Examples of association types are occurrence, obligation, permission, function and so on. The triple store holds the complete representation of the business application, and is updated whenever relevant occurrences take place; such occurrences may include both changes to contractual provisions and transactions such as the payment of a bill or the delivery of an order.

EDEE could express and represent patterns which could be matched in any situation against the contents of the triple store. Patterns are implemented by queries; these queries can themselves be stored. In any situation each query will yield a set of results; in particular, a query that tests occurrences defines a set of occurrences. Contracts are represented by establishing their structure and defining queries to correspond to their provisions.

The EDEE prototype used an Access database wrapped with active database-style triggers. When an occurrence takes place those queries that may be relevant are evaluated in order to determine which contract provisions cover it. In this way all contracts are actively monitored so that business policy can be enforced. Query evaluation is an inefficient process, and the EDEE implementation did not scale to the activities of a large organisation. EDEE has been reimplemented in the past year as CamPACE (Policy Analysis, Checking, and Execution Environment), available at http://www.cl.cam.ac.uk/users/asa28/CamPACE_Install. With David Eyers’ input, significant performance improvement has been achieved through replacing Java by Prolog in the rule engine, yet a triple store and wrapping are still used. We see the potential for further improvements that will allow the system to scale to large applications.

B.2 Extending CamPACE functionality
Contract definition and storage. A contract definition language will be provided to allow business developers to input and store contract specifications. A contractual provision mandates particular actions or construals upon the occurrence of a certain pattern of events or states. The mechanisms prototyped in EDEE can be exploited to store contracts and to determine when provisions apply and when a contract is fulfilled. In addition a simple contract presentation language is required in order to express machine-parsable contracts in a form that humans can read.

Assessing the completeness of contracts. It is important that contracts should specify the consequences of violating obligations and prohibitions. We shall develop algorithms and tools to ensure that in this sense contracts are fully specified. These tools will be applied automatically when a contract is defined or extended.

Conflict representation, detection and resolution. In some circumstances more than one provision may be applicable, especially if contracts are modified. New provisions may introduce inconsistency. We shall provide analytic consistency checks to detect contradictory provisions, in so far as this is possible.

Contract provisions define obligations that will be incurred in particular circumstances. These circumstances are described in terms of queries on occurrences. In order to detect conflicts amongst specified provisions it is necessary to determine overlaps between sets of occurrences described by queries. The coverage-checker implemented by Abrahams and Eyers in our EDEE prototype can compute both static and dynamic semantic relationships between stored queries. More work is required in this area.

Conflicts between obligations may be of at least two kinds:
1. **Mutually exclusive obligations**: in this case a meta-policy will define a decision metric to select the single term that prevails. Weaker obligations are inapplicable and unenforceable in the circumstance. In the light of the choice of relevant provision, the remaining provisions in the conflict set are null in the circumstance (though not null in general) and are not construed as violated by non-fulfilment. For instance, in the case of an obligation to discount bulk purchases by 10% and an obligation to refrain from providing discounts to customers in overdraft, a meta-policy may determine that the latter provision is the sole determiner of discount for an overdrawn customer making a large purchase. Consequently the obligation to provide a discount of 10% is irrelevant and is not violated in this circumstance by the absence of any discount.

2. **Co-existing but conflicting obligations**: in this case fulfilment of one obligation forces violation of another co-existing and enforceable obligation. For instance, the obligation to deliver vehicle registration FHK802GP to Mr Jones on 1st May must be violated if the obligation to deliver that vehicle to Mr Smith on 30th April is to be fulfilled, and the obliged party is liable to penalty for failure to fulfil an active obligation. A decision module will select the strongest of a set of contradictory obligations, subject to a meta-policy such as *most-recent-overrides*, *most-authoritative-overrides*, *most-specific-overrides*.

**Scheduled enactment.** The enactment scheduler determines which system components can fulfil the prevailing obligations determined by contract analysis in the current circumstance. The scheduler attempts to invoke the appropriate procedures at the relevant time.

**Resolution of queries on hypothetical scenarios.** Business managers need to be able to determine the legal consequences of possible future actions or scenarios. For instance, it would be necessary to resolve queries such as ‘what obligations is the company subject to if it does not deliver the goods by 26th July?’ or ‘what refund is the delegate entitled to if they cancel their registration 5 days before the start of the conference?’. Earlier authors [5] tagged opaque textual subunits with keywords such as ‘right’ and ‘duty’ and used a simple index-based keyword lookup without regard for sentence contents or the semantics of the scenario presented. We propose to use semantic lookup using relationships between formally described sets of occurrences.

### B.3 Providing a scalable and distributed implementation

**Using an active predicate store.** An active extension to the PostgreSQL object-relational database management system has recently been developed in the Laboratory for access control policy storage and enforcement. Using this support it will be possible to notify contract enforcement managers when occurrences that match a particular template take place. This should yield significantly better performance. We shall evaluate alternative storage systems.

**Developing a database schema for contract management.** The strength of the *CamPACE* system lies in its semantic power. The generic store used to represent business applications models a semantic network directly as a set of triples. However, the procedures that enforce contracts distinguish between *associations* based on their *type*. We shall explore ways of representing contract data that will allow us to exploit such distinctions while retaining the full expressive power of the *CamPACE* model.

**Integrating Access Control** So far, our focus has been supporting application developers in expressing business contracts. When the system is used to control simultaneous use of applications by users, access control will be needed. David Eyers has experience in exporting RBAC to a number of projects and will evaluate and integrate OASIS RBAC for this purpose.

**Distributing contract and occurrence data to specialist nodes.** The complexity of medium and large enterprises makes it infeasible to enforce all contracts centrally. In the interests of efficiency and to reflect management autonomy it is appropriate to partition related contracts into stores monitored by specialist agents. Cooperating nodes must be aware of such specialist monitors so that they can forward relevant business occurrences to the nodes responsible for ensuring contract conformance in that sub-area of business. Contractual provisions will be exchanged through SOAP-based web services.

**Contract collation from semi-autonomous nodes.** The contracts governing the behaviour of an organisation and its agents constitute a dynamically changing set. Semi-autonomous subdivisions may independently define contracts with provisions that are mutually inconsistent, or that conflict with organisational or departmental goals. Independently specified policies must be collated and then analyzed using conflict detection and resolution mechanisms.

**Contract monitoring.** Relevant business occurrences, constituting workflow and contracting events as well as dynamic states of business entities, will be fed into an active database (ADB) that records business
policies. The ADB will compare inbound occurrences against the stored terms of its contracts, firing consequent obligations and permissions, and bringing about conventionally accepted states of affairs (such as the institutionally-recognized state of owning, according to the provisions of a particular clause) attainable by empowered parties.

For large scale applications it is essential that actions initiated by the terms of contracts are managed efficiently. The incremental, continuous query mechanism implemented by our CamPACE environment can determine the marginal effect of new occurrences on system state, which allows immediate detective policy enforcement. In some circumstances instant preventative actions must be scheduled when a violation is detected; for applications that are not time-critical it is preferable to reduce the load on the system by queuing consequent actions for later batched execution. We intend to investigate the trade-offs by implementing a number of trial applications.

**Evaluation.** We have a bloodbank application in place and this will be used for early evaluation and feedback. We will subsequently select one or more e-commerce applications; Alan Abrahams has contacts, and contract specifications, resulting from an internship placement, see below.

**C Relevance to beneficiaries**

At the end of his first year of PhD research Alan Abrahams took a three month internship at the headquarters of the electronic commerce division of a prominent FTSE-100 info-tech company. His task was to explore ways of reducing the effort of translation from user requirements specification to implemented code in a variety of e-business applications. As part of this project he gathered specifications of e-commerce applications from a number of domains including insurance, conference management, advertising media reservation and billing, customer loyalty and sales incentive schemes, and credit provision. Analysis identified common primitives of permission, obligation and power across these specifications, and showed that explicit representation of contract semantics, and a supporting contract management infrastructure, would assist business system implementation in many settings.

The wide applicability of these proposals across the domains analysed suggests that a variety of commercial organisations can benefit significantly from our approach. We intend to design usable tools to support contract definition, management, monitoring and enforcement, in order to enable software developers to implement contract conformant business applications. In conventional software development processes there is an impedance mismatch between requirements documentation and implemented code. We believe that the unification of specifications and code through contract representation and execution primitives will ensure mutual understanding between business units and software development teams.

There is therefore a pressing need to undertake further research in order to develop scalable prototype implementations that support our approach.

**D Dissemination and exploitation**

We have presented our initial results in policy management for business applications at a variety of refereed fora, see http://www.cl.cam.ac.uk/Research/SRG/opera/publications/index.html, under the topic “business contracts”. We will continue to publish, present and discuss our approach and findings. In particular, as well as collaborating closely with Alan Abrahams at Wharton UPenn, we will maintain contact and collaborate with colleagues at Imperial and King’s Colleges, London, who have worked previously on macro- and micro-level contract assembly tools [5, 6].

Before Alan Abrahams leaves the Computer Laboratory he will make CamPACE available under Open BSD licence conditions. We will maintain and update this software as improvements are made and as applications are developed in addition to the current “bloodbank”.

**F Management and resources**

**Staff.** We request a postdoctoral Research Associate post for David Eyers for three years to work in the Computer Laboratory to collaborate closely with the University of Pennsylvania’s Wharton Business School. Please see Professor Kimbrough’s letter of support. We request 10% of a Computer Officer’s time for making successive versions of CamPACE publicly available and maintaining it, a separate activity from the research software development and evaluation by the PDRA. We request support for a server and storage for this purpose and a PC for the PDRA.

**Travel.** We request funds to maintain contact with groups working in this area, in the UK and abroad, especially Wharton UPenn (see section D), to present our work at conferences and workshops. (£8000).
References


