1-1-2005

Designing a regional e-logistics portal

Adrian Collins
University of Wollongong

George Ditsa
University of Wollongong

Aditya K. Ghose
University of Wollongong, aditya@uow.edu.au

Peter N. Hyland
University of Wollongong, phyland@uow.edu.au

Sim K. Lau
University of Wollongong, simlau@uow.edu.au

Designing a regional e-logistics portal

Abstract

A variety of optimization and negotiation technologies hold the promise of delivering value to the logistics processes of businesses both small and large, yet they tend to remain inaccessible to SMEs (largely due to price and complexity concerns). This paper describes the early-phase steps in a project to develop a regional e- logistics portal. The project seeks to make constraint-based optimization and automated negotiation technologies accessible to SMEs within a portal that also serves their information needs. The paper highlights several novel aspects of the design of the portal, as well as a novel requirements gathering process involving community consultation.

Keywords

Designing, Regional, Logistics, Portal

Disciplines

Business | Social and Behavioral Sciences

Publication Details

Ghose, A. K., Hyland, P. N., Collins, A., Lau, S. K. & Ditsa, G. (2005). Designing a regional e-logistics portal. In D. Bunker & B. Campbell (Eds.), Australasian Conference on Information Systems (pp. 56-58). Sydney, Australia: Australasian Chapter of the Association for Information Systems.

Designing a Regional E-Logistics Portal

Aditya Ghose[1]
Peter Hyland[1]
Adrian Collins[1]
Sim Kim Lau[2]
George Ditsa[2]

[1]School of Information Technology and Computer Science
University of Wollongong
Wollongong, NSW
Email: aditya_ghose@uow.edu.au

[2]School of Economics and Information Systems
University of Wollongong
Wollongong, NSW

Abstract

A variety of optimization and negotiation technologies hold the promise of delivering value to the logistics processes of businesses both small and large, yet they tend to remain inaccessible to SMEs (largely due to price and complexity concerns). This paper describes the early-phase steps in a project to develop a regional e-logistics portal. The project seeks to make constraint-based optimization and automated negotiation technologies accessible to SMEs within a portal that also serves their information needs. The paper highlights several novel aspects of the design of the portal, as well as a novel requirements gathering process involving community consultation.

Keywords

Portal, E-Logistics

INTRODUCTION

This paper presents the early-phase steps in a project to develop a regional e-logistics portal. The project seeks to make accessible to small and medium enterprises (SMEs), as well as larger firms within a region, a novel combination of optimization, logistics and negotiation decision support tools as well as e-marketplace functionalities. These tools would enable seamless integration of the processes of seeking trading partners (buyers, suppliers and logistics providers), coordination of logistics functions across the supply chains thus formed and negotiation of contracts across these supply chains. The manner in which this integration is achieved in novel in several ways. Also novel is the tight coupling of the optimization and negotiation functionalities, so that the two form a closed loop system where the optimization modules inform the negotiation decision support system and vice versa. The target set of stakeholders, typically regional SMEs without the financial or technical wherewithal to acquire and deploy higher-end decision support technology, present unique challenges. These include stringent requirements for the resulting system to be accessible to non-technically proficient users. Requirements engineering for such a system also presents unique challenges, since the stakeholder base is largely unfamiliar with the technologies being brought to bear and is thus unable to articulate their goals from such a system (which would require some understanding of the functionalities possible with such technology).

We shall motivate our discussion via two scenarios.

Scenario One: Truck Right is a small trucking company running a fleet of small and large trucks from 4 depots, spread over a small geographical region. Business growth in the region has led to increased jobs for the trucking company, making it more difficult to schedule the fleet. At the same time, the introduction of a number of similar sized competitors into the field has meant that Truck Right cannot afford to charge as much as once it did. This has increased pressure to schedule trips as efficiently as possible. It is simply not profitable to send out the wrong size truck or trucks that are not loaded as near to capacity as possible. Having a trucking schedule that is as near to 100% utilisation as possible reduces costs of both driver's wages and consumables such as petrol. At the same time, the fleet must be properly maintained to reduce downtime on trucks from unexpected breakdowns. Despite this, breakdowns do occur and the schedule must be able to be revised as quickly as possible to ensure that deliveries are not delayed and that new jobs that arise unexpectedly can be accepted.

Truck Right would prefer to dispatch trucks from the nearest depot but would also prefer to dispatch trucks with a full load. So if a pickup is close to one depot but they already have an unfilled truck on the road from another depot, it may be cheaper to use the remaining capacity from the unfilled truck, rather than despatch another truck, which would also be unfilled, from a nearby depot. Similarly, a shipment that was too large for a small truck could be split across two small trucks if one of those trucks could also pick up additional deliveries to complete its load.

Scenario Two: Perfect Produce is a wholesaler of fruits and vegetables, purchasing goods at two or three centralised produce markets and distributing them to large supermarket chains and independent greengrocers over a wide area. Perfect Produce operates its own fleet of trucks but frequently makes use of other trucking companies, such as Truck Right, when it has more produce to ship than its own small fleet can handle. This typically occurs when Perfect Produce is able to source a product at a reduced price and can identify buyers for the product at that "special price". These special prices occur when there is a glut in the market for a product or when the shelf life of a product is low. For example, Perfect Produce may be offered an excellent buy price on a large quantity of strawberries that are reaching the end of their shelf life. If Perfect Produce can find a buyer for these strawberries and have them delivered immediately, there is a handsome profit to be made.

Perfect Produce also has an optimisation problem but that optimisation is less critical for Perfect Produce because of a surplus of trucking resources provided by companies like Truck Right and its competitors.

Perfect Produce also has a negotiation problem. The buy price for Perfect Produce on "special offers" depends on the volume of sales – the more units they buy, the cheaper each unit becomes. However, if they buy more units than they can ship, they will suffer a loss on unshipped units, which will offset the discount that is gained by purchasing in bulk. For example, the buy price for 100 crates of strawberries might be $5000 ($50 a crate) while the buy price for 200 crates might be $9000($45 a crate). If Perfect Produce buys 200 crates but can only ship 170 crates in time, they lose 30 x $45 = $1350. Since this exceeds the total value of the discount gained by purchasing the larger order, it is not in Perfect Produce's interest to take the larger order in the first place. Perfect Produce must negotiate with the suppliers of various products for the best discounts on prices

It can purchase goods at reduced prices but it will only profit on those goods if they are delivered immediately so Perfect Produce must negotiate with Truck Right as part of the decision to purchase the goods. At the same time Truck Right may not be able to supply all of the carrying capacity that Perfect Produce requires. However, Truck Right may be prepared to reschedule other deliveries Perfect Produce will pay a slightly higher rate than usual for the delivery of these particular goods. This of course would mean that Truck Right may lose the original jobs and so Truck Right needs to factor that possible loss into the price that they would find acceptable to do this special shipment for Perfect Produce.

If the price for the shipment becomes too high, it will offset the benefit to Perfect Produce of the special price. Perfect Produce would be wise to take that additional shipping cost into account when purchasing the goods in the first place. It would be no real benefit to Perfect Produce if all the profits gained from the special price were consumed in extra shipping charges.

At the same time, Truck Right would be unwise to charge so much for the shipping that Perfect Produce could not make a profit on the deal in the first place. If that happened, Perfect Produce would only buy 100 cases of strawberries and so Truck Right would not only lose out on the high rate that Perfect Produce was prepared to pay but would lose out on the delivery altogether.

Ideally, both companies want to ensure that a) the additional goods are bought, shipped and sold and that b) both companies make more from entering into the deal than they would if they had not entered into the deal.

The examples presented above highlight the three essential elements of an e-logistics portal:

- information (visibility and exchange)
- optimization
- negotiation

The rest of the paper explores how these functionalities might be integrated and deployed in a seamless and usable fashion. In Section 2, we examine these three elements in greater detail, motivating the case for their integration. In Section 3, we present our design of the integrated e-logistics portal. In Section 4, we describe briefly a novel requirements engineering approach being used in the context of this project. Section 5 present concluding remarks.

THE THREE PILLARS OF INTEGRATED E-LOGISTICS

It is widely recognized that *information visibility and exchange* is critical in effectively managing supply chains (achieving this is also recognized to be a difficult organizational problem). For precisely these reasons, supporting information sharing across the relatively small and dynamically constructed supply chains that are common when SMEs inter-operate within a regional context is a key functionality of the e-logistics portal. Sellers must have at least partial access to buyers' inventory levels and demand constraints. Buyers must have partial access to sellers' production/delivery schedules and constraints. Similarly, logistics service providers (such as trucking companies) must have partial access to the operational schedules of both buyers and sellers.

The Merriam-Webster Dictionary defines *optimization* as "an act, process, or methodology of making something (a design, system or decision) as fully perfect, functional or effective as possible". The effective deployment of *optimization technologies* has emerged, in recent times, as a key determinant of business profitability and organizational efficiency. Optimization techniques have the potential to support and improve all aspects of business operations. Such techniques are typically used to help a business decide how much to buy, from whom and when, how much to manufacture, when and where, how much to hold in inventory etc. Businesses are increasingly relying on optimization software to streamline product flows through their supply chains, plan production in the medium- and long-term, schedule production (over short-term planning horizons) and plan transportation, amongst others. Optimization technologies trace their roots back to *operations research* (OR) techniques, originally developed in the 1940s to solve military logistics problems. The examples above clearly highlight the critical need for optimization technology for businesses of all sizes, including SMEs.

The third and final element of functionality that an e-logistics portal must implement is the ability to support *negotiation*. Negotiation is typically defined as the process through which multiple parties might come to an agreement. Typical examples of agreements are clearing prices for commercial transactions. Other forms of agreement obtained through negotiation include contracts (where the agreement involves not only price but also delivery schedules, quality metrics and criteria, service schedules etc.) or wage agreements (obtained, for instance, via negotiation between a labour union and the management of an organization). Price and more complex contracts are both routinely negotiated by SMEs. We believe that automated decision support for negotiation is critical for SMEs, for several reasons.

- The scale of the negotiation problem, even for SMEs, can be large enough to warrant automated support. Traditionally, SMEs tend to inter-operate with a fixed (and typically small) set of suppliers, customers and logistics providers. A regional e-logistics portal must, in the first instance, incorporate the functionalities of a B2B market, allowing member organizations to conduct business with other member organizations. For a given SME, this can dramatically broaden the range of potential suppliers, customers or logistics providers. A range of market mechanisms are feasible in this context, such as forward or reverse auctions, continuous double auctions, one-on-one negotiation (see Wurman (2002)) for a survey of alternative mechanism designs). Almost all of these mechanisms present daunting challenges of scale (large numbers of messages/bids, complex computation for winner determination and auction clearing).

- Negotiation in settings such as these is a long-duration process, which may be difficult and expensive, if not impossible, to support manually. To fully leverage the benefits of an e-logistics portal, even a relatively small enterprise might find itself engaged in negotiating several hundred transactions (these could be buy-side, sell-side or logistics-related) concurrently. A useful means of comprehending the complexity of such an exercise is to imagine having to manage bidding in several hundred e-bay auctions concurrently, and under far more stringent sets of business constraints than the average bidder on e-bay. The bidding process may extend over several days, and requires continual monitoring, as well as the deployment of complex bidding strategies to achieve optimal outcomes. Automated support for negotiation is clearly important from this perspective.

- The negotiation process must be sensitive to real-time changes in the business context. The optimal negotiating behaviour of a firm must depend on real-time information on metrics such as inventory levels for various products, current production plans and the current set of commitments/contracts. Thus a firm may choose to withdraw from the bidding process in a reverse auction for a certain product if it determines that manufacturing the goods under consideration would not be feasible given another recently concluded deal. The volume of and complexity of relevant information, as well as the complexity of the reasoning required is likely to exceed the cognitive capabilities of individuals (who might have manually conducted the negotiation process on behalf of the firm).

A key insight from the requirements gathering exercise conducted for this project was the need to support the interleaving of negotiation and optimization. We shall discuss this in greater detail in the following section.

INTEGRATED E-LOGISTICS: A SYSTEM DESIGN

We present key elements of the design of the e-logistics portal in this section. In the previous section, we have identified three key elements of the functionality of such a portal: *information visibility (and exchange), optimization* and *negotiation*. We shall highlight some the important guiding principles that must inform any design that supports these functionalities, and shall present some design decisions in the light of these principles. An interesting initial observation is that the support for these functionalities ends up being tightly interleaved. Of particular interest are some novel issues that arise form the interleaving of negotiation *and optimization.*

Supporting information visibility and exchange poses three major challenges:

- Data storage: Information sharing requires that production schedules for suppliers, production (and hence, demand) schedules for customers and shipping schedules for logistics providers must be stored. To support semantic search for products and services in the online marketplace, our design involves semantic markup (in OWL (http:://www.w3.org/tr/owl-features)) of most of this information. We shall discuss semantic search in greater detail later in this section.
- Data visualization: Information visibility throughout a supply chain is required not only by the automated decision support machinery, but also by human users. A key challenge is to determine how logistics decisions (such as production or delivery schedules and intermediate or final negotiation agreements) might be visualized in meaningful ways. Our current design uses interactive Gantt charts as the basic visualization tool. The interactive Gantt chart pops up text boxes when the cursor is moved over specific portions of the chart containing relevant details such as the agreed selling price of the product under consideration, quality criteria etc. The interactive Gantt chart permits users to alter schedules via a ``drag-and-drop" interface, thus supporting mixed-initiative reasoning (we will discuss this in greater detail later in this section). The complexity of the data being visualized entails that multiple Gantt charts, from different viewpoints, will be required to completely represent a firm's production or delivery schedule, or a complete negotiation outcome.
- Data access policies: Logistics information relating to a given firm can only be partially revealed to other firms. Firms typically require that only the minimal subsets required to coordinate schedules, or to arrive at successful negotiation outcomes be revealed to other parties. The information revelation policies are also context-specific, i.e., these policies specify what information can be revealed to which organizations and under what circumstances. Every time information is revealed, whether for users to visualize or in response to automated messages, inference is performed over the rules encoding access policies to determine what information may be revealed.

We shall discuss the design of the optimization support component of the system by first explaining our choice of optimization technique and engine and then discussing several ways in which we extend the standard design of such engines. A wide range of technological choices exist for any designer of an optimization system. The operations research community has developed linear and integer programming, as well as several variants. Artificial intelligence (AI) research has resulted in the development of rule-based systems, genetic algorithms and constraint programming. Our design relies on constraint programming technology, which traces its roots back to the class of *constraint satisfaction problems* (Mackworth, 1977) that were first studied in the 1970s in the context of picture processing.

A constraint satisfaction problem (CSP) typically involves a set of variables, a corresponding set of domains from which these variables may be assigned values and a set of constraints that restrict the possible combinations of values that may be assigned to these variables. A solution to a CSP involves the assignment of values to these variables that satisfy all of the specified constraints. A common variant of CSPs are *constraint optimization* problems, where the intent is to find solutions which maximize or minimize the value of a specified objective function.

The constraint programming paradigm has been successfully deployed in a wide variety of domains, including hardware verification, concurrent computing, databases, graphical interfaces, combinatorial optimization, programming language design and implementation, as well as well-known artificial intelligence problems such as agent planning and configuration.

The observation that a key component of program execution in logic programming languages (such as PROLOG) is a special case of constraint solving led to the development of the constraint logic programming paradigm (Jaffar and Lassez, 1987) and a large number of successfully deployed constraint logic programming languages (Jaffar and Maher, 1994; Jaffar et al, 1992; Hentenryck 1989).

Several factors make constraint programming the technology of choice in an exercise such as this. First, constraint programming supports a robust knowledge representation language which permits all of the relevant domain knowledge/constraints to be brought to bear on the problem solving process. Traditional operations

research techniques are able to generate optimal solutions relatively efficiently, but insist on a rigid input specification format that often makes it impossible to represent relevant domain constraints. Thus, the optimal solutions generated by these techniques are sometimes solutions to (possibly poor) approximations of the real problem. Second, constraint programming offers a robust and flexible modelling language for specifying constraints. Linear and mixed integer programming techniques, on the other hand, place a heavy burden on the user in modelling problems in an inflexible language that insists on all constraints being expressed as linear relationships. Third, the flexible knowledge representation language supported by constraint programming can, in some cases, permit the specification of constraints that significantly reduce the search space and dramatically speed up the solving process.

Several recent studies have reported performance improvements of several orders of magnitude in constraint programming formulations of specific problems (presented in the literature as particularly intractable challenge problems) relative to their best competing linear or integer programming formulation. Fourth, constraint programming techniques permit the deployment of domain-specific heuristics that can further improve performance. Finally, the knowledge representation language permits the logical interpretation of each constraint as a sentence in a first-order language. This permits the integration of constraint programming with a variety of other formal systems derived from the field of artificial intelligence (such as logic programming), with significant benefits in domains with uncertain and changing information. Constraint programming also compares favourably with other AI techniques such as rule-based systems (which offer limited optimization facilities and suffer serious performance degradation problems in dynamic domains) and genetic algorithms (which require a large population of solutions to be effective and which do not provide any means of verifying that a given solution is indeed optimal).

Our proposed design of the optimization support component of the e-logistics portal relies on an industry-standard constraint solving engine (ILOG SOLVER), but involves several significant extensions to its functionality. We discuss each of these in turn below.

*Over-constrained solving***:** Most existing constraint programming tools assume that the input set of constraints is solvable (i.e., a solution exists that satisfies all input constraints). In most supply chain and logistics optimization problems (as with several other application domains) this is often not the case. Solvability of the input set of constraints is also difficult to guarantee via manual scanning by operators or planners entering these constraints. Existing constraint programming tools must therefore be augmented to support such *over-constrained* problems (Jampel et al, 1996; Freuder, 1992)**.** Over-constrained problems may be solved using a variety of strategies: by identifying the largest subset that can be satisfied (Freuder, 1992), by minimizing the degree of violation of the unsatisfied constraints **(**Wilson and Borning, 1993) or by aggregating local specifications of preference (over partial solutions) into global specifications of preference over complete solutions (Bistarelli et al, 1997)**.** Our current design involves supporting the first of these three strategies (this is often referred to as the Max-CSP approach). A more complex strategy that we propose to subsequently implement is the third approach, where each constraint is specified via preferences on alternative sets of value assignments to variables. The ``allowed'' sets of value assignments in the classical CSP approach are the most preferred assignments in this approach, but other, less preferred sets of assignments are also specified so that if the most preferred assignments cannot be consistently achieved, a partially-ordered set of less preferred value assignments can progressively explored until a solution at the “best level of preference” is identified. A detailed discussion of this approach is outside the scope of this paper, but it is useful to note that the preferences themselves are specified using abstract preference values that are structured as an algebraic semiring (where the associated additive multiplicative operators are used for comparison and combination of preference values, respectively). These abstract preference values lend themselves to several useful practical realizations - thus permitting complex preference specifications that combine rankings on multiple orthogonal dimensions using heterogeneous scales (some of which may be qualitative, while others might be quantitative). This approach is of particular interest from the perspective of integrating decision support for optimization and negotiation (that will be discussed in greater detail later in this section), because these same abstract preference values can be used to specify user preferences over alternative negotiation outcomes.

Dealing with dynamic problem domains: Existing constraint programming tools assume that the input problem is static and that each problem variable is a *decision variable* (for which the system decides and assigns a value). Most real-life optimization problems are dynamic and involve a mix of decision variables and *parameters* (variables whose values are decided by the external environment) (Fargier et al, 1994). Such settings might require tools which generate *conditional* solutions that are contingent on certain assumptions about the external environment being true. Our design involves an extension to the constraint solving engine to deal with variables whose values are decided by the external environment.

Incremental solution reuse: In dynamic settings such as supply chain and logistics planning, the problem inputs go through frequent (but typically small) changes. Re-computing the solution in response to every change would result in unacceptable degradation in system performance. Effective tools must thus incorporate techniques

which incrementally adapt and reuse prior solutions to deal with changes to the input specification. Our design incorporates features that permit us to localize the impact of incremental, marginal changes so that re-solving is required of only smaller sub-problems.

Mixed-initiative reasoning: The key idea in mixed-initiative reasoning is to formulate a problem-solving process that is driven in equal measure by the system and the operator. This is useful because the user is able to intervene at critical decision points (such as those that require managerial approval). As well, user feedback is used to rule out unlikely possibilities, thus reducing the search space and improving system performance. Mixed-initiative reasoning is built into the design of the machinery for generating optimal production and delivery schedules in two ways. First, certain variables are designated as *user settings* - at key points in the search process, the user is queried for the values of these variables. Note that there are critical differences between this approach and one in which the values of the user settings are determined at the start of the search process. Given that the process of generating a logistics schedule can extend over a significant duration of time, it is important that the values of these user settings be context-dependent and determined as late in the process as possible. Second, we designate a set of variables as *conditional user settings*. These are variables whose values can be determined by the user given a partially-constructed solution, i.e., the values that the users decide for these variables are contingent on the specific partial solution that is presented to the user.

Real-time planning/scheduling: Operations planning and scheduling systems must typically satisfy stringent time constraints. Ideally, such tools must use *anytime algorithms*. An anytime algorithm is one which satisfies the following three conditions: First, it must be possible to interrupt it at any time to obtain a useful partial solution. Second, it must be pre-emptively schedulable (i.e., it must be able to save its state when interrupted so that it can resume computation from that point when re-started). Third, it must guarantee that the quality of the partial solutions improves monotonically with time (i.e., it must provide better solutions given longer processing time). Constraint-based optimization procedures support anytime reasoning in a straightforward manner. After generating the first feasible solution, these procedures climb the optimisation gradient, producing progressively better solutions. When interrupted, the procedures can thus return the best solution obtained thus far. However, the complexity and hierarchical structure of integrated planning and scheduling problems pose special challenges. While a solver at any given level of the hierarchy might support anytime reasoning, it is not clear how the partial solutions might be propagated to different levels of the hierarchy. The hierarchical structure leads to inherently non-monotonic behaviour (a consistent high-level plan might be invalidated by scheduling level constraints, for instance), making it difficult to guarantee monotonic improvement of solution quality.

Support for reasoning with hypothetical scenarios: Most planning/scheduling systems require support for reasoning with hypothetical scenarios (often referred to as ``what-if'' analysis) to explore the impact of decisions without committing to them. Ideally such systems should support the merging of multiple hypothetical scenarios to generate a candidate solution. However, such merging operations remain poorly understood. As above, the hierarchical structure of the problem also presents difficult challenges.

The design of the *negotiation decision support* component currently supports the following modes of negotiation.

- Forward and reverse Vickrey auctions: In a forward Vickrey auction, a seller invites sealed bids from potential buyers. The highest bidder wins, at the second-highest price. A well-known result from the economic theory of auctions is that this auction mode is strategy-proof (i.e., the optimal strategy is to submit honest bids). In a reverse Vickrey auction, buyers invite sealed bids from sellers, and the lowest bid wins at the second lowest price.
- Forward and reverse English auctions: A forward English auction is the most common auction mode that most people are familiar with. Such auctions are open cry (bids are visible to all buyers in addition to the auctioneer/seller) and ascending price, with the highest bid winning. The auction clears when no higher bids are received within a fixed time-out period, or at a fixed deadline (such as in online auction sites like eBay). In reverse English auctions, the buyer replaces the seller in the role of auctioneer (ignoring the role of third-party auctioneers), are descending price, and the lowest bidder wins.
- Forward and reverse Dutch auctions: Forward Dutch auctions are open cry descending price auctions.
- Continuous double auctions.

A key component of the negotiation techniques being designed involves the integration of logistics parameters (such as delivery deadlines) into each of these mechanisms. Thus an ascending bid might relax not the offered price but the required delivery date (i.e., accepting a later delivery). We do not discuss these here due to space constraints.

Optimization and negotiation techniques integrate in the following ways:

- Optimization engines can help identify the cost/efficiency impact of alternative negotiation outcomes (e.g., how would a commitment to delivering in 2 weeks impact my equipment utilization levels? How expensive would it be to reschedule to production to ensure that a high-priority order is processed within 2 days?). In this mode, the optimization engine
- The negotiation engine may operate as a "subroutine" to an optimization engine. In this instance, some of the inputs to the optimization engine derive from the negotiation engine. For instance, negotiated delivery deadlines and prices will determine optimisation outcomes.
- A negotiation front-end will be informed by an optimisation back-end.
- Negotiation changes the optimization land-scape (i.e. the set of constraints that are considered in computing a solution).

REQUIREMENTS ACQUISITION AND MODELING

We have devised a novel approach to requirements collection and analysis in this project. The stakeholders for this project are SMEs in the local region. Given the relative sophistication of the technologies being brought to bear, and the unfamiliarity (and technical inaccessibility) of these technologies, it is difficult for these stakeholders to envision the possibilities offered by these technologies, thus making it difficult to elicit requirements for the proposed system. We have devised a three-pronged approach to the problem, involving a combination of conceptual modelling in an expressive framework (we are using the *i** framework for agent-oriented conceptual modelling (Yu, 1997), rapid prototyping and the use of technology exemplars from distinct problem domains to clarify the possibilities offered by the technologies being brought to bear. Interaction with stakeholders has been structured into a series of SME consultation evenings, the first two of which have already been held in late 2003.

CONCLUSION

We have presented a novel design of an integrated regional e-logistics portal. We believe that systems that support collaborative logistics in the sense described here will become increasingly important. The portal has been partially implemented, and we expect to report results from that exercise at a later date.

REFERENCES

Bistarelli, S., Montanari, U., and Rossi, F. (1997) Semiring-based constraint solving and optimisation, *Journal of the ACM*, 44(2): 201-236

Fargier, H., Lang, J. and Schiex, T. (1994) Mixed constraint satisfaction: a framework for decision problems under incomplete knowledge, in *Proceedings of AAAI-94: The 1994 National (U.S.) Conference on Artificial Intelligence*, Portland, OR., USA

Freuder, E. C. and Wallace, R. J. (1992) Partial constraint satisfaction, *Artificial Intelligence*, 58(1-3): 21-70

http:://www.w3.org/tr/owl-features/

Jaffar, J. and Lassez, J.L. (1987) Constraint logic programming, in *Proceedings of the 14th Annual ACM Symposium on Principles of Programming Languages*

Jaffar, J. and Maher, M. (1994) Constraint logic programming: a survey, *Journal of Logic Programming*, 19-20

Jaffar, J., Michaylov, P., Stuckey, P. and Yap, R. (1992) The clp® language and system, *ACM Transactions on Programming Languages and Systems*, 14(3)

Jampel, M., Freuder, E. and Maher, M. (1996) *Over-constrained systems*, Springer Verlag Lecture Notes in Computer Science

Mackworth, A.K. (1977) Consistency in networks of relations, *Artificial Intelligence*, 8:99-118

van Hentenryck, P. (1989) *Constraint satisfaction in Logic Programming*, MIT Press

Wilson, M. and Borning, A. (1993) Hierarchical constraint logic programming, *Journal of Logic Programming*, 16:277-318

Wurman, P.R., Wellman, M.P. and Walsh, W.E. (2002) Specifying rules for electronic auctions, *AI Magazine*, 23(3): 15-23

Yu, E. (1997) Towards modelling and reasoning support for early-phase requirement engineering, in *Proceedings of 3rd IEEE Intl. Symposium on Requirements Engineering*, 226-235

COPYRIGHT