InterNeg

InterNeg Research Papers INR06/06

Revised version appeared in: *Proceedings of the Montreal Conference on Technologies (MCeTech)*, L. Logrippo, H. Mili and A.S. Salah (Eds.), Montreal, May 17-19, 2006, (43-52).

# Developing Context-Independent E-Negotiation Systems using Software Protocol and Model-View-Controller Design Pattern

[1]JinBaek Kim, [2]Gregory Kersten, [3]Stefan Strecker and [2]Ka Pong Law

[1]Concordia Institute for Information Systems Engineering
[2]John Molson School of Business, Concordia University
[3]Information Systems and Enterprise Modelling, University of Duisburg-Essen, Germany

**Abstract**

A major challenge in developing an e-negotiation system (ENS) is that the context of negotiation such as negotiators' characteristics, negotiation rules and processes are different case-by-case. This context dependency makes it difficult to develop a general ENS which is applicable to a wide variety of negotiation problems. In this paper, we propose to adopt the component-oriented software protocol approach and adopt Model-View-Controller design pattern in order to mitigate the context dependency issue. In this approach, an ENS is developed first by designing a high level e-negotiation protocol which specifies which page composer should be called at each state. The page composer contains the logic that controls software components such as queries, buttons, display text and format to compose a page. The activities of the negotiators are guided by activities allowed in a page and change into another page. This approach allows one to easily develop and modify ENSs which fit into the negotiation context very well. We prove validity and usefulness of our approach by quickly re-developing two existing ENS's - SimpleNS and Inspire – by defining e-negotiation protocols on a general purpose ENS platform, which executes the protocol and controls the software components.

Keywords: e-negotiation systems, context dependency, negotiation protocol, negotiation process design, component-based approach, software protocol.

**Acknowledgments:** The publication of the InterNeg Research Papers has been supported by the Natural Sciences and Engineering Research Council, the Social Sciences and Humanities Research Council, and the J. Molson School of Business, Concordia University.

## 1. Introduction

There are many factors that hinder negotiators from exploring all available options and reaching an efficient outcome. For example, the limited information processing and cognitive capability of human beings restrict the range options that can be examined. Also, socio-emotional biases towards the counterpart often negatively affect the negotiation outcome [1, 2].

Negotiation Support Systems(NSS) are networked decision support tools for improving intra-personal and inter-personal activities in negotiations[3]. Studies show that Negotiation Support Systems can increase the joint outcome and improve fairness[4]. Human information processing capability can be complemented by systematic elicitation and application of preferences[4, 5]. Objective and holistic view of alternatives presented by the system helps negotiators to overcome their cognitive limits[6]. It has been also reported that negotiation through electronic channels can alleviate socio-emotional biases by depersonalization[1].

Potential functionalities of NSSs include decision support in preparation (e.g. formulation of negotiation as a problem, modeling preferences, analyzing counterpart), execution (e.g. evaluating offers, advising strategies, visually presenting information), and post-settlement (e.g. analyzing efficiency of the outcome, suggesting improvement) as well as communications [7] [8].

As the potential for e-negotiation systems is continuously growing, we observe development of many commercial as well as research-purpose e-negotiation systems. Some of them focus on supporting evaluation and decision making (e.g. SmartSettle.com), while others mainly consider communication support (e.g. WebNS[9], ElectronicCourthouse.com). Many recently developed e-negotiation systems consider both decision making and communications [10, 11]. Domains of application are also becoming diverse including trade agreement, insurance claim, and dispute resolution. [12] provide a good survey of commercial e-negotiation systems and business models.

The focus of this paper is e-negotiation systems(ENS), NSSs which allow and help people to conduct negotiations on the web. Among many forms of ENSs, the ENS considered in this paper are the ones used by humans where the key negotiation activities such as sending and reading offer are not automated, although auxiliary activities which traditionally have been conducted by software, such as computation of utilities, simulations, scenario generation, remain conducted by software.

In face-to-face (F2F) negotiations, the parties may freely choose from complementary activities at any stage of the negotiation. The parties may decide on the information they use, the strategies and tactics, decision rules, and so on. They also may undertake activities which they did not consider at the beginning of the negotiation. The flexibility that negotiators have in F2F encounters can be matched when only communication tools (e.g., email and videoconferencing) are provided. The more support there is from an ENS and the more actions the system undertakes autonomously, usually the more restrictions are imposed on the negotiators' own activities. The *negotiation protocol* is a formal model, often represented by a set of rules, which control and guide the activities undertaken on one hand by the negotiators and on the other by the ENS.

A major challenge in developing an e-negotiation system (ENS) is that the context of negotiation such as negotiators' characteristics, negotiation rules and processes are different case-by-case. One possible approach to mitigate this context dependency issue is to separate model for negotiation processes or negotiation protocol model from the system platform that executes it. Through this way, the most appropriate rules that guide activities of negotiators, or e-negotiation protocols, can be created new or by modifying existing e-negotiation protocols. In order to achieve this, the conceptual protocol model should be abstract enough to help users focus only on essential negotiation activities while it should have logical consistency liked to the details for implementation of ENS.

In this paper, we propose to adopt the component-oriented software protocol approach and adopt Model-View-Controller design pattern in order to develop ENSs mitigating the context dependency issue. The organization of this paper is as follows. In the next section, we explain software engineering foundations for developing context-independent ENSs. Component-oriented approach, software protocol, and Model-View-Controller software design patterns are introduced. In section 3, we present e-negotiation protocol model that allows developing ENS simply by defining the protocol which uses reusable components. In section 4, system architecture of e-negotiation system is proposed. In section 5 and 6, we show implementation ENSs using this approach and evaluation of the proposed development approach and implementation. In section 7, we conclude with summarizing the contribution and discussing future work.

## 2. Software Engineering Foundations for Context-Independent E-Negotiation Systems

### 2.1 Component-Oriented Approach and Software Protocol

The component-oriented approach to e-negotiation system development allows us to take advantage of many well-known merits of component-based systems such as flexibility in the system, easy maintenance and upgrade, reducing the development cycle time by reusing existing components, and so on.

A software protocol is defined as a set of rules that determine what or how unrelated objects or components communicate with each other [13]. The software protocol for component-oriented systems determines which component is executed and in what order. The software protocol also decides what is to be done when a component's execution results in success or in failure; it invokes another component to be executed or—if it cannot determine a component—chooses one of the "fall-back alternatives" (e.g., it terminates the system's execution).

Component-oriented approach to ENS will lead to viewing the e-negotiation protocol as a software protocol. From this perspective, e-negotiation protocol is the set of rules allowing, forbidding, and forcing negotiation activities through enabling, hiding, and invoking components.

Component is a relatively defined term. Sometimes a whole system may mean a component while a component may refer to a small piece of code. The granularity of components to be used by the e-negotiation protocol depends on the activities modeled in the e-negotiation protocol because the foremost goal of the e-negotiation protocol is to support negotiation activities. The World Wide Web is designed based on the metaphor of pages and hyperlinks [14]. Therefore, page becomes an easily identifiable unit of activity in web-based applications such as e-negotiation systems. Considering the page-level as the granularity of the activity, e-negotiation protocol can be viewed as a software protocol that controls page-level components.

It should be noted that activities and components defined at the page-level can be further decomposed into minuscule ones. In other words, an activity on one page may be broken into several smaller elements (i.e. actions), and likewise, a component for a page may be decomposed into smaller components (i.e. sub-components). For example, the construct offer page may comprise the form component which writes the submitted offer to the database and the display component that reads the most recent offer from the database and presents it to allow a user to read the most recently received offer and construct a counteroffer. This case shows that the page-level activity, construct offer, can be decomposed into the minuscule actions of read offer and write offer, and the page-level component, offer construction, should contain the subcomponents of offer display and offer construction.

We propose to separate modeling of the page-level activity and the detailed composition of the page. There are advantages in doing so. First, modeling detailed actions composing each page can be left out of the scope when designing e-negotiation protocol. Therefore, the implementation of ENS can be divided into the protocol designer who focuses on high level coordination among page-level activities and the component developer who implements page-level components by combining sub-components for various negotiation actions.  Second, the features on individual pages can be upgraded or changed without changing defined e-negotiation protocol using them. Third, component designers can have more flexibility in designing and implementing page-level components because those components are encapsulated and separated from the e-negotiation protocol. An example of the software protocol controlling the execution of four components (A, B, C, and D) is presented in Figure 1.

## 2.2 Model-View-Controller (MVC) Pattern

Model-View-Controller (MVC) is programming design pattern which has its roots in Smalltalk. MVC pattern decomposes software into three types of components - model, view and controller. The model represents data and the logic that governs access to and updates of this data. The view specifies how the data contents of a model should be presented. The controller translates interactions with the view into actions to be performed by the model. Based on the user interactions and the outcome of the model actions, the controller responds by selecting an appropriate view[15].

In a web based application such as ENSs, database queries and computation modules fall into the model component, HTML and CSS into the view component, and the controller component includes modules that links model and view components.

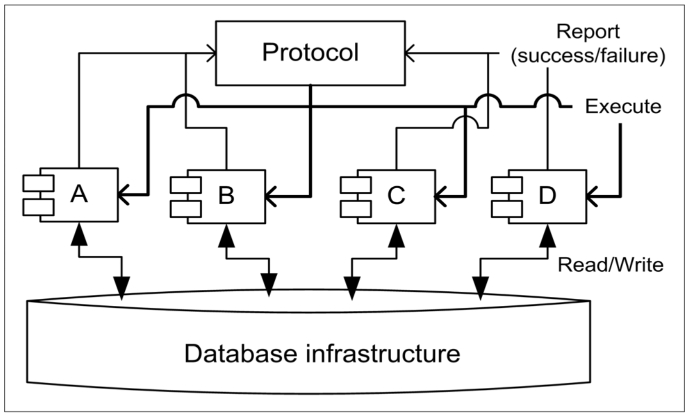


Figure 1. Protocol, components and database infrastructure

Using the MVC pattern improves development cycle time, reusability, testability, and maintainability of the ENS by allowing easy creation and change of various page-level components through configuration of existing model and view components through the controller. Another advantage is to allow ENSs to easily support new types of client such as mobile devices and survive technology changes with minimal rework such as HTML replaced by Flash, MS Access replaced by MS SQL Server through encapsulating each of the three components [16].

## 3. E-Negotiation Protocol Model

The ENS should be applicable and appropriate to many different contexts without changes as far as the difference is minor. In order to accommodate minor difference in contexts, the conceptual e-negotiation protocol should allow defining protocols that give some degree of freedom in selecting activities. For example, during the process, the negotiators may wish to review the problem, modify their preferences, and add or remove issues. The freedom of performing these activities should be allowed to fully fit into the context. On the other hand, however, the e-negotiation protocol should be able to force the negotiators to undertake certain activities in order to reach a better outcome. For example, forcing the negotiators to learn about the negotiation problem, consider their own objectives and preferences, and evaluate the counterpart's offer before making their own offers can improve the negotiation outcome by encouraging the negotiators to make informed decisions. In these situations, the selection of a particular activity opens a new path of activities which have to be contiguous; every possible path selected by the user and/or ENS has to be connected and geared towards the desired negotiation outcomes.

The dilemma between restricting the bargainers' possible activities and providing the flexibility in activities is the main issue of negotiation protocol design. The protocol model for ENS or e-negotiation protocol model should allow both considerations to be incorporated when designing the e-negotiation protocols. Adopting component-oriented software protocol approach, we propose to model e-negotiation protocol as *sequences* (i.e. sets of interrelated components), *states* of the sequences, *exit links* (i.e. directed links between sequences), and intervening rules (i.e. rules of handling information exchange) as a feasible model for this problem.

ENSs should support intra-personal and inter-personal activities to improve negotiations. Therefore, e-negotiation protocol should model both types of activities. As mentioned in the previous section, the level of activities considered in our e-negotiation protocol model is page-level activities and the component considered in ENS is the software module associated with composing a page.

In the proposed e-negotiation protocol model, intra-personal activities are guided and controlled by the *sequence-state-exit link* model which specifies components accessible by the user and change of those available components. A set of interrelated components which together determine all possible activities in a given negotiation situation is called *sequence*. Every sequence has at least one component and it is possible that one component belongs to more than one sequence. *Exit link* is defined as binary relation between two sequences indicating a user can move from one to the other. Components in a sequence are classified into *initial, mandatory, optional, and hidden optional states* depending on their roles. *Initial state* is a component which a user is forwarded to when entering a sequence is the initial state of the sequence. *Mandatory state* is a component which the user has to enter in order to exit to another sequence is the mandatory state of the sequence. *Optional states* are the components accessible to the user in a sequence. *Hidden optional states* are the components not accessible to the user until some conditions are met.

The needs for initial states and optional states are clear. When there are many components in a sequence, the component to be invoked when a user first enters a sequence should be specified. Also, the reason for having optional states are evident – components that can be visited from any other component from the sequence – from the reason to cluster components into a sequence. Because a user should access the initial state, the initial state must be an optional state.

After trying to model different types of protocols by components and sequences, we found in general there is a component that should be invoked before leaving a sequence. For example, it is desirable to allow moving from the exchange_offer sequence to the agreement sequence only when the read_offer component is being executed, because this will ensure that the user knows which offer he agrees. Mandatory state of a sequence models such a component that should be visited before leaving a sequence. The mandatory state may not be an optional state because sometimes it is desirable not to allow the user to leave the sequence until some conditions are satisfied.

We made a design choice of allowing one mandatory state in a sequence, because by doing so the system can consistently apply rules of displaying all exit links at a mandatory state, without considering which links to display on which mandatory state. This rule of the unique mandatory state affects the granularity of the sequences in an e-negotiation protocol. After trials of modeling e-negotiation protocols, we found it leads to a reasonable level of granularity. Figure 2 visually describes the relationship between components, sequences, exit links, and states.

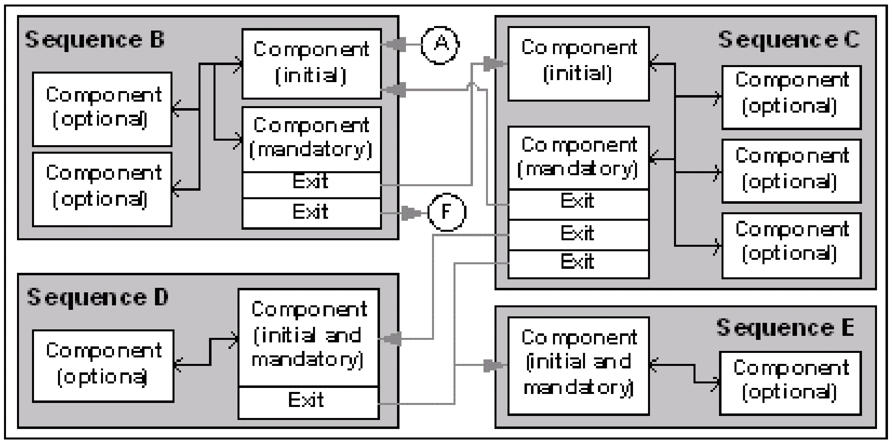


Figure 2. An example of six sequences (A,B,C,D,E,F)

We propose to model intra-personal activities by information exchange or intervening model. *Intervening* is defined as sending information to the counterpart and change the status of the counterpart's system. E-negotiation protocols should consider how to handle intervening or the intervening rules. Well defining how to handle intervening (i.e. intervening rules) is critical in the e-negotiation protocol because intervening is a key activity in negotiations. Intervening rules are the function of information type received. Common information types found in most negotiations are offer, message, agreement, and termination.

*Intervening rules* affect intra-personal activities through executing, enabling, or invoking components in response to the received information type. In other words, intervening rules define the way to change the current activity and/or allowed activities when a specific type of information is received.

The e-negotiation protocol model defines the start of intervening by executing a component in the sequence (e.g. by sending offer or message) or by exiting to a different sequence linked through an exit link (e.g. by moving to agreement or termination sequence). If the execution of the component or exit link sends a specific type of information to the counterpart, the component or exit link is associated with the information type.

After trials of modeling different protocols, we identify three types intervening rules described below are most commonly required. They are

- Activate hidden optional states in sequence into optional states

- Update initial state of a sequence

- Forward a user to a specific sequence

More details on the model and implementation of behavior patterns commonly observed in ENS can be found in Kim et al. [17].

## 4. E-Negotiation System Architecture

Adopting the concepts and models discussed in the earlier sections, we developed Invite platform which runs the e-negotiation protocols defined conforming to the model presented in the previous section.

The system is composed of model (negotiation protocol database, protocol queries, negotiation content database, content queries, graphic generation and computation components), view (protocol layout, content format), and controller (page composer). Negotiation protocol database stores various e-negotiation protocols which are defined using the proposed e-negotiation protocol model. Negotiation protocol queries read from and insert/update into the negotiation protocol database in order to generate links (or buttons) to other activities, change the available activities, and force some activities. Negotiation content database stores the information generated during the negotiation such as offers, messages, preferences, and agreement. Negotiation content queries read from and insert/update into the negotiation content database in order to display information and interact with the counterpart.

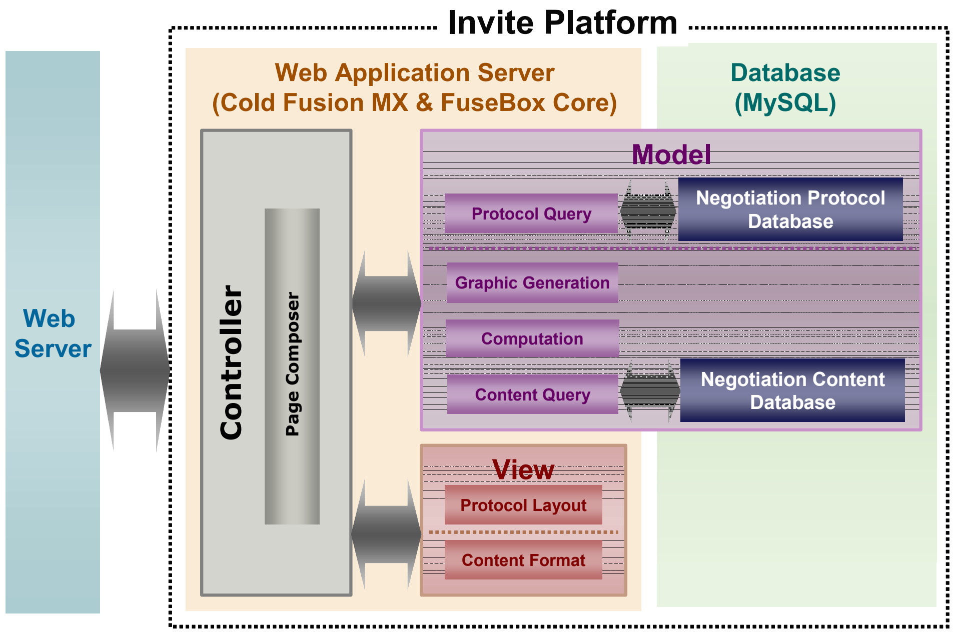


Figure 3. ENS architecture

Graphic generation component generates visual information such as offer history and negotiation dance graph by transforming the output from negotiation content queries. Computation components are used to transform user input into more complex data and analyze the negotiation outcome. They are used for preference elicitation, offer evaluation, analyzing Pareto efficiency of the negotiation outcome. Content formatting components format the data from content queries to present to the user. Negotiation content data is structured and formatted using tables, lines, colors, fonts, etc. Protocol layout components specify the format that should be consistently maintained throughout the negotiation protocol such as header, footer, links, and page – style and location of display.

User interface or a page is generated by the page composer which combines the output of negotiation content query, graphic generation component, and computation component and is presented to the user after formatting it according to the HTML formatting component and protocol-dependent page layout beside the output of the negotiation protocol query.

Figure 3 shows the proposed architecture of the ENS. For implementation we used Macromedia's ColdFusion Web application server and FuseBox MVC core. FuseBox core helps following the MVC pattern by supporting separation of code files into model, view, and controller folders depending on whether the code is display/layout related, query related, or controller related. As for the databases, we used MySQL server. Graphic generation and computation modules are implemented in Java.

The e-negotiation protocols following the model explained in the previous section can be represented by tables, and consequently, implemented by a relational database. Implementation of the e-negotiation protocol requires design-time tables and run-time tables.

An e-negotiation protocol is specified in the design-time protocol tables which describe initial setting of sequences, states, and exit links as well as intervening rules. For each optional state and exit link, the information type should also be specified if execution of it invokes intervening. Design time protocol tables also include three tables needed for describing the intervening rules: 1) activating hidden optional states, 2) updating initial states, and 3) forwarding to a sequence. All of them should have the type of received information as the condition for applying the rules. The hidden optional state activation table contains information on which state to activate in which sequence, the initial state update table describes which component to become an initial state in a specific sequence, and the sequence forwarding table defines the origin (i.e. from sequence) and the destination (i.e. to sequence) to which the user who received information is forwarded.

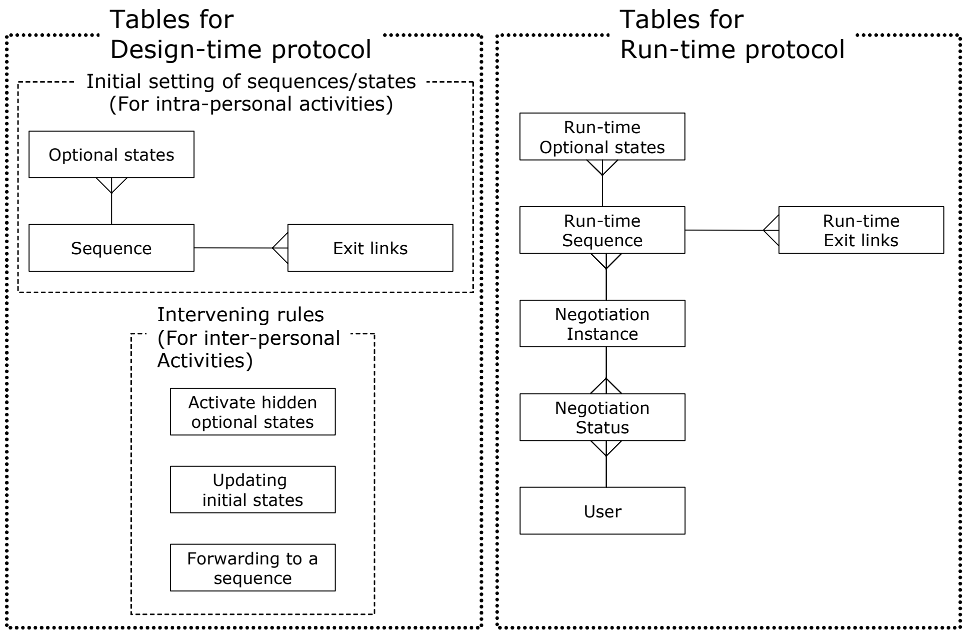

Tables for Design-time protocol
Initial setting of sequences/states (For intra-personal activities)
Optional states
Sequence
Exit links
Intervening rules (For inter-personal Activities)
Activate hidden optional states
Updating initial states
Forwarding to a sequence
Tables for Run-time protocol
Run-time Optional states
Run-time Sequence
Run-time Exit links
Negotiation Instance
Negotiation Status
User

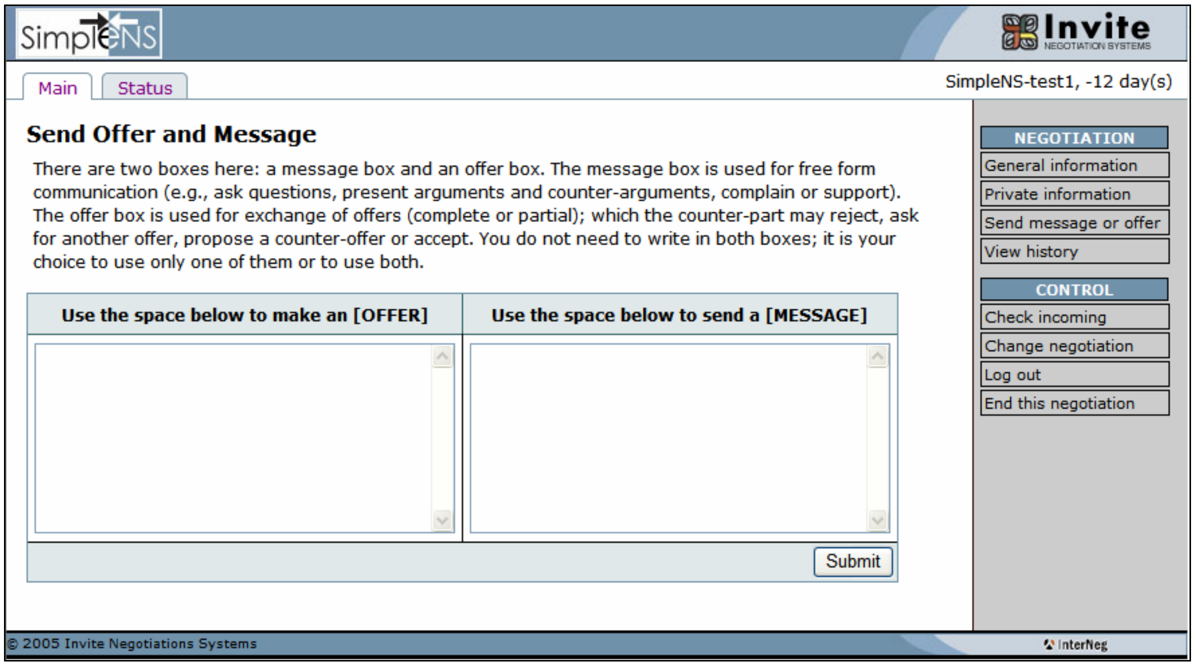
Figure 4. Data model for e-negotiation protocols

Operationalization or instantiation of the negotiation protocol is supported by the tables recording the status of each negotiation instance and each user. Instantiation requires copying the sequence, state, and exit link setting from the design-time protocol table to run-time protocol table. Then, each negotiation instance and user starts with the initial setting of sequences, their states, and exit links which change as the negotiation proceeds according to the intervening rules. The ER diagram in Figure 4 presents the data model for design-time and run-time protocols.

## 5. Implementation

In this section, we validate ideas proposed in the previous sections by illustrating how two previously developed ENS's, SimpleNS and Inspire, can be re-implemented.

SimpleNS is a communication and process support oriented ENS which does not offer analytical support to the negotiators. It has been developed for teaching and comparative studies on the use and effectiveness of different ENS's (http://mis.concordia.ca/SimpleNS). It provides a virtual negotiation table which allows its users to exchange offers and messages. This system displays the negotiation case and other information required to conduct the negotiation, presents a form in which users write messages and offers, and shows the negotiation history in which all messages and offers are displayed in one table with the time they were made. It has been used in teaching at the University of Ottawa, Concordia University, Vienna University, Austria and National Sun-Yat Sen University, and Taiwan. The table-based representation of the underlying e-negotiation protocol model emulating the SimpleNS system is presented in Appendix 1.



SimpleNS-test1, -12 day(s)

**Send Offer and Message**

There are two boxes here: a message box and an offer box. The message box is used for free form communication (e.g., ask questions, present arguments and counter-arguments, complain or support). The offer box is used for exchange of offers (complete or partial); which the counter-part may reject, ask for another offer, propose a counter-offer or accept. You do not need to write in both boxes; it is your choice to use only one of them or to use both.

| Use the space below to make an [OFFER] | Use the space below to send a [MESSAGE] |
|---|---|
| | |

© 2005 Invite Negotiations Systems

Figure 5. Screen shot of the SimpleNS system implemented on the Invite platform

Table 1. Reusability of Page-level Components

| Page-level components | SimpleNS | Inspire |
|---|---|---|
| Read public case | Used | Used |
| Read private case | Used | Used |
| Send (structured) offer | Not used | Used |
| Send (unstructured) offer | Used | Not used |
| Read (structured) offer | Not used | Used |
| Read (unstructured) offer | Used | Not used |
| Send message | Used | Used |
| Read message | Used | Used |
| View (structured) history | Not used | Used |

Figure 5 shows the screen shot of the SimpleNS system implemention on the Invite platform. The screen is the page for constructing an offer and/or message. The user is in the exchange_offer sequence, where the read_public_case, read_private_case, construct_offer_and_message, and view_history components are optional states. The main content of the page in Figure 5, is the result of invoking the send_offer_and_message component, and the links on the right column indicate accessible components from the current page send_offer_and _message. These links are generated by getting al the optional states of the current sequence from the negotiation protocol database.

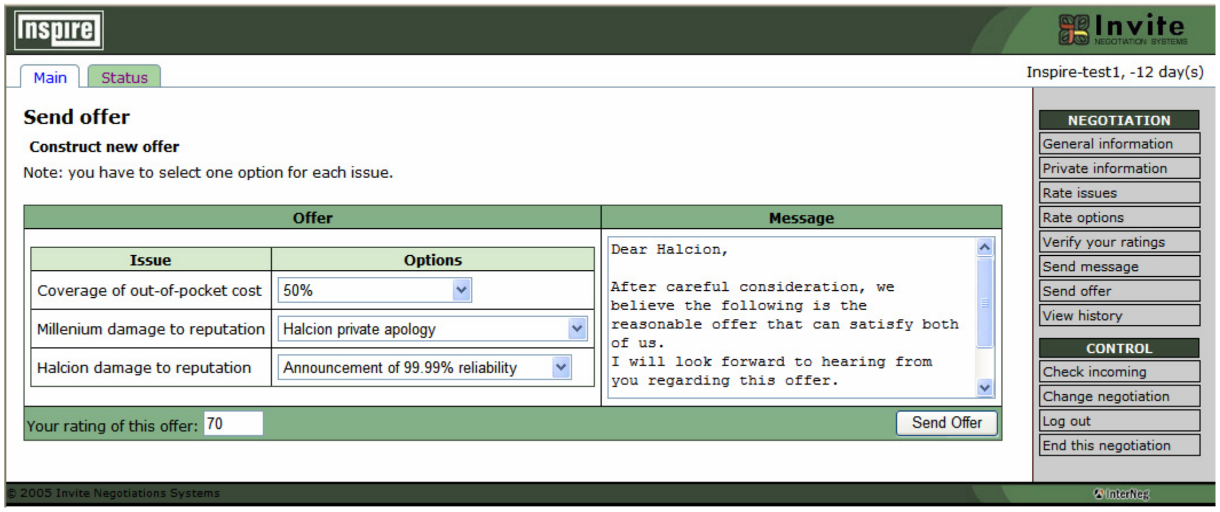
Table 2. Reusability of MVC Level Components

| Type | Role | Used by (page-level components) |
|---|---|---|
| Content query (Model) | Reading the case | Read public case<br>Read private case |
| Content format<br>(View) | Formatting the case | Read public case<br>Read private case |
| Content query (Model) | Reading the most recently received offer | Read offer<br>View history<br>Send offer<br>Agreement |
| Content query (Model) | Reading the most recently sent offer | Send offer<br>View history |
| Content query (Model) | Insert a new message | Send offer |

| | | |
|---|---|---|
| | | Send message |
| Computation module (Model) | Check Pareto efficiency | Agreement<br><br>Send post-settlement offer |
| Graphic generation module (Model) | Display the history of interactions | View offer<br><br>Agreement |

Inspire is a bilateral ENS developed by the InterNeg research group based on decision and negotiation analysis theory [12]. Its main purpose is to investigate cross-cultural negotiations and to provide a teaching tool in negotiation courses. Inspire views a negotiation as a process that occurs in a particular context. The system uses a simplified 3-stage process model: pre-negotiation analysis, negotiation, and post-settlement analysis.

In the pre-negotiation phase, Inspire provides tools for preference elicitation based on the additive utility model. The preference elicitation is performed in three steps. First, the user specifies his preferences over pre-defined issues by distributing 100 points among issues. Once the user assigns preference over the issues, he proceeds to assign scores on the options in each issue. The next step of the pre-negotiation is generation of packages followed by the calculation of ratings for these packages and by the user's verification of these ratings. If the user changes the displayed rating values the least-square procedure propagates these changes to all remaining ratings.



Inspire-test1, -12 day(s)

Send offer

Construct new offer

Note: you have to select one option for each issue.

| Issue | Options |
|---|---|
| Coverage of out-of-pocket cost | 50% |
| Millenium damage to reputation | Halcion private apology |
| Halcion damage to reputation | Announcement of 99.99% reliability |

Message:

```
Dear Halcion,

After careful consideration, we
believe the following is the
reasonable offer that can satisfy both
of us.
I will look forward to hearing from
you regarding this offer.
```

Your rating of this offer: 70

Figure 6. Screen shot (the offer construction page) of the Inspire system implemented on the Invite platform

The negotiation phase involves exchange of messages and offers, evaluation of offers, and the review of the progress of the negotiation. In support of the offer exchange activity, the system presents a drop down menu for each issue, so that user can select for each issue only one option. Once an offer has been constructed, the rating is displayed based on the rating function, constructed from user's preferences earlier.

In the post-settlement phase, the system first determines the rating values of the achieved compromise for both users. Then it checks whether the compromise is a Pareto efficient outcome. If not, the system searches for up to five efficient packages and display them to the user so that they can re-negotiate and improve inefficient compromises.

A negotiation in the Inspire system is terminated when (1) an agreement has reached among both parties, (2) one party terminates in any phase during the negotiation, or (3) the period allocated for talk is expired. The e-negotiation protocol model for the Inspire system deployable on the Invite platform can be found in Appendix 2.

Figure 6 shows the screen shot of the Inspire system implemented using the Invite platform. Like in SimpleNS, links on the right column are generated by the optional states defined in the current sequence (i.e. exchange_offer).

## 6. Evaluation

Through the implementation of two different ENS – SimpleNS and Inspire – on a single platform called Invite system, feasibility of the proposed e-negotiation protocol model and system architecture is proved. So far, we had lab experiments of 44 users (22 negotiation instances) for analyzing effects of various decision support tools using the Invite platform. We observed the system is flexible enough to change negotiation protocols or replace decision support tools with only minor efforts. User feedbacks indicate that this is achieved without sacrificing usability.

Reusability of the components is one of the key benefits we experienced during the implementation. Reusability can be found at both page-level and sub-page (MVC) level. Table 1 shows some of the page-level components and how they were used in SimpleNS and Inspire ENS. It shows many components could be reused by SimpleNS and Inspire despite contrasting difference between them.

Table 2 shows how sub-page level components or MVC level components were reused in implementing the Inspire system. We could observe that components corresponding to the model part of the MVC patterns are reused while view components are rarely reused.

Beside reusability, encapsulation of components allowed us to reduce development lifecycle by enabling concurrent development and improving the process of testing, maintenance and upgrade.

## 7. Discussion

Context dependency is one of the key issues that hinder development of general purpose ENS and adoption of ENS in practice. However, context dependency has been an important issue of many other types of information systems. In the database arena, a breakthrough improvement in handling context dependency was achieved by separating the conceptual data model, usually represented by the ER-diagram, from physical implementation of the database system. In the workflow arena, similar was made possible by separating the conceptual process model from the physical implementation layer.

Noting these, we propose to separate conceptual e-negotiation protocol model from the implementation of ENS. As explained in [17], the unique nature of negotiation processes prevents us from directly applying the workflow models to ENS. First, while ENS should deal with unstructured cyclic activities, workflow systems have focused on repetitive execution of a highly structured flow of activities. Second, a process in a workflow system usually assumes cooperative relationships among users while in most negotiations, the relationships between negotiators are somewhere in-between competition and cooperation.

In this paper, we propose an e-negotiation protocol model based on the component-oriented software protocol approach. And we also proposed to adopt MVC design pattern for increasing reusability and reducing system development cycle time. According to this model, an ENS is developed first by designing a high level e-negotiation protocol which controls interactions between users and ENS by executing page-level components, routing the user to a specific page, and generating links to relevant activities. The protocol also controls inter-personal activities based on the rules on how to handle information received from the counterpart.

The separation of the conceptual model from physical implementation of database and workflow was possible only because it is supported by the lower level infrastructure systems, DBMS and WfMS, which bridge them and execute the conceptual model. In order to support the e-negotiation protocol model, we developed a generic purpose e-negotiation platform called Invite platform which can run multiple e-negotiation protocols defined using the proposed model.

Although our work focuses on negotiations, the contribution is not limited to negotiation systems only. It can give a useful model for developing other general web-based applications having similar nature such as collaboration and group decision support systems.

Table-based representation of e-negotiation protocols are database and development friendly, but not very user friendly. One of the important future works to be done is to develop tools for visual representation and modeling of e-negotiation protocols. We are studying methods to map visual representation into the table-based representation. GUI tools for visual design of e-negotiation protocols will be implemented based on the method.

## 8. References

Foroughi, A. and M.T. Jelassi. NSS solutions to major negotiation stumbling blocks. in Proceedings of the 23rd Annual Hawaii International Conference on System Sciences. 1990. Hawaii.

Foroughi, A., A Survey of the Use computer Support for negotiation. Journal of Applied Business Research, 1995.

Lim, L. and I. Benbasat, A Theoretical Perspective of Negotiation Support Systems. Journal of Management Information Systems, 1992. 9(3): p. 27-44.

Lim, J., An experimental investigation of the impact of NSS and proximity on negotiation outcomes. Behaviour & Information Technology, 2000. 19(5): p. 329-338.

DeSanctis, G. and R.B. Gallupe, A Foundation for the Study of Group Decision Support Systems. Management Science, 1987. 33(5): p. 589-609.

Foroughi, A., W.C. Perkins, and M.T. Jelassi, An Empirical-Study of An Interactive, Session-Oriented Computerized Negotiation Support System (NSS). Group Decision and Negotiation, 1995. 4(6): p. 485-512.

Starke, K. and A. Rangaswamy, Computer-mediated Negotiations: Review and Research Opportunities, in eBusiness Research Center Working Paper. 1999, The Smeal College of Business Administration, The Pennsylvania State University: University Park, PA.

Kersten, G.E., E-negotiation systems: interaction of people and technologies to resolve conflicts. The Magnus Journal of Management, 2004. 1(3): p. 71-96.

Yuan, Y. and J.B. Rose, A Web-based Negotiation Support System. Electronic Markets, 1998. 8(3): p. 13-17.

Kersten, G.E. and S.J. Noronha, WWW-based Negotiation Support: Design, Implementation, and Use. Decision Support Systems, 1999. 25: p. 135-154.

Schoop, M., A. Jertila, and T. List, Negoisst: A Negotiation Support System for Electronic Business-to-Business Negotiations in E-Commerce. Data and Knowledge Engineering, 2003. 47(3): p. 371-401.

Yuan, Y. and O. Turel, A business model for e-negotiation in electronic commerce, in Working paper INR 02/04, I.R. Group, Editor. 2004, http://interneg.concordia.ca/interneg/research/papers/index.html.

Wikipedia, Protocol (object-oriented programming), GNU, Editor, http://en.wikipedia.org/wiki/Protocol_%28object-oriented_programming%29.

Berners-Lee, T. and R. Cailliau, World Wide Web: Proposal for a HyperText Project. 1990, http://www.w3.org/Proposal.html.

Java, Model-View-Controller pattern. 2006, http://java.sun.com/blueprints/patterns/MVC-detailed.html.

Shadovitz, D., ColdFusion and Software Engineering, in Presentaiton slides at SCCFUG-LA. 2002.

Kim, J., et al. Towards a Theory of E-negotiation Protocols. in Group Decision and Negotiation. 2005. Vienna, Austria.