Basha Systems LLC
117 Oneida Avenue
Croton on Hudson, NY 10520
914.827.9173
sgr@bashasys.com

Basha Systems LLC

Bringing Up Rita – The Re-Education of a Lawyer

Artificial Intelligence Modeling of Complex Legal Transactions

Presented by Seth G. Rowland, Esq. on March 16, 2002 at International Association for Artificial Intelligence and Law's 2002 Practical AI Workshop

Bringing Up Rita – The Re-Education of a Lawyer

Artificial Intelligence Modeling of Complex Legal Transactions

Prologue

Lawyers and AI programmers speak different languages and approach problems from different perspectives. In attempting to build expert drafting systems that model the legal thinking process and produce legally acceptable documents, the AI programmer must adjust his thinking and at the same time readjust the lawyer's thinking.

Rita is a lawyer. Not just any lawyer, but a brilliant lawyer at the top of her field. Her specialty is the creation of complex security instruments, known to others by the shorthand, "derivatives." She has the ability to grasp complex financial trends, the nuances of international tax, the hedging of risk, and build a collection of transactional documents that effectuates the goals of her clients.

Rita understands the meaning (and nuance) of every word in the derivatives she creates. She knows all the moving parts. She knows the implications of changes in one passage and how they would ripple through hundreds of other passages and dozens of other documents. As a junior partner at a major law firm, Rita is tasked with educating a group of associates to produce the lucrative transactional documents for her well-heeled financial clients.

One of Rita's clients offers her the opportunity to bid on all of its derivative work if Rita can do the work for a flat transactional fee. This promises to be a 10-fold increase in work that would overwhelm Rita's current tangible resources (i.e. bodies). A survey of her historical billing records for these derivatives shows that even if she had the bodies, the proposed fee (20% below the average historical fee for production of the derivatives) would put her department at risk and threaten the current profit margin.

Rita attends a session on artificial intelligence and the law and decides to work with a consultant to build an expert drafting system to model the creation of the documentation for her client's derivatives. She recognizes the substantial investment of her time that will be required as well as capital resources. She also recognizes that such an expert system will let her handle a 10-fold increase in work without increasing her legal body count. Combining the flat-fee with the volume guarantee of her client, this could make her team of lawyers the most profitable team at her firm.

So begins the education of Rita.

Linguistics and Thought Processes

Rita meets retain Marc, an experienced AI programmer. In their initial discussions, Rita explains the documentation to Marc. Rita starts with a broad conceptual discussion, explaining financial markets, the different parties to the transaction, and the different types of debt instruments. She explains the tax provisions that permit these transactions to function in a tax free status. She explains how the hedges work. Then Rita presents Marc with hundreds of pages of documentation. She proceeds to go through the documents in minute detail, as she would with a junior associate. Marc interrupts her and begins her education.

Lawyer Meets Programmer:

To a programmer the classic legal approach to documentation is fuzzy and soft. There are no hard and fast rules. At each paragraph, sentence, and phrase, the language is reexamined and crafted. There are shades of gray and nuance in every phrase. The inflexion of a sentence can be vitally important. Some statements are made in the affirmative. Other requirements are effectuated by the absence of a negative.

Contract language that appears precise to a lawyer is hopelessly imprecise to a programmer. The precision that comes from an understanding of the other moving parts of a document is not self-evident to the programmer. For example, many lawyers define terms in context, rather than in a section called "Definitions". Lawyers sometimes purposely foster ambiguity in documents through imprecise language. Also, since many documents are evolutionary, concepts can be improperly labeled. Rather than creating new section numbers, lawyers will often add proviso on proviso.

When it comes to extracting the "Rules", the programmer will discover there are multiple and overlapping conditions. Some of the conditions will be "nested" whereas others will just be "concurrent". There will often be errors in the logic that have been perpetuated. In sets of documentation that can run over 300 pages, there are bound to be mistakes that get perpetuated, most of them inconsequential (but some important). Since a programmer needs to address every single clause, the AI programmer is bound to find most if not all of these inconsistencies in the development process.

Furthermore, there is a complex thread of consequences that ripples through the entire documentation. The document is like a complex tapestry where the lawyer and AI programmer need to follow every thread. It is the blend of overlapping threads that produces the tapestry. Once the threads are separated and the patterns discerned, the real work begins. Many of these patterns are non-evident on the face of the tapestry.

For a lawyer, the "form" of the document, not just the words, is also vitally important. The document must look "official". Indices and a table of contents must be produced. Each lawyer has his own unique paragraph numbering scheme. Many a project has been considered a "partial failure" because the document required cleanup by a secretary.

Finally, and perhaps most frustrating to the AI programmer, is the ad hoc manner in which the documents were created. Lawyers react to client requests and changes in the law, building options into their master template to address known scenarios or past scenarios. Lawyers are reactive, not proactive.

Lawyers generally do not build options into their documents to address the hypothetical, for the simple reason that they are paid by the hour to address the current need. This is not to say that lawyers are not forward thinking and don't think about hypotheticals. Rather, the current structures do not compensate a lawyer for drafting a complex financial instrument that will not be used.

Programmer Meets Lawyer:

To a lawyer, the classic programming approach is rigid and cumbersome. Programming elevates rules above text, and then reduces the rules to symbols. Programming requires rigid adherence to proper syntax. Programming is a foreign language that seems ill-suited to the nuanced legal approach.

Syntax errors will cause an application (or template) to fail. Templates will contain "code". Blocks of code will be reduced to "subroutines". A programmer will then use the "subroutines" as shorthand for blocks of code.

A programmer will also building in error checking to constrain mistakes and prevent the template from failing.

The template must be marked to show every rule and every result. Rules must be explicit, returning either a "true" or a "false". There will be a consequence for every rule. The absence of a consequence *will* leave a hole in the document. Rules are evident, instead of non-evident. The syntax will often be cumbersome and hard to follow.

Programming is "structured". For every IF statement, there is or should be an ELSE statement, or an awareness that no ELSE statement is required. When it comes through handling nested conditions, the programmer needs to unpeel the tapestry and rebuild it in exactly the right order. Minor changes should allow the application to build an entirely different tapestry without producing a jumble of threads.

Finally, programming is by definition proactive. In programming you are constantly asking, "What if …." The programmer needs to address all scenarios and find the appropriate textual consequences for combinations of choices that have not been made.

Education begins in Ernest:

Rita's background explanation on the derivatives transaction was informative to Marc. However, Marc saw that the detailed explanation of the documents would not be a fruitful discussion unless Marc focused the discussion on the information he needed to build the application.

Markup Techniques:

Marc explained that he needed to understand the "moving parts" of the transaction. If a portion of the documentation never changed, Marc had no need to understand it. In fact, much of the lesson in Derivatives 101 would never find its way into the expert drafting system because those lessons applied to "boiler plate" that would be found in all variants of the transaction.

Marc asked Rita to step outside the text of the document and develop "business rules" that described the preconditions to the legal language that would change. Marc asked Rita to review the documentation and mark all text that was optional or subject to change. For each block of text, Marc asked Rita to create a footnote or endnote that explained in plain English the circumstance under which that text was included or excluded. Similarly, where there were blanks or transaction specific details, Marc similarly asked her to drop a footnote that explained how the blank would be filled in. And with respect to exhibits and ancillary documents, Marc asked her to define the conditions under which such documents were to be included.

Marc asked that Rita be explicit in her explanation of the business rules. He suggested she could use "shorthand" for the rules, but admonished her to be consistent, and describe the rules consistently across the documents. Marc asked Rita to send the annotated documentation to him and promised to schedule a follow-up meeting a week after he received the documentation. At this meeting, the real work would begin.

A Lesson in Abstraction:

Marc met with Rita a week after he received her mark-up. Marc handed Rita a spreadsheet printout. He had organized all her business rules on a spreadsheet. He had grouped them together by concept. Marc had taken the liberty to restate the business rules in the form of questions or prompts and to assign to those rules a variable code (or codes). Where the rules were composites or nested, Marc had also defined the circumstances under which the business rules would be asked.

The spreadsheet also contained a number of inquiries. In refining the business rules, Marc had discovered gaps in logic and inconsistencies in the mark-up. These gaps were raised at the abstract level.

Together, Marc and Rita reviewed the spreadsheet, further refining the rules. There were areas where Marc had addressed scenarios which would never occur. These options were paired down. There were also areas where there were gaps in the documentation. Rita would have to revise her documentation to address those gaps. However, there was no requirement for the documents to be revised at this stage. Marc would put placeholders in the template to keep the project moving.

Where Rubber Meets the Pavement:

Marc and Rita also discussed the document. Some of Marc's refinements would require certain passages to be rewritten. For example, definitional language might be moved to a central location. Certain provisos would be moved into subparagraphs. Descriptions of parties, assets and other collections might be changed syntactically.

Rita reviewed the spreadsheet carefully. She was concerned that she could understand the "code". There was discussion of variable naming schemes. Every prompt was revised and clarified. Rita stressed that the coding be "transparent" so that any lawyer with a "key" could understand the template. The key or "Rosetta stone" would include the "code" and the business rule in plain English.

Most important, Marc and Rita defined a series of special business rules to handle combinations of conditions. This was done to simplify the coding of the template so that Rita could quickly identify the business rules surrounding optional text. The special business rules, called "computations" or "objects" would go a long way to hiding and simplifying the programming syntax.

Rita also stressed the resulting format of the assembled document. The document must conform to "legal standards". There should be a minimal amount of post-assembly word processing.

Working Together

Over the next few weeks Marc and Rita developed a common language. Through the use of the Rosetta stone embodied by the spreadsheet, Rita revised the document markup and added language where omissions had been pointed out by Marc.

Marc in turn refined his coding rules and built a collection of information gathering dialogs. He applied all the business rules discussed to properly thread his way through the prompts.

Marc met with Rita to review and refine the dialogs. He then took Rita's revised markup of the documentation and applied the codes to the templates to finish the application. Over the next week, Rita and Marc had several phone calls to reviews issues that arose as Marc coded the documents. Some of these were done over a remote PC session so both parties could look at the language together.

Since Marc and Rita had developed a common language and approach, the conversations were brief and focused. In short order the document set was completed and the application delivered.

Epilogue:

Rita developed an intake questionnaire to gather transactional information from her client. Her client filled out the questionnaire on-line. One of Rita's associates would contact the client to clarify any discrepancies in the answers given and to address any additional requirements not addressed in the questionnaire.

The associate would then process the answers through the expert system Marc and Rita had built. The system would produce an abstract of the transaction and the complete set of documentation in short order. The associate would review the abstract and transaction with Rita, make further refinements and turn the document around the same day.

Needless to say, it was a happy result for everyone.