Supporting Negotiation and Dispute Resolution with Computing and Communication Technologies

Leon J. Osterweil
University of Massachusetts
LASER
Dept. of Computer Science
+1.413.545.2186
ljo@cs.umass.edu

Lori A. Clarke
University of Massachusetts
LASER
Dept. of Computer Science
+1.413.545.1328
clarke@cs.umass.edu

ABSTRACT

Careful study of the application of computer and communications technologies to negotiation and dispute resolution can lead to a more harmonious and functional society, but also to more effective technology for software development and evaluation, and to new facilities for pursuing social science research. Work in this area has initially been spurred on by rapid growth in the number and variety of disputes in the world, which seem to be increasing as the number of parties grows and the variety of their interactions increases. Initial application of software technology to support dispute resolution has shown encouraging success, and suggests a broader research program in which computer science and social science research enrich each other. The resulting mature technological support for dispute resolution should, moreover, have important benefits for software development, a domain in which constructive approaches to resolving disputes should be of enormous value.

Categories and Subject Descriptors

D.2.9. Software Process Models, H.1.2. Human information processing, H.5.3 Collaborative Computing, K.4.3. Computer Supported Collaborative Work

General Terms

Management, Design, Experimentation, Human Factors, Languages.

Keywords

Dispute resolution, process definition and execution, human-computer collaborative systems.

1. INTRODUCTION AND BACKGROUND

The existence of disputes seems to be inherent to the human condition. Disputes typically arise in dealings between labor and management, between parents and children, between plaintiffs and defendants, between the stakeholders in the development of a software system, between political parties, and the members of those parties, between the users of a web site, and between the governments of different countries. As the number of humans, and groups of humans, increases, the number of disputes can be expected to increase. Moreover, as the kinds of interactions among humans become more diverse (e.g. as interactions through the internet proliferate) the variety of disputes can also be expected to increase. In particular, note that while most of the disputes just described typically involve only two parties, some of them (e.g. disputes among governments and disputes among software system developers) are more likely to involve multiple parties.

Disputes are not necessarily bad or harmful. A dispute simply indicates the existence of a difference of opinion, perception, or perspective. A dispute that remains unresolved can indeed be harmful if it leads to anger and hostility, and becomes an obstacle to progress. But a dispute can be distinctly advantageous when its resolution leads to the effective synthesis of important ideas and the bringing to bear upon a problem of a number of different ideas in a way that might never have emerged from the mind of any single person. Thus effective contributions to facilitating dispute resolution can be expected to be of great value to society.

There have been a number of approaches to dispute resolution. Most commonly disputes have been resolved by having the disputing parties meet face-to-face to have conversations and discussions aimed at resolution. Often resolution guidance is provided by human mediators, trained in how to facilitate resolution. Organizations as such as the US National Labor Relations Board (NLRB) and the US National Mediation Board (NMB) have been set up to institutionalize support for the resolution of particularly complex and important disputes. Their mediators often use approaches such as brainstorming and Interest Based Bargaining (IBB) to facilitate their efforts [1].

As the number and complexity of disputes has grown, however, these organizations have recognized the need to support human dispute resolution experts with technology. The NMB, for example is discovering that the use of modern communications can facilitate dispute resolution even when the disputing parties are separated in time, or in place, or both. Data processing facilities can provide further support by expediting the management of the exchange of data between the parties to disputes. The term Online Dispute Resolution (ODR) has been applied to the approach of applying computer and communications technology to the support of dispute resolution

FoSER 2010 November 7-8, 2010, Santa Fe, New Mexico, USA.
Copyright 2010 ACM 978-1-4503-0427-6/10/11…$10.00.

[2]. Technology has been used more dramatically in cases where enormous numbers of disputes must be resolved. Thus, for example, internet-based companies such as eBay exploit technology guidance for dispute resolution to resolve millions of disputes every day [3]. While this success is notable, it seems to derive substantially from the fact that the large majority of eBay disputes fall into one of a very small number of categories, each of which is typically relatively straightforward to understand and guide by software. It seems interesting and important to investigate the extent to which technology might be used to support the resolution of the far wider variety of disputes that are typically more complex and challenging. Indeed this seems particularly significant for software development, as it is increasingly recognized that software development is an activity in which disputes abound. Approaches to expediting constructive resolutions to software development disputes could greatly facilitate such key activities as requirements specification, test and evaluation, and selection of overall development approaches.

Some early results of using technology to support the resolution of complex disputes have been encouraging. The NMB is discovering that processes such as brainstorming and IBB can be codified rather clearly and precisely using process definition languages [4]. In addition, automated execution of these processes seems to be effective in providing support for performing these processes [5]. This has demonstrated the promise of using software engineering ideas and technologies to provide support for dispute resolution activities.

Important additional research must be addressed, however, in order to create the technologies that are needed for realizing the promise of effective software support for resolving such complex disputes. Among the issues to be explored are:

•What approaches to dispute resolution seem well suited to which kinds of disputes?

•Which details of which dispute resolution processes are really important, and which are not?

•What should be the role of human dispute resolution agents, and what jobs are better left to automated (e.g. software) systems?

• Under what circumstances is face-to-face bargaining important, and when may it be counterproductive?

•What should be the role of privacy and anonymity in fostering frank expression of opinions?

We note that social scientists have long been interested in questions such as these, and thus computer scientists and software engineers would be joining an important longstanding research area. It is important to note, however, that computer scientists and software engineers will bring to this research area perspectives, technologies, and tools that seem likely to bring a new dimension to the ongoing social science research. Conversely, preliminary work in this area has demonstrated that while some computer science and software engineering technologies are unexpectedly effective, other technologies seem to have serious shortcomings whose investigation is reflecting important light back on these technologies. In addition, exposing computer science and software engineering researchers to social science research modalities is also demonstrating the relevance of social science experimental research approaches to computer science and software engineering research.

Thus, a vigorous program of research into the application of computer and communications technologies to the broad area of dispute resolution should be of great value to social science in providing it with very promising new research tools, and of great value to computer science and software engineering in forcing needed reconsideration and enhancement to some basic technologies and tools. These advantages are in addition to the advantages that will accrue to society from expedited resolution of its disputes, and to software engineering through facilitation of the basic activities of software development.

2. THE PROMISE OF SOFTWARE TECHNOLOGY

Early success in using process definition, analysis, and execution technologies to support dispute resolution processes such as IBB has encouraged the belief that such technologies could be an effective basis for developing dispute resolution as an engineering discipline. While in the past processes may have been considered not to be amenable to precise definition, recent successes in devising process definition languages has suggested that processes can indeed be defined precisely, and can indeed be considered tangible objects that can be engineered. Early precisely-defined processes were typically aimed at such pursuits as the manufacture of tangible products, or the management of administrative paperwork flows. But more recent research seems to indicate that rigorously defined process definition languages can also be effective in supporting the definition of processes aimed at producing less tangible results, such as the resolution of a dispute. Thus, for example, the Little-JIL process definition language [6, 7, 8] was used to support the definition of the IBB process that is used at the NMB. A system supporting the execution of this process, STORM2 [9], focuses most of its detailed specification upon the Brainstorming part of the NMB's dispute resolution process. STORM2 is an executable system that integrates the activities of participants who represent the disputing parties, a mediator, and software systems such as a repository of participant inputs, and a system for enforcing varying degrees of anonymity on the participants' inputs. STORM2 is currently used as a training aid at the NMB and has supported some negotiations of actual disputes between labor and management.

The need to support NMB's actual IBB process necessitated a number of enhancements to the Little-JIL process language. Some enhancements were necessitated by the fact that, although this process definition language purports to treat both human and non-human participants in the same ways, there are interesting ways in which this is pragmatically difficult or impossible. Thus user interface technologies had to be created and used to augment the language. Important enhancements to the Little-JIL runtime execution system had to be made in order to assure appropriate separation of concerns that would facilitate the interactions between human users and process definitions. This work strongly brought home some key difficulties that needed to be resolved in making the desired synergy between humans and automated devices a reality. In addition, the flow of control in NMB's IBB approach, which seems quite straightforward at first, was in practice far harder to define in a computer language like Little-JIL, necessitating the enhancement of the language with some important new semantic features such as the ability for one process step to preempt the execution of another.

An initial case study project using STORM2 also demonstrated the relevance of some social science evaluation approaches and modalities to software engineering research. All of these have suggested the importance and potential significance of this collaboration between computer science and social science research. The case study project was aimed at determining where, when, and how anonymity should be supported in dispute resolution processes such as NMB's IBB. To investigate this range of questions, STORM2 was enhanced by the imbedding of a software system capable of accepting directives for when to enforce different levels of anonymity and for then enforcing them. The case study entailed repeating the execution of a specific Little-JIL defined process twice, each time with a different anonymity setting. The results of the two executions were then evaluated using qualitative evaluation approaches adapted from social science. Because the two different dispute resolution process executions were driven by the same Little-JIL process definition, social science researchers had the assurance that the only way in which the two process executions differed was in the level of anonymity provided to participants. This added solidity to the evaluation results that would be hard to achieve if a less formally defined process had been used (as is typically the case in social science research). Conversely, the computer science researchers obtained the advantage of using well-established social science research modalities and approaches in evaluating this case study, thereby obtaining understandings of how to carry out this kind of evaluation, which is of considerable importance in supporting the evaluation of much software engineering research that is aimed at facilitating the work of humans.

It was noted, moreover, that the structure of the Little-JIL process definition language particularly facilitated inserting monitoring and instrumentation capabilities into the execution of a process. Such monitoring could potentially be used to gather data, either automatically (e.g. how long did a take a participant to respond to a particular request) or through human action (e.g. by responding to a questionnaire). Thus, the computer science technologies used in STORM2 seemed effective in facilitating the gathering of insights into human reactions to process issues at various levels of process abstraction. This could help identify ways to facilitate the synergy between humans and automated devices that is expected to be of great importance in building the complex heterogeneous system that will increasingly be the backbones of our society in the near future.

3. A RESEARCH AGENDA

The following subsections enumerate a selected set of examples of research activities that might be undertaken, indicating their value to the field of dispute resolution, to software engineering, and to social science.

3.1 Process definition language development

Using a well-defined process definition language to define social science processes such as dispute resolution should serve both to make these processes clearer and also to indicate improvements needed in such a language. It has already been noted that defining NMB's IBB process demonstrated the need for such process definition language features as preemption. Work in this area has also sharpened focus on the importance of artifact specification and management, as a complement to the current primary focus on activity specification in process definition languages. In addition, the need to support actual execution of such process definitions raises many new issues. Some issues relate mostly to human interfaces. But other issues address the need to understand how best to provide for shared control of process execution by human agents and process definitions. We note that there has been a considerable amount of work in modeling processes, but far less work in supporting the actual execution of such process models. The suggested research initiative would make a substantial contribution to the key research needed in this area.

3.2 Qualitative Evaluation applied to Software Engineering research

Software engineering research has long been disadvantaged by the relative difficulty of carrying out statistically significant quantitative research. Especially in areas such as requirements specification and software process, it is difficult or impossible to compare the results obtained from applying alternative technological approaches to the large numbers of projects needed to assure statistical significance. Social science researchers have grappled with analogous problems for quite some time, and have evolved qualitative approaches that should be of interest and value to software engineering researchers. The research program proposed here should acquaint software engineering researchers with these approaches and suggest ways in which they might be synergized with existing empirical research approaches from software engineering to create more effective software engineering experimental research methodologies.

3.3 Synergizing the Actions of Humans and non-Humans in Complex Systems

Dispute resolution is a good example of an important kind of process that works best when it is most effective in synergizing the efforts of humans and automated agents. But it is only one of many such examples. Indeed it seems likely that such systems will among be the most important in the society that is now being created. Thus there should be considerable value in carrying out careful studies of how to build such systems so that they are most effective in assuring that each type of participant is best supporting in doing the kind of work for which it is best suited. The proposed program of research should be useful in providing insights into these higher-level issues by providing some carefully done cases-in-point in the form of systems that provide automated support for the resolution of disputes among humans.

3.4 Facilitating Requirements Engineering

Requirements engineering is an area of software engineering that would seem to benefit substantially from the application of dispute resolution support technology. It is well accepted that a key challenge in arriving at the specification of a solid specification of the requirements for a complex software system is achieving a consensus among the different stakeholders in such a system. Customers, users, developers, regulators, and investors are all examples of groups that are often stakeholders in large software systems. Their views and priorities can be expected to diverge from each other at least in the early stages of negotiation for the requirements of the system. An effective approach to expediting the resolution of disputes arising from disagreements among these stakeholders would seem to be of great value to software development.

The resolution of two-party disputes is currently being facilitated by approaches such as IBB, and systems such as STORM2, as used by the US NMB. These approaches feature the participation of human mediators whose roles include such activities as precipitating the articulation of issue statements, soliciting approaches, and encouraging comments. While these kinds of activities would seem to be relevant to the process of resolving multi-party disputes, it is not clear how the nature of such processes might have to differ. Research in this area would presumably entail the specification of different multi-party dispute resolution approaches patterned after established approaches such as IBB, and then the evaluation of the strengths and weaknesses of these approaches.

The existence of some degree of agreement about the components of a requirement specification (e.g. Functional, Robustness, User Interface, and Performance requirements), and the need for them to be consistent with each other in various ways could provide much-needed structure to processes for resolving disputes in this domain. This is an observation that could greatly facilitate requirements specification, and also help in the better understanding of what it might take to facilitate multi-party dispute resolution in general.

3.5 Elucidating Software Development Process Issues

Software development itself is an example of a social activity in which disagreements are to be expected, and in which effective resolution should be of considerable importance. Here too, disputes might often involve multiple parties, could arise along many different dimensions and might require resolution on an ongoing basis. Thus, technologies and understanding derived from applying dispute resolution to requirements specification would most likely also have strong relevance to dealing with disagreements about overall software development methodologies and approaches.

3.6 Multi-party dispute resolution

Most of the literature and experience with dispute resolution seems to address disputes between two parties. Yet we have noted that there are many important kinds of disputes that involve more than two parties (requirements specification is an important example). Relatively little seems to be known about how to address supporting the resolution of such disputes. Some approaches from two-party resolution might indeed scale up to deal effectively with disputes among larger numbers of parties, but others may not. Initial research should address the question of which approaches can be expected to scale. For those that do not seem to scale, subsequent research will be needed to devise new dispute resolution approaches and technologies.

4. ACKNOWLEDGMENTS

This material is based upon work supported by the US National Science Foundation under grant # CCF-0905530. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the view of the National Science Foundation.

5. REFERENCES

[1] Fisher R, Ury WL, Patton B, Getting to Yes: Negotiating Agreement Without Giving In, 2nd edition, 1992 Penguin.

[2] Katsh, E. and Rifkin, J., Online Dispute Resolution: Resolving Disputes in Cyberspace. Jossey-Bass, San Francisco, 2001

[3] eBay Resolution Center (2010) URL http://resolutioncenter.ebay.com/

[4] Katsh, E., Osterweil, L. and Sondheimer, N.K., Process Technology for Achieving Government Online Dispute Resolution. in National Conference on Digital Government Research, (Seattle, WA, 2004).

[5] Clarke, L.A., Gaitenby, A., Gyllstrom, D., Katsh, E., Marzilli, M., Osterweil, L.J., Rainey, D., Sondheimer, N.K., Wing, L. and Wise, A., Realizing Online Dispute Resolution in a Federal Agency through Process-Based Methods. in The National Conference on Digital Government Research, (San Diego, CA, 2006).

[6] Wise, A. Little-JIL 1.5 Language Report, Dept. of Comp. Scie, U. of Massachusetts, Amherst, MA 2006, http://laser.cs.umass.edu/publications/index.jsp.

[7] Cass, A.G., Lerner, B.S., McCall, E.K., Osterweil, L.J., Sutton, S.M. Jr., and Wise, A.E., Little-JIL/Juliette: A Process Definition Language and Interpreter, 22nd International Conference on Software Engineering (ICSE 2000), Limerick, Ireland, pp. 754-757, June 2000.

[8] Wise, A., Aaron, C.G., Lerner, B.S., McCall, E., J., O. and Sutton, S.M., Using Little-Jil to Coordinate Agents in Software Engineering. in Automated Software Engineering Conference, (Grenoble, France, 2000), IEEE Computer Society 155.

[9] Simidchieva, B.I, and Osterweil, L.J., STORM2: Process Guided Online Dispute Resolution, Dept. of Comp. Sci. U. of Massachusetts Tech. Report, Amherst MA 2009, http://laser.cs.umass.edu/publications/index.jsp