https://doi.org/10.48009/3_iis_2017_53-61

Issues in Information Systems
Volume 18, Issue 3, pp. 53-61, 2017

SOURCES OF CONFLICT BETWEEN SOFTWARE DEVELOPERS AND TESTERS: A COMPARISON AND AN INTEGRATION OF TWO MODELS

Xihui Zhang, University of North Alabama, xzhang6@una.edu
Tony J. Hunkapiller, University of North Alabama, thunkapiller@una.edu

ABSTRACT

Conflict between software developers and testers is inevitable and has important implications to software development outcomes. Understanding sources of such conflict, therefore, is critical for improving the effectiveness of its management with the ultimate goal of improving the quality of software development outcomes. In the information systems (IS) literature, there are two leading models of sources of conflict between software developers and testers. In this paper, we compared these two models and developed an integrated model from them. It is our hope that IS researchers and practitioners can use this integrated model to better understand the sources of conflict between software developers and testers. Increased understanding of the sources of such conflict is fundamental to developing strategies and tactics for managing it, ensuring positive developer-tester working relationships, efficient application of managerial effort, and more consistent achievement of both software development effectiveness and organizational outcomes.

Keywords: Conflict, Interpersonal Conflict, Conflict Sources, Software Development, Testing

INTRODUCTION

Conflict between software developers and testers is inevitable and has important implications to software development outcomes (e.g., Barki & Hartwick, 2001; Cohen et al., 2004; De Dreu & Weingart, 2003; Dhaliwal et al., 2011; Jehn, 1995; Jehn et al., 1997; Moeller et al., 2012; Sawyer, 2001; Wang et al., 2005; Zhang et al., 2013; Zhang et al., 2014). Understanding sources of such conflict, therefore, is critical for improving the effectiveness of its management with the ultimate goal of improving the quality of software development outcomes.

In the information systems (IS) literature, there are two leading models of sources of conflict between software developers and testers. The first conflict model, proposed by Cohen et al. (2004), has three conflict layers: *Process, people, and organization*; for each layer, two conflict sources are identified, resulting in a total of six conflict sources. The second conflict model, proposed by Zhang et al. (2014), has three categories: *Communication, people, and process*; the three categories were then further divided into five subcategories.

Two research questions arise: (1) What are the similarities and the differences between the two leading models of sources of conflict between software developers and testers? (2) Is there a way to consolidate the two leading models into an integrated model? As such, this research has two objectives: (1) to better understand sources of conflict between software developers and testers by comparing the two leading models in the IS literature, and (2) to build an integrated model of sources of conflict between software developers and testers by consolidating the two leading models.

Our research goal is to learn the sources of conflict between software developers and testers. Based on this knowledge, what we hope to accomplish is to build an integrated model of sources of conflict between software developers and testers so that IS researchers and practitioners can better understand the sources of such conflict. Increased understanding of the sources of such conflict is fundamental to developing strategies and tactics for managing it, ensuring positive developer-tester working relationships, efficient application of managerial effort, and more consistent achievement of both software development effectiveness and organizational outcomes.

RELATED LITERATURE

Even though research studies on conflict are abundant (e.g., Barki & Hartwick, 2001; Cohen et al., 2004; De Dreu & Weingart, 2003; Dhaliwal et al., 2011; Jehn, 1995; Jehn et al., 1997; Moeller et al., 2012; Sawyer, 2001; Wang et al., 2005; Zhang et al., 2013; Zhang et al., 2014), those with a focus on conflict sources, especially in the context of software development and specifically between software developers and testers are scarce. Not all references cited in this paper are of equal importance to this research, and the three most important ones are: Cohen et al. (2004), Moeller et al. (2012), and Zhang et al. (2014).

In the Introduction section, we have described the content of Cohen et al. (2004) and Zhang et al. (2014), which contain the two leading models of sources of conflict between software developers and testers. The first model is from the work of Cohen et al. (2004). By analyzing in-depth field interviews with 10 software testing professionals, they categorized the sources of conflict between software developers and testers into three conflict layers: *Process, people, and organization* (see Figure 1). For each layer, two conflict sources were identified, resulting in a total of six conflict sources (see Table 1).

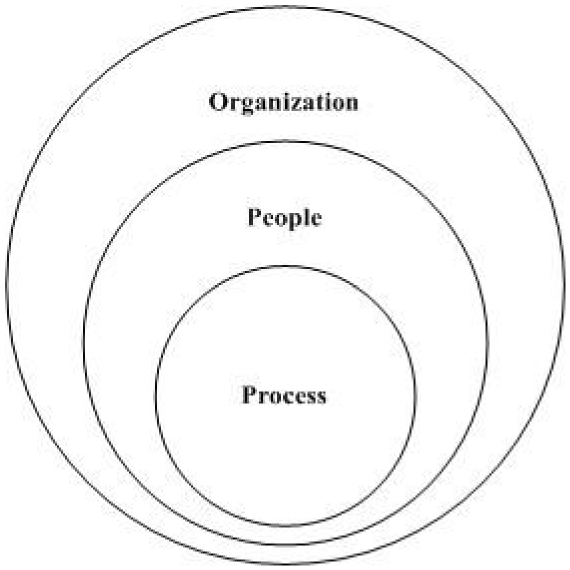

Figure 1. Cohen et al. (2004) Conflict Model

Table 1. Sources of Conflict (from Cohen et al., 2004)

Layer	Sources of Conflict
Process	Developer and tester competition for the scarce resource of project time
	Testers focusing more on user requirements versus developers focusing more on technical requirements
People	Differences in tester versus developer "mental process and personality attributes" related to the process of software development
	Many developers being too protective of their code when testers find defects in it
Organization	The role of differential power and politics, e.g., for some testers, "the struggle for recognition becomes part of the job itself"
	The critical role of managers, e.g., managers who set the tone for the importance of testing and use a collaborative working style make a difference

The second model is from Zhang et al. (2014). They collected 50 written conflict scenarios from both software developers and testers, of which 43 were validated by subsequent methodological steps. Their data analysis resulted in the development of an alternative three-category conceptual structure: *Communication, people, and process* (see Figure 2). The three layers or categories were then further divided into five subcategories based on the results of two rounds of Q-sort and an online survey (see Table 2).

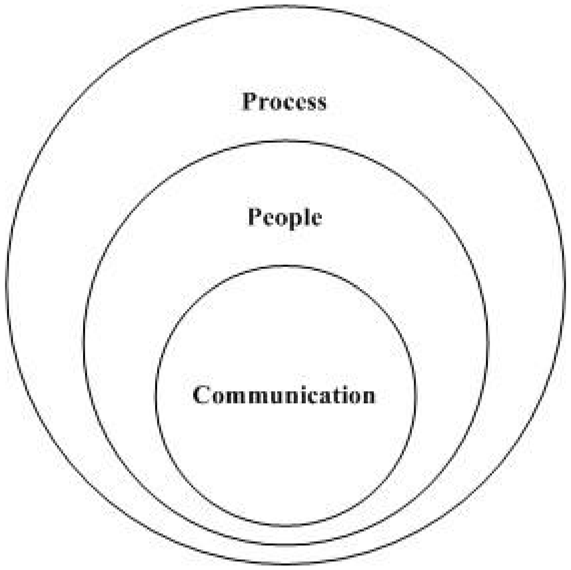

Figure 2. Zhang et al. (2014) Conflict Model

Table 2. Categories, Subcategories, and Summaries of Conflict Scenarios (from Zhang et al., 2014)

Category, Subcategory	Summary of Conflict Scenario
Communication, Knowledge Sharing	Developer refuses to share knowledge to support assertion that a defect is invalid.
	Errors not detected because test lead was left out of the communication loop for code maintenance.
	During test, developer and tester disagree over coding standards, which were not discussed during prior reviews.
	Developers deliver non-test ready hardware because testers failed to communicate their expectations.
	Late/unclear communication of new ‘‘release to production’’ criteria results in disagreement about implementation readiness.
	Lack of communication between developer and tester results in delay and poor work quality.
	Developer fails to communicate status of incomplete code; results in test failure.
	Intermittent testing failures result from multiple code changes that are not communicated to the test team.
	Developer does not inform testing of code changes, resulting in invalid test cases and test failure.
People, Emotions/Attitude/Personality	Developer’s emotional outburst challenges test design, resulting in abandonment of level 2 testing.
	Developers dislike working with tester who is perceived as gloating and bragging about critical defects.
	During code review, a heated argument results when developer reacts emotionally to identification of defects in ‘‘your code.’’
	In response to a request for clarification, developer makes a belittling remark about tester’s professionalism.
	Tester’s accusatory manner in reporting an environmental issue triggers development team reluctance to acknowledge responsibility.
	Developer believed to have a low opinion of testers refuses to assist in designing a specially assigned test of his code.

	During testing, developer hovers, makes intrusive verbalizations, and refuses to leave tester's workspace when requested.
	Developer rudely rejects tester's request for a test tool option.
People, Knowledge/Experience	Tester opens invalid defects due to lack of application knowledge.
	Tester lacks understanding of the requirements.
Process, Documentation	Developer fails to document defect fix and claims tester error.
	Invalid defect results from undocumented development change in test execution procedure that was not communicated to the test team.
	Inadequate requirements result in tester request for changes after code completion.
Process, Process Compliance or Noncompliance	Developer fails to follow the correct defect management process.
	During a walkthrough, testers criticize design and code and recommend compliance with a coding standard.

Moeller et al. (2012) provided a critical analysis of the antecedents of interpersonal conflict in information systems development (ISD), an almost identical context as in software development. In their proposed "ISD Interpersonal Conflict Model," they maintain that conflict antecedents include: (1) Project characteristics (requirements, resources, visibility, risk), (2) Structural characteristics (organization), (3) Cultural characteristics (organization, team), and (4) Individual characteristics (perceptions, expectations, attitudes, personality, values, demographic, education, etc.). In their model, they also proposed some moderating factors to the relationship between conflict antecedents and interpersonal conflict, including (1) Team structure (roles, role relationships), (2) Project process (project management, documentation), (3) Team communication (frequency, quality/effectiveness), and (4) Individual behavior (leadership, conflict management, other behaviors). In this research, we plan to use the research findings of Moeller et al. (2012) as an overarching conceptual model when attempting to integrate the two leading conflict models of sources of conflict between software developers and testers proposed by Cohen et al. (2004) and Zhang et al. (2014), respectively.

COMPARISON OF TWO MODELS

In this section, we will compare the two leading models of sources of conflict between software developers and testers proposed by Cohen et al. (2004) and Zhang et al. (2014). The comparison will be comprehensive, both broad and deep. The dimensions to compare of the two research studies include: motivation, design, method, results, as well as theoretical and practical implications.

Cohen et al. (2004) conducted in-depth field interviews with 10 software testing professionals from four different U.S. companies: two large, one mid-size, and one small. These interviews led them to categorize three basic conflict layers - *process, people, and organization* (see Figure 1), as mentioned in previous sections.

Cohen et al.'s (2004) study is interesting and their findings are informative. However, their study has three limitations: (1) the interviewees were all testers, which makes it very likely that only one side of the story was covered, (2) the research lacks a theoretical basis, which makes it difficult to advance knowledge in the conflict domain, and (3) the findings are not backed up with quantitative data, which makes it hard to assess their reliability and validity. These limitations suggest that additional studies are needed.

Inspired by Cohen et al. (2004) and attempting to overcome the limitations of its research design, Zhang et al. (2014) took a different approach to address the same research question. They collected 50 developer-tester conflict scenarios from both software developers and testers and used content analysis techniques to broadly sample the domain of potential constructs of developer/tester conflict. They then used two rounds of Q-sort analysis to distill underlying dimensions of conflict sources, and finally conducted an online survey for confirming and validating purposes. Their data analysis resulted in the development of an alternative three-category conceptual structure: *Communication, people, process* (see Figure 2).

The research design of Zhang et al. (2014) was significantly influenced by Cohen et al. (2004); therefore, it is instructive to compare methodology and results between the two studies. In both studies, a qualitative approach utilized open-ended questions to gather information about conflict experiences from professionals involved in software development projects. While Cohen et al. (2004) conducted in-depth field interviews with 10 software testing professionals, Zhang et al. (2014) collected 50 written conflict scenarios from both software developers and testers, of which 43 were validated by subsequent methodological steps. Based on their analysis, Cohen et al. (2004) constructed the three-layer conflict model depicted in Figure 1. Similar to Zhang et al. (2014)'s process layer (Figure 2), "organization" appears to function as a contextual base for the remaining layers. In contrast to Zhang et al. (2014)'s model, however, "process" is contained within "people," and "communication" in Zhang et al. (2014)'s study is not featured as a major conflict source.

We believe that both Zhang et al. (2014)'s data (in terms of sample size and characteristics) and more particularly, their model represent a significant advance from the departure point provided by Cohen et al. (2004). Zhang et al. (2014)'s 50 written scenarios from both developers and testers employed by industry-leading companies compare favorably with the earlier study's interviews which involved only 10 software testing professionals. Regarding model construction, Cohen et al. (2004) do not provide a rationale for the spatial relationship of their three conflict layers (see Figure 1) or the nature of the relationships between them, nor are the conceptualizations informing the arrangement readily inferred. While "organization" appears to function legitimately both as a major conflict source and as an overall context within which team conflict arises, positioning of the other two layers is problematic because it is difficult to support the assertion that "process" is primarily a function or subset of "people." Rather, people act (or fail to act) in compliance with process, a key organizational feature. Nor is it inevitable that a formal process will be significantly altered if the people assigned to a project are replaced; the usual expectation is that process will remain relatively stable regardless of team composition. This suggests that the spatial relationship structure should be altered so that "process" functions as the second layer, contained by "organization" while constraining the behavior of "people."

It is true that in their capacity as structural agents (Giddens, 1986) project team members can and often do function as owners of (rather than just clients constrained by) formal or informal processes existing at a variety of levels: organizational, departmental, project, and subteam. It is also true that when an established process does not adequately serve the needs of a particular project, or when incompatible processes interfere with team effectiveness, team members may choose to solve the problem with formal or informal process innovations. Just the same, vis-à-vis Cohen et al. (2004), Zhang et al. (2014)'s layer ranking recommendations stand: by its very nature, organizational politics influences process, and process constrains individual behavior so as to produce predictable results. That is the entire purpose of creating, documenting, and enforcing the process.

A final contrast between Zhang et al. (2014)'s study and Cohen et al. (2004) is presented by Zhang et al. (2014)'s finding (consistent with the general conflict literature) that communication functions as a major source of conflict between software developers and testers. However, as noted by Zhang et al. (2014), all but one of the communication-related scenarios appeared to be symptoms of an underlying cause: lack of unified, appropriately designed and effectively applied project structure and process. Thus, through second layer mediation ("People," Figure 2), process gaps or flaws can negatively impact project communication, arguably rendering suspect the decision to classify such issues as arising from the "communication" layer rather than the "process" layer. It also worth noting that Zhang et al. (2014)'s classification decision was guided by the results of two rounds of Q-sort followed by an online survey, and the online survey was conducted for confirmation and validation purposes.

DEVELOPING AN INTEGRATED MODEL

In this section, we will attempt to integrate the two leading models of sources of conflict between software developers and testers proposed by Cohen et al. (2004) and Zhang et al. (2014). The integration process will use the analysis we have provided from previous sections, especially in the section of "Comparison of Two Models." We will analyze each and every conflict layer or category. We will rename categories or layers if necessary. All these tasks are performed under the guidance of results from our comprehensive literature search and our critical literature review.

An intuitive way to integrate these two models is to create a four-layer conflict model. The four layers are (from inside out): *Communication, people, process, and organization* (see Figure 3).

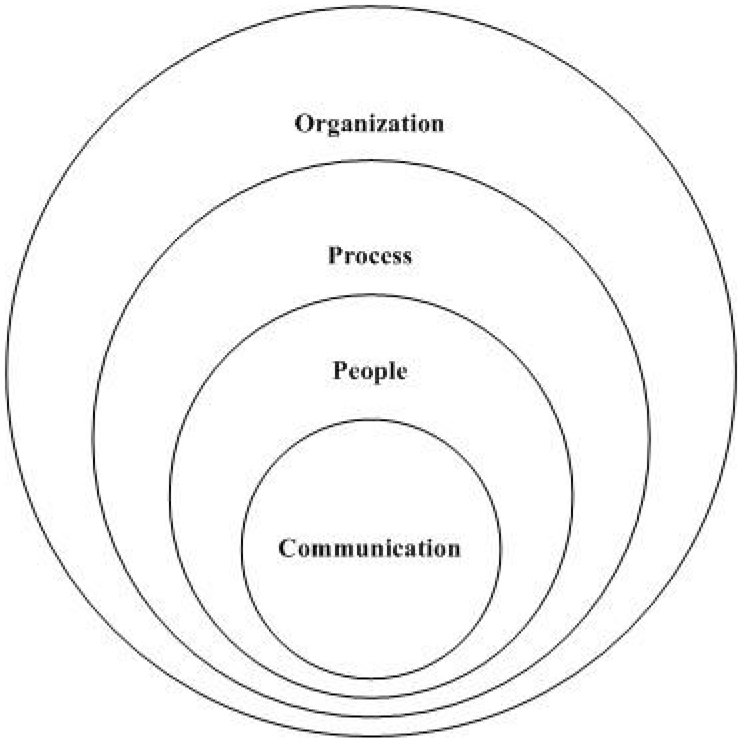

Figure 3. An Integrated Conflict Model

As mentioned in previous sections, the people layer and the process layer are supported by both models. The communication layer is only supported by Zhang et al. (2014), and the organization layer is only supported by Cohen et al. (2004). We keep the communication layer for two reasons: (1) it is supported by Zhang et al. (2014); (2) prior research has identified communication as a critical success factor for software development with conflict resulting from miscommunication and misunderstanding between stakeholders, particularly users and developers (e.g., Furumo, 2009; Wang et al., 2005; Zhang et al., 2014). In virtual teams, both communicating asynchronously and lacking of media richness can increase "the likelihood of the communication being misunderstood" (Furumo, 2009, p. 66). The misunderstanding problem between developers and users can be minimized when developers work closely and consistently with users so that minor points can easily be clarified throughout the development process, while relationship building and increasing levels of trust defuse emotional negativity and conflict escalation (Wang et al., 2005). Furthermore, the general conflict literature also suggests that communication is a double-edged sword - when mishandled, it can promote conflict, rather than prevent or resolve it. All in all, the general conflict literature (e.g., Barki & Hartwick, 2001; De Dreu & Weingart, 2003), the conflict literature in software development (e.g., Robey et al., 2001), and the conflict literature pertaining specifically to software developers and testers (e.g., Zhang et al., 2014) all suggest that communication is a critical success factor for software development outcomes. Thus, we keep the communication layer in our integrated model.

Similarly, we keep the organization layer for three reasons: (1) it is supported by Cohen et al. (2004); (2) it is supported in the conflict literature in software development (e.g., Barki & Hartwick, 2001; Robey et al., 1989; Robey et al., 2001; Sawyer, 2001; Yeh & Tsai, 2001); and (3) it is partially supported by Zhang et al. (2014). If you take a closer look at the data analysis process in Zhang et al. (2014), you shall realize that after two rounds of Q-sort, there were still four primary categories, including *communication, people, process, and organization*. They removed the organization category only after the online survey because "of the 24 scenarios meeting the criteria for model inclusion, none fell into the [organization] category" (Zhang et al., 2014, p. 16). However, if you look at the online survey results (i.e., Appendix E in Zhang et al., 2014, p. 25), you shall find that the organization category received a total 208 votes out of 3000 votes for 30 conflict scenarios, which indicates that it is reasonable to add the organization layer back in order to better consolidate the two models into an integrated model. Thus, we keep the organization layer in our integrated model.

DISCUSSION

A Brief Summary

The purpose of this study was to compare and integrate two leading models of sources of conflict between software developers and testers, proposed by Cohen et al. (2004) and Zhang et al. (2014), respectively. We first compared the similarities and the differences between the two models, and then consolidated the two models into an integrated model. The newly integrated model has four conflict layers; these layers are (from inside out): *communication, people, process, and organization*. This suggests that developers and testers experience conflict in the software development and testing process because: (1) they do not communicate appropriately, effectively, or efficiently, (2) they engage in, or are recipients of, strongly negative behaviors and consequently develop negative perceptions of each other, (3) they do not document their work properly or follow standardized procedures correctly, and/or (4) they struggle with organizational power and politics.

Theoretical and Practical Implications

The findings of this paper have important theoretical implications. The conflict between software developers and testers appears to originate from four layers, including *communication, people, process, and organization*. This helps us understand that the antecedents of conflict between software developers and testers are multi-dimensional. This paper clearly demonstrated that factors at layers of *communication, people, process, and organization* could impact the level of conflict between software developers and testers. As such, when assembling measurement items of antecedents of conflict between software developers and testers, we need to be certain that all four layers should be adequately represented by the items. Otherwise, the measurement scale for the antecedents of the conflict between software developers and testers will be partial, possibly resulting in biased assessment outcomes and prejudiced conclusions.

The findings of this paper also have important practical implications. Since conflict between software developers and testers originates from four layers, including *communication, people, process, and organization*, corporate executives and IT directors can focus on these four layers when dealing with conflict between software developers and testers. Specifically, corporate executives and IT directors should do the following in order to manage conflict between developers and testers in the software development and testing process: (1) improve communication effectiveness and efficiency between software developers and testers, (2) improve working relationship and personal relationship between software developers and testers, (3) encourage software developers and tester to document their work properly and follow standard procedures correctly, and (4) make software developers and testers less struggled with organizational power and politics.

Limitations and Future Research

This study has several limitations. First, it is based on only two leading models, possibly limiting the generalizability of its findings. Second, neither study included standard control data (e.g., age, gender, tenure, and education) of interviewees or conflict scenario providers, possibly hindering the identification of sub-constituencies within the developer and tester groups. And third, neither study reported what software development methodologies the interviewees and conflict scenario providers were using when they participated in their respective studies; we believe that software development methodologies may moderate the relationship between developer-tester conflict and conflict sources.

Future research should look into all available conflict models pertaining to software developers and testers and try to consolidate them into an integrated model. In addition to addressing conflict sources, researchers might address conflict resolution and construct innovative ways in managing conflict. Future studies should also collect demographic data in order to improve the opportunities of the identification of sub-constituencies within the developer and tester groups. Future research can also focus on the potential moderating role of software development methodologies in the relationship between developer-tester conflict and conflict sources.

Concluding Remarks

Conflict between software developers and testers has important implications to software development outcomes. As such, understanding sources of such conflict is critical. In this paper, we compared two leading models of sources of conflict between software developers and testers and developed an integrated model from them. Consolidated from the two leading models, our newly integrated model contains four conflict layers: *communication, people, process, and organization*. Corporate executives and IT directors can use this integrated model to better understand the sources of conflict between software developers and testers so that they can achieve both software development effectiveness and organizational outcomes.

REFERENCES

Barki, H., & Hartwick, J. (2001). Interpersonal conflict and its management in information system development. *MIS Quarterly*, *25*(2), 195-228.

Cohen, C. F., Birkin, S. J., Garfield, M. J., & Webb, H. W. (2004). Management conflict in software testing. *Communications of the ACM*, *47*(1), 76-81.

De Dreu, C. K. W., & Weingart, L. R. (2003). Task versus relationship conflict, team performance and team member satisfaction: A meta-analysis. *Journal of Applied Psychology*, *88*(4), 741-749.

Dhaliwal, J., Onita, C. G., Poston, R., & Zhang, X. (2011). Alignment within the software development unit: Assessing structural and relational dimensions between developers and testers. *Journal of Strategic Information Systems*, *20*(4), 323-342.

Furumo, K. (2009). The impact of conflict and conflict management style on deadbeats and deserters in virtual teams. *Journal of Computer Information Systems*, *49*(4), 66-73.

Giddens, A. (1986). *The constitution of society: Outline of the theory of structuration*. Berkeley, CA: University of California Press.

Jehn, K. A. (1995). A multimethod examination of the benefits and detriments of intragroup conflict. *Administrative Science Quarterly*, *40*(2), 256-282.

Jehn, K. A., Chatwick, C., & Thatcher, S. M. B. (1997). To agree or not to agree: The effects of value congruence, individual demographic dissimilarity, and conflict on workgroup outcomes. *International Journal of Conflict Management*, *8*(4), 287-305.

Moeller, G., Zhang, X., & Richardson, S. M. (2012). Understanding antecedents of interpersonal conflict in information systems development: A critical analysis. *Journal of Information Technology Management*, *23*(3), 12-43.

Robey, D., Farrow, D. L., & Franz, C. R. (1989). Group process and conflict in system development. *Management Science*, *35*(10), 1172-1191.

Robey, D., Welke, R., & Turk, D. (2001). Traditional, iterative, and component-based development: A social analysis of software development paradigms. *Information Technology & Management*, *2*(1), 53-70.

Sawyer, S. (2001). Effects of intra-group conflict on packages software development team performance. *Information Systems Journal*, *11*(2), 155-178.

Sawyer, S., & Guinan, P. J. (1998). Software development: Processes and performance. *IBM Systems Journal*, *37*(4), 552-568.

Wang, E. T. G., Chen, H. H. G., Jiang, J. J., & Klein, G. (2005). Interaction quality between IS professionals and users: Impacting conflict and project performance. *Journal of Information Science*, *31*(4), 273-282.

Yeh, Q.-J., & Tsai, C.-L. (2001). Two conflict potentials during IS development. *Information & Management*, *19*(2), 135-149.

Zhang, X., Dhaliwal, J. S., Gillenson, M. L., & Stafford, T. F. (2013). The impact of conflict judgments between developers and testers in software development. *Journal of Database Management*, *24*(4), 26-50.

Zhang, X., Stafford, T. F., Dhaliwal, J. S., Gillenson, M. L., & Moeller, G. (2014). Sources of conflict between developers and testers in software development. *Information & Management*, *51*(1), 13-26.