Research

Antidotes for high complexity and ambiguity in software development

Stephanie Watts Sussman\textsuperscript{a, \*}, P.J. Guinan\textsuperscript{2, b}

\textsuperscript{a} Weatherhead School of Management – MIDS, Case Western Reserve University, 10900 Euclid Ave., Cleveland, OH 44106-7235, USA
\textsuperscript{b} Babson College – Information Systems, Babson Park, Wellesley, MA 02157-0310, USA

Received 18 February 1998; accepted 8 November 1998

Abstract

In a longitudinal study of 47 software development teams, we investigate interactions between team and technology factors and the degree of complexity and ambiguity of the projects themselves. From the literature, we propose a theoretical model that identifies a characteristic of the technology (modularity) and a characteristic of the team process (conflict resolution) used during system development, as effective for minimizing the adverse effects of high task-based complexity and ambiguity (those tasks for which multiple acceptable solutions exist). We hypothesize that modularity and conflict-resolution techniques will account for a significant amount of the variance in user satisfaction for highly complex and ambiguous projects, but that this will not be the case for simple and unambiguous projects. Our findings confirm this hypothesis, indicating that effective conflict resolution and modularity are associated with significantly higher client satisfaction six months after implementation for all projects. An explanation for these findings is offered, followed by implications for theorists and practitioners. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Complexity; Ambiguity; Conflict resolution; Modularity; Task–technology fit; Software development; Teams; Survey research; Moderation analysis

1. Introduction

The team tasks of software design and development are intrinsically complex and often exacerbated by incomplete user requirements, changing environmental demands, and high levels of interdependence within teams [12]. Many large-scale software development projects do not function as intended or fulfil their potential [11, 12]. Clearly, research into improving the software development process is still warranted [15].

Managing development teams is a non-routine task characterized by high levels of complexity [30], especially when these teams are medium or large. Task complexity or uncertainty has been studied for its effects on various types of systems, such as expert systems [55], decision-support systems [23, 45], end-user computing [9], and project management in general [35]. What can be done to mitigate the negative impact of high task complexity on the software development process? The structural contingency perspec-
ative of organizational design [48, 52] prescribes an appropriate fit between organizational structure and environmental uncertainty for reducing task complexity and uncertainty. According to the information processing perspective, more routine, less complex environments process information most efficiently with hierarchical control structures [20], while complex and uncertain environments benefit from organismic organizational forms [7] and rich information channels [13].

In addition to high complexity, theorists also identify ambiguous or equivocal task environments as a threat to effective information processing. Such task environments are those in which no single correct outcome is inherently correct, and potentially acceptable multiple solutions exist [14]. The creation of shared systems of symbols can help reduce task ambiguity [53], and provide a horizontal coordination mechanism [35] by forcing stakeholders to undertake an explicit process of convergence on diverse desired outcomes.

From an information processing perspective, the impact of task complexity has been widely documented [18, 54], but less research has been conducted on the role of task ambiguity in the context of software development. Nor has there been any recent empirical work relating to influences of task complexity and task ambiguity on software development teams.

Although theoretical prescriptions for mitigating the adverse effects of task complexity and ambiguity are helpful, they are very general in nature, and do little to guide practitioners in selecting particular technologies and processes for software development from the myriad of options available. It is important to investigate which specific technologies and processes are most effective under conditions of high complexity and ambiguity, since it is under these conditions that traditional methods of coordination and control are least effective [20]. Successful identification of such factors contributes to our theoretical understanding of these two organizational problems at a higher level of granularity than has been achieved thus far, and can benefit practitioners faced with real task uncertainty and ambiguity.

Section 2 presents past research on task-based complexity and ambiguity and develops hypotheses around specific team and technology characteristics identified as having the capacity to absorb the negative effects of complexity and ambiguity. Section 3 describes the method used to analyze these hypotheses, based on a survey of 47 medium-sized software development teams. The resultant findings and discussion are presented in Section 4. Conclusions and implications of these results are discussed in Section 5.

2. Background

Organization design research posits that organizational forms can be fitted to characteristics of tasks in order to optimize organizations’ ability to deal with external forces [21, 49]. Team problem-solving processes are clearly affected by the type of task involved [32, 8]. These task characteristics are distinct from the technologies employed to transform inputs into outputs, most frequently described on a continuum of routine/non-routine [40].

Two major forces external to organizational subunits influence the ability of their members to process information: uncertainty and equivocality [13]. Uncertainty is defined as the absence of information [2], and is reduced by new task-related information [50]. Ambiguous task environments are those where multiple and conflicting interpretations of existing information occur, such that they are resolved through enactment of shared interpretations of the task at hand [53].

2.1. Task complexity

Work-related uncertainty can be viewed as an objective characteristic of the task [8] that imposes behavioral requirements on individuals facing it. Task complexity is one widely accepted source of work-unit uncertainty [50]. Task uncertainty has been associated with lower levels of software project performance and higher levels of software performance risk [35].

Perceived complexity is widely acknowledged to be an important factor affecting the software development process [42]. Software development project teams are often faced with high levels of complexity, since it is not unusual to build a system which has multiple interfacing components and is measured in millions of lines of code [6]. Theory states that task complexity can be reduced by the acquisition and use of additional new information [2], thus organizational
forms and processes that promote maximum information processing capacity should be most beneficial in highly complex tasks [20], such as software development.

2.2. Task ambiguity

Unlike task complexity, task ambiguity refers to those tasks for which multiple acceptable solutions exist, as perceived by those with different frames of reference [2]. Task information that is clear and directed leads to similar interpretations, while task information that is ambiguous leads to multiple interpretations which must be resolved in order to develop a shared understanding of how to perform the task. At the team level, the team leader must work to integrate the separate perceptions, frames and knowledge bases of team members [51], who are often working on different development tasks, especially in the case of large projects [56]. A system development project in which different stakeholders have divergent goals and views of the final system presents an ambiguous task to its developers [5]. Typical ambiguous systems are those spanning different functional areas, such that managers from the different functions have particular and divergent needs for the system. Such conflicts create task ambiguity when they are not identified and resolved.

Organizational forms which promote the exchange of rich information have been prescribed as most effective for reducing task ambiguity, since this type of information increases the capacity of the information exchanged to move individual perceptions toward a shared perception [13]. Alternatively, we can view the problem from the perspective of social cognition: Externalized knowledge is less ambiguous than tacit knowledge, since it is explicit and codified. Externalization of tacit knowledge held by team members occurs through social processes which make use of metaphors, analogies and models [36], and by the use of explicit models of the system-to-be. Cognitive maps, prototypes and scripts are used to represent individuals’ tacit cognitions [26] by cognitive scientists. In the same way, technologies which enable physical depiction of the future computer system – such as application module management systems – enable the articulation of shared team perceptions of the system-to-be; articulations that then become manifestations of shared, externalized knowledge. The process of building these models reduces task ambiguity by requiring that the divergent perspectives of the various individuals involved become enmeshed as their tacit knowledge becomes public and represented. Thus, rather than information richness, we look to another prescription for the reduction of task ambiguity – the creation of systems of symbols which embody shared perceptions of an unambiguous task outcome.

The concept of task-technology fit is widely applied to system utilization theory [22] at the individual level. We build a related contingency model at the group level to hypothesize an appropriate fit between these two characteristics of the task, complexity and ambiguity, and a group process factor – team conflict resolution – and a technology factor – modularity – during the software development process, as described below.

2.3. Team conflict resolution

A substantial body of research has investigated the impact of conflict resolution across team boundaries, especially in interaction with users participating in development [43]. This research has focused primarily on medium-sized teams consisting of 8–15 members. Less work has been done in the area of within-team conflict resolution. Team-member conflict-resolution behaviors during system development have been directly associated with project success [44]. For this reason, and as a consequence of the theoretical argument presented below, we have selected conflict resolution as the key group-process variable to include in our model.

Although many definitions of conflict exist, it is generally agreed that conflict occurs around incompatible goals or interests among members [5]. Since goal interference represses information processing capacity, conflict resolution can enhance information processing capacity by facilitating goal congruence. Networked communication environments enable the “adjustment and continual re-definition of individual tasks through interaction with others” [7] that is necessary for high levels of information processing. As a mechanism for encouraging differentiation and integration of diverse views of the various team members, horizontal coordination requires that team mem-
bers resolve conflicts effectively through generation of agreement and consensus [43]. Conflict resolution is one highly effective coordination mechanism that can increase communication within the team and so mitigate the adverse effects of complexity by increasing the quantity of information available to the team [52, 56].

In these ways, conflict-resolution behaviors can increase the information processing capacity of a team [47]. Since teams facing high levels of task complexity need to process greater quantities of information than those facing lower levels of task complexity [50] more successfully than those without effective conflict resolution behaviors. And since conflict resolution capabilities are also mechanisms that teams can use to generate shared perceptions of the desired outcome, conflict resolution skills can be helpful for resolving the inconsistent views of ambiguous projects. Thus, conflict resolution may mitigate the impacts of both task complexity and ambiguity.

### 2.4. Modularity

Where organizations reach the limits of their capacity to manage complexity, information technologies can be used to reduce and manage this complexity [28]. Technologies include the programs and routines generated from tool use, as well as the actual physical tools. Systems development methodologies and the production of necessary artifacts require that the development team articulate its shared perceptions of the system-to-be. We have argued above that ambiguity can be reduced through the creation and use of shared systems of symbols [53], so technologies that require the team to make explicit its tacit perceptions of the system through generation of modules or other physical artifacts should be most effective for complex and ambiguous processes.

Explicit shared perceptions of the system under development include the whole system, its parts, and the relationships of these parts to the whole [51]. Under the guise of reuse, many systems are developed in parts or modules, a system characteristic we refer to here as ‘Modularity’. When teams design modules, they are sharing their views of the system parts. Therefore, we consider the extent to which modularization technology is utilized during system development to be a measure of the degree to which a team makes its tacit knowledge explicit – through module creation and selection. The benefits of modularity are well documented in the literature [37, 19, 39], especially in the context of object-oriented technologies [38, 16] but not fully theoretically grounded [1, 4]; we offer a theoretical explanation for these benefits: The process of deciding and resolving multiple views about which modules to build and to include in the final system externalizes individuals’ tacit knowledge so that a group knowledge convergence can occur and be made explicit. In this view, the greater the degree of modularity used by the team, the greater the likelihood that task ambiguity will be reduced in the process. In addition, by committing to include certain modules and not others in the final system, modularity also mitigates task ambiguity by reducing the number of future possible options available to the team. And by making explicit the relationship of parts to the whole, modularity generates additional information which may be helpful in reducing task-based uncertainty.

### 2.5. Information system success

Client satisfaction or client information satisfaction is the most widely used single measure of IS success [17]. As DeLone and McLean state: “It is hard to deny the success of a system which its users say that they like.” The clients’ perceptions of the value, appropriateness, and timeliness of the information that the system produces is an extremely important determinant of overall system success [27], and is used in this study to measure system success six months after installation.

### 2.6. Model development

Fig. 1 illustrates a model derived from the literature, reflecting the relationships between the two task...
factors, the two process factors, and the dependent measure of user information satisfaction.

From this model we hypothesize that modularity and conflict resolution moderate the negative impacts of complexity and ambiguity on user information satisfaction:

**Hypothesis 1**: Taken together, effective conflict resolution and high levels of modularity will result in high user information satisfaction (six months after project implementation) in highly complex projects, but not in less complex projects.

**Hypothesis 2**: Taken together, effective conflict resolution and high levels of modularity will result in high user information satisfaction (six months after project implementation) in highly ambiguous projects, but not in less ambiguous projects.

By investigating both process factors together, we can compare the extent to which they moderate the task variables, to see if one is more salient than the other under the different task conditions. Since we have no theoretical basis on which to judge the relative impacts of the moderating variables under the two task conditions, this aspect of the analysis is data driven and, thus, exploratory.

3. Research method

This study investigated 47 software development teams from 15 organizations. The research design is based on a cross-sectional field study. These teams consisted of eight-to-fifteen members each, so can be considered ‘medium-sized’. The unit of analysis is the IS design team.

For each software development project, two sets of questionnaires were distributed. The first questionnaire was distributed to team members and completed during the design and code phase of software development. It contained items on the degree of conflict resolution and modularity, as well as perceived task complexity and ambiguity. The second questionnaire was distributed to actual users of the developed systems, six months after implementation.

In the first questionnaire, data on the independent and moderating variables were collected directly from members of each team. Wherever possible, all team members were surveyed. When it was not possible to reach all team members, teams provided a representative sample of key informants from the team. Using key informant techniques for data collection has been found to be effective for survey research in organizations [41]. In the second questionnaire, a minimum of two users per project were asked the dependent measure – to what extent the information provided them by the system meets their precise information needs. By surveying both team members and users six months apart, we prevent self-report and recall bias in the dependent measure.

Only projects estimated to have twelve-to-fifteen-month development schedules were selected, from organizations representing a range of industries: insurance, financial services, high technology, heavy industry, transportation, and petroleum. Table 1 presents demographic information.

### 3.1. Measures and procedures

Excerpts from standard indicators were used to measure the constructs of this study. All indicators

<table>
<thead>
<tr>
<th>Industry</th>
<th># Cos</th>
<th># Teams</th>
<th># Respondents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insurance</td>
<td>4</td>
<td>11</td>
<td>41</td>
</tr>
<tr>
<td>Transportation</td>
<td>2</td>
<td>12</td>
<td>79</td>
</tr>
<tr>
<td>High technology</td>
<td>1</td>
<td>7</td>
<td>64</td>
</tr>
<tr>
<td>Financial services</td>
<td>4</td>
<td>11</td>
<td>58</td>
</tr>
<tr>
<td>Petroleum</td>
<td>2</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>Heavy industry</td>
<td>1</td>
<td>3</td>
<td>31</td>
</tr>
<tr>
<td>Total</td>
<td>15</td>
<td>47</td>
<td>291</td>
</tr>
</tbody>
</table>
and constructs are in the form of seven-point Likert-type scales, and are listed in Appendix A. Question order was randomized and items were reverse-scored in the actual instrument to reduce response order effects. Client information-satisfaction items were taken from Ives et al. [27]. Indicators for conflict resolution were drawn from Hackman [24], while those for modularity were taken from Henderson and Cooprider [25]. Task ambiguity was operationalized using Daft and Macintosh’s [14] measure of work-unit information equivocality. Task complexity items were drawn from Van de Ven, Delbecq and Koenig [52].

4. Results and discussion

The unit of analysis is the team, so values for each construct reported represent the average value of that construct for the team. Table 2 presents descriptive statistics, and the Cronbach αs indicate acceptable levels of reliability for all constructs.

Univariate analyses confirm acceptable levels of skewness and kurtosis for all variables. Pearson correlation matrices indicate no significant correlations between independent constructs (see Table 3), except between ambiguity and complexity. Thus, the two hypothesized moderators are not entirely independent or strictly orthogonal. However, their correlation of 0.34 is of medium magnitude and within acceptable parameters for a weak form test of orthogonality. And based on the sample size, analysis of the effects of the two moderators is performed independently, so this correlation does not impact it. To investigate convergent validity, confirmatory factor analysis with varimax rotation and Kaiser normalization was used. All items loaded onto their respective constructs with no overlap (see Table 4).

In order to assess both, the form and the strength of the hypothesized moderators, it is necessary to employ two types of analysis – moderated regression analysis performed on the full sample, and the split-sample analysis of subgroups divided by the moderator [46]. These two types of analyses are presented in the following.

4.1. Moderated regression analysis

Moderated regression analysis (MRA) avoids the loss of information resulting from the artificial transformation of a continuous variable into a qualitative one. It enables the identification of all types of moderators, except homologizers which operate through the error term to affect the strength rather than the form of the predictor-criterion relationship. Subgroup analysis in Section 4.2 will be used to determine if the hypothesized task moderators are homologizers.

Table 2
Descriptive statistics

<table>
<thead>
<tr>
<th>Factor</th>
<th>Mean/median</th>
<th>S.D.</th>
<th>No. indicators</th>
<th>α</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguity</td>
<td>3.96/4.08</td>
<td>0.79</td>
<td>3</td>
<td>(0.8117)</td>
</tr>
<tr>
<td>Complexity</td>
<td>4.97/5.00</td>
<td>0.85</td>
<td>2</td>
<td>(0.8398)</td>
</tr>
<tr>
<td>System success</td>
<td>5.17/5.67</td>
<td>1.44</td>
<td>3</td>
<td>(0.9472)</td>
</tr>
<tr>
<td>Modularity</td>
<td>2.12/2.00</td>
<td>1.57</td>
<td>3</td>
<td>(0.8640)</td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>5.12/5.17</td>
<td>0.81</td>
<td>2</td>
<td>(0.8839)</td>
</tr>
</tbody>
</table>

Table 3
Correlation matrices using whole sample

<table>
<thead>
<tr>
<th></th>
<th>Ambiguity</th>
<th>Complexity</th>
<th>Modularity</th>
<th>Conflict resolution</th>
<th>UIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambiguity</td>
<td>1.00</td>
<td>0.34**</td>
<td>−0.08</td>
<td>−0.12</td>
<td>−0.19</td>
</tr>
<tr>
<td>Complexity</td>
<td>1.00</td>
<td>−0.05</td>
<td>−0.004</td>
<td>−0.10</td>
<td>−0.10</td>
</tr>
<tr>
<td>Modularity</td>
<td>1.00</td>
<td>−0.06</td>
<td>0.22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>1.00</td>
<td></td>
<td></td>
<td>0.37**</td>
<td></td>
</tr>
</tbody>
</table>
MRA involves comparing three regression equations for equality of the regression coefficients [46]. The first of these compares regressions that contain the independent variables and the moderator with one containing these and an additional interaction term.

To create interactions terms between conflict resolution and modularity, and complexity and ambiguity, task complexity and ambiguity were coded as dummy variables, with ‘zero’ representing below-average complexity and ambiguity, and ‘one’ representing above-average complexity and ambiguity. These dummy variables were then multiplied by the respective independent factors in each model to create cross-product terms, and regressions were performed on the entire sample (n = 47) using these interaction terms, the hypothesized moderators, and the independent variables. This technique was used to determine the extent to which the independent constructs vary in a linear fashion with the continuous variables of complexity and then ambiguity. Significant outcomes generated in this way may allow us to say, for example, that not only does modularity significantly impact client satisfaction under conditions of high ambiguity, but also that the more the ambiguity present, the greater impact of modularity on client satisfaction, and vice versa.

A regression (Table 5(a)) was performed using the two independent factors of conflict resolution and the extent of modularity, the interaction terms of these two factors with the complexity dummy variable, and complexity. The overall model was significant \( F(5,41) = 2.83, p < 0.05 \) with an adjusted \( r^2 \) of 0.17. No interaction factors were significant. The hypothesized moderators of conflict resolution and modularity were significant, while complexity is not, although its impact is negative, as predicted from theory. The \( \beta \) coefficients of both the interaction terms are positive, however.

Table 5(b) presents results of the same analysis for ambiguity. The overall model was significant

| Table 4 |
| Factor analyses using whole sample |
| Rotated factor matrix from factor analysis of independent variables |
| Ambiguity | Factor 1 | Factor 2 |
| P4S5Q4 | 0.87630 |
| P4S5Q18 | 0.87607 |
| P4S5Q20 | 0.81088 |
| Complexity | Factor 1 | Factor 2 |
| P4S5Q14 | 0.93321 |
| P4S5Q10 | 0.93071 |

| Rotated factor matrix from factor analysis of the moderator variables |
| Modularity | Factor 1 | Factor 2 |
| P2S1Q4 | 0.94558 |
| P2S1Q8 | 0.92372 |
| P2S1Q12 | 0.91714 |
| Conflict resolution | Factor 1 | Factor 2 |
| S3Q8 | 0.92876 |
| S3Q26 | 0.88392 |
| S3Q16 | 0.86345 |

Table 5 |
Whole sample regression analysis |
| \( \beta \) | \( T \) | Significance of \( T \) |
(a) A complexity dummy in interaction with other variables \(^a\) |
| Complexity | −0.07 | −0.30 | 0.7639 |
| Modularity | 0.31 | 1.89 | 0.0656 |
| Modularity × complex | 0.02 | 0.10 | 0.9176 |
| Conflict resolution | 0.44 | 2.96 | 0.0051 |
| Conflict resolution × complex | 0.11 | 0.34 | 0.7377 |
(b) An ambiguity dummy in interaction with other variables \(^b\) |
| Ambiguity | −0.12 | −0.50 | 0.6188 |
| Modularity | 0.18 | 1.05 | 0.2991 |
| Modularity × ambiguity | 0.27 | 1.11 | 0.2736 |
| Conflict resolution | 0.48 | 3.31 | 0.0020 |
| Conflict resolution × ambiguity | −0.18 | −0.67 | 0.5086 |

\(^a\) Adjusted \( r^2 \) of 0.17, \( F(5,41) = 2.83, p = 0.028 \).
\(^b\) Adjusted \( r^2 \) of 0.20, \( F(5,41) = 3.24, p = 0.0149 \).
The negative impact of ambiguity is greater than that of complexity, but is not significant. The only significant individual factor is conflict resolution, although the modularity-ambiguity interaction term has a strong positive $\beta$.

Under MRA, for complexity or ambiguity to be pure moderators, significant differences must be apparent between the regression coefficients of the regressions above that include the interaction terms, and those below (Table 6(a and b)), that do not include them. No significance was found, so we can conclude that task complexity and ambiguity are not pure moderators in this model.

A final comparison in MRA is of regression coefficients between regressions without the moderator and those that include the moderator. Table 7 depicts the regression without the moderator and those that include the moderator. Table 7 shows that complexity and ambiguity are not pure moderators in this model.

A final comparison in MRA is of regression coefficients between regressions without the moderator and those that include the moderator. Table 7 depicts the regression without the moderator and those that include the moderator. Table 7 shows that complexity and ambiguity are not pure moderators in this model.

4.2. Subgroup analysis

In the split-sample analysis, median values for complexity and ambiguity were used to divide the sample into high and low complexity and ambiguity, as is typical for split sample analysis [33, 29, 31]. For complexity, the median reported value was 5.0, but since seven cases reported exactly this value, these seven were eliminated, leaving 19 teams with above-the-median complexity value of 5.0, and 21 teams reporting below this average. For ambiguity, the reported value of median was 4.0, but one case reported exactly this value, so this team was eliminated, leaving 24 teams above this median value and 22 below it.

Following standard practice [34], Chow tests [10] were performed to determine whether the models based on the split samples were significantly different from one another. Regressions were performed on the four subsamples defined by the model – high and low task complexity, and high and low task ambiguity.

To investigate Hypothesis 1, conflict resolution and modularity were regressed in separate regressions onto the dependent variable of client satisfaction, first using only high complexity teams, and then using only low complexity teams (H1). For Hypothesis 2, this process was repeated using only high ambiguity teams, and then using only low ambiguity teams (H2).

Hypothesis 1 was then investigated, and for teams reporting above the median in task complexity, the model is highly significant ($F(2,15) = 11.44, p < 0.001$), with an adjusted $R^2$ of .55, as is the individual conflict factor ($T = 4.43, p < 0.001$). Modularity is also significant ($T = 2.19, p < 0.05$), but not substantially different from that revealed in the whole sample analysis. For those teams reporting below average complexity, neither the model nor either individual indicator are significant.

Moderation is difficult to confirm with small sample sizes, but results of the Chow test confirmed that the regression coefficients of the high and low complexity subsamples were significantly different ($F = 3.083, p = 0.05$). Thus, split-sample analyses confirm...
Hypothesis 1 to the extent that the model is highly significant under conditions of high task complexity, and insignificant under conditions of low task complexity. However, this effect is due to the impact of conflict resolution (which is almost ten times more significant than in the whole-sample analysis), and not the effect of modularity. Thus, Hypothesis 1 is true for conflict resolution, but not for modularity. [See Table 8(a and b).]

Hypothesis 2 was then investigated, and for teams reporting above the median in task ambiguity, the model is highly significant (\(F(2,21) = 9.48, p \leq 0.0012\)), with an adjusted \(r^2\) of 0.42. Both, conflict resolution and modularity are individually significant (for conflict, \(T = 3.47, p < 0.01\), and for modularity, \(T = 3.07, p < 0.01\)). For those teams reporting below average ambiguity, neither the model nor the indicator is significant. However, the Chow test comparing regression coefficients was not significant (\(F(2,20) = 0.70, p = 0.51\)). Despite the fact that the model is significant under conditions of high ambiguity, but not low ambiguity, we cannot confirm Hypothesis 2. [See Table 9(a and b).]

Table 10(a and b) illustrate the effects of conflict resolution and modularity on user information satisfaction for those teams with high and low average levels of both, complexity and ambiguity, respectively. As in the foregoing, the model was significant for the high complexity and ambiguity sample (\(F(2,8) = 5.72, p < 0.05\)), with an adjusted \(r^2\) of 0.49, but not significant for the low ambiguity and complexity sample. Again, conflict resolution explained a larger portion of the variance than did ambiguity. The Chow test did confirm that the two regression coefficients are significantly different (\(F = 3.34, p = 0.05\)).

### 4.3. Discussion

This study utilizes a parsimonious, theoretically grounded model derived from the organizational design and software development literatures. Its contribution lies in its simplicity, and also in its combined investigation of the two important task variables, both of which can sabotage effective software development. It operationalizes theoretical prescriptions for managing complexity and ambiguity in its selection of two critical process factors – conflict resolution and modularity. These factors together do not explain any of the variance in client information satisfaction six months subsequent to installation for unambiguous, low-complexity projects. But they explain 55\% of the variance in client satisfaction for projects with above-average complexity, and 42\% of the variance in client satisfaction for projects with above-average ambiguity. As compared to their impact across all projects, conflict resolution plays a greater role than modularity.

### Table 8
Split sample regression analysis

<table>
<thead>
<tr>
<th></th>
<th>(\beta)</th>
<th>(T)</th>
<th>Significance of (T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Model using teams with above-median complexity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>0.72</td>
<td>4.43</td>
<td>0.0005</td>
</tr>
<tr>
<td>Modularity</td>
<td>0.36</td>
<td>2.19</td>
<td>0.045</td>
</tr>
<tr>
<td>(b) Model using teams with below-median complexity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>0.10</td>
<td>0.46</td>
<td>0.6499</td>
</tr>
<tr>
<td>Modularity</td>
<td>0.25</td>
<td>1.13</td>
<td>0.2731</td>
</tr>
</tbody>
</table>

\(^a\) Adjusted \(r^2\) of 0.55, \(F(2,15) = 11.44, p < 0.001\).
\(^b\) Adjusted \(r^2\) of \(-0.03, F(2,19) = 0.695, p = 0.511\).

### Table 9
Split sample regression analysis

<table>
<thead>
<tr>
<th></th>
<th>(\beta)</th>
<th>(T)</th>
<th>Significance of (T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Model using teams with above-median ambiguity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>0.55</td>
<td>3.47</td>
<td>0.0023</td>
</tr>
<tr>
<td>Modularity</td>
<td>0.49</td>
<td>3.07</td>
<td>0.0058</td>
</tr>
<tr>
<td>(b) Model using teams with below-median ambiguity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>0.26</td>
<td>1.14</td>
<td>0.2680</td>
</tr>
<tr>
<td>Modularity</td>
<td>0.14</td>
<td>0.64</td>
<td>0.5307</td>
</tr>
</tbody>
</table>

\(^a\) Adjusted \(r^2\) of 0.42, \(F(2,21) = 9.48, p = 0.0012\).
\(^b\) Adjusted \(r^2\) of \(-0.03, F(2,20) = 0.70, p = 0.51\).

### Table 10
Split sample regression

<table>
<thead>
<tr>
<th></th>
<th>(\beta)</th>
<th>(T)</th>
<th>Significance of (T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Using teams with both, above-median complexity and ambiguity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>0.60</td>
<td>2.65</td>
<td>0.0292</td>
</tr>
<tr>
<td>Modularity</td>
<td>0.42</td>
<td>1.83</td>
<td>0.1043</td>
</tr>
<tr>
<td>(b) Using teams with both, below-median complexity and ambiguity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict resolution</td>
<td>0.11</td>
<td>0.39</td>
<td>0.7059</td>
</tr>
<tr>
<td>Modularity</td>
<td>0.16</td>
<td>0.55</td>
<td>0.5951</td>
</tr>
</tbody>
</table>

\(^a\) Adjusted \(r^2\) of 0.49, \(F(2,8) = 5.72, p < 0.05\).
\(^b\) Adjusted \(r^2\) of \(-0.13, F(2,12) = 0.30, p = 0.82\).
in apparently mitigating the effects of complexity, and modularity positively impacts highly ambiguous projects. The converse is not true – conflict resolution does not strongly interact with ambiguity to counteract its negative effects, nor does modularity interact strongly with complexity to mitigate its effects.

The whole-sample MRA analysis indicates that this effect operates through the error terms, by affecting the strength rather than the form of the relationship of the independent variables to client information satisfaction. Conflict resolution and, to a lesser extent, modularity, are homologizer moderators with no direct relationship to either the independent or dependent constructs.

For practitioners, the lesson is clear. First, take time to identify those projects that are high in complexity and ambiguity. Team members and leaders are often aware of these conditions and are a good source for this information. Users also may well understand the potential complexities and ambiguities of a software development effort. Such identification processes should become a routine aspect of application portfolio development. Once these projects have been identified, corrective action can be taken, as suggested by this research. Foster conflict resolution capabilities in project teams faced with highly complex projects. This could include formal training, tacit learning, and modifications to incentive systems. Many human resource departments provide training in this area, as do many consulting organizations. Team members should be made aware of the purpose for the trainings and the importance of conflict resolution, particularly for projects high in complexity.

Practitioners should also work to ensure that modularity is being utilized in highly ambiguous software development projects. It is widely acknowledged that modularity can save time and, therefore, money during development. This study shows that modularity can increase system effectiveness also, for reasons consistent with current organizational theory. There are a number of CASE (computer-aided software engineering) tools available in the market that have features that promote modularity of design and development. Team members in these teams should be instructed regarding the meaning of ambiguity, its potential deleterious effects on software development success, and the role that modularity can play in mitigating these negative effects.

4.4. Limitations

This study does not confirm that feedback and conflict resolution behaviors actually reduce complexity, nor that modularity reduces ambiguity; only that, in the presence of high complexity, conflict-resolution behaviors have a positive impact on subsequent client satisfaction, and that, in the presence of high ambiguity, modularity also has a positive impact on information satisfaction. A theoretical argument has been presented which associates these positive impacts with their capacity to mitigate or absorb the negative effects of task complexity and ambiguity, but the evidence does not go so far. Future research in this direction could confirm these theoretical propositions through controlled experimental manipulations.

Client satisfaction is critical for systems success, and information satisfaction is especially important in this era of knowledge work and organizational learning. However, another path for future research would be the addition of non-perceptual dependent measures and metrics to identify impacts of this model on technical success.

This study does not investigate mechanisms for complexity and ambiguity reduction at the boundary of the team. It seems likely that much effective ambiguity reduction occurs in interaction with those outside the team. Thus, the impacts of these behaviors by boundary spanners (externally-oriented team members) is another potentially fruitful direction for future research. Nor does it investigate small or large teams, focusing as it does on medium-sized teams. While we have no reason to believe that these results cannot be generalized to larger or smaller software development teams, our results cannot speak on this issue.

5. Conclusion

Organizational theorists have long looked to organizational design for ways to improve the effectiveness of organizational information processing. The present study builds on this theory base to take an empirical look at information-processing behaviors at the team level of analysis. An important contribution is the combined examination of both, task complexity and ambiguity, confirming theory that identifies them as distinct constructs, each posing their own chal-
lenges. That which moderates complexity does not moderate ambiguity and vice versa. By operationalizing conflict resolution and modularity as an information processing behavior and a technology, respectively, this research confirms theoretical organizational design prescriptions in a model that is parsimonious, yet highly significant under the task conditions specified.

Communication via rich information channels has been cited as a way to reduce task ambiguity. Within teams, much interaction occurs face-to-face and is thus already rich, so it is important to investigate alternative mechanisms for reducing task-related ambiguity. We do this by looking to the modules produced with process technologies as external manifestations of shared perceptions of desired outcomes. This conceptualization of modularity provides a theoretical rationale for the widely acclaimed benefits of modularity technologies and contributes to the growing body of research that ties tools and techniques to theoretical reasons for their effectiveness. The growing popularity of object-oriented technologies motivates our continued theoretical interest in the phenomenon of modularity.

For practitioners, we present support for a contingency model which states that under conditions of high task complexity, conflict resolution is an effective team capability, while under conditions of high task ambiguity, modularity technologies are beneficial, especially in conjunction with conflict resolution. While further research is needed to confirm these prescriptions, this study paves the path for practitioners to enhance system development effectiveness by providing modularity capabilities to teams facing ambiguous projects, and by enabling conflict-resolution behaviors within those teams facing highly complex projects.

Appendix A

Survey items (all seven-point Likert scales)

Ambiguity: (α = 0.8117)

Please characterize the extent to which your project has the following characteristics:

1. To what extent do multiple views exist of how the final system should look?

Complexity: (α = 0.8398)

Please characterize the extent to which your project has the following characteristics:

1. How technically complex is the system being developed?
2. To what extent are the technical problems for this system particularly complicated?

System success: (α = 0.9472)

Please provide your impressions of how the present system satisfies your needs:

1. The system provides the precise information that I need.
2. The information content meets my needs.
3. I get the information I need on time.

Modularity: (α = 0.8640)

How often does your project team perform this activity on the current project:

1. Develop modular, component-oriented application parts?
2. Maintain an inventory of reusable application components?
3. Create application system units from individual components?

Conflict resolution:

Please evaluate to what degree you feel the following statements describe your project team:

1. In general, we get along very well.
2. This team resolves conflicts that exist among team members in a timely manner.

3. This team finds ways to minimize tensions between members.

N/A – strongly disagree – neither agree nor disagree – strongly agree

References


Stephanie Watts Sussman is an Assistant Professor of Information Systems at the Weatherhead School of Management at Case Western Reserve University. She received her doctorate from Boston University, where the research for this paper was performed. Her research interests include systems development, technologies for knowledge transfer, and computer-mediated communication.