December 2002

National Culture Influences in Cross-Cultural Software Development Teams: an analysis of social networks

Regit Young
University of Western Australia

Nick Letch
University of Western Australia

National Culture Influences in Cross-Cultural Software Development Teams: an analysis of social networks

Regit Young

Nick Letch

Department of Information Management and Marketing
The University of Western Australia
Western Australia, Australia
ryoung@ecel.uwa.edu.au

Abstract

Team dynamics and ingroup behaviour are critical factors in software development projects. In Western organisations, software development cultures promote rational values to be pursued at industry, development, and team levels of analysis. While Western values dominate the culture of software teams, increased globalisation has led to more IS being developed by teams with a cross-cultural make-up. In this paper, propositions regarding the interactions of software teams comprising members from Western and Chinese cultures are developed. These propositions highlight potential conflicts that may arise within cross-cultural development teams. A social network analysis of a small software development team is presented in order to investigate these propositions.

Keywords

Software development, National culture, Social network analysis

INTRODUCTION

Software development is essentially a group activity involving interaction between various stakeholders including users, analysts, programmers and senior management. As globalisation continues and organisations develop and integrate their information systems worldwide, the workgroups and teams that form to develop information systems will increasingly involve members from different cultural backgrounds. During systems development, interactions among team members can highlight the cultural commonalities and differences that can often impact the group processes and outcomes (Punnet and Shenkar, 1995).

Research on cross-cultural workgroups has identified several issues that can impact group processes. For example, in group dynamics there is “an apparent universal tendency to view one’s own group as superior and more trustworthy than another group (Brewer, 1986); intragroup anxiety may occur resulting in interaction avoidance or excessive politeness (Stephen, 1994). Even racism, often expressed in beliefs that the distribution of resources are uneven between groups (McConahay, 1986); or symbolic racism that emphasises ingroup values to justify unequal treatment of the outgroup (Kinder and Sears, 1981) have been identified at a time where blatant prejudice and discrimination are socially unacceptable. Foreign consultants working in local cultures with which they are unfamiliar may find differences in IS practices to be problematic (Thanasankit and Corbitt, 1999; Kumar and Bjørn-Andersen, 1990; Dagwell and Weber, 1983). Differences in IS practices which are contingent on the diversity of national culture need to be managed (Shore, 1996).

Despite being recognised as an important issue, culture, like many organizational issues, is usually given a low priority and only “lip service” is paid to its treatment (Doherty and King, 1998). In this paper, propositions regarding the interactions of software teams comprising members from Western and Chinese cultures are developed. These propositions highlight potential conflicts that may arise within cross-cultural development teams that may need to be addressed to facilitate project success. These propositions are investigated through the interpretation of the social networks of a small software development team.

SOFTWARE DEVELOPMENT TEAMS

Software development projects frequently face a variety of problems including cost overruns, buggy releases, maintenance underestimations and delays in proposed release dates (e.g. DeMarco, 1995; Brooks, 1987). While software developers have since the 1960s been inundated with technological innovations that are intended to boost their productivity (e.g. CASE tools, framework designers, code generators, etc.), there is little evidence to suggest that there is a silver bullet that will deliver productivity boosts or improve the quality of the software (Guinan *et al*., 1997; DeMarco, 1995; Kane, 1992; Brooks, 1987). In addition, the effects of management techniques such as Total Quality Management to address software productivity have ranged from modest improvements to complete abandonment of projects (Anthes, 1997; Williamson, 1997; Paulk *et al*., 1993). Rather than the use of advanced tools and practices, it has long been recognised that the critical factors in software development productivity relate to the attributes and interactions of development team members (Carmel and Sawyer, 1998; Krishnan, 1998; Brooks, 1993; Boehm, 1981).

Team size is often cited as a factor in the productivity of software development teams with small teams not necessarily being more effective, but large teams being more likely to have lower productivity (Brooks, 1993). This is due to complexity and communication cost of a project rising with the square of the number of developers, while actual work performed rises linearly (Brooks, 1993). However, the success of Linux indicates that large teams can also be effective (Raymond, 1999) and rather than team size, it is the attitude of the developers that is critical. This has prompted research aimed at identifying the drivers of "good attitudes" among software developers (Carmel and Sawyer, 1998) and the effect of differences among developers in a software project (Krishnan, 1998; Kemerer and Patrick, 1993). In particular, Krishnan (1998) identified, personal abilities and experiences of individual team members, facilities in support of effective communication, team composition, team member commitment, and the ability to build a shared vision as critical factors in successful development teams.

Some studies of software development teams have considered how the organisational culture of software development teams influences productivity. Carmel and Sawyer (1998) examine packaged software development teams in terms of different levels of analysis contributing to the cultural milieu. In their analysis, the packaged software industry is marked by immense time-to-market pressures and success is measured by profit and market share. The development environment is characterised by the unique positions that the developers hold. Developers in packaged software project typically tend work in a highly unstructured setting, working on small parts of the project relying on frequent coordination to "piece things together". Packaged software developers work in a bizarre style (Raymond, 1999) and would readily reject the custom software industrial-engineering-oriented notion of software development as being bureaucratic, boring, and stifling of critical innovation (Carmel and Sawyer, 1998). The development teams of packaged software are more cohesive, motivated and jelled in comparison. To some extent, this is due to the large rewards commonly associated with package software projects. The work culture of packaged software developers is entrepreneurial and highly individualistic. They are deeply influenced by the hacker culture that values the intangible of their own ego satisfaction and reputation among other hackers (Raymond, 1999).

While analyses such as these assist in characterising the organisational culture in which packaged software developers work, they do not address how the national cultural background of individual team members can impact the dynamics of the development team.

NATIONAL CULTURE AND SOFTWARE DEVELOPMENT TEAMS

Culture can be broadly conceptualised as a complex web of norms, values, assumptions, attitudes and beliefs that are characteristic of a particular group and these are reinforced and perpetuated through socialisation, training, rewards, and sanctions (Lytle *et al*., 1993). Hofstede (1991) describes culture as patterns of thinking, feeling, and acting. At the national culture level of analysis, Hofstede's individualism/ collectivism dimension for differentiating cultures is of particular significance for cross-cultural software development teams. This dimension can be used to understand the behaviour of social groups in software

development, as well as reflecting the differences in symbolic codes (or values) across cultures.

Individualism versus collectivism refers to the tendency of people to look after themselves and their immediate family or "ingroups" and neglect the needs of the society (Hofstede, 1991; Robertson and Hoffman, 2000). The concept of ingroup is highly relevant to the activities of software development teams. An ingroup is simply a social group. In a collectivist culture, there is normally one stable ingroup (e.g. family, band, tribe, etc) while in an individualistic culture, there are many more ingroups (e.g. family, clubs, motorcycles gangs, etc.). The concept of an ingroup means very different things in different cultures. The biggest difference between the ingroups of two cultures is the quality of relationships between the members within an ingroup (Triandis *et al*., 1988). Differences in the quality of relationships are a good indication of the degree of acceptance by group members of symbols like loyalty and trust. While a software development team forms a group of co-workers, the loyalties of individual team members may lie with their ingroup that is outside of the development team, with subsequent impact on their behaviour within the team. The influences on ingroup behaviour of collectivist and individualist cultures (Triandis *et al*., 1985) are summarised in Table 1.

Collectivism	**Individualism**
Ingroup regulation of behaviour	Individual regulation of behaviour
Interdependence	Self-sufficiency
Subordination of personal goals to goals of ingroup	Ingroup and personal goals are unrelated
Ingroup harmony is important	Confrontation within ingroup may be good
Shame control	Guilt control
Sense of common fate with ingroup	Person fate
Ingroup is centre of psychological field	Person is centre of psychological field
Ingroup is extension of the self	Self is distinct from ingroup

Table 1. Collectivist vs. Individualist cultures: Ingroup behaviours (Triandis et al., 1985)

WESTERN AND CHINESE CROSS-CULTURAL SOFTWARE DEVELOPMENT TEAMS

To further investigate how a mix of individualist and collectivist cultures can influence behaviour within cross-cultural software development teams, this section briefly examines the values or "symbolic codes" (Luhmann, 1982) that pervade, on the one hand Chinese culture (collectivist), and secondly "Western" culture (individualist). From these analyses, three propositions with respect to Chinese/ Western software development teams are put forward. These propositions are then examined through an interpretation of events in a small cross-cultural software development team.

Symbols of Chinese Culture

In Chinese culture, philosophy has become profoundly inseparable with life, theory and practice. The most important basic principles of Chinese philosophy, as identified by Moore (1967), are summarised below.

Firstly is the notion of achieving "sageliness within and kingliness without". That is, striving for superiority through interests for all. Secondly, in Chinese culture the predominance of ethical consciousness (both inner and outer) is the highest goal for man. When one achieves the fullest possible development of ethical characteristics and innate goodness, one is deemed to attain spirituality. Similarly, the classical Confucian notion that "the investigation of things begin with sincerity of will, leading to personal moral integrity, the well-established family, the well-ordered state, and peace in the world", sees ethical consciousness as the lens through which Chinese view the path to the ideal world. Finally,

the doctrine of filial piety possibly represents the very essence of Chinese ethical and social life. The respect of immediate elders illustrates the power perception that is rooted in the family structures.

The pervasiveness of principles such as filial piety and the notion of ethical precedence over of self, family, nation and world suggest that Chinese culture is collectivist since Chinese would readily consider the immediate family's welfare as part of their own ethical development. Therefore, values of trust and loyalty can be seen as strong influences on the behaviour of Chinese software developers.

Symbols of Western Culture

"Western" culture has undergone many transformations from the ancient Mediterranean cultures, through medieval Europe and the Renaissance and on to modern history. It is therefore difficult to epitomise. However, the root of Western philosophy can be traced to "Theocentricism" – meaning that God is at the centre of all viewpoints (Bourke and Kelkel, 1992). Laws, values, definitions of all sorts are established according to what God views as important. Coupled with the classical Christian view that humans are incapable of reaching perfection, there exists an inherent need to guide people in differentiating between (divine) good and evil. That is, there is an inherent need for rules governing peoples' actions.

From the early formation of monarchy and despotism through to the later developments of corporatism and bureaucracy, together with the agriculture (over environment), industrial (over machinery) and management (over people) revolutions, power and status have become inseparable with Western idealism.

Power endows the ruling classes with the ability to formulate roles that humans fulfil to ensure the survival of society. Through the establishment of multiple interrelated abstract systems (e.g. legal, financial, etc.), whereby humans are "specialised" into activities in exchange for recognisable returns (e.g. money), Western culture is somewhat more functionalistic when compared to others. However, through the proliferation of sub-systems, humans within Western culture are also presented with more choices of systems for interaction and the investment of deep, personal emotion is often not required. Western culture therefore has more freedom and flexibility for people in their dealing of daily lives (Triandis *et al.*, 1988; Luhmann, 1982).

PROPOSITIONS FOR CHINESE/ WESTERN DEVELOPMENT TEAMS

Given that Chinese and Western cultures promote different symbols and value systems, this section puts forth three propositions regarding behaviour in Chinese/ Western software development teams.

Proposition 1: The absence of deep, personal relationships amongst Western development teams will prevent Chinese developers from sharing their knowledge openly

The wider societal legal framework can affect developers individually. Copyrights, software rights and intellectual property rights for instance promote and protect an individualist and ego-laden style of programming. However, the same may not have an impact on Chinese programmer. Their inherent collectivist background suggests that Chinese software developers might be more willing to share code and algorithms. According to Gerald Weinberg (1971) in "The Psychology of Computer Programming", a Chinese programmer may practice what is termed "egoless programming", in that they are not territorial about their code and would encourage sharing and collaboration. However, while they may be more willing to share, they will be highly selective of the people they share with. Of the many reasons for this selectivity, the most important one is to improve the relationships with these people. Hence, Chinese developers often try to invest a personal stake with the people they work with. This may be problematic because while Chinese developers tend to be generous with team members who relate closely with them, the lack of relationship-cultivation mechanisms within Western organisations (since Western developers often do not invest in personal relationships) may prevent them from wider sharing. Such behaviour is also noted by Triandis *et al.*, (1988) who states that while cooperation is high in ingroups of a collectivist culture, it is unlikely when with other people.

Proposition 2: Communication within a cross-cultural software development team is handicapped since the symbolic codes used in communication are different

While they may not invest personal emotion, Western developers are also said to be cohesive, motivated and jelled (Carmel and Sawyer, 1998). The two driving factors are financial incentives and commitment. Financial incentives are highly generalised symbolic codes that are inherited from the wider societal system and used to integrate subsystems. The financial rewards for high quality software developers are enormous and are used to promote and maintain hard work and dedication of developers and "glue" (or integrate) team members together in a particular project.

Another form of incentive pervasive amongst software developers is status. Being "the best", "the first", "the one", etc., is deeply influenced by the hacker sub-culture. Both financial incentives and status can be considered as universal symbols since their values are acknowledged on a wider societal context. However, these symbols may only be able to motivate Western developers, in that Chinese may not always value them. Because of the Chinese strong ethical consciousness, Chinese developers may be quite ready to abandon money, power and status for symbols like trust, loyalty and morality. Trust, loyalty and morality are particular symbols in that their values are subjective. This can be problematic for a cross-culture software development team as it may prevent effective communication. According to Luhmann (1982), symbols are transmitted via communication to guide the selection of complexity reduction mechanisms. The different views on valid symbolic codes may potentially distort communication. For example, one developer may not wish to communicate with another if the topic is of little significance to him.

Since coordination and communication are important in distributed-style development projects, the failure to communicate will create difficulties in reducing the complexities of projects and hence be detrimental to their success.

Proposition 3: To Chinese developers, having a closely tied ingroup is more important than having a high performance team

Another important factor for a successful development team is the team composition. Teams with talented members having strong application domain and computer language experience are said to increase the success rate of a project. However, following the first previous propositions, Chinese developers' high propensity for particular resources is likely to undervalue those members with valuable universalistic resources. With their strong inclination toward ingrouping, team composition should perhaps focus on personal relationships rather than skill sets.

A SOCIAL NETWORK ANALYSIS OF A CROSS-CULTURAL SOFTWARE DEVELOPMENT TEAM

This section investigates these propositions for Chinese/ Western cross-cultural development teams by interpreting the behaviour of a Chinese developer working in a predominantly Western software development environment. The data for this study was collected primarily through participant observation and interviews with key actors over a twelve-month period (June 2000 to June 2001). Secondary data is drawn from documentation including the project development diary, group emails and memos.

In order to aid interpretation of the case, our analysis uses techniques for mapping and analysing the social networks related to the software development team. Social Network Analysis (SNA) is widely used in the social and behavioural sciences to examine relationships among social entities. SNA techniques have been applied in a variety of research contexts including communication among group members, transactions between corporations and treaties among nations (Burt and Minor, 1983; Wellman and Berkowitz, 1988; Wasserman and Faust, 1994). In relation to information systems, SNA techniques have been applied in the study of the computer-mediated communication, diffusion of innovations, IS implementation, and the study of the social influences of communications and information technologies (Rice and Aydin, 1991; Wellman *et al.,* 1996; Graham, 1998; Hislop *et al.,* 1998; Zack, 2000). Rather than examining the attributes of individual agents within the social system, SNA pays attention to the ties, contacts and meetings between agents that connect them in the larger relational systems (Scott, 1991). Sophisticated

quantitative techniques for the analyses of network structures have been developed and concepts such as “structural holes” and “network social capital” have emerged through their application.

In this paper, we borrow some principles and analytical techniques from SNA rather to support our interpretation, rather than performing a detailed structural analysis of the social networks. Specifically, we examine three networks of relations. Firstly, the network of relations directly associated with the project-related tasks of team members is investigated in order to study communication patterns between team members. Secondly, a social status network is constructed to provide an indication of how network members are perceived by others. Finally, the ingroup or personal network of the central actor in this case is examined. By focusing our interpretation on these three networks, the cross-cultural for software development team propositions are investigated.

Case Background

The software development project was conducted as part of a research initiative within an academic department of an Australian university. The project involved developing a generic research facility for tracking the choices and online behaviour of users of web-based systems. The team that was formed to develop the system drew upon expertise available within the department, the faculty-wide IT support group, and an external contractor. The development team consisted of five core members:

Name	Role	Description
James	Project designer and manager	The project owner; a department member; controls the project budget and all key decisions; a Westerner (USA)
Freddy	IT System administrator; programmer	Provided network and infrastructure advice; Faculty-wide IT support member; Westerner (Australian)
Fang	Software programmer	Involved in technical design; software selection; interface programming; Chinese (Singapore)
Kelvin	Programmer	Contracted for programming; Westerner (Australian) Added later to the project.
Charles	Project Sponsor	Funded the project; interested in project as part of longer term applications; Westerner (USA)

Table 2. Project Team Members

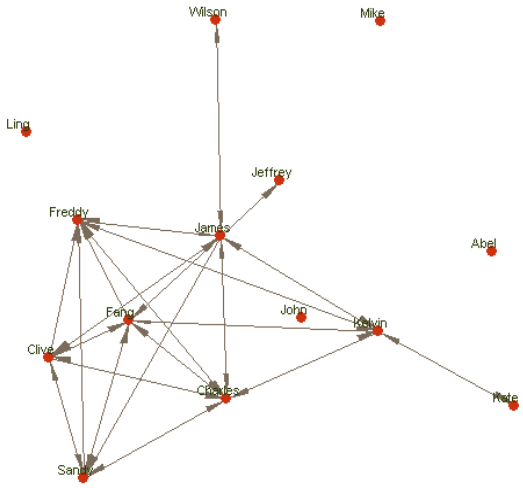
These actors are the central players in this case. However, in order to address the wider social networks of team members, additional actors are also considered in this analysis. Those actors who provided project-related advice to one or more of the team members are included, as well as, those actors who are associated with team members by virtue of organisational and social relations that are not directly related to the project.

James was both the manager of the project and the client. The impetus and funding for the project were derived from his particular research interest in the behaviour of online game players. It was further envisaged that the software developed for this project would have wider application in other research areas related to online behaviour. Initially, only James and two other actors (Fang and Freddy) formed the core development team. During the first four months, the project proceeded smoothly and beyond expectations. Various software modules and interface implementations were ready for beta testing months ahead of schedule. At some point however, Fang began to show signs of disinterest in the project and withheld knowledge relating to the technical viability of the project. He also withheld knowledge relating to possible solutions to technical problems that arose. Fang’s behaviour significantly slowed the progress of the project and subsequently, Kelvin was contracted by James to assist in the programming effort.

As the only non-Western team member and the apparent cause of the slowed productivity, Fang becomes the central figure in this analysis.

Software Development Team Social Network

Effective communication is regarded as a critical factor to both software development (Krishnan, 1998) and knowledge management (Gold *et al.*, 2001). The key to effective communications is the interaction of group members (McGrath, 1991). Therefore, in assessing communication among the development team members, a social network based on the affiliation of actors with software development tasks and the frequency of their interactions was constructed. Affiliations between actors were based on their joint involvement in software development tasks such as code design, project management, software testing and results in connectivity between ten members of the wider social network (Figure 1a).



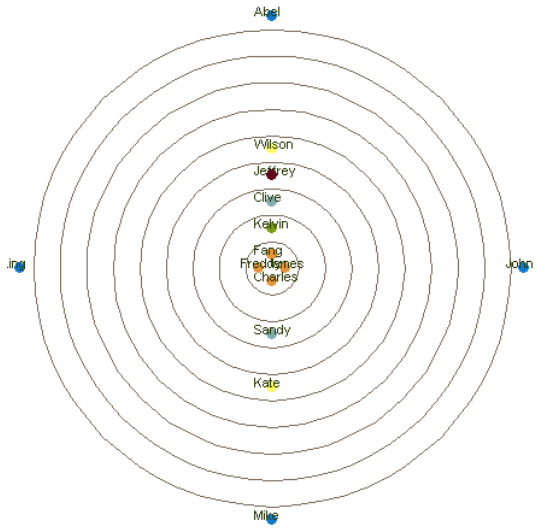


Figure 1a: Connectivity – Adjacency Diagram Figure 1b: Centrality – Closeness Diagram

The core team members (Fang, James, Freddy, Kelvin and Charles) are all closely connected through the project related tasks, suggesting that there are no structural constraints on the interaction of team members. The centrality-closeness diagram (Figure 1b) emphasises the close task-related relationship between the core team members. This analysis indicates that project failure was not due to a lack of interaction between group members.

Status Social Network

Status is identified as an important source of motivation in software development (Raymond, 1999) and knowledge management (Huber, 2001). A status social network based on the extent to which actors recognise the capabilities and expertise of members within the development team can provide some indication regarding how individual team other actors perceive member's competence in the wider network. For the purposes of this study, the focus is on Fang's perceived status within the network.

The analysis shows that Fang has the highest "out-degree" score (8) of all network actors. That is, Fang is perceived to be competent in relation to his software development duties by eight other network actors. Thus his status as a competent software developer is widely recognised within the network. Furthermore, throughout interviews with various network actors, team members acknowledged his expertise by citing him as the "guru", "the man" and "he's the one to look for" in getting the job done. Under this analysis, Fang's behaviour that led to the stalling of the project cannot be attributed to other team members failing to acknowledge his skills, expertise and contribution to the project.

Ingroup Social Network

Ingroup behaviours are dependent on the cultural variations across Hofstede's (1991) individualism/ collectivism dimension (Triandis *et al.*, 1985). Given that Fang is the only non-Western member of the development team, his ingroup behaviour is examined in a network

that represents the strength of personal ties between the team members. This network is constructed based on whether actors in the network had personal affiliations with each other rather than task related affiliations.

A "clique" and "subgroup" analysis of this network shows that Fang is a member of three cliques (K1, K2 and K3 in Figure 2a).

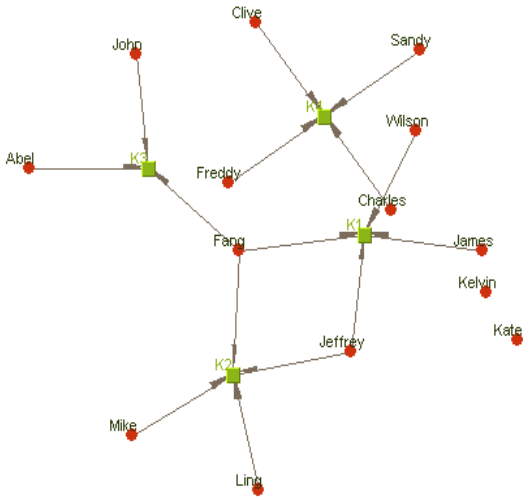


Figure 2a: Subgroup-Cliques Diagram

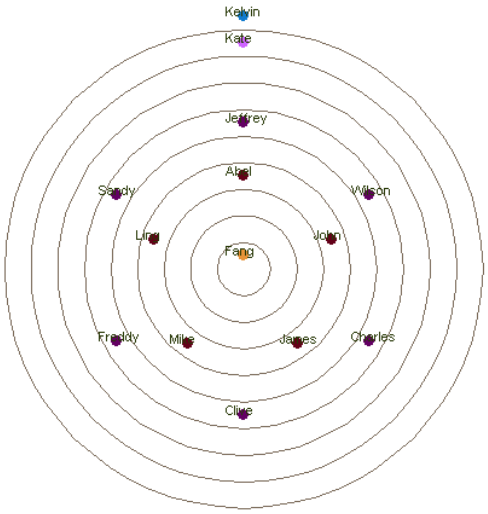


Figure 2b: Centrality-Closeness Diagram

However, his relationships with the members of the three cliques are not uniform. From the centrality-closeness analysis, actors who are likely to be ingroup members can be identified from within the inner circle of Figure 2b (Abel, John, Ling and Mike). During interviews with network actors, a strong negative relationship between one of Fang's close ties (Abel), and the key project owners (Charles and James) was identified. The animosity between these actors was not directly related to the software development project. Working from the premise that the behaviours exhibited by members of an ingroup is group regulated rather than self-regulated (Triandis *et al.*, 1985) Fang's negative behaviour toward the project can be attributed to the influence of Abel. It was not through any direct experiences that Fang's relationships with other team members were soured. However, given the collective opinions of respected peers (Abel and John) and some negative "stories" he was told by friends (Ling and Mike), Fang's attitude toward the project was coloured. Consequently, Fang's ingroup behaviour represents a plausible explanation for his detachment from the project.

PROPOSITIONS FOR CHINESE/ WESTERN SOFTWARE DEVELOPMENT TEAMS REVISITED

Proposition 1: The absence of deep, personal relationships amongst Western development teams will prevent Chinese developers from sharing their knowledge openly

While there was a high degree of interaction among team members, their interactions were restricted to project-related tasks and not of a personal nature. The collectivist culture of Chinese programmers suggests that they are inclined to share code and algorithms with people with whom they have developed strong team relationships. In this case, no mechanisms were in place to develop strong team relationships and no effort was made on behalf of the project owners to cultivate a team environment.

Proposition 2: Differences in the recognition of symbolic codes can undermine the effectiveness of management practices based on universal symbols and norms

Status is a universal symbol since its value is acknowledged in a wider societal context. However, Chinese developers may not always value status and may be quite ready to abandon money, power and status for symbols like trust, loyalty and morality. These are particular rather than universal symbols in that their values are subjective. Differences

between team members about which symbolic codes are viewed as important may therefore affect team member practices.

In software development, driven by the influence of the hacker sub-culture (Raymond, 1999), being “the best”, “the first”, “the one”, etc., can instigate an individual to put in extra effort. Similarly, in knowledge management, being the “thought leader” may be “enough to incent an individual to contribute to a knowledge base” (Rappleye, 2000). In the case described above, Fang was widely acknowledged by network actors as being highly competent and his status as a software developer was never questioned. However, recognition and monetary reward for this level of status did not ensure Fang’s dedication to the project.

Proposition 3: To Chinese developers, having a closely tied ingroup is more important than having a high-performance team

Although team composition is an important factor in successful development teams (Krishnan, 1998) and having members with appropriate skill sets increases the degree of knowledge sharing and dissemination (Zack, 1999), the Chinese software developers’ inclination to particular rather than universal symbols suggests that they are likely to dismiss those members who value universal resources. In this case, while Fang worked within an environment with high degree of shared contextual knowledge, it was the regulation of his ingroup members that dictated his willingness to share and disseminate his knowledge.

CONCLUSION

This paper has highlighted potential problems that can arise in cross-cultural software development teams. As organisations increasingly adopt global strategies and source software development expertise from different countries, software development teams comprising members from different cultural backgrounds are likely to increase. This paper has specifically focussed on potential problems that could arise in teams with a mix of Chinese and Western developers. Our aim has been to illustrate that cultural factors can play a role in the effectiveness of software development activities. Furthermore, we have investigated the utility of adopting a social network approach to interpreting team member interactions and behaviour. By identifying different underlying cultural values and symbols of Chinese and Western cultures, three propositions have been put forward regarding the behaviour of Chinese/ Western software development teams. These propositions have been investigated in an interpretive analysis of a small software development team with a Chinese/ Western composition using social network analysis techniques. This analysis did show some support for the propositions that were put forward but we do not wish to rule out that alternative explanations of Fang’s behaviour are possible. Our limited application of social network analysis techniques also shows some promise as a methodological tool for further investigation of cultural influences in software development environments.

Managers of software development projects that use cross-cultural teams cannot assume that the typically Western-oriented management practices and structures will guarantee project success. Further research needs to be conducted in order to investigate the forms of rewards, structures and practices that will facilitate teams with a cross-cultural composition.

REFERENCES

Anthes, G. H., (1997). Quality? What’s That? Computerworld, October 13, 1997, pp. 75.

Boehm, B. W., (1981). Software engineering economics. Prentice-Hall, Inc. Englewood Cliffs, NJ.

Bourke, V. J. and Kelkel, A., (1992). Augustine’s Love of Wisdom: An Introspective Philosophy. Purdue University Series in the History of Philosophy, Purdue University Press

Brewer, M. B., (1986). The role of ethnocentrism in intergroup conflict. In Worchel, s. and Austin, W. G., Psychology of intergroup relations. Nelson-Hall, Chicago.

Brooks, F., (1987). No silver bullet: essence and accidents of software engineering. Computer, Vol. 19, No. 5, pp. 10-19.

Brooks, F., (1993). The mythical man month. Addison-Wesley, Reading. MA.

Burt, R.S. & Minor, M.J. (1983) Applied Network Analysis. Sage Publications, London.

Carmel, E. and Sawyer, S., (1998). Packaged software development teams: what makes them different? Information Technology and People, Vol. 11, No. 1, 1998, pp. 7-19.

Dagwell, R. and Weber, R., (1983). System Designers' User Models: A Comparative Study and Methodological Critique. Communications of the ACM, Vol. 26, No. 11, November 1983, pp. 987-997.

DeMarco, T., (1995). Why does software cost so much? And other puzzles of the Information age. Dorset House Publishing, NY.

Doherty, N. F. and King, M., (1998). The importance of organizational issues in systems development. Information Technology and People, Vol. 11, No. 2, pp. 104-123.

Gold, A. H.; Malhotra, A. and Segars, A. H., (2001). Knowledge Management: An Organizational Capabilities Perspective. Journal of Management Information Systems. Vol. 18, No. 1, pp. 185-214.

Graham, I. (1998). The Construction of a Network Technology: Electronic Livestock Auction Markets. International Journal of Innovation Management. Vol.2(2), pp. 183-199.

Guinan, P., Cooprider, J. and Sawyer, S., (1997). The effective use of automated application development tools. IBM Systems Journal, Vol. 36, No. 1, pp. 124-39.

Hislop, D., Newell, S., Scarborough, H. & Swan, J. (1997). Innovation and Networks: Linking Diffusion and Innovation. International Journal of Innovation Management. Vol.1(4), 427-448

Hofstede, G., (1991). Cultures and Organizations: Software of the mind. McGraw-Hill, London.

Hofstede, G., (1998). Attitudes, Values and Organizational Culture: Disentangling the Concepts. Organizational Studies, Vol. 19, No. 3, pp. 477-492.

Huber, G. P., (2001). Transfer of knowledge in knowledge management systems: unexplored issues and suggested studies. European Journal of Information Systems, No. 10, pp. 72-79.

Kane, E. J., (1992). Implementing TQM at Dun and Bradstreet Software. National Productivity Review, Vol. 11, No. 3, Summer 1992, pp. 405-416.

Kemerer, C. F. and Patrick, M. W., (1993). Staffing factors in software cost estimation models. In Keyes, J., Software Engineering Productivity Handbook. McGraw-Hill.

Kinder, D. R. and Sears, D. O., (1981). Prejudice and politics: Symbolic racism versus racial threats to the good life. Journal of Personality and Social Psychology, 40, 414-431.

Krishnan, M. S., (1998). The role of team factors in software cost and quality. Information Technology and People, Vol. 11, No. 1, 1998, pp. 20-35.

Kumar, K. and Bjørn -Andersen, N., (1990). A Cross-Cultural Comparison of IS Designer Values. Communications of the ACM, Vol. 33, No. 5, May 1990, pp. 528-538.

Luhmann, N., (1982). The Differentiation of Society. Columbia University Press, NY.

Lytle, A. L., Brett, J. M., Barsness, Z. L., Tinsley, C. H. and Jansen, M., (1993). A paradigm for confirmatory cross-cultural research in organizational behaviour. Research in Organizational Behaviour, 17., 167-214.

McConahay, J. B., (1986). Modern racism, ambivalence, and the modern racism scale. In Dovidio, J. F. and Gaertner, S. L., Prejudice, discrimination, and racism. Hardcourt Brace Jovanovich, Orlando.

McGrath, J. E., (1991). Time, Interaction and Performance (TIP): A Theory of Groups. Small Group Research, Vol. 22, No. 2, pp. 147-174.

Moore, A. C., (1967). The Chinese Mind. East-West Center Press, Honolulu.

Paulk, M. C., Weber, C. V., Garcia, S. M., Chrissis, M. and Bush, M., (1993). Key practices of the capability maturity model, version 1.1, Technical Report, CMU/SEI-93-TR-25, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.

Punnett, B. J. and Shenkar, O., (1995). Handbook for international management research. Cambridge, MA.

Rappleye, W. C., (2000). Knowledge Management: A force whose time has come. Across the Board: The Conference Board Magazine. January, pp. 59-66.

Raymond, E. S., (1999). The Cathedral and the Bazaar. O'Reilly, New York.

Rice, R & Aydin, C., (1991). Attitudes toward New Orgnizational Technology: Network Proximity As a Mechanism for Social Information Processing. Administrative Science Quarterly. Vol 3, pp. 219-244.

Robertson, C. J. and Hoffman, J. J., (2000). How Different Are We? An Investigation of Confucian Values in the United States. Journal of Managerial Issues, Vol. 12, No. 1, pp. 34-47.

Scott, J., (1991). Network Analysis: A Handbook. Sage, CA.

Shore, B., (1996). A Conceptual Framework to Access Gaps in Information Systems Cultures Between Headquarters and Foreign Subsidiaries. In Palvia, P. C., Palvia, S. and Roche, E. M., 1996. Global Information Technology and System Management: Key Issues and Trends. Ivy League Publishing, Westford.

Stephen, W. G., (1994). Intragroup anxiety. Proceedings at Society of Experimental Social Psychology, Lake Tohoe, CA.

Thanasankit, T, and Corbit, B., (1999). Towards an Understanding of the Impact of Thai Culture On Requirements Elicitation. Conference on Information Technology in Asia: CITA '99. The Asian Regional Conference of IFIP WG 9.4.

Triandis, H. C., Leung, K., Villareal, M. J. and Clack, F. L., (1985). Allocentric versus Idiocentric Tendencies: Convergent and Discriminant Validation. Journal of Research in Personality, 19, 395-415, 1985.

Triandis, H. C.; Bontempo, R. and Villareal, M. J., (1988). Individualism and Collectivism: Cross-Cultural Perspectives on Self-Ingroup Relationships. Journal of Personality and Social Psychology, Vol. 54, No. 2, pp. 323-338.

Wasserman, S. & Faust, K. (1994) Social Network Analysis: Methods and Applications. Cambridge University Press, Cambridge.

Wellman, B. & Berkowitz, S.D. (1988) Social Structures. Cambridge University Press, Cambridge.

Wellman, B., Salaff, J., Dimitrova, D., Garton, L., Gulia, M. & Haythornthwaite, C., (1996). Computer Networks as Social Networks. Annual Review of Sociology, Vol. 22: 213-38.

Williamson, M., (1997). Quality Pays. Computerworld, August 18, 1997, pp. 78-81.

Weinberg, G., (1971). The Psychology of Computer Programming. Van Nostrand Reinhold Press.

Zack, M. H., (1999). Managing Codified Knowledge. Sloan Management Review, Vol. 40, pp. 45-58.

Zack, M. H., (2000). Researching Organisational Systems using Social Network Analysis. Proceedings of the 33rd Hawaii International Conference on System Science. January, 2000.

COPYRIGHT