Ad-Hoc Leadership in Agile Software Development Environments

Yael Dubinsky
IBM Haifa Research Lab
Mount Carmel
Haifa, 31905, Israel
dubinsky@il.ibm.com

Orit Hazzan
Department of Education in Technology and Science
Technion – Israel Institute of Technology
Haifa 32000, Israel
oritha@techunix.technion.ac.il

ABSTRACT
Leadership is the ability to influence people, leading them to behave in a certain way in order to achieve the group's goals. Leadership is independent of job titles and descriptions. Usually, however, in order to lead, leaders need the power derived from their organizational positions. There are different leadership styles, like task-oriented versus people-oriented, directive versus permissive, autocrat versus democrat. In this paper, we examine the leadership concept in software development environments and focus on leadership in transition processes to agile software development. Specifically, based on our comprehensive research on agile software development, we suggest a leadership style – ad-hoc leadership – that usually emerges in such change processes. We present the characteristics, dynamic and uniqueness of this leadership style and illustrate its usefulness for the analysis of representative scenarios.

Categories and Subject Descriptors
K.6.1 [**Project and People Management**]: Management Techniques; K.6.3 [**Software Management**]: Software Process.

General Terms
Management, Human Factors.

Keywords
Leadership, agile software development, change leader, ad-hoc leadership

1. INTRODUCTION
Leadership is a social phenomenon required for achieving group's goals [20]. Various definitions for leadership exist. The evolution of these definitions across the last century reflects a change in leadership conception: from leadership conception as a unilateral ability or process to control people, to leadership conception as an influential relationship between leaders and followers [4, 6]. In other words, leadership definitions moved from the description of a unique person, who was born as a leader, to holistic definitions for leadership that refer to leaders and followers, the interaction among them and the task in hand [24].

In software development environments, leadership is required for the management of different activities in different situations e.g, in the case of leading partially distributed teams [21], in the case of coordinating large scale development [2], or in the case of collaborating with testing managers [18]. Management of a software project includes team management, process management, quality management and cost management [15, 17, 19] and consists on constant assessment of its own activities [22, 23]. Table 1 outlines these four categories; for each category, several main activities are described. For example, time management is part of the process management. Time is managed on different levels of the organization and for different process phases. Specifically, one can manage his or her personal time within a day or within a specific iteration/release; a team manages its time within iteration/release; and a team that is composed of several teams may manage and coordinate its time resources within iteration/release.

Table 1. Software Development Management

Management category	Description
Team management	Increasing awareness to human factors Consolidating group and introducing organization values Arranging suitable development environment Updating organization and group knowledge base Enhancing communication Defining and implementing training programs
Process management	Detailing process activities Managing time Implementing a role scheme Obtaining metrics regarding time, resources and events Improving the process continuously
Quality management	Defining quality goals Defining and implementing quality procedures Setting metrics regarding quality Improving quality procedures
Cost management	Defining and implementing a cost model Referring to project, product, resources

CHASE '10, May 2, 2010, Cape Town, South Africa

	and personnel aspects Using a costing model as a metric for process status

Agile software engineering adopts a leadership style that empowers the people involved in the development process. For example, instead of promoting the idea that leaders should keep the power to themselves in order not to loose it, the agile approach fosters the idea that leaders gains power by its distribution and sharing. This idea is expressed, among other ways, by the transparency of agile software development processes that makes information accessible to anyone and enables each team member to be fully accountable and involved in the development process.

The transition to agile software development involves a conceptual and organizational change. This shift is expressed, for example, in role redefinition, a tighter development rhythm and an enhanced cooperation level. In our comprehensive research on the implementation of agile software development (see, for example, [13, 9,10]), we found that each transition process can be identified with a person who initiates the change and accompanies it during the different transition phases. We call this person as the *Change Leader* (CL) and in this paper we report on a research work that focuses on the positioning of the CL role and on its relationships to other entities involved in transition processes to agile software development. Specifically, we present a leadership model – *ad-hoc leadership* – that reflects the CL role's positioning and dynamics during the transition and change process.

The contribution of our research includes the a) definition of the *CL role* and its position in transition processes to agile software development; b) presentation of a new leadership style, *ad-hoc leadership*, which is usually emerged in cases in which temporary leadership is needed and c) analysis of field scenarios by the *ad-hoc leadership* model.

In what follows, we elaborate on leadership and leadership styles and present the CL role. Then, we describe the research setting, present the *ad-hoc leadership* model and illustrate its usages.

2. LEADERSHIP AND CHANGE LEADERSHIP IN AGILE PROJECTS

Management and leadership are described as a continuous act that aims to keep the balance between characteristics that on the one hand, dominate and control and on the other hand, are inspiring and strive for creativity. Table 2 (adopted from [16] originally from [7]) presents the evolution of leadership models, indicating a shift in leadership perception.

Table 2. Evolving Models of Leadership (source: [7]

	Ancient	Traditional	Modern	Future
Idea of Leadership	Domination	Influence	Common goals	Reciprocal relations
Action of Leadership	Commanding followers	Motivating followers	Creating inner commitment	Mutual meaning making
Focus of the Leadership Development	Power of the leader	Interpersonal skills of the leader	Self-knowledge of the leader	Interactions within the group

In software development environments, "Leadership is generally taken to mean the ability to influence others in a group to act in a particular way to achieve group goals" [15, p. 222]. In [1], an adaptive leadership is suggested for agile teams including first, the ability to adapt to changes and second, the problem-solving approach that considers all stakeholders to be skilled and valuable, relies on autonomous teams and minimizes up front planning to be able to adapt to changes. It is important to note that autonomous agile teams also have leaders who work in a leadership-collaboration environment [5]. This means that in many cases, team members experience collegial relationships with the team leader and not hierarchical ones.

Referring to Table 2, the *agile* approach for software development fits the 'modern' and 'future' leadership styles. In what follows, we elaborate on this conception.

With respect to the *Idea of Leadership*, the notion of common goals in agile teams is mainly expressed by the customer on going collaboration during the entire development process and by the information transparency, which enables each team member to be exposed to the common goals and to participate in planning and presentation meetings related to these goals. Reciprocal relations in agile teams relate to high levels of cooperation, confidence and trust among the team members. In [12], for example, we use game theory to explain reciprocation in software development environments by employing the prisoners' dilemma.

With respect to the *Action of Leadership* (Table 2), inner commitment is created and enhanced by using a role scheme by which each team member has an additional specific role that assists the project leadership [9]. Though team members are committed, mutual meaning is still needed, i.e., the commitment should produce a relevant and meaningful product.

The *Focus of the Leadership Development* aspect in Table 2 shows how the leader position should be developed in order to improve leadership. Specifically, while the three first columns focus on the leader, the 'future' column deals with the group and its interactions. As the level of leadership increases, the group interactions lead the team, i.e., the way team members communicate, reflect, and collaborate enables the team to lead itself as if there is no leader, while, in practice, high quality leadership exists.

2.1 Leadership Styles

There are several leadership styles in software development environments [3, 15].

The first and most common style ranges from dictatorship or autocracy to democracy. The autocrat leader decides alone while the democrat leader involves the team members in decision making processes. In between these two poles we have the analytic leader who makes decisions based on data collection and analysis, and the opinion-seeking leader who seeks for the team members' opinions in order to make a decision.

The second scale ranges from directive to permissive leadership. The directive leader closely supervises the implementation, while the permissive leader lets team members to have latitude. This scale also includes the level of delegation a leader uses, i.e., the degree to which he or she delegates tasks to team members.

Another scale is known as task-oriented leadership versus people-oriented leadership. The former relates to leaders' focus on the

task in hand while the latter relates to leaders' focus on the people involved.

Finally, emotional-intelligence is a relative new leadership style scale [11]. Since human beings' behavior is influenced by emotions, leaders can elevate emotions for leadership improvement.

As stated before, in the continuation of this paper, we present a new leadership style – *ad-hoc leadership*. Unlike the above leadership styles, which are characterized by two poles, the *ad-hoc leadership* style is characterized by three poles.

2.2 The Change Leader in Agile Projects

In general, an agile change leader, whom we call the methodology Change Leader (CL), can be found in almost any transition process to agile software development. The CL is an insider who knows well why change is required in his or her specific environment and finds it worthwhile to adopt the agile spirit either as a whole or only several of its practices. He or she is aware of the problems in the organization with respect to software development, can, in most cases, analyze them, and has reached the conclusion that the agile approach can solve them with some degree of success. The CL can belong to the organization management or to one of the software groups. In any case, the CL is busy convincing the management to make the initial decision towards adopting the agile approach. This process can sometimes take months and even years. In practice, the CL is a kind of mediator between the different parties involved in the transition process. The CL understands all sides involved and can put him or herself in the shoes of either side.

The CL role evolves gradually, as the CL, as well as the other people involved in the process, learn while implementing the agile approach on a daily basis. A CL should have enough patience and strength to lead a learning process that has its ups and downs, in which, on the one hand, everyone knows that the CL is there for them, and on the other hand, there are some cynics who just wait to watch him or her fall. In addition, the CL establishes a network of team-level methodologists[1], who together serve as the backbone of the transition process and later on support the sustainability phase of the transition process.

3. THE RESEARCH

3.1 Research Background

As aforementioned, the research described in this paper is a part of our comprehensive research on the implementation of agile software development in software teamwork. We started the examination of leadership in agile software development environments by defining a role scheme for agile teams [9] and, specifically, by examining the project-supervisor role in an academic setting of agile teams [8]. Later, in our book [14] we present the HOT – Human, Organizational, and Technical – scale, which provides a multifaceted perspective for the description and analysis of software engineering case studies. The *Human* perspective includes cognitive and social aspects, and refers to learning and interpersonal processes between teammates, customers, and management. The *Organizational* perspective includes managerial and cultural aspects, and refers to software project management and control. The *Technological* perspective includes practical and technical aspects, and refers to design, testing, and coding, as well as to integration, delivery, and maintenance of software products.

[1] A methodologist is a team member whose role is to guides the team members how to follow the principles of the software development method and is responsible for the method implementation [8].

3.2 Research Setting

In the work presented in this paper, we focus on the industry setting and investigate how leadership is expressed in change processes which are required in the transition process to agile software development. Specifically, we focus on the *Change Leader* (CL) – the person who leads the transition to agile software development. In the five cases on which the paper is based, this person initiated the work with us as consultants and cooperated with us during the consulting period.

Data for this research were gathered in five software projects developed in five companies: one in a service company (small-medium size) and four in product companies (two small-medium companies and two large-scale enterprises). The subjects of the projects vary from financial to security and electronics fields. The five CLs we worked with (two women and three men) have different titles in their respective organizational hierarchy: from a team leader to a vice president of research and development. The time of our engagement with these projects varies from half a year to two years. Our engagement in all these projects was external and involved an average of 15-20 hours per month, which were dedicated mainly for training, participation in central events and retrospective facilitation.

3.3 Research Tools

Data were collected by the following research tools:

- Observations in significant meetings and specifically business days [14] in which after each short iteration of development, the team presents to the customer the features that were developed in the previous iteration, a customer feedback is provided, a reflection on the previous iteration is facilitated, and a planning session is performed in which the customer prioritizes and decides upon the scope of the next iteration.
- Questionnaires distributed to team members and management representatives.
- Interviews with key roles in the organizations and occasional conversations with team members.
- Correspondence with the CLs.
- Artifacts collected from retrospective sessions.
- Researcher notes.

3.4 Main Observations

From the leadership perspective, the data analysis yielded three main observations:

1. The role of the CL is crucial in transition processes to agile software development. Though the CLs keep learning the agile approach during the transition process, they are perceived as agile supporters and resistance to agile development is directed, in many cases, personally towards them. Their success as CLs relates, among other factors, to

the support level they receive from higher management and its consistency.

2. CLs can be characterized by a leadership style that we name *ad-hoc leadership*. The main characteristic of this leadership style is its temporal manner. It emerges when there is a specific problem for which the CL conceives of the agile approach as an appropriate solution; it terminates when the CL exhausts its power and authority with respect to the change process. Then, a pullback can sometimes be observed in some elements of the transition process and, in many cases, additional steps of the transition process can not be further made.

3. The *ad-hoc leadership* in general and the interactions of the CL with the other roles in the transition process in particular, can be described by a model that can be used for the analysis of organizational and team scenarios that occur during transition processes to agile software development. This model is presented and illustrated in the next section.

4. THE AD-HOC LEADERSHIP MODEL

In this section, we described the *ad-hoc leadership* model, which reflects the interaction between the CL and the different players involved in transition processes to agile software development. The model has been constructed based on the analysis of data gathered in the above-mentioned five software projects.

Figure 1 presents the *ad-hoc leadership* model. As can be seen in Figure 1, in addition to the CL, the other organizational entities participating in this model are the management, the customer, and the software team or teams. The consultant entity, that most companies hire to guide transition processes, appears as an external entity to the organization. The consultant role is to collaborate with the CL in fostering the implementation of the agile methodology.

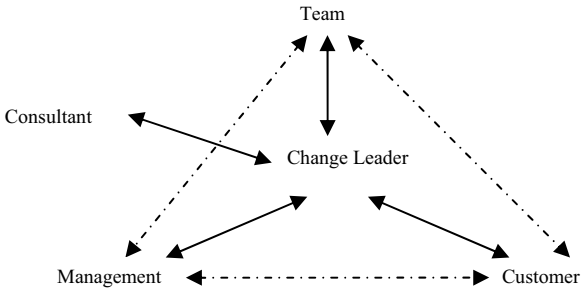

Figure 1. The Ad-hoc Leadership Model

The CL role, which is placed at the heart of the model, plays a key role in the transition process, as is described in what follows:

- The CL serves as a *knowledge* center, who is aware of all information related to the transition process, including plans, problems, and documents.
- The CL serves as a *leadership* center in the transition process, who has the ability to lead the process and the people involved in directions he or she sees as being the best.
- The CL serves as a *communication* center, who relays various kinds of messages among the different entities.

Each of the relationship lines in Figure 1 represents a different kind of interaction and serves different purposes. The dashed lines represent interactions in the organization that are not made via the CL; the solid lines represent interactions with the CL. Note that since the CL and the consultant are new to the organization and appear as a result of the transition process, the only interactions that existed prior to the transition were those depicted by the dashed triangle between the management, customer, and the development team. Still, since this model represents the transition, the relationship content of this triangle is changed by the transition process to agile development. We note also, that this model highlights also the fact that the customer in agile software development environments has a significant role, which is *not* part of the management role or the team.

4.1 Relationships in the Model

We describe the different CL-related relationships in the model as they are manifested in the management-team-customer triangle. In general, the CL conveys the management's needs to the other people involved in the process, and returns feedback from the process to the management. In other words, the CL establishes a communication and feedback channel among the entities, as follows.

Management ↔ CL ↔ Customer. Since the customer role is redefined in agile projects, the role of the CL is, on the one hand, to guide the customer to perform according to the new responsibilities and, on the other hand, to guide the management to accept and cooperate with the customer as an additional stakeholder. Specifically, in the Management ↔ CL ↔ Customer chain:

- Management ↔ CL: The CL works with the management in order to reach an understanding of the customer's role and authority scope, including the definition of the internal or external contract between the organization and the customer. In general, the CL is usually more concerned with the management's side.
- Customer ↔ CL: The customer, guided by the CL, learns how to plan requirements, how to give constructive feedback to the team and which measures to use to gauge the process success. The customer learns how to cooperate with the management, becoming aware of business aspects, such as, risks and constraints.

Team ↔ CL ↔ Customer. The main and most noticeable characteristic of this relationship is "presence"; the CL is present in as many meetings as possible between the team and the customer, especially during the business days in which the team presents, receives feedback, and participates together with the customer, in the planning session for next iteration. Specifically, in the Team ↔ CL ↔ Customer chain:

- Team ↔ CL: The CL is involved in the selection of the most appropriate team to start the process in the organization, as well in the selection of any subsequent teams. The CL encourages the team during the process, which is guided by the consultant, and delivers the message that the transition is part of the management initiative. The CL also listens to the teammates in order to understand their difficulties with respect to the transition process (some of them can be solved using the organization's internal resources).
- Customer ↔ CL: The CL conveys feelings, problems and requests that comes from the customer and works with the

management and team to increase customer satisfaction. In a sense, the CL serves as the organization's "sensor" especially with respect to the fulfillment of customer requirements.

Team ↔ CL ↔ Management. The Team ↔ Management relationship exists under all conditions and, in the transition process, receives additional attention in order to ensure success. Specifically:

- Team ↔ CL: The CL brings the management's vision to the team and shares with it the management's praises as well as its concerns. The team uses this interaction to deliver their fears and concerns. The CL also searches for team methodology leader (Methodologist) in the different teams undergoing the transition process. Thus, the CL builds a network of methodology CLs and receives constant feedback from the field on the agile implementation.
- Management ↔ CL: The CL delivers feedback from the team to the management and together with the management deals with special events concerning the team. The CL is the management's representative in the presentation and planning session of each iteration, and updates the management regarding the customer's level of satisfaction.

4.2 The Model Dynamics

The entities in the model are all people who are undergoing transition; thus, some level of dynamics can be expected. The CL entity, in particular, can be observed as a dynamic entity that is committed to a specific subject matter pending the circumstances. The three entities that constantly interact with the CL and are inherently significant to the transition process are the team, the management, and the customer. Using these entities, we describe the model dynamics by three categories, each represents a special focus on the respective entity, and illustrate this dynamics by scenarios taken from our research field.

Category I – The Team. The first category is *The Team*, that, among different changes, goes through a redefinition process of specific roles, responsibilities and authorities. An example for such a change is the team leader role, for whom, in agile software development, the daily work distribution is eliminated from the role definition. The following scenario, illustrates this category.

*Illustrative Scenario***.** Several teammates talk with the CL about their concerns with the way their role is changed in the transition process to agile software development. The CL realizes that in order to improve the software development process of the team, these changes in role definitions are essential. Therefore, the CL tries to encourage the team members to see the positive sides of the change; some of them, however, still feel that they wish to leave the organization. The CL decides that this situation requires the involvement and help of both the management and the consultant. This approach reflects how the CL leans towards the Team in order to solve the problem, by interweaving both desires, i.e., keeping a high technical process while supporting the people involved in the process (see Figure 2).

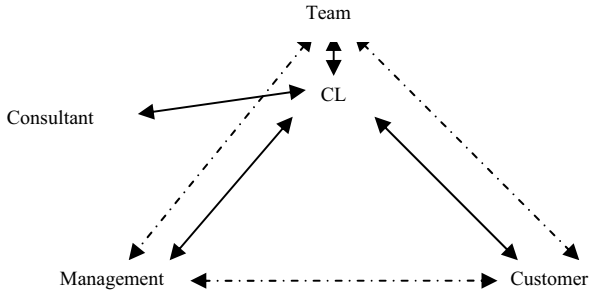

Figure 2. Leaning Towards the Team

Category II – The Customer. The second category is *The Customer*. The emphasis placed on customer collaboration in agile projects causes several changes. Specifically, in agile development environments, the customer defines the requirements, decides on their priorities and provides feedback to the team with respect to the developed scope after each short iteration of 2-4 weeks. Such highly collaborative relationships with the customer cause the CL to be highly sensitive to the customer feedbacks and satisfaction. The following scenario illustrates this category.

*Illustrative Scenario***.** During one of the presentation and planning session, the customer expresses deep concern about the development progress of a production release of a specific component of the developed product. The CL feels that not enough has been done to address this concern and suggests to construct a specific plan to ensure that the production release is dealt with within the next few iterations. The CL further stresses that this is the management's wish. The consultant suggests a measure to be used specifically for the advancement of this component's progress, and the team's tracker adds it to the team board in the collaborative workspace for daily viewing. Figure 3 shows the CL dynamics in this case.

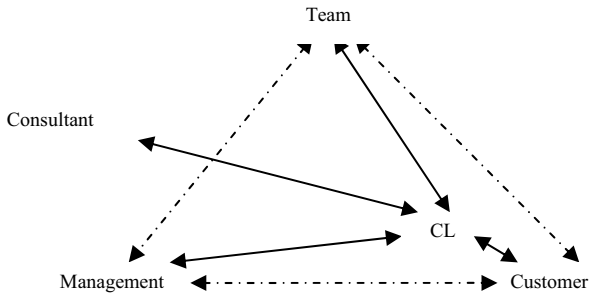

Figure 3. Leaning Towards the Customer

Category III – The Management. The third category is *The Management*. Naturally, the CL wants the transition process to be conceived as a success. Though the CL gets a reasonable level of support from the management during the entire transition process, the management keeps reminding the CL that the agile approach should still prove itself, i.e., the CL is asked to provide evidence that agile software development benefits with the business and the organization. The following scenario illustrates this category.

*Illustrative Scenario***.** The management group hears that a senior member of *The Team* strives to define one person from the

customer group to act as the primary customer with respect to the communication with the team. Further, the senior member from *The Team* asks to accommodate the development team, which sits in separate development areas, in one space. Both requests are not approved by the management. The CL understands the management concerns and in order to increase the cooperation from the people involved, decides to work together with the management in order to reach a clearer management message with respect to its support in agile software development. Since the management decides to stop temporarily the transition process, the CL arranges a meeting with the management, the senior member from *The Team,* and the consultant, in order to explain what happened and to get an approval to resume the transition process. Figure 4 reflects the CL dynamics in this case.

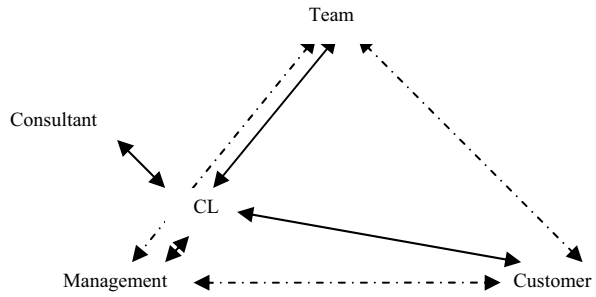

Figure 4. Leaning Towards the Management

5. CONCLUSION

This paper deals with leadership in agile software development environments in general and change leadership in transition processes to agile software development in particular. Specifically, in this paper we present a leadership style – *ad-hoc leadership* – that, based on our comprehensive research, fits for such situations. A model is presented that reflects this leadership style and its dynamic and it is illustrated with scenarios taken from case studies analyzed in our research. We highlight two of the model characteristics:

a) *Ad-hoc Leadership – The Scale*. Unlike other leadership styles which are usually characterized by two-pole scales, the ad-hoc leadership style is characterized by three poles: the team, the customer, and the management. Further, this leadership model is not constant and it allows a movement towards one pole or another, pending on the nature of specific scenarios emerged during the transition process.

b) *Ad-hoc Leadership – The Life Cycle.* The ad-hoc leadership *starts* with the CL's willingness to lead the change when the transition process to agile software development is initiated. It *continues* during the transition process, when the movement between the three poles reflects the different meaningful events that take place. As the frequency of the movements between the poles decreases, the need for this leadership style is diminishing, and may be replaced by other leadership styles.

c) We suggest that the *ad-hoc leadership* model may be used for the interpretation of the CL's position at different scenarios. Such interpretations, we suggest, may contribute to the analysis of transition and change processes.

6. REFERENCES

[1] Augustine, S., Payne, B., Sencindiver, F., and Woodcock, S. "Agile project management: steering from the edges", *Communication of AC,* 48(12), 2005, pp. 85-89.

[2] A. Begel. Effecting change: Coordination in large-scale software development. In Workshop on Cooperative and Human Aspects of Software Engineering, pages 17–20, Leipzig, Germany, May 2008. ACM.

[3] Bruce, A., and Langdon, K *Project management*, New York: Dorling Kindersley, 2000.

[4] Ciulla. J.B. *Ethics: The Heart of Leadership*, Praeger, 1998.

[5] Cockburn, A., and Highsmith, J. "Agile Software Development: The People Factor", *IEEE Software* 34:11, 2001, pp. 131-133.

[6] Cooper, C.L. *Leadership and Management in the 21st Century: Business Challenges of the Future*, Oxford University Press, NY, 2005.

[7] Drath, W.H. "Approaching the Future of Leadership Development", In: C. D. McCauley, R. S. Moxley, and E. Van Velsor (eds.), *The Center for Creative Leadership*, San Francisco, CA, Jossey-Bass, 1998, pp. 403–432.

[8] Dubinsky, Y., and Hazzan, O. "eXtreme Programming as a framework for student-project coaching in computer science capstone courses", *Proceedings of the IEEE International Conference on Software – Science, Technology & Engineering*, Herzelia, Israel, 2003, pp. 53-59

[9] Dubinsky, Y., and Hazzan, O. "Using a role scheme to derive software project quality", *Journal of System Architecture* 52(11) 2006, pp. 693-699.

[10] Dubinsky, Y., Talby, D., Hazzan, O. and Keren, A. "Agile metrics at the Israeli air force", Proceedings of the Agile 2005 Conference, IEEE computer society, Denver, Colorado, 2005, pp. 12-19.

[11] Goleman, D. "What makes a leader?", *Harvard Business Review*, 76(6) 93, 1998.

[12] Hazzan, O. and Dubinsky, Y. "Cognitive and social perspectives of software development methods: The case of Extreme Programming", *Proceedings of the 6th International Conf. on Extreme Programming and Agile Processes in Software Engineering*, 2005A, pp. 74-81.

[13] Hazzan, O. and Dubinsky, Y. "Clashes between culture and software development methods: The case of the Israeli hi-tech industry and Extreme Programming", Proceedings of the Agile Conference, IEEE computer society, Denver, Colorado, 2005B, pp. 59-69.

[14] Hazzan, O. and Dubinsky, Y. *Agile Software Engineering,* Undergraduate Topics in Computer Science Series, Springer-Verlag London Ltd, 2008.

[15] Hughes, B., and Cotterell, M. *Software Project Management*, 3rd edition, McGraw-Hill, 2002.

[16] Huff, A.S., and Moeslein, K. "An Agenda for Understanding Individual Leadership in Corporate Leadership Systems", In Cooper, CL. (Eds.) *Leadership and Management in the 21st Century: Business Challenges of the Future*,Oxford,248-270.

[17] Humphrey, W.S. *Introduction to the Team Software Process*, Addison-Wesley, 2000.

[18] Jones, J.A., Grechanik, M., van der Hoek, A. "Enabling and enhancing collaborations between software development organizations and independent test agencies," chase, pp.56-59, 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering, 2009.

[19] Mayrhauser, V.A. *Software Engineering Methods and Management,* Academic Press, 1990.

[20] Nirenberg, J. *Global Leadership*, Capstone Wiley, 2002.

[21] Plotnick, L., Ocker, R., Hiltz, S.R. & Rosson, M.B. Leadership in partially distributed emergency response software development teams. Proceedings of Information Systems for Crisis Response and Management: ISCRAM 2008.

[22] Pulford, K., Combelles, K.A., and Shirlaw, S. *A quantitative Approach to Software Management,* Addison-Wesley, 1996.

[23] Putnam, L.H., and Myers, W. *Industrial Strength Software – Effective Management Using Measurement*, IEEE Comp. Soc. Press, 1997.

[24] Topping, P.A. *Managerial Leadership*, McGraw-Hill, 2002.