OVERTIME EFFECTS ON PROJECT TEAM EFFECTIVENESS

Brandon Olson
Department of Computer Information Systems
The College of St. Scholastica
Duluth, Minnesota 55811
bolson1@css.edu

David Swenson
Department of Management
The College of St. Scholastica
Duluth, Minnesota 55811
dswenson@css.edu

Abstract

The fast-paced, competitive, and continually changing business environment is driven by information technology projects. The changing scope, requirements, and time pressure of projects requires overtime and long sprints. As commitments to an implementation date are established, the project team is left to absorb the additional time required to address project issues, new requirements and constraints. The extra load requires the project team to invest additional resource hours in order to achieve the target implementation date. In this paper, the impacts of overtime on the project team are considered as well as the implications on the project output. Recommendations are made on organizational improvements to structuring overtime more effectively, more effective team and personal coping mechanisms for dealing with overtime stress, in order to lessen the influence of overtime on the quality of the project deliverables.

1 Introduction

Software engineering projects are thought-intensive endeavors involving the careful coordination of highly-specialized information technology professionals. These efforts are carried out in a project environment that must balance the delivered functionality with the available project resource budget and project schedule. Additionally, as with every project environment, uncertainty is present and strongly influences the plans established by the project team.

The software development project team encounters risk issues such as inaccurate task estimates, inadequate risk management strategies, fluid requirements, increasing project scope, and poorly negotiated baseline variables. In this paper, these risk issues are viewed as environmental variables impacting the project schedule and the work patterns of the project team. The schedule impacts on the project team and project deliverables are identified and assessed in terms of software quality and the psychological impacts on the project team. The paper concludes with recommendations for mitigating the schedule impacts on the project deliverables and project team through changes to project team, project management, and individual behaviors.

2 Software Development Process

Organizations developing software often follow methodologies that align with the systems development life cycle (SDLC). This life cycle consists of the series of phases across software development projects. Hoffer, George, and Valacich (2011) offered a common SDLC consisting of planning, analysis, design, implementation, and maintenance phases. Each of these phases represents a unique set of activities and stakeholder involvement.

Early software development methodologies applied the SDLC to a waterfall approach where, as described by Whitten and Bentley (2007) each phase is initiated upon the completion of the prior phase (see Figure 1). Using this approach the software design was initiated once the entire system was defined and development of the software was initiated only after the design was completed. Finally, the entire system was delivered as a single-phased project. This approach encouraged early definition and design to ensure a stable set of requirements before programming started.

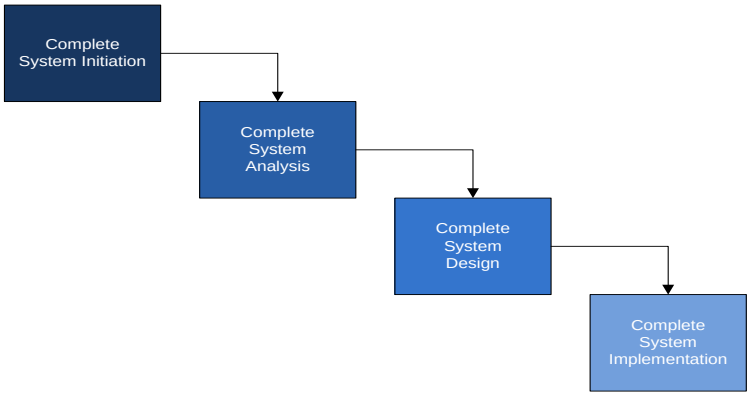

Figure 1. Waterfall approach to SDLC

As the software engineering field matured and software was moved to smaller and less critical applications, issues with the waterfall approach began to appear. These issues included inflexibility in accommodating changing requirement (Brewer & Dittman, 2010); challenges in scaling down for smaller projects (Boehm & Turner, 2004); more overhead due to a more formalized process, difficulty in effective risk management in later phases, and longer period of time before any software is produced (Munassar & Govardhan, 2010). In order to address these issues, a more iterative approach can be applied. This iterative approach, described as a phased approach by Dennis, Wixom, and Tegarden (2005), as illustrated in Figure 2, executed the SDLC multiple times throughout the project. Munassar and Govardhan (2010) noted that this more iterative approach supports a more rapid delivery of critical functions, better resource cohesion, and higher thresholds for changing requirements or evolving requirements.

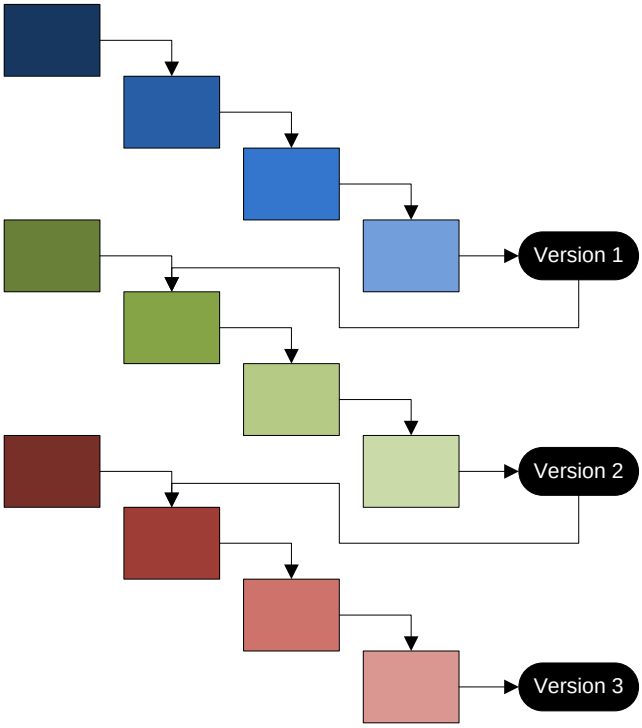

Figure 2. Iterative approach to SDLC

Unlike the waterfall approach to software development, the iterative model requires the system to be designed across multiple iterations where individual functional modules or portions of modules are designed as part of an iteration and the modules are slowly integrated to form the complete system. As a result, the system design is carried out over a longer period of time than the waterfall method and requires integration of the designed components as the system evolves and requirements emerge.

3 Time Pressure

There is an inherent uncertainty with software development projects. Uncertainty over new technologies, new business processes, new project members, and the many other environmental

factors impact the project and the project team. Through scope creep and fluid requirements, these uncertainties create opportunities for deviations in the quality or availability of project deliverables, project budget, or project schedule. While concerns over quality, functionality, and budget are important to the project team, the focus of this paper is on the schedule concerns; particularly where uncertainties create delays in the project schedule.

It is common for IT projects to conclude with a final push of project deliverables in a short period of time in order to meet the project milestone date. This period before the significant milestone date creates additional pressure for the project team. The team must continue to produce high quality project deliverables while the milestone date becomes more apparent and the project team attempts to complete an increased volume of activities before the date arrives. This final push creates additional pressure on the project team through higher workloads and anxiety over errors and a missed milestone date.

3.1 Sources of Pressure

There are several factors contributing to the risk of information technology projects. These risk factors lead to delays in producing software and result in a higher volume of project deliverables required to complete at the end of a project period. Some of the risk factors identified by Schmidt, Lyytinen, Keil, and Cule (2001) include poor estimating or simulation of task durations, immature risk management practices, fluidity of system requirements, scope creep and change management deficiencies, and a premature commitment to project variables.

3.1.1 Poor Resource Estimates

During project planning, the project team carries out estimation activities to determine the approximate duration required for each project task. The task durations may be derived through estimates from the project team or through a type of simulation model. Estimates from the project team rely on the experiences of project team members and their ability to apply their experience to the context of the current project while factoring in any observed differences between prior experience and the current project. Task durations obtained from simulation models rely on the proper identification and application of simulation variables and for the correct fit of the model to the project. Regardless of the source for the task durations, assumptions are made and the estimates are simply that - estimates. The actual time duration is known only after the task has been completed. Many factors influence the execution of the task so an absolute estimate cannot be determined and the project team must rely on best guesses.

3.1.2 Immature Risk Management

Beyond the uncertainty of resource estimates and task durations, many other uncertainties influence the project schedule. The project team must be able to address these uncertainties to lessen the impact to the project schedule. Proactive planning and response to the project uncertainty is referred to as risk management. The Project Management Institute (2008) includes

risk identification, qualitative and qualitative risk analysis, risk response planning, and risk monitoring and controlling as the main processes in mature risk management practices. Project teams with inadequate risk practices are not prepared to identify and address risk events as they occur or anticipate risks. The missed risk mitigation opportunities may result in unexpected work for the project team and extend the project schedule or, in the event of a fixed schedule project, increase the workload of the project team. In the event of a fixed-schedule project, this additional workload creates further pressure on the team. More mature risk management and practices are needed to ensure risks events such as scope creep and fluid requirements do not impact the project schedule or create unanticipated workload for the project team.

Another common issue creating pressure for the project team is the existence of schedule constraints associated with an early commitment to a project timelines. Whitten and Bentley (2007) identified this premature commitment to project variables, such as the project schedule, as a common source for software development project failures. Additionally, Cerpa and Verner (2009) found the schedule impact on the development process as the primary reason for project failure and a factor in 93% of project failures. If the project team commits to a project schedule before the project is sufficiently defined and proper estimates are obtained, the schedule may not reflect the actual time needed to complete the project. In such events, the project team is required to increase the workload in order to deliver all of the functionality within the time constraints. This premature commitment to the project schedule results in increased pressure to meet timelines that are not reflective of the scheduled work. This may result in additional overtime or shortcuts to the software development methodology.

3.2 Impact of Pressure

Information systems evolved from simple standalone and encapsulated systems into complex, highly distributed environments. As a result of the evolution, information systems are more flexible and open, but this progress comes at the cost of the increased complexity. This complexity forces the system engineers to design systems containing a multitude of layers. The level of effort required to create a cohesive design has greatly increased with this increase in complexity. While the individual components may not, individually, be difficult to design and construct, the multiple layers of the modern environment increases the difficulty of the system (Schneberger, 1997). As a result, the overall complexity of these systems increases the difficulty of formulating a cohesive design across application layers.

3.2.1 System Design

Furuyama (1994) conducted a study to measure the effects of stress on software faults. The author found that 75% of the faults in software development projects were generated during the design phase of the project and 71% of all faults were caused by long-term stress factors such as short development time and human nature. Human nature was attributed to situations where engineers skipped a part of the methodology as a shortcut and resulted in a fault in the design or code.

The Furuyama (1994) study showed that software faults occur in the phase where a significant level of conceptualization and deeper thinking is required. When an engineer is under pressure to produce results faster, and with incomplete or changing requirements, a higher rate of design faults are produced. The study also suggested that development of the software code was not affected by pressure to the same degree as in the design phase. The stress placed on the developers was less than the designers due to the less thought-intensive task of creating the code required to satisfy the design specifications. While the total number of faults associated with the implementation activities was lower than the design phase, the study noted that stress and human nature contributed to 73% of the implementation faults. This result is similar to the percentage of faults found in the design phase. It is clear that most faults are linked to stress and human nature in both the design and implementation phase and these faults can be associated with the time and workload pressures placed on the project team.

The Furuyama (1994) study demonstrated that faults play a significant role in the quantity of faults found in software. The shortening of design and implementation timelines lead to stress on the project team and also shortcuts to the process which results in further propagate the increase in software faults. As software becomes more complex and the need for faster development cycles increases, more software faults can be expected due to the deeper thinking required and added pressure placed on the software engineers. These faults, and their corresponding resolution, not only cause delays to delivery of the software, they also increase the costs of software.

3.2.2 Iterative Development

Compounding the issue of errors in software design is the placement of the design phase in the current iterative model to software development. In the waterfall approach (see Figure 1), system design was conducted earlier in the development process. As a result, the time pressure occurring late in the project had little or no impact on the system design activities. However, moving to a more iterative approach (see Figure 2), the design phase is carried out in multiple iterations resulting in design activities take place closer to the implementation date. Additionally, the iterative nature of the software development process requires integration of the designed components thus adding integration to the level of complexity for the designer and increasing the risk of downstream errors in not only the individual software components but the integration of these components.

3.2.3 Software Costs

Developing software is an expensive endeavor. As information systems become more complex and integrated, the time required to design and develop these systems increase. As a result, we should expect to find the costs associated with the development of these systems to increase. Additionally, the technical specialists required to develop these systems come at a premium price. As demonstrated by the pursuit of offshore development (Apte et al., 1997), the high cost of the technical specialists is contributing significant factor in the costs associated with information technology projects.

The amount of faults contained in the software directly relates to the costs of the project. As faults are introduced into the software, rework is required to redesign, redevelop, and redeploy modifications to the software. Unfortunately, these rework activities must be performed by the more expensive resources, therefore, dramatically increasing the costs (Westland, 2004). The costs associated with software faults are further exacerbated by the process required for software development. If a fault is detected once the software has been completed, the entire set of analysis, design, development and deployment activities must be performed. If faults are detected earlier in the software development process, fewer steps are required to be repeated. As a result, the costs associated with the fault can be dramatically reduced through early detection.

In a 1976 study, Boehm demonstrated the cost savings of early detection of software faults. This results indicated an exponential relationship between the cost of the software fault and the phase where the fault was detected. Faults detected later in the development process are much more costly to resolve than those detected earlier in the process (see Figure 3). This significant increase in costs is attributable to the resources required to complete an additional iteration of the development processes for the fault. This additional rework increases the costs of the project and lead to schedule delays and increased workload.

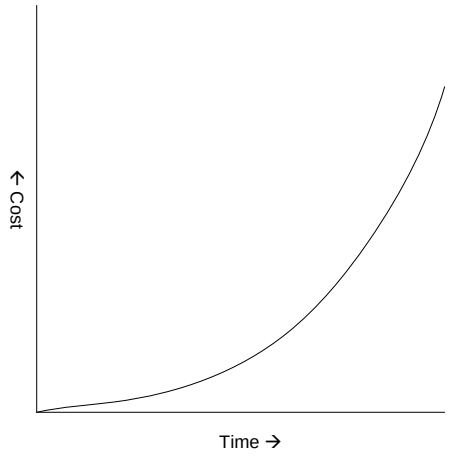

Figure 3: Relationship between fault detention and cost

3.3 Implications of Project Stress on the Way People Work

3.3.0 Programmer stress

Stress has been shown to be higher among software engineers than other professions (Kumashiro, Kamada, & Miyaki, 1989), and is ubiquitous throughout the phases of the software development lifecycle (Glass, 1997). During the requirements phase stress tends to produce anxiety and depression related to not having clear and consistent definitions and outcomes, and inaccuracy in anticipating needs of end-users. During the design phase, stress-related errors are even more prominent, with up to 42% of design flaws due to programmer stress. This is attributed to the level of deep reflection and thought required during this stage that is vulnerable to stress. During coding, irritability and declining morale emerge as programmers try to integrate

the minor details hidden under higher level designs. Finally, during testing, all forms of stress occur as programmers rush to integrate the previous phases into a final product, and overtime is more likely (Fujigaki, 1993; Furuyama, Arai, & Lio, 1993).

3.3.1 Overtime

Project stresses often lead project managers to rely on overtime as a tactic to meet project mileposts and deadlines. In some areas of software development such as the gaming industry, *crunch periods* of overtime were reported as common by as many as 60% of programmers. In addition, 47% said they were not compensated and only 3% indicated that the overtime was even counted (Not all fun and games, 2005). Overtime seems a logical way to meet the demands, and it seems to be a widely held expectation about the nature of work. For example, Swanson (2011) described a software development programmer who was both a project manager and team member. He intentionally worked 16-hour days in order to impress management by delivering the project earlier, as well as believed that he was a good role model and was providing other team members with the opportunity to gain overtime income. He also considered overtime as a way to identify programmers who excel under stress so he could reward them with bonuses to maintain such behavior.

3.3.2 Disruption of work/home life balance

Spillover theory (Zedeck, 1992) proposes that work and family microsystems significantly influence each other. Working excessive or mandatory overtime is not only disruptive for the programmer but also for the programmer's family and social life (Hill, Hawkins, Ferris, & Weitzman, 2001). When the transitions between work and home become disrupted, people tend to overlap work and home functions that lead to more stressful and less efficient multitasking (Floro & Miles, 2003). Such schedules can adversely affect couple's time together, create role conflicts, work overload, and unequal division of labor for household responsibilities and childcare (Crouter, Bumpus, Maguire, & McHale, 1999).

Even when the couple is together, there may be so much fatigue, low mood, or physical complaints that sleep and rest replace social and recreational activities together. In addition, the frustration from project pressure or irritability from insufficient sleep can create sensitivity such that even little issues lead to conflict that undermines the spousal support usually available (Golden & Wiens-Tuers, 2008). Single persons also find less time for creating or developing relationships.

4 Research on Overtime

4.1 Adverse effects of overtime

Scheduled overtime tends to produce progressive deterioration in performance. In one study (National Electrical Contractor Association, 1989) after six nine hour days performance slipped to 88%, seven eight hour days reduced it to 75%, and seven 12-hour days to 47%. When overtime is as high as 60 hours a week, performance may decrease as much as 20% (Nevison, 1992). Sustained overtime can produce fatigue that increases errors and reduces productivity (Lyneis & Ford, 2007; Reichelt & Lyneis, 1998). Projects that have late stage delays often shift into what have been called *death marches* in which programmers spend long overtime hours. It is during these extended periods that errors are most common, and error removal is one of the most time-consuming phases of the product development lifecycle, thereby adding further delays.

4.1.1 Cognitive effects

Unfortunately, people are not good judges of their degree of fatigue or sleepiness, usually rating themselves as more alert than they are (Sasaki, Kurosaki, Mori & Edo, 1986), and the correlation between subjective report and physiological measures is only .18 (Johnson et al., 1988). Mental workload caused by time pressure can have immediate as well as delayed impact on the project. Initially it can be stimulating and challenging to which programmers respond with interest and high motivation. However, as overtime continues, mental fatigue likewise increases, and gradually impairs work engagement, quality and productivity (Putkonen, 2009; Weinberg 1971).

Software programming is primarily a cognitive activity, and cognition is one of the first lines of impact of overtime fatigue. Alertness, attention, and concentration are among the first abilities to suffer (Folkard, Monk & Lobban, 1978). The average adult can demonstrate sustained attention to a routine or mundane task for about 20 minutes, although interest and motivation can maintain it for much longer. However as the task progresses, attention declines and is particularly affected by fatigue. Such deficits can result in overlooking errors, missing instructions, and more easily losing one's place on lines of code. At this point, efforts for innovation are gone, and programmers are simply trying to complete a task.

Under fatigue, memory and recall tend to be slower (Balkin & Badia, 1988), short term memory is worse (Rubin, Orris, Lau, et al. 1991), and even after a full day of recovery following an all-night sprint, memory may still show impairments (Meijman, van der Meer & van Dormolen, 1993). Logical and arithmetic reasoning have more errors, and decision making is slower (Balkin & Badia, 1988). These kinds of impairments, however slight, may interfere with a programmer's recall of what has been done, logic to be used, and whether and how to approach a particular coding problem.

4.1.2 Mood and interpersonal effects

During final phases of software development, overtime is more likely used to meet deadlines. This quantitative and qualitative overload produces anxiety, depression and low self-esteem (Cooper, Dewe, & O'Driscoll, 2001). As fatigue increases, and particularly when sleep quality is further disturbed, people tend to become more irritable, impatient, and easily frustrated (Pilcher & Huffcutt, 1996). Such sensitivity can decrease tolerance of ambiguity and complexity on a task, as well as interfere with working relationships as a team.

Conflict has been reported by some as pervasive in the information system development process. Conflict can include professional differences of opinion, incompatibility of personal work styles, and conflicting goals. These may occur within the team, or between the team and other stakeholders. Conflict disrupts the orderly process of problem solving, the product is not as high quality as planned, and the project is more likely to fail meeting budgetary goals, schedule and scope (Liu, Chen, Klein, & Jiang, 2009). Conflicts can distract team members from the task, redirect attention to personal issues, and even cause side-taking of team members over disagreements. Depression has also been identified as one of the risks of subjective time pressure related to overtime (Roxburgh, 2004), and such low mood can reduce energy and attention or increase irritability. Extensive overtime can also contribute to poor morale that subsequently affects absenteeism and team spirit (Kerin, 2003; Shepard & Clifton, 2000).

4.1.3 Health effects

Short term stress does not particularly have an adverse effect on health, but when it is recurrent, prolonged, or very intense it can affect physiological processes that may persist (Guerts & Sonnentag, 2006). Briefer periods of stress temporarily increase muscle tension, blood pressure, increased heart rate, and blood glucose levels for available energy. However, prolonged muscle tension can result in distracting postural aches and pains and muscle tension headaches. Frequently elevated blood pressure due to stress can remain high, and combined with sedentary lifestyle, poor diet, and increased heart rate can portend heart disease. During stress the digestive system tends to slow down, resulting in lower appetite and constipation, exacerbated by smoking and generally poor diet of fast food and energy drinks (Van der Hulst, 2003).

The combination of persistent cognitive stimulation as a project overtime continues over days or weeks, poor diet and exercise, digestive upset, and emotional sensitivity also interferes with sleep quality. Increasing sleep deficit then completes the feedback loop for further fatigue and its consequences. Meijman and Mulder's (1998) Effort-Recovery Theory proposes that overtime work involves a prolonged effort expenditure that results in physiological stress (e.g., increased heart rate, blood pressure, fatigue, etc.) and reduced recovery time. While recovery would typically occur during off-work hours, the extended hours prevent refreshment, thereby resulting in continued fatigue. Under these circumstances, what were short term physiological responses to stress can become exacerbated and become sleep disorders, chronic hypertension, and irregular heart rate. Even when the physical demands are over for the day, cognitive processes may continue as the programmer continues to think about the coding, have intrusive thoughts or flashes of insight on coding problems, reflect on the difficult day or anxiously anticipate the next

(Brosschot, Pieper, & Thayer, 2005). Only when sufficient recovery can occur are demands of the job related to positive feelings of work engagement (Demerouti, Bakker, Geurts, & Taris)

4.2 Effects on team performance

The early experiences in team building and development often set the foundation for later team interaction as well as influence team effectiveness (Ericksen & Dyer, 2004). When new teams are challenged by time pressures or organizational culture to rely on overtime, it becomes an expectation, coping strategy, and a team norm.

Fatigue resulting from sleep deprivation has been linked obviously to people going without sleep for days at a time, but also to those who have sleep restriction for as little as six hours a night over two weeks (Van Dongen, Maislin, Mullington, & Dinges, 2003). This latter duration is not uncommon for many people working in many industries. Although the fatigue that accompanies frequent or prolonged overtime and subsequent interference with normal sleep patterns has been extensively researched on individuals, but correspondingly little on teams (Harrison & Horne, 2000). Recent research by Barnes and Hollenbeck (2009) summarizes much of what is known about team fatigue. They defined sleep-deprived teams as those consisting of less than 50% rested members.

Akula and Cusick (2008) examined the effects of overtime and stress on software quality. They identified six ways in which fatigue-effected defects on quality can have a dampening effect:

- Poor quality can redirect teams away from their core project activities.
- Quality assurance and development teams may mistrust each other.
- Discussions of defects can add to the task pressure and personal stress.
- Worst case outcome can result in critical delay of the project.
- Blaming others over the defect undermines team spirit.
- Teams may spend more time repairing defects than on developing business critical features.

5 Recommendations

5.1 Organization level

By its very nature, knowledge work is ambiguous and complex, that makes it difficult to plan decisively. Although requirements are usually stated at the outset, it is common for new requirements and other changes to emerge, making the planning quite dynamic. As described above, the organization can more realistically adapt to these ongoing changes by pursuing an agile approach to software development, and making adjustments in scheduling as needed. Improved scheduling and better realization of the organization’s project capacity can be established through the proper implementation of project portfolio management (2008). Effective management of the project portfolio helps the organization to avoid scheduling too many overlapping projects at one time such that a bind on one will affect performance on the

others. Nonetheless, it is a given that the best plans will often show flaws in later development (Moran, 2010).

Runaway projects can also be reduced by realizing that they are based on poor estimations and communication. Glass (2003) suggested that much software estimation occurs at the wrong time and is often done by the wrong people. He further considered that managers and programmers do not communicate as clearly or often as they need to. This may be partly due to using a manufacturing model for software development that emphasizes management by schedule, rather than one based on learning and communication (Fujigaki, 1993). It also highlights the need to involve both managers and programmers who have stronger interpersonal needs and skills, so they can communicate what they need and provide more effective support to each other.

The Human Resource Department of the organization is in an important role to provide policy and services. Family-friendly services, such as childcare assistance (Berg, Kalleberg, & Appelbaum, 2003), sick child care, school tutorials for youth, wellness massages, and onsite fitness center (Berg, Kalleberg, & Appelbaum, 2003) tend to promote more employee committment to the job and reduce turnover intention (Grover & Crooker, 1995). HR can also support practices and policies that involve employees in schedule planning and decisions, monitor frequency and duration of overtime to keep it from being chronic, and allow flextime.

5.2 Team Level

As stressful as overtime fatigue can be there are also several moderators of its impact. Such task structure characteristics as variety, task demands, and sense of control or autonomy can reduce the immediate effects. Research by Van der Hulst, Van Veldhoven and Beckers (2004) and Beckers, Van der Linden, Smulders, et al., (2004) both concluded that high task pressure combined with low responsibility for decision making, low autonomy and low task variety led to higher fatigue than when these variables were high (Beckers et al., 2008). This suggests that self-directed teams may be more resilient when they can shift or rotate task responsibilities from time to time, use individual discretion in some levels of decision making, and have some protection from continuous high task demands.

Beckers et al., (2008) has also identified control over overtime as an important moderator. Worktime control is related to the balance between work and homelife balance as well as health, and to the adverse effects of fatigue such as work dissatisfaction, absenteeism, and job burnout (Fenwick & Tausig, 2001). These results suggest that it is important for team members to exercise as much control as feasible over the duration and distribution of worktime.

Teams may also be able to compensate for fatigue errors by using methods such as pair programming. For example, in XP programing two people sit at one computer with one reviewing the other coder's work for potential errors. As Cockburn and Williams (2001) pointed out, while this reduces individual productivity, it catches errors that can become even more costly and time-consuming at later stages. This is also similar to Microsoft's use of a *tester buddies* who keep writing test cases for the coding (Cusumano, 2007).

Communication with and support from others is another essential component of stress management for software developers. While many programmers tend to have an introverted style that may serve their intense focus on programming, the need for them to assertively communicate with each other, project managers, and end users is increasingly emphasized in businesses (Glen, 2003; Humphrey, 1997). Project managers often assume that if they are not hearing from programmers that everything must be on schedule. At the same time, programmers may be reluctant to admit they are behind, or simply do not have a clear idea of the time schedule since it keeps changing. It is advantageous for programmers to keep project managers informed of potential delays and changes so the manager's can make alternate changes in scheduling, remove barriers, and provide resources.

5.3 Personal Level

Even with high interest and novelty of a task, it is physiologically challenging to maintain continuous vigilant performance for periods greater than 30 minutes. For programmers working over and beyond the regular schedule can increase fatigue and consequent errors (Circadian.com, 2011). Although vacations and time off following intense projects can aid recovery to some degree, prevacation levels of stress tend to recur rapidly after return to work (Westman & Etzion, 2001). Consequently, the ability to recover daily and on weekends is more important for aiding recovery, health and performance. Whether overtime is voluntary or involuntary, the sense of control programmers feel over their time and workload can affect fatigue and work satisfaction (Beckers, et al. 2008).

Although the organization and team can do much to lessen the pressure, it is up to individuals to maintain a healthy and resilient lifestyle outside of work hours. Fatigue is the primary stressor and it can be reduced by making as much time for sleep as possible during the days or weeks of an intense work sprint. Going to bed at the same time each night enables the circadian rhythm that regulates wakefulness and sleep to become consistent. Finding relaxing rituals before bed, such as quiet reading or music, warm bath, meditation, or other calming activity can facilitate sleep. During the day, taking a 10-second break for attention or simply getting up and stretching every 10 minutes, and taking a 5-minute break and walk around every hour can refresh attention (Cohen, 1997).

Various forms of exercise are highly recommended to maintain alertness and vitality during stressful periods. Taking brief walks as well as strenuous activity can improve mental alertness, self confidence, and sense of self reliance. In addition, stretching, yoga, and t'ai chi can have a calming influence and help release tension (Mack, 2010; Ross & Thomas, 2010).

Nutrition is also related to moderating stress levels, especially fatigue. Commonly espoused guidelines are recommended to reduce stress effects: limit caffeinated drinks such as coffee and energy drinks, as well as alcohol, both of which can interfere with restful sleep. Snack on healthy foods such as whole grains and vegetables; sugared snacks will provide rapid energy often followed by an energy crash. Small and frequent meals help maintain energy, while skipping meals can exacerbate anxiety and fatigue. Take at least 20 minutes for relaxed eating to get as much nutrition out of the food, foster digestion, and relax (Neville, 2006).

Conclusion

The challenge to an organization of how to maintain competitive positioning in a changing marketplace sometimes leads them to pursue more projects than they should, or inaccurately estimate the requirements, especially for scheduling. Such practices lead to creating a culture in which functional managers, project managers and the project team come to expect poorly specified or frequently changing requirements and timelines. This contributes to delays in progress so that there is excessive reliance on overtime. While overtime may not be a problem if it occurs infrequently, it can be a serious problem when it becomes the mode of operating.

The resulting stress and fatigue that overtime produces can affect cognitive performance such as attention, concentration, memory, and logic errors. The emotional consequences can leave the project team frustrated, anxious, depressed and irritable, that may complicate team interaction. If prolonged, this stress can produce physical discomforts and even illness. Organizations can prevent many of these adverse effects by monitoring realistic project load and scheduling through proper project portfolio management, offering family-friendly services, and encouraging more communication among all stakeholders. Teams can also improve their resiliency by keeping project managers informed earlier of potential delays and changes, thereby enabling project managers to remove barriers. Finally, the project team needs to accept personal responsibility for their own lifestyle habits, and during stressful periods, find ways to practice healthy sleep and rest, relaxation, exercise, and diet.

References

Apte, U. M., Sobol, M. G., Hanaoka, S., Shimada, T., Saarinen, T., Salmela, T., & Vepsalainen, A. J. (1997). IS outsourcing practices in the USA, Japan and Finland: a comparative study. *Journal of Information Technology, 12*(4), 289-304.

Akula, B., & Cusick, J. (2008). *Impact of overtime and stress on software quality*. 4th International Symposium on Management, Engineering, and Informatics: MEI 2008. Orlando, FL, June 29-July 2. http://www.jamescusick.net/pubs/Cusick_MEI08_StressQuality.pdf

Barnes, C. M., & Hollenbeck, J. R. (2009). Sleep deprivation and decision-making teams: Burning the midnight oil or playing with fire? *Academy of Management Review, 34*(1), 56-66.

Beckers, D. G. J., & Van der Linden, D., Smulders, P. G. W., Kompier, M. A. J., Taris, T. W., & Geurts, S. A. E. (2008). Voluntary and involuntary? Control over overtime and rewards for overtime in relation to fatigue and work satisfaction. *Work & Stress, 22*(1), 33-50.

Beckers, D. G. J., Van der Linden, D., Smulders, P. G. W., Kompier, M. A. J., Van Veldhoven, M. J. P. M., & Van Yperen, N. W. (2004). Working overtime hours: relations with fatigue, work motivation, and the quality of work. *Journal of Occupational and Environmental Medicine, 46*, 1282-1289.

Berg, P., Kalleberg, A. L., & Appelbaum, E. (2003). Balancing work and family: The role of high-committment environments. *Industrial Relations, 42*(2), 168-188.

Boehm, B. W. (1976). Software engineering. *IEEE Transactions on Computers, 25*, 1226-1241.

Boehm, B., & Turner, R. (2004). *Balancing agility and discipline: A guide for the perplexed*. Boston: Addison-Wesley.

Brewer, J. L., & Dittman, K. C. (2010). *Methods of IT project management*. Boston, MA: Prentice Hall.

Brosschot J. F., Pieper S., & Thayer J. F. (2005). Expanding stress theory: Prolonged activation and perseverative cognition. Psychoneuroendocrinology, *30*, 1043–1049.

Brown, J. T. (2008). *The handbook of program management: How to faciliate project success with optimal program management*. New York, NY: McGraw Hill.

Cerpa, N., & Verner, J. M. (2009). Why did your project fail?. *Communications of the ACM, 52*(12), 130-134.

Circadian.com (2011). Corporate shiftwork strategy. http://www.circadian.com/pages/414_ corporate_shiftwork_strategy.cfm?searchterm=overtime

Cockburn, A., & Williams, L. (2001). The Costs and Benefits of Pair Programming. In G. Succi & M. Marchesi (Eds.), *Extreme programming examined*, Boston, MA: Addison-Wesley Longman, pp. 223-243.

Cohen, S. (1997). De-stress for success. *Training and Development, 51*(11), 76-80.

Cooper, C. L., Dewe, P. J., & O'Driscoll, M. P. (2001). *Organizational stress*. Thousand Oaks, CA: Sage.

Crouter, A. C., Bumpus, M. F., Maguire, M. C., & McHale, S. M. (1999). Linking parents' work pressure and adolescents' wellbeing: Insights into dynamics in dualearner families. *Developmental Psychology, 35*(6), 1453-1461.

Cusumano, M. (2007). Extreme programming compared with Microsoft-style iterative development. *Communications of the ACM, 50*(10), 15-18.

Dawson, T., Heitman, A., & Kerin, A. (2004). Industry trends, costs and management of long working hours. NIOSH Safety and Health Topic: Work schedules: Shift work and long work hours. http://www.cdc.gov/niosh/topics/workschedules/abstracts/dawson.html.

Demerouti, E., Bakker, A. B., Geurts, S. A. E., & Taris, T. W. Daily recovery from work-related effort during non-work time (2009). In S. Sonnetag, P. L. Perrewe, & D. C. Ganster (Eds.,), Current perspectives on job-stress recovery, pp. 85-123.

Dennis, A., Wixom, B. H., & Tegarden, D. P. (2005). *Systems analysis and design with UML version 2.0: An object-oriented approach*. Hoboken, NJ: J. Wiley.

Erickson, J., & Dyer, L. (2004). Right from the start: Exploring the effects of early team events on subsequent project team development and performance. *Administrative Science Quarterly, 49*, 438-471.

Fenwick, R., & Tausig, M. (2001). Scheduling stress. Family and health outcomes of shift work and schedule control. *American Behavioral Scientist, 44*, 1179-1198.

Floro, M. S., & Miles, M. (2003). Time use, work and overlapping activities: Evidence from Australia. *Cambridge Journal of Economics, 27*(6), 881-904.

Fujigaki, Y. (1993). Stress analysis: A new perspective on peopleware. *American Programmer, 6*(7), 33-38.

Furuyama, T., Arai, Y., & Lio, K. (1993). Fault generation model and mental stress effect analysis. In Proceedings of the Second International Conference on Achieving Quality in Software, Venice, Italy, October 18-20.

Geurts, S. A. E., & Sonnentag, S. (2006). Recovery as an explanatory mechanism in the relation between acute stress reactions and chronic health impairment. *Scandinavian Journal of Work, Environment & Health, 32*, 482-492.

Glass, R. L. (2003). *Facts and fallacies of software engineering.* Boston, MA: Pearson.

Glass, R. L. (1997). The ups and downs of programmer stress. *Communication of the ACM, 40*(4), 17-19.

Glen, P. (2003). *Leading geeks: How to manage and lead people who deliver technology*. San Francisco, CA: Jossey Bass.

Golden, L., & Wiens-Tuers, B. (2008). Overtime work and wellbeing at home. *Review of Social Economy, 66*(1), 25-49.

Grover, S, L., & Crooker, K. J. (1995). Who appreciates family-responsive human resource policies: The impact of family-friendly policies on the organizational attachment of parents and non-parents. *Personnel Psychology 48*, 271-88.

Harrison, Y., & Horne, J. A. (2000). The impact of sleep deprivation on decision making: A review. *Journal of Experimental Psychology—Applied, 6*, 236 –249.

Hill, E. J., Hawkins, A. J., Ferris, M., & Weitzman, M. (2001). Finding an extra day a week: The positive influence of perceived job flexibility on work and family life balance. *Family Relations, 50*(1), 49-54.

Hoffer, J. A., George, J. F., & Valacich, J. S. (2011). *Modern systems analysis and design* (6th ed.). Upper Saddle River, NJ: Pearson Prentice Hall.

Howick, S., Ackerman, F., Eden,C., & Williams, T. (2009). Understanding the causes and consequences of disruption and delay in complex projects: How system dynamics can help. In Meyers, RA (Eds.), *Encyclopedia of Complexity and System Science*. Springer Verlag.

Humphrey, W. S. (1997). *Managing technical people: Innovation, teamwork, and the software process*. Reading, MA: Addison Wesley.

Kerin, A. (2004). Shiftwork practices 2004. Lexington, MA: Circadian Technologies.
Kumashiro, M., Kamada, R., & Miyaki, S. (1989). Mental stress with new technology at the workplace. In M. J. Smith & G. Salvendy (Eds.), *Work with computers: Organizational, management, stress and health aspects*, pp. 270-277, Amsterdam: Elsevier.
Liu, J. Y. C., Chen, J. V., Klein, G., & Jiang, J. J. (2009). The negative impact of conflict on the information system development process, product, and project. *Journal of Computer Information Systems, 49*(4), 98-104
Mack, J. B. (2010). Stress: Part 2-- Physical and psychological benefits of exercise. *American Fitness, 28*(6), 59.
Meijman, T. F., & Mulder, G. (1998). Psychological aspects of workload. In P. J. D. Drenth, H. Thierry, & C. J. de Wolff (Eds.), *Handbook of work and organizational psychology* (2nd ed., pp. 5-33). Hove: Psychology Press.
Moran, D. (2010). The challenges of managing knowledge workers. *Supervision, 71*(5), 18-21.
Munassar, N., & Govardhan, A. A. (2010). A comparison between five models of software engineering. *International Journal of Computer Science Issues, 7*(5), 94.
Neville, K. (November, 2006). Nutritional boost for medical stress. *Environmental Nutrition, 29*(11), 1/6.
Not all fun and games (2005). News Track, *Communications of the ACM, 48*(2), 9.
Pilcher, J. J., & Huffcutt, A. I. 1996. Effects of sleep deprivation on performance: A meta-analysis. *Sleep, 19,* 318 –326.
Project Management Institute. (2008). A guide to the project management body of knowledge (4th ed.). Newtown Square, PA: Author.
Putkonen, A. (2009). Predicting the effects of time pressure on design work. *International Journal of Innovation and Learning. 6*(5), 477-492.
Ross, A., & Thomas, S. (2010). Health benefits of yoga and exercise: A review of comparison studies. *Journal of Alternative and Complementary Medicine, 16*(1), 3-12.
Roxburgh, S. (2004). There just aren't enough hours in the day: The mental health consequences of time pressure, *Journal of Health and Social Behavior 45*(2), 115-31.
Schmidt, R., Lyytinen, K., Keil, M., & Cule, P. (2001). Identifying software project risks: An international delphi study. *Journal of Management Information Systems,* 17(4), 5-36.
Schneberger, S. L. (1997). Distributed computing environments: Effects on software maintenance difficulty. *Journal of Systems and Software, 37*(2), 101-116.
Swanson, S. (2011). The midnight oil: Overtime and the problem it causes with software development. Careworks Technologies, News. http://www.careworkstech.com/news/the-midnight-oil.
Toole, T. M. (2005). A project management causal loop diagram. Presented at 2005 ARCOM Conference, London, UK. September 5-7.
Van der Hulst, M. (2003). Long work hours and health [review]. *Scandinavian Journal of Work, Environment & Health, 29*, 171-188
Van der Hulst, M., Van Veldhoven, M., & Beckers, D. (2006). Overtime and need for recovery in relation to job demands and job control. *Journal of Occupational Health, 48*, 11-19.
Van Dongen, H. P. A., Maislin, G., Mullington, J. M., & Dinges, D. F. (2003). The cumulative cost of additional wakefulness: Dose-response effects on neurobehavioral functions and sleep physiology from chronic sleep restriction and total sleep deprivation. *Sleep, 26*, 117–126.

Westman, M., & Etzion, D. (2001). The impact of vacation and job stress on burnout and absenteeism. *Psychology and Health (special issue on burnout), 16*(5), 595-606.
Whitten, J. L., & Bentley, L. D. (2007). *Systems analysis and design methods* (7th ed.). Boston, MA: McGraw-Hill/Irwin.
Zedeck (1992). *Work, families, and organizations.* San Francisco: Jossey-Bass.