26th Annual INCOSE International Symposium (IS 2016)
Edinburgh, Scotland, UK, July 18-21, 2016

# Merit of Adaptable Pairing as an Agile Systems Engineering Knowledge Management Practice

Chris Leroux
Firefly Space Systems
cleroux.tx@gmail.com

Rick Dove
Stevens Institute of Technology
rdove@stevens.edu

**Abstract.** Driven by customer demands and global competition, systems engineers often experience dramatic project shifts and rapid technological advances that can place great strains on an engineering organization. In addition, project teams also deal with changing personnel, mistakes or rework, and slow engineering response times to unplanned needs. These external and internal factors can lead to reduced quality, increased costs, and slipped schedules. An effective knowledge management architecture is a necessity in today's engineering environment, and should be treated as an important project activity. Due to constantly evolving projects and project teams, knowledge management must be able to adapt based on current needs and resources. The purpose of this paper is to present knowledge management issues that must be addressed during an engineering project, and to propose a useful operational architecture for knowledge management. This architecture will provide an image of how agile knowledge management can be used by systems engineers to counter the turbulence in today's technical work forces, and how it can be integrated into existing organizational structures. The proposed architecture is best described as *adaptable pairing*, where two or more individuals engage in knowledge sharing and assignment swapping to complete a specific project task. The pairing architecture will address three common knowledge management problems that may negatively impact project quality, cost, and schedule - uninformed decision making, lengthy response delays, and rework. Intents and traits of *adaptable pairing* may be appropriate for any industry, however, evidence will focus primarily on technology industries engaged in contract projects. Historical evidence of successful pairing in the world of software engineering is presented at the end of the paper.

## Introduction

For the past twenty five years, the knowledge age has empowered people to seek out information and create new knowledge that can be shared within a large domain. Both the individual worker and the organization have found competitive advantages in not only keeping up with the current state of technology, but also creating new technological knowledge that is used to deliver the best products and services to the customer. The late management consultant, Peter Drucker, described knowledge as being the "only meaningful economic resource." Indeed, organizations can be viewed most importantly as knowledge-creating entities who's most valuable resource is knowledge generated through human experience (Nonaka 2000). Technology industries perhaps best exemplify this idea, where innovation can drive companies to the top of their field and technological complacency can quickly make a company obsolete. Knowledge management within organizations seeks to provide some organizational structure to knowledge creation, transfer, and assimilation. It is described by (Malhotra 2001) as the management processes of acquisition, conversion, and application of knowledge. For systems engineering and the systems engineer, effective knowledge management is an essential activity to be managed throughout a project or program. Failure to effectively manage knowledge resources can result in extending schedules and adding cost to thin profit margins. The first part of this paper seeks to define and provide insight into the knowledge base resource and necessary knowledge management activities.

Knowledge management decisions and activities are often reactions to changes in the environment, such as technology advances, shifting market demands, and workforce transitions. There is significant research into how workforces can be designed to respond effectively to change. The term *agility* has been applied to workforces that are in "a continual readiness to change" (Goldman 1995). An agile work force is often a necessity rather than simply a particular approach (Alavi 2013). This paper proposes that discussions of the agile work force are intrinsically linked to discussions of knowledge management, as it is the human worker that is necessary to derive knowledge from mere information. The second part of this paper addresses the dynamic environment where knowledge management decisions and activities take place.

Finally, to build on existing research areas in knowledge management and workforce agility, an agile adaptable-pairing approach to knowledge management is presented. The operational architecture of the approach enables and facilitates effective transfer, assimilation, and application of knowledge in unpredictable and unstable systems engineering environments.

## Knowledge Base and Knowledge Management

An organization's and individual's knowledge base consists of both explicit and tacit knowledge.

Explicit knowledge includes company standards, manuals, specifications, formulas, proofs, and data. Explicit knowledge is fairly easy to access and transfer between individuals, and should require minimal interpretation with respect to intent and application. Kogut distinguishes information as a type of explicit knowledge, in that it can be "transferred without the loss of integrity" (Kogut 1992). Codifiability is another adjective that may be used to describe explicit knowledge. Explicit knowledge may comprise the majority of an organization's core knowledge, accessible to all employees as guidance for most day to day activities. For structural engineers, this may be a manual that includes descriptions of how to analyze typical structures and proprietary design allowables for types of joints and materials. For a production mechanic, explicit knowledge may include documents on how to fabricate and assemble different types of parts. Knowledge offered by technical standards should encompass sufficient information to guide specific task accomplishment. New or less experienced employees may need more senior members of the team to explain some areas in more detail, but the standards should be clear enough that they are not open to interpretation.

Tacit knowledge is derived from human experience. The idea of tacit knowledge was introduced by Polanyi in 1958 (Polanyi 1958). Kogut describes tacit knowledge as human "know-how". Tacit knowledge is not formally documented, and may not be part of the common knowledge available to all employees. It is accumulated skill or expertise (von Hippel 1988). Tacit knowledge is more difficult to transfer between employees, because it can be subjective and may be difficult to describe or transfer as a concise knowledge package. Examples of tacit knowledge are beliefs, perspectives, mental models, and ideas (Nonaka 2000). Generally, tacit knowledge comes from first-hand experience of how to handle a given situation. There is evidence that tacit knowledge plays a vital role in organizations. Davenport claims that important decisions are more likely to be made using knowledge in the heads of staff rather than information from other channels (Davenport 2005). It is likely that the most significant technical challenges (complexity) faced by an organization do not have clean solutions that can be referenced in a document of book. Entire organizations, from the leadership team to the junior employees, rely heavily on tacit knowledge from key personnel.

A common distinguisher between knowledge and information or data, is that knowledge involves human judgement. "Knowledge consists of truths and beliefs, perspectives and concepts, judgements and expectations, methodologies and know how" (Lefrere 1997). Knowledge is also context specific (Nonaka 2000). It is not enough for an organization to have a library of available resources if it does not have the personnel to distinguish what information is necessary and how it should be applied to a given situation. This is one reason tacit knowledge may be particularly difficult to capture and assimilate. As a consequence, tacit knowledge is often neglected in knowledge growth "blueprints" that explain what to learn, but contain little insight into how to apply it (Kogut 1992).

An adequate knowledge base is necessary for the success of any systems engineering project. Both tacit and explicit knowledge have their utility within the engineering organization, and should be monitored to ensure the knowledge base is meeting the demands placed on the company. For this reason, knowledge management activities in an organization are essential. The key output of knowledge management is readily available knowledge that drives the success of a project. There must be a process that asks for the right type of knowledge at the right time, and a process that delivers the right knowledge in a timely manner. Malhotra describes knowledge management as acquisition, conversion, and application processes (Malhotra 2001). Knowledge must first be acquired before it can be used. The conversion process ensures that knowledge is easily available to its users by integrating it into an organizational structure. There must also be structures that allow knowledge to be stored and retrieved. Finally, knowledge must be applied effectively. A key to effective knowledge application may be the ability to distinguish the right time to use knowledge so that decisions are made with the maximum number of known variables.

A central theme with knowledge management is the effective integration of knowledge within an organization. Companies acquire information and knowledge in many different forms. Some forms, such as airworthiness directives for a commercial airline, may be delivered in a predictable form that allows routine and quick integration. Soon after the directives have been received, they are sent to the relevant people who know how to act on them. Integration of other forms of knowledge may not be as straight forward. A company that is looking to branch off into a new technology market may hire new employees with experience in that field. The integration of that new knowledge into the existing organizational structure and culture may be the most significant challenge of the knowledge acquisition process.

The take-away from the knowledge management discussion are:

- Many different types of knowledge, both explicit and tacit, form the knowledge base and contribute to the success of a systems engineering project. Human interpretation and judgement are often necessary to convert information into knowledge. Different types of knowledge have unique benefits and challenges with respect to effectively applying knowledge to a particular task.
- Knowledge management provides structure to knowledge related activities and answers key questions such as how much knowledge to acquire and when, how

knowledge should be stored and retrieved, and how to match the best knowledge with the tasks at hand.

The next section investigates the operational environment of knowledge management for a systems engineering project.

## Knowledge Management Environment

For systems engineers in technology fields, the management of knowledge for any given project faces several principal challenges. Each of the challenges presented stems from an unsteady knowledge environment. Unsteady environments can be created through unpredictability, uncertainty, risk, variation, and evolution. (Dove 2001).

- Unpredictability: the randomness among unknowable possibilities
- Uncertainty: the randomness among known possibilities with unknowable probabilities
- Risk: the randomness among known possibilities with knowable probabilities
- Variation: the randomness among knowable variables and knowable variance ranges
- Evolution: gradual and successive developments

Changes in the project environment may affect the ability of a team to apply the correct knowledge, at the right time, to an appropriate issue. Over the project life cycle, poor knowledge management may lead to significant cost, schedule, and quality penalties. Activities to counter these challenges may be either proactive or reactive. In either type of response, the goal is to make the correct strategic moves (timely, cost-effective, and appropriate) to best position the team to make knowledgeable decisions.

The first challenge faced by knowledge managers is that work forces are not static. Individuals are a complex unit of the work force and are constantly in motion. Individuals may choose to accept a role within an organization, change roles, or exit an organization. The movement and growth of individuals can be classified as both functional and numerical change (Atkinson 1984). Functional change is the movement of individuals between different types of tasks. In the context of the operational environment, functional change can be described as a **variation**. Functional change often involves movement across organizational boundaries. An example may be a design engineer taking on the tasks of a production or liaison engineer as a project matures past the conceptual and design phase. Functional changes may occur due to workforce flexibility initiatives. Individuals with broader skills base and cross-training may allow a company to have easier variable capacity in different knowledge areas than traditional department organizations. This is particularly common in start-ups where employees are expected to wear different hats. If the work load in one department becomes low, individuals with knowledge of other disciplines can be more easily re-allocated. The idea is to allow work forces to be fluid so that they can constantly re-align themselves with the goals and needs of the company. With each movement of individuals, there is some shifting of knowledge and some variation in the knowledge management environment. For a system’s engineer, the movement of workers can add or subtract knowledge from a project team. Subtractions can create knowledge voids that must be filled by other team members. Failure to do so in a timely manner can cause schedule slips.

Numerical change is associated with an organization’s headcount. Work force reductions are often implemented when work load is low. Similarly, hiring phases are often associated with the award of new contracts. A sudden decrease or increase in work force is a **risk** to the knowledge management of a project. Like functional changes, there are efforts to minimize the negative impacts of numerical workforce change. The use of contract workers, for example, may allow a company to quickly fill a knowledge gap or add capacity to a short-term project. Because the most important knowledge is often found in the heads of employees, numerical changes can dramatically increase, decrease, or change a company’s knowledge base. Consequences to a project may be a sudden lack of knowledge support, or disruptions to the team organization.

A second project **risk** that may arise from poor knowledge management is unnecessary mistakes and rework due to a lack of knowledge or mistakes in correct knowledge application. This may result from a lack of direct supervision or from routine errors that can be made by an engineer of any skill level.

Another challenge of knowledge management deals with the changing state of technology. With new or evolving technology, existing knowledge bases may become insufficient, irrelevant, or obsolete. This process can be described as **evolution** within the knowledge management environment and is perhaps most noticeable on long term projects. To remain competitive, companies must embrace or generate their own new technology (innovation). Research into agile organizations shows there may be competitive benefits from cultures that embraces change, rather than cultures that rely on traditional “sweet spots” or “core competencies”. From the knowledge management perspective, this means being constantly aware of knowledge areas that will provide the best advantage to a project or knowledge areas that may need to be supplemented to allow the integration of new technology into a project.

Knowledge management is also **unpredictable**. Technology industries deal with customers who often have changing requirements, priorities, and wants. Ever-changing demands may pull a project into unfamiliar knowledge territory. For example, customers may request a certain type of output from a system that was not originally considered. This request may require new or outside knowledge in order to be fulfilled.

One final knowledge challenge may be posed by shifting market opportunities. Similar to a customer’s change request, business leaders may realize a market opportunity that requires unique knowledge that may not be well documented or currently only exists outside of the organization. This can be categorized as **uncertainty** in the environment.

## The Knowledge Management System

Now that the mission and operational environment of knowledge management activities has been set, the focus of this paper can shift to present an effective approach to knowledge management in the context of a systems engineering project. Due to its challenging and demanding environment, any knowledge management architecture must be agile. Agility is a system’s ability to make strategic moves to counter or take advantage of a constantly changing environment. An agile knowledge management system must account for the fluidity of the work force, technology progressions, and shifting customer wants. This contrasts a conventional knowledge management approach that gathers all the perceived knowledge necessary to complete a project and struggles to adjust to environment changes as the project progresses.

At the onset of a project it is common for managers to create an organized team that hopefully possesses the skills and knowledge necessary to meet the project requirements. This team can be labeled as the baseline knowledge profile. For an aircraft modification team, the baseline knowledge profile may consist of design, stress, aerodynamic, electrical, thermal, and systems engineers. There will likely be a variety of backgrounds and experience levels. There will generally be some level of functional and integrated leadership assigned and tasks will be delegated as deemed appropriate. If this is the extent of knowledge management, then the project may suffer from many of the environment challenges discussed previously.

The proposed architecture for agile knowledge management can best be described as adaptable pairing. The value of adaptable pairing will be more efficient knowledge sharing for informed decision making, less rework from mistakes, and added redundancy for shifting personnel.

## *Agile Knowledge Management*

From the previous *knowledge base* discussion, it was shows that tacit knowledge (individual knowledge) is often the most valuable, but also the most difficult form of knowledge to transfer. The successful sharing of tacit knowledge also includes the transfer of human interpretations and personal experiences. The pairing of two individuals so that tacit knowledge can be shared dates back centuries. The "master-apprentice" relationship grew during the middle ages for many craft and guild workers and still exists formally in many fields today (Lee 2012). Traditionally, this relationship ensured that valuable skills and trades could be passed between generations. Apprenticeships can effectively facilitate the transfer of knowledge between individuals because the apprentice is not only taught theories, but also observes their application in solving real problems. In today's technology centered organizations, the master-apprentice exchange still exists on an informal basis. Green engineers are often paired with mentors so that knowledge can be passed down and oversight can be provided. The pairing may also serve as a two way street that allows the green engineer to observe how theories are applied in the real world, and at the same time allows the green engineer to provide fresh ideas and perspectives to the more experienced mentor. The problem for engineers working on real world systems, that have tight cost and schedule constraints, is that this traditional apprenticeship approach often fails to generate timely and effective engineering responses to daily challenges. As a result, mentor type relationships are often seen as a hindrance to project success.

There may be many reasons for this, but for development of the agile knowledge management architecture, the focus will be on the environment where technical decisions must be made.

It is assumed at the start of a project that each team member either possesses or will be directly provided with knowledge necessary to make informed decisions. The knowledge management problem really begins after program leadership believes they have all necessary knowledge management in place. Once a project has been set in motion what guarantees the baseline knowledge management structure is adequate to account for daily environment changes? The answer is that without an agile form of knowledge management, there is no guarantee. The knowledge structure may break down as engineers relocate or are tasked with challenges that require knowledge that could not have been predicted. The strain on both experienced and new engineers increases with decreasing levels of adequate knowledge support.

The knowledge management of systems engineering projects must instead be designed to always operate in the present. It must facilitate rapid sharing of knowledge, fill sudden voids, and catch early mistakes. Adaptable pairing has the potential to meet these needs. A pair will be the base unit of a project team. Vital to the success of these working pairs is having effective pairing plans and clearly defined relationships. Because there are no “all-knowing” gurus, an engineer presented with a problem outside his own area of expertise really needs access to a much wider pool of knowledge, sometimes located in different disciplines or different organizations. Any form of pairing must then be based on a particular task and applicable knowledge. Individuals should be allowed to move freely and have independence to form their own working relationship outside of traditional functional working groups. Establishing and maintaining complimentary pairs throughout a project will maximize the benefits of pairing. The management of pairing plans may be the responsibility of project leadership, or it may rely more on self-organization within the team. In either case, pairings should be treated as a necessity for informed decision making, quick response times, and minimal rework. Pairings may change between projects, or even multiple times throughout a project. An individual may also participate in multiple pairing relationships simultaneously. Daily decision making will naturally be based on experiences from multiple individuals. The absence of an individual will not require expensive “up to speed” times as engineers scramble to fill voids. Redundancy is key to solid knowledge management. There must be constant feedback between working pairs and project leadership responsible for ensuring an adequate knowledge base. If specific knowledge required to provide an efficient solution is lacking, new knowledge may need to be brought in from outside the organization and integrated into the pairing structure.

The general pairing relationship will involve knowledge transfer through discussions, and work checks through assignment swapping. For this discussion, an assignment is a subtask assigned to a single individual at a time. Within a working pair, each individual will provide and receive tacit knowledge through discussions to solve daily challenges. Assignment swapping may differ depending on the pairing type. For software engineers, it may involve writing two pieces of code concurrently, where each code is routinely swapped between engineers. For analysis engineers, it may involve swapping individual jobs required for a complete analysis. The benefits of assignment swapping from the project leadership perspective are two-fold. First, mistakes are more likely to be caught early in the project as work is constantly reviewed and improved. This contrasts a work review process that often begins after design has been finalized. The second benefit is that there are generally two engineers with intrinsic knowledge related to each defined assignment. This improves daily response times, as well as reduces the risk of unexpected voids if an employee leaves a project. Assignment swapping contrasts the idea of having two engineers performing the work of one, as is often the case with traditional mentorships. Thus, a pairing relationship should increase overall productivity by producing a similar amount of work with higher quality.

Complimentary has been used to describe a successful working pair. Because the pairing relationship is formed to transfer knowledge and swap work assignments, a complimentary pair may consist of two engineers with a relatively similar skillset. Engineers of vastly different levels of expertise will likely form a relationship more in line with traditional mentorships, where immediate costs benefits are questionable. Pairing plans must account for the type of knowledge that is required to meet a project need, and the existing skillset of the individuals.

The direct benefits of adaptable pairing to a project team should be clear. Pairs that match a knowledge request with a knowledge source are an effective way of transferring tacit knowledge to meet immediate needs. This improves response time and quality of daily project decision making. Effective pairings can also reduce the amount of mistakes and limit the impact of shifting team members. However, there are also long term benefits of adaptable pairing that will only briefly be mentioned. As discussed in the introduction, a solid knowledge base is vital to a company’s success. Pairings will allow valuable knowledge to spread between individuals as a byproduct of daily informed decision making. This will reduce the need for separate classes or work study groups to transfer knowledge. Motivation is key to effective knowledge transfer, and solving immediate, real problems provides high motivation for all parties. There is also credible evidence to suggest pairing type relationship increase individual performance and improve job satisfaction (Eby 2009). This may reduce employee turnover that is costly to both current and future projects.

## A Proposal to Systems Engineers

The merit of pairing based teams has been discussed, but how does a project engineer establish and maintain effective pairings throughout the life of a project? The first step will be to establish a baseline pairing plan founded on defined tasks and team members. This should occur immediately following the creation of project requirements. Pairing plans require thought and understanding of the necessary tasks and available project knowledge. This contrasts manpower lists that simply ensure there will be an adequate number of engineers available to complete the entire scope of work. The primary considerations of the pairing plan are to create pairs that will make sound decisions in relation to their tasks and minimize their mistakes or rework.

Effective pairing plans should:

- Clearly define task boundaries for each pair.
- Create self-sufficient pairs – pairs that possess or have immediate access to the tools and knowledge necessary to complete assigned tasks.
- Create pairs with some common core knowledge – pairs that can engage in useful knowledge sharing and assignment swapping.

Effective pairing plans should not-

- be limited to engineers of the same title or functional groups
- consist of traditional mentor relationships that may hinder response times

As a project progresses, required tasks and personnel evolve. These evolutions require modification to the pairing plan. The culture and relationships of the project team should facilitate easy changes to the pairing plan. The project engineer needs to establish a feedback loop to constantly evaluate the performance of each pair and knowledge needs of each task. Pairs should also be free to report any knowledge or task concerns to the project engineer. At any point in time effective pairing plans should be maintained by the project engineer. Individuals should be expected to fully engage in pairing activities and view them as a necessary to maximizing project success.

Within a pair, each individual should:

- Leverage all available knowledge to solve daily challenges.
- Fully understand all assigned tasks and work performed by the pair.
- Create a work swapping plan.

The activity of monitoring and adapting pairing plans should be performed throughout the project. Often, the cost of mistakes and poor decision making increases as a project progresses. Thus, there could be severe penalties if there is resistance to baseline knowledge management change.

## *The Knowledge Management Concept of Operations*

A simple Concept of Operations diagram of the proposed knowledge management concept is illustrated Figure 1. The system's purpose is to pair complimentary individuals to engage in activities that reduce the time and cost of project tasks. Different pairs can be created from a common pool of knowledge. Pairings are formed to engage in knowledge transfer and work sharing, while adding redundancy to daily tasks.

As a project progresses, tasks and personnel evolve. At any point in time, at least two engineers are actively engaged as a pair.

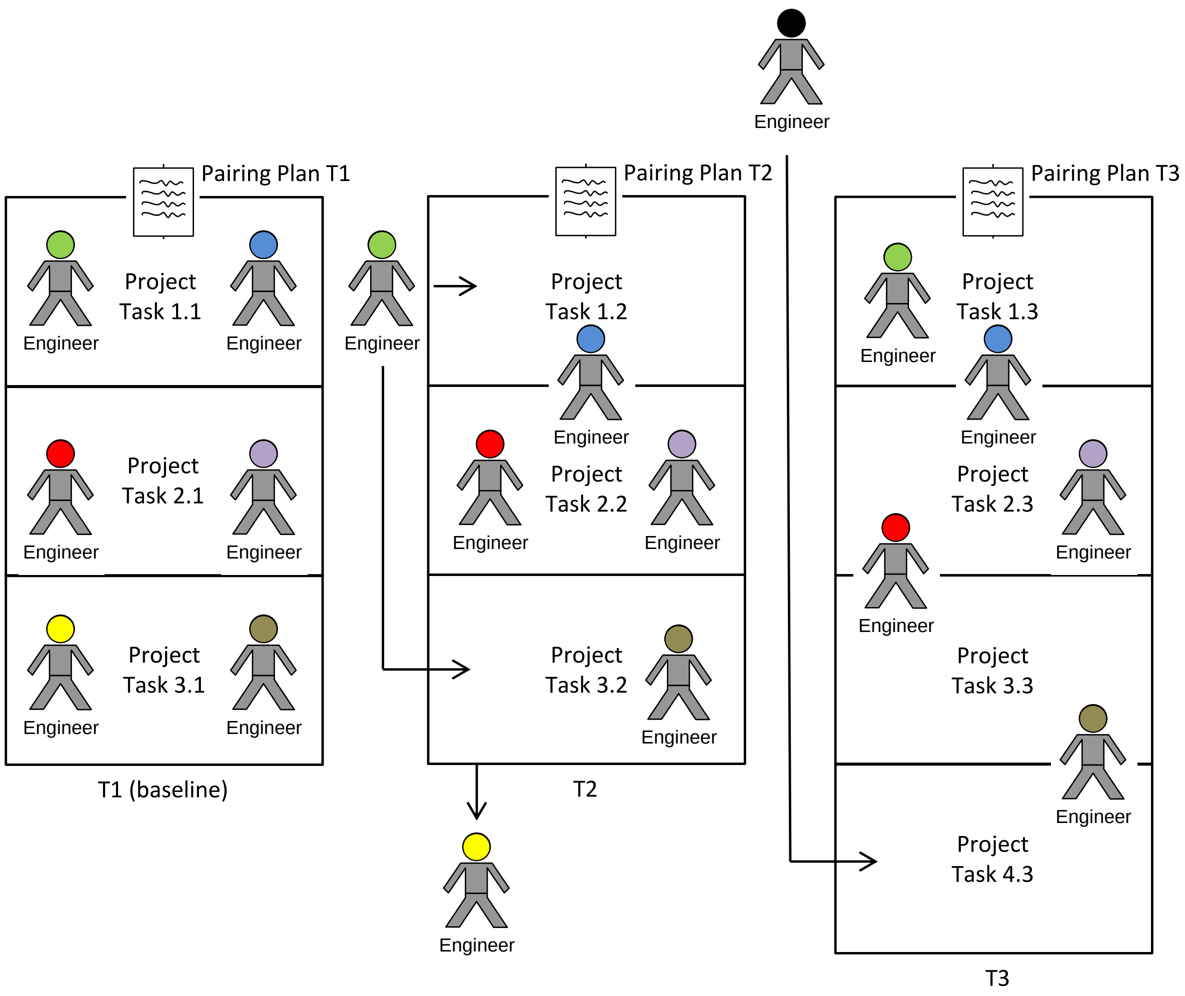

Figure 1. Pair-Based Agile Knowledge Management Concept of Operations

At the project onset (T1), three unique tasks are defined. An initial pairing plan delegates each task to two engineers. Each pair engages in knowledge sharing and assignment swapping to progress on their assigned task.

After a period of time (T2), the Yellow Engineer assigned to Task 3 is reassigned to a different program. It is decide that the Green Engineer is the best fit to fill the open spot on Task 3, while still performing a pairing role on Task 1. Also at T2, the scope of Task 2 increases and requires additional knowledge support that can be provided by the Blue Engineer. The pairing plan has shifted to account for changes in personnel and knowledge requirements.

Towards the end of the project (T3), the customer has requested an additional amount of work to be performed (Task 4). Pairing for this new task is accomplished by shifting the Brown Engineer to an additional Task, and introducing a new engineer (Black Engineer) to Task 4. At the same time, Task 3 has progressed to a new stage where the expertise of the Red Engineer is desired. The Green Engineer shifts back to working full time on Task 1.

## *Evidence for Pairing*

Although the pairing concept presented in this paper differs from traditional apprenticeships and mentorships, it is not a new idea in the field of software engineering. There is evidence in both research and practice to support the claim that work pairs can be beneficial (Williams 2000). Pair-programming is a practice for generating software. It can be described as “two programmers working side by side at one computer on the same problem” (Cockburn 2001). The actual benefits of pair programming are no doubt task specific. For example, pair programming has shown to be more beneficial for higher complexity tasks (Dyba 2007). Some of the claims made in support of pair programming are that it-

- Reduces the risk of errors and debugging time
- Provides more in depth reviews
- Provides an opportunity to share knowledge

In addition to reducing errors and increasing knowledge sharing, there is evidence to suggest that pair programming can actually reduce project times. “By working in tandem, the pairs completed their assignments 40% to 50% faster”. This paper does not try to assert that pairing engineers will increase the rate that work will be completed, but it is interesting counter evidence to the notion of “two workers doing one job” being less efficient.

Pair programming has its roots in the Extreme Programming methodology. Extreme Programming is a “light-weight methodology for small to medium size teams developing software in the face of vague or rapidly changing requirements” (Beck 2004). A quantitative survey has shown that extreme programming has been adopted in real world projects and that pair programming is a major component of Extreme Programming (Rumpe 2002). The survey was conducted in 2001 and results, shown in Figure 2, include responses from 45 evaluated questionnaires. The left side shows which of the 12 components of Extreme Programming were most used. Pair Programming ranked as the fourth most used (ranked 0-9). The right side shows how much each component contributed to the success of the project. Pair programming ranked as the seventh most helpful with a rating of about 85% helpfulness.

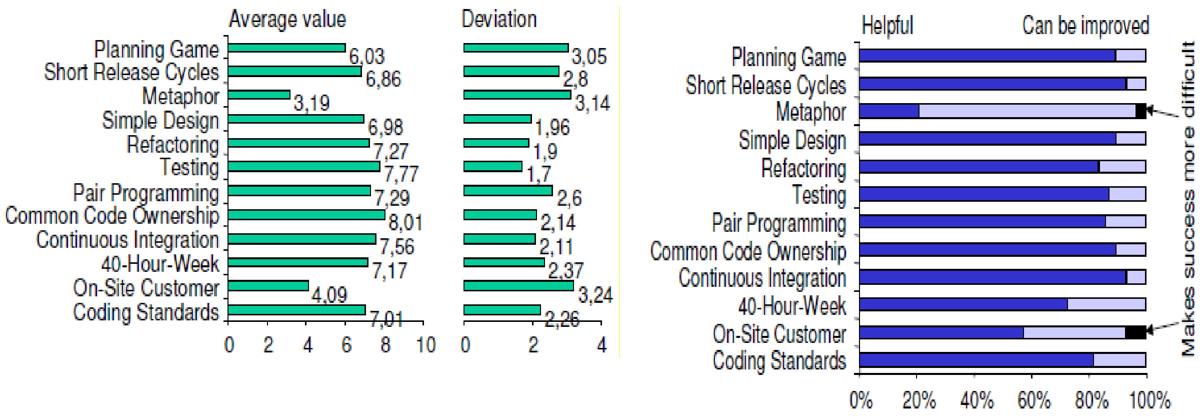

Figure 2. Survey results from (Rumpe 2002)

A book by Cockburn and Williams (Cockburn 2001) includes excerpts from real world applications of pair programming. Much of the feedback hi-lighted the success of pair programming after an initial adjustment period: mistakes caught early, collaborative learning opportunity, increased job satisfaction.

The value of pairing in an Extreme Programming environment can be debated, but there are no doubt situations where it has merit. It is reasonable to assert that pair programming benefits can be extended beyond the world of programming and be applied to other types of systems engineering development.

## Conclusions

Knowledge is grouped into tacit and explicit categories. Both types of knowledge form the knowledge base of an organization. Tacit knowledge predominately resides in the heads of individuals and is often difficult to quantify and transfer. Knowledge management is an important and challenging activity for a systems engineer in a technology organization. Individuals are the most important components of the knowledge management system. Knowledge management operates in an unsteady and often fast paced environment where it may be hard to recover from a slow response or lack of foresight.

An agile approach is necessary to deal with the constant personnel, technology, and customer driven changes. The proposed agile architecture for knowledge management consists of adaptable pairs or working groups. The value proposition of constantly evolving pairs is increased project quality, lower project cost, and more predictable schedule performance.

Beneficial activities of pairing are assignment swapping and knowledge sharing. These activities lead to fewer errors and rework, faster resolution of issues, and organizational resiliency to changing personnel. Evidence from pair programming is used to support these claims.

The creation of pairing plans and management of pairs are vital to the success of the agile knowledge management architecture. Pairings should form primarily for the benefit of the project – increases quality, lower costs, and reduced schedule time. Secondary benefits such as knowledge growth and work satisfaction are also important to the sustainability of a technology company. Over time, pairings will increase the skillset and experience of the workforce for successive projects. With each pairing experience, the knowledge base of the organization evolves and becomes more aligned with the current challenges facing the organization.

## References

Alavi, Somaieh and Dzuraidah Wahab. 2013. A Review on Workforce Agility. Research Journal of Applied Sciences, Engineering and Technology 5(16): 4195-4199.

Atkinson, John. 1984. Manpower Strategies for Flexible Organizations. Personnel Management. August 28-31.

Beck, Kent. 2004. *Extreme Programming Explained.* Addison-Wesley.

Cockburn, Alistair and Laurie Williams. 2001. *The Costs and Benefits of Pair Programming*. Addison-Wesley Longman Publishing Co., Inc.

Davenport, Thomas and Lawrence Prusak. 2005. Working Knowledge: How Organizations Manage What They Know. Ubiquity.

Dove, Rick. 2001. *Response Ability - The Language, Structure, and Culture of the Agile Enterprise*. Wiley.

Dyba, Tore, et al. 2007. Are Two Heads Better than One? On the Effectiveness of Pair Programming. IEEE, November/December.

Eby, Lillian, Tammy Allen, Sarah Evans, Thomas Ng, and David DuBois. 2009. Does Mentoring Matter? A Multidisciplinary Meta-Analysis Comparing Mentored and Non-Mentored Individuals. Journal of Vocational Behavior, 72, 254-267.

Goldman, Steven, Roger Nagel, and Kenneth Preiss. 1995. *Agile Competitors and Virtual Organizations. Strategies for Enriching the Customer.* Nostrand Reinhold.

Kogut, Bruce, and Udo Zander. 1992. Knowledge of the Firm, Combinative Capabilities, and the Replication of Technology. Organization Science. August 1992.

Lee, Dionne. 2012. Apprenticeships in England: an overview of current issues. Higher Education, Skills and Work-Based Learning 2(3), 225-239.

Lefrere, Paul. 1997. Knowledge management: A strategic agenda. Long Range Planning, June.

Malhotra, Arvind and Albert Segars. 2001. Knowledge Management: An Organizational Capabilities Perspective. Journal of Management Information Systems. Summer.

Nonaka, Ikujiro, Ryoko Toyama, and Akiya Nagata. 2000. A Firm as a Knowledge-creating Entity: A New Perspective on the Theory of the Firm. Industrial and Corporate Change 9 (1): 1-20.

Polanyi, Michael. *1958. Personal Knowledge: Towards a Post-Critical Philosophy.* University of Chicago Press.

Rumpe, Bernhard and Astrid Schroder. 2002. Quantitative Survey on Extreme Programming Project. Third International Conference on Extreme Programming and Flexible Processes in Software Engineering, Alghero, Italy, May 26-30.

von Hippel, Eric. 1987. Cooperation between rivals: Informal know-how training. Research Policy 16:291-302.

Williams, Laurie, et al. 2000. Strengthening the Case for Pair Programming. IEEE, July/August.

## Biography

Christopher Leroux has systems engineering and structural analysis experience with Continental Airlines, United Space Alliance, L-3 Communications, and Firefly Space Systems, and is currently working on a project team developing hardware for the Firefly Alpha launch vehicle. Christopher has a Bachelor of Science in Aerospace Engineering from Texas A&M University and a Masters of Engineering in Systems Engineering from Steven's Institute of Technology.

**Rick Dove** is CEO of Paradigm Shift International, and an adjunct professor at Stevens Institute of Technology teaching graduate courses in agile and self-organizing systems. He chairs the INCOSE working groups for Agile Systems and Systems Engineering, and for Systems Security Engineering. He is author of *Response Ability, the Language, Structure, and Culture of the Agile Enterprise.*