Building Successful Software Development Teams Using TSP and Effective Communication Networks

Dr. William R. Nichols
Bechtel Bettis, Inc.

Social network models can help explain how and why some organizational structures and practices work. Moreover, network analysis is accessible to engineering practitioners and is particularly effective in helping us understand the value of Team Software Process℠ (TSP℠). Networks not only offer an explanation of how TSP works with respect to communication, but also suggest that as we scale beyond a team of teams, new organizational structures will be required. The role manager structure sets TSP apart. Teams that use role managers take advantage of a proven communication pattern that scales as teams grow. Successful work is facilitated by effective communication, which can be improved with specific network structures. These structures can take shape through the self-organization of teams around TSP role managers. Unlike the traditional tree hierarchy that you see on most organizational charts, the more flexible, self-organizing network can respond quickly to the demands of a fast-paced workplace.

Every software development organization strives to build successful project teams. But almost anyone who has been part of a growing organization has seen formerly successful teams fail as coordination, communication, and decision making were impeded by increased team size.

As a Team Software Process℠ (TSP℠) coach and team lead, I have struggled with the problems of getting the right people talking through the requirements, synchronizing schedules, and working through the problems such as design and configuration control. The time demands upon a lead in the middle of these decisions become overwhelming. Fortunately, TSP encourages role managers to guide self-directed teams toward making decisions and completing work.

Role managers act as the conscience of the team within certain domains: planning, design, quality, customer interface, implementation, test, support, and process. Role managers need not *do* the associated domain tasks, but rather serve as points of contact and ensure that the work is done and done well.

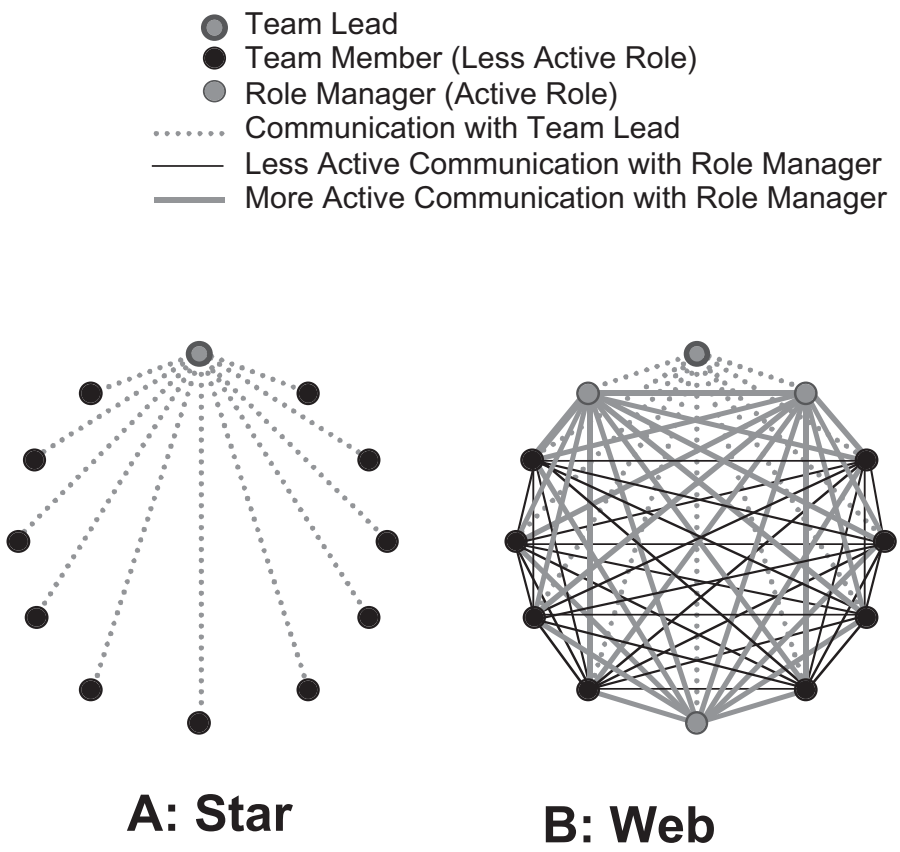
Watts Humphrey [1] describes the reasons for role managers. I found that although these reasons seemed sound, many of the team members, including myself, could not internalize this explanation. The epiphany for me came after I used a pen and notepad to sketch out the principle patterns of communication within my team and among the several teams on our project. I discovered that there is another and more compelling reason that roles are important in making TSP work.

The patterns were based on subjective rather than actual measurement; nonetheless, I discovered that I could demonstrate important characteristics of our group interactions by graphically representing the paths of communication. As if it were a physical network, I sketched an idealized version of a team reliant upon its lead for passing information and making decisions, shown in Figure 1A. I represented team members as *nodes* and the communication *links* as lines joining the nodes. The hub and spoke topology formed a *star* with the team lead in the central role.

I next sketched Figure 1B to show several role managers, for example, the planning, design, and customer interface role managers assuming responsibility for these important knowledge domains. The role manager links distributed communication, thus opening new paths and reducing the communication traffic load through the team lead. Finally, I added some less-active roles to Figure 1B, (testing becomes more active late in a project, design less active) and the result was a complete network following a web. I reasoned that this was the team that would keep running even if one or two key members became unavailable.

Using this graphical approach, I was able to qualitatively show, by following the Figure 1A pattern, how team leads had become overloaded. With this graphic, I

Figure 1A and 1B: *Star and Web Network Technologies*



℠ Team Software Process and TSP are service marks of Carnegie Mellon University.

was able to show how the team leads had become bottlenecks, and how reinvigorating the role managers' activity would change the communication patterns to our benefit.

Role Managers Add Communication Links

A necessary condition for a self-directed team is that the team manages all tasks. TSP teams designate role managers to assume cognizance of important task-based information domains within the team. But the benefits of role managers extend beyond the individual team.

Multiteam TSP (TSPm) scales the practice of TSP to larger projects containing more than one team. Across the project, teams of role managers – one from each team – form *affiliation groups*, directing information for a given domain through a team knowledge node – the role manager – to the greater project, thus distributing project information traffic. Figure 2 shows three teams (clusters) communicating through multiple channels. Communication traffic is heavier within teams and lighter between teams. The primary paths of communication between teams are through team leads and role managers.

Because communication becomes more efficiently directed, there is less communication *traffic*, and nodes no longer get as many *busy signals* when seeking information. There is no strategically placed node that, when lost, would cause a catastrophic failure. Moreover, the information most commonly needed is under the cognizance of someone who knows or can gain access to that information when it is needed.

For example, a new team member may need clarification on requirements. The customer interface manager may or may not know the answer, but should know whom to ask. The power of networks is leveraged through this selective specialization, thus information becomes readily available through the network. The team leaders are still important, but they no longer stand out or create bottlenecks in the network topology. By creating role teams, we have enabled the *team of teams* to function as a *small world*. Everyone in another team is either a *friend* or a *friend of a friend*. Information flows within and among the teams with very few intermediate connections.

This has fundamentally changed the group structure and dynamic. The smaller groups, teams, and role teams can invest in social capital (spend time cultivating relationships) required to form tightly knit

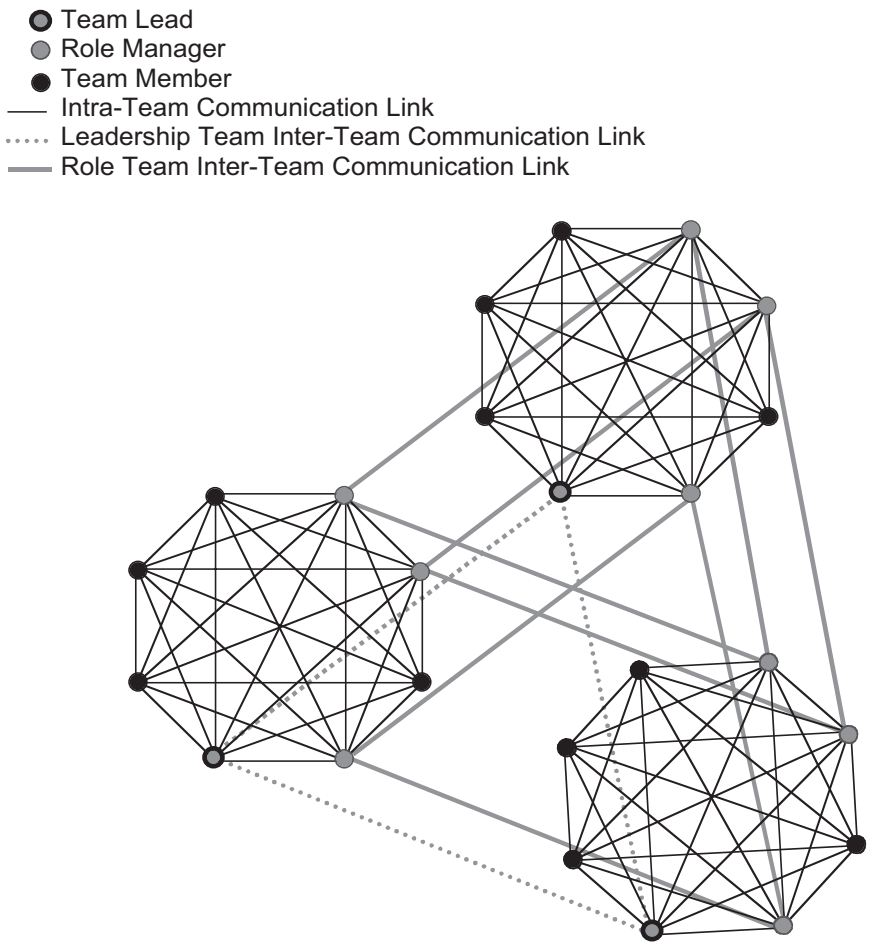


Figure 2: *Three Teams (Clusters) Connected by Team Leads and Role Managers*

teams. Where it is necessary to pass information between groups, we know who the *go-to guy* is. A structure has emerged that does not show up on the organizational chart.

Team and Project Size Limits

Physical networks have physical and engineering constraints. Routers, for example, can process only a limited bandwidth; transformers in an electrical grid can carry only so much current. Similarly, human networks have constraints that must be considered. It is not possible to work closely with a large number of people simultaneously. As working groups become large, communication requires more overhead.

In "The Mythical Man Month," [2] Brooks points out that there are n(n-1)/2 potential links, leading to an n-squared scaling with team size. A team of 15 has 50 percent more links than a team of 12, more than twice the links of a team of 10, and five times as many links as a team of seven. Time is limited, and each relationship requires time for maintenance. Because of this, it is practical to keep working groups small [3]. Some claim the *sweet spot* is around a team as small as seven [4]. For our purposes, we will place the practical upper limit at 12, based on an observed sociological phenomenon known as an *empathy group* [5]. Larger teams can exist, but they will usually factor into sub-teams.

It may not be just the total number of potential links. Figure 1A suggests that the number of direct links any individual can support may limit team size. In Figure 1A, the team lead is the central node. The communications to other team members are represented by links (communication channels) to other nodes (team members). In Figure 1A, a star network appears to be simple and elegant, and team members communicate primarily through the lead who is a *hub* linking the nodes. Most questions are resolved by asking the lead for clarification or guidance. There is no interaction between most nodes. However, the hub creates a communication bottleneck. For example, if the person acting as the hub is out sick, no communication can take place, no guidance can be provided, and work cannot move forward. If the hub needs to manage too much communication, some will not take place. If you limit your team communication to flow through one centralized point, it rapidly

becomes less effective, and because of the bottleneck effect, team size can no longer scale.

To achieve maximum scaling within a team, encourage efficient, point-to-point communication. Distribute the communication traffic in a decentralized network. By distributing the communication traffic through many node-to-node pairs, the traffic becomes balanced and bottlenecks are eliminated. If one node is removed, others can easily replace it. The cost is the time needed to build and maintain the relationships. But the resulting *Web* network is strong and flexible; it provides ongoing, efficient communication that keeps information flowing and tasks moving forward.

Consider Figure 1B as an alternative. In this communication model, the team lead has delegated responsibility. TSP does this naturally through the role managers who assume responsibility not only for domain knowledge, but also for tracking tasks within that domain. Adding several highly active roles significantly increases the available routes for moving information between any two points. The communication traffic is distributed so that the team lead no longer stands out in the network topology.

Inclusion of the less active roles permits most sorts of information essential to project success to be communicated directly. In the language of networks, we have converted a network with average node-to-node degree of separation (the number of links that a message must traverse between two nodes) of nearly two, to one with an average degree of separation of one. This small degree of separation, along with plenty of direct communication, is another key to project success.

Degrees of Separation and Small Worlds

The degree of separation is important for the team and project because it affects the flow and accuracy of information that teams need to be successful. Noah Friedkin of the University of California at Santa Barbara has shown that the limit of observability in organizations is only about three degrees of separation [6]. This is intuitively consistent with how we may use the *friend of a friend* (two degrees of separation) to gather information or to access other parts of the organization. However, at three degrees, the view becomes cloudy; at four, it becomes opaque. This makes sense if you consider some common barriers to clear communication:

1. **Messages are imperfect.** The sender and receiver can understand an ambiguous or vague message differently.
2. **When information is directed through a node, that node acts as a filter.** The message is filtered through that person's experience, knowledge, and priorities. Each node can change a message in subtle ways that, when added together, result in an original sender and final recipient understanding very different meanings.
3. **The technical means of communication are imperfect or incomplete.** Most communication channels include signal loss or noise. The telephone loses facial cues. E-mail loses facial and vocal inflection. Video conferencing has inconsistent sound and visual signal delays. Any of these can cause unintended interpretations of communication.

"Teams of role managers not only make the network a small world, but also serve to make the network searchable, greatly shortening the average communication path."

The upshot is that a functioning team must be kept to an average of three or fewer degrees of separation, much like the *small world* network described by Duncan Watts and Steven Strogatz [7]. The essence of a small world is that everyone knows everyone else through a very short chain of *handshakes*. This recalls the well-known concept of *six degrees of separation* that was based on a famous experiment by Stanley Milgram [8]. Watts and Strogatz described transforming a network into a small-world network by adding only a small number of random links. Local clusters, in our case teams, are the smallest of small worlds. Their world becomes even smaller by adding a few role managers. Role managers direct information through standard and commonly understood channels. Teams of role managers not only make the network a small world, but also serve to make the network searchable, greatly shortening the average communication path. This becomes particularly important as projects and teams grow.

Scaling Up to Teams of Teams

When the project size scales up, teams must deal with the stresses that come with the increased numbers. The British anthropologist Dunbar [9, 10, 11] noted that group size tends to saturate at around 12, similar to the *empathy* group described by Buys and Larsen [5]. This saturation occurs when the necessary investment in social capital becomes too large; at that point, the groups then fission into smaller groups. Dunbar also noted that the larger social network is limited to about 150, which is due to the human capacity to recognize and track personal facts about all members of a group.

Below 150 group members, a relatively informal structure is sufficient because peer pressure and personal loyalty are adequate to maintain discipline and control. Larger groups need a formal command structure to maintain order. For example, the Hutterites, a rural North American group that practices communal living, limit each community to 150 members [12]. Throughout history, basic military battle groups, comparable to a modern army company, remained near this limit. Many working groups and businesses fail at this point as efficient communication, knowledge, and informal control structures break down. Interestingly, Dunbar noted that where groups exceeded the nominal upper bounds, it was typical that *roles* had evolved, (e.g., sheriff, minister) that permitted people to interact appropriately with the role.

The size thresholds of 12 and 150 can be used as *rules of thumb* – heuristic guidelines – where we expect a new social order to accompany increased group size. When combined, the thresholds at 12 and 150 have implications for successful development teams. Virtual teams of role managers or team leads, drawn from each of the product teams, are the glue that binds a project into a small world. What happens as these virtual teams grow in size?

It is interesting to note that role teams reach a size of 12 (12 teams) at about the same time that the project reaches a size of 150, a number that, after all, is very close to 12 teams of 12. In this way, the rules of 12 and 150 converge. TSPm, on a modest-sized project, fits within limits imposed by these rules of 12 and 150. However, scaling TSPm beyond this size, perhaps to many hundreds or thousands, becomes problematic when the role teams that deal with inter-team coordination become too large. The next level of scaling appears to require either additional communication structures or substantial independence of subprojects.

The Organizational Chart Versus Self-Organization

Organizational structure can be a powerful factor in a project's success. In any organization, there are charts that show the official organizational hierarchy, but they probably do not represent the interaction patterns and functional organization through which work gets done – the result of self-organization that occurs in successful software development teams. Traditional organizational hierarchies are effective for imposing structure and control, but they are not effective for managing creative work in frequently changing environments such as those common to software development. Therefore, to manage effectively, we must work with the actual, self-organized network through which work gets done.

Fostering self-organization and flexible communication within a commonly understood structure solves this problem. The TSP role managers and role teams satisfy this need, providing a way to organize a project's information patterns for maximum efficiency and effectiveness. This form frees team leaders from constantly managing communication, allowing them to focus on strategic issues.

Formal leadership retains its importance for managing resources and setting business goals, but assumes a different role with respect to information, communication, and getting work done. A changed environment requires the network to change as well. Formal hierarchies are slow to change, which is insufficient in a dynamic environment. Self-organizing networks, however, are flexible. They adapt to a dynamic environment and can lead to success where less adaptable, formal hierarchies fail. Instead of trying to constantly restructure our formal hierarchies, we should look for ways to leverage the phenomenon of the self-organizing network. Self-organization within the role-team framework becomes the key to flexibility and meeting goals in an ambiguous and changing environment.

Conclusion

The key actions for building successful team communications are to identify organizational needs, encourage the right roles, support self-organization, and coach individuals. In addition to encouraging the self-organization of role teams, consider the organizational priorities and support necessary to encourage and sustain the right roles. From a TSP standpoint, we should coach teams and projects to tailor roles so there is a central focus for the project or organizational priorities.

Also consider additional ways to reduce the organizational path lengths. For example, encourage customer interface managers to form user groups, which reduce your path length to the user, probably to as few as two or three degrees. TSP coaching is another resource that must be kept to a short path length. Most effective is one degree of separation. For a successful effort, people need ongoing, one-on-one coaching.

As shown earlier, relying on a traditional organizational hierarchy makes a project vulnerable to single node failure and information bottlenecks, and does little to reduce path lengths. But we can use what we have learned about networks to address these problems in a flexible team environment.

Keep small teams tightly coupled with many internal links, as shown by the web network in Figure 1B. Fewer links between teams are adequate to maintain short, inter-team path lengths throughout the organization as shown in Figure 2. This model fits within human limits and scales up to a team of teams. Strong ties support the detailed and creative work within teams. Some team-to-team links are necessary to convert a project or organization into a small world. Weaker ties bind the teams to a project and make the network searchable. Role teams build communication paths starting from the context of important task domains. Role teams are a natural method for TSP to add these cross-team links.

The network model shows us how valuable functioning role managers can be to the success of small-world, self-organized networks. They can balance information flow, provide alternate paths if congestion develops, and make information easier to find. Fostering self-organization within teams and critical role-manager communication among teams can be highly motivating. Properly motivated and prepared teams are capable of extraordinary things.◆

References

1. Humphrey, Watts. Introduction to the Team Software Process℠. Addison-Wesley, 1999.
2. Brooks, F.P. The Mythical Man-Month. Addison-Wesley, 1975.
3. Kruchten, Philippe. "Scaling Down Large Projects to Meet the Agile Sweet Spot." IBM, 13 Aug. 2004 <www.106.ibm.com/developerworks/rational/library/content/RationalEdge/aug04/5558.html>.
4. Putnam, Lawrence H., and Ware Myers. "Team Size: Development Productivity Index." Cutter Information Corp., Aug. 1998 <http://jeffsutherland.com/objwld98/development_productivity.html>.
5. Buys, C.J., and K.L. Larsen. "Human Sympathy Groups." Psychological Report 45 (1979): 547-553.
6. Friedkin, N.E. "Horizons of Observability and Limits of Informal Control in Organizations." Social Forces 62 (1983): 54-77.
7. Watts, D.J., and S.H. Strogatz. "Collective Dynamics of 'Small World' Networks." Nature 393: 440-442.
8. Travers, J., and S. Milgram. "An Experimental Study of the Small-World Problem." Sociometry 32 (1969): 425-443.
9. Dunbar, Robin I.M. "Neocortex Size as a Constraint on Group Size in Primates." Journal of Human Evolution 20 (1992): 469-493.
10. Dunbar, Robin I.M. "Co-Evolution of Neocortex Size, Group Size, and Language in Humans." Behavioral and Brain Sciences 16.4 (1993): 681-735.
11. Dunbar, Robin I.M., and M. Spoors. "Social Networks, Support Cliques, and Kinship." Human Nature 6 (1995): 273-290.
12. Hardin, Garrett. "The Tragedy of the Commons." The Concise Encyclopedia of Economics. The Library of Economics and Liberty, 1968 <www.econlib.org/library/Enc/TragedyoftheCommons.html>.

About the Author

William R. Nichols, Ph.D., is a Personal Software Process℠ instructor and Team Software Process℠ coach, certified by the Software Engineering Institute. He currently leads a software development team at the Bettis Laboratory near Pittsburgh, Penn., where he has been developing and maintaining engineering and scientific software for 14 years. He has a doctorate in physics from Carnegie Mellon University.

Bechtel Bettis, Inc.
PO Box 79
West Mifflin, PA 15122
Phone: (412) 476-5667
Fax: (419) 781-9750
E-mail: wnichols@bellatlantic.net

℠ Personal Software Process is a service mark of Carnegie Mellon University.