Breaking Up Is Hard To Do

Patterns for Team Dispersal

by
Don S. Olson
AG Communication Systems
2500 West Utopia Road
Phoenix, Arizona 85027
Tele: (602) 581-4972
Fax: (602) 581-4862
email: olsond@agcs.com

Copyright © 1997AG Communication Systems Corporation
Permission is granted to copy for the PLoP-97 Conference

Introduction

This is a small collection of patterns for self-directed, software development teams which have demonstrated some measure of success but are at the end of their product development cycle or run and must be disbanded for the benefit of other teams in the organization. In our experience, the following practices have proven to benefit the receiving teams to a noticeable extent. It should be noted that in our organization we have been extremely fortunate to have enlightened managers who have shared risks in allowing exploration of new organizational models. The successes we have experienced are theirs to claim as well.

These patterns are meant as a supplement to the work of James Coplien in *Pattern Languages of Program Design* (Coplien, Schmidt, 1995), called "A Generative Development-Process Pattern Language." Cope's excellent work provided many patterns which we used as a basis for our initiation into self-directed work teams in the Intelligent Networks group at AG Communication Systems. By incorporating these patterns into our project strategy and plan and by becoming evangelists for the use of patterns in software design and team organization, we were able to create several successful teams in terms of financial gain and software quality, and one of our teams has been judged by the Organizational Change and Development Group to be the highest performing self-directed work team out of more than 40 in the company.

There is nothing revolutionary to be found here but rather small nudges to the normal organizational direction that may bring about, indirectly, an improvement in the quality of process and product of a software development team. This indirection is the essence of the generativity of patterns, what some of us in the community believe to be the patterns movement's most powerful contribution. Rigid hierarchical organizations and cast iron processes work less and less well in the fluid business and technical environments we inhabit, and "light hands" management requires that teams be quickly assembled and enabled to become productive business units in their own right. Patterns have proven their worth in our organization in helping us to reach this goal, and if any experience we can share through these patterns is beneficial, then the seed is cast, perhaps to bloom anew.

Pattern Map

[1]*Sowing Seeds* and [2] *By Request Only* address the need for judicious spreading of the word in regard to team success, but with a caveat to those who think that trumpeting *anything* as a success makes it so. [3] *Sever Along the Joints* concerns the way in which a team is to be dismembered, allowing for human nature, natural selection, and the organic nature of team development. [6] *Let 'Em Choose* discusses how to decide where team members should go for maximum benefit. [5] *Members Follow Model*, [4] *Pairs or Better to Open* , and [7] *Jumper Cables* deal with the way in which to preserve knowledge effectively when it's transferred between teams.

1. Sowing Seeds

PROBLEM: How do you preserve technical and organizational expertise when a team must be disbanded after a project?

CONTEXT: The team has proven successful as a self-directed work team, and the software architecture within which it has developed its products has been equally successful. The disbanding of the team and the dispersion of the team members is required for business purposes as new projects come along.

FORCES:

- You want to preserve the essential qualities of this team.
- You must distribute their expertise throughout the organization.
- Synergy actually worked on this team; the whole was greater than the sum of its parts.
- There is fear that individual team member's effectiveness will be diluted in a new or established team.

SOLUTION: Make the team's successes visible through presentations, meetings, and exchanges prior to team disbandment.

RESULTING CONTEXT: When tools, techniques, architectures, frameworks, and faces are known within an organization, seeds are sown in other teams that may begin to grow on their own. Because the people doing the original work in a successful team are known and available for questions and casual chat regarding their work, new ideas do not seem threatening or oppressive. Success attracts interest, but the sources of the success must be visible and available to share. Additionally, teams adopting technical or organizational approaches may enrich them with their own spin or interpretation or additions. However, a danger lurks in the overzealous presentation of successes, or alleged successes, so the reader is asked to also consider [2] *By Request Only*.

RATIONALE: In our organization, one team in particular showed extraordinary success as a self-directed work team, as a software product center, and as a healthy, happy, and fun group. How they managed themselves was of much interest as their successes were publicized. The teams "Project Strategy and Plan," a document which explained their way of working, became a model for other teams. The software architecture they used, known internally as SeRBA (Service Request Broker Architecture), became an object of interest for the way it mapped system components to individuals, permitting *Code Ownership* (Coplien) and solving integration problems early in the development cycle. Several teams adopted some, much, or all of this, and contributed additional knowledge and improvement to both architecture, code frameworks, team management, and external relations.

It should be noted that parading any old thing before a trapped audience is not what is called for here. As an antidote to such tendencies, the pattern [2]*By Request Only* suggests how information of team successes be shared.

2. By Request Only

PROBLEM: How do you promulgate ideas, architectures, processes, and otherwise promote successes without turning such exercises into showcases for those who value form and performance over substance?

CONTEXT: Things have worked very well for a particular team, and some of their experience may be valuable to other teams or to the development organization at large.

FORCES:

- You have information - technical, process, or organizational - that you wish to make known to the organization at large, in the hope of repeating successes.
- People, particularly busy technical people, are highly suspicious of showcases for alleged successes.

SOLUTION: Only give presentations by request from outside parties.

RESULTING CONTEXT: It adds a great deal of credibility to a presentation when someone is interested enough to ask for it and set aside the time and arrange the schedules of people to see it. It's only possible to apply [7] *Jumper Cables* effectively when the driver wants the car to start in the first place.

RATIONALE: It's hard to hide true success. However you might try to keep your light under a bucket, some will always leak out and people in search of help will always seek you out. Too often, monuments to the organizational ideal, what Norm Kerth calls "pageants," are staged before a cynical crowd of employees assembled on a monthly or quarterly basis under the guise of "information sharing." At a previous company, such extravaganzas were held on a regular basis, within a structure not unlike that of beauty pageants, with winners going to higher levels ultimately to compete for prizes at a corporate-wide contest. Such was the nature of the organizational mentality that managers put terrific pressure on teams to prepare for such competitions, as executive leaders held great store in these monumental productions which trumpeted the superiority of the company's policies. Needless to say, the persons most attracted to such showcases were those least needed by the actual working teams. Unfortunately, these same highly visible personalities were rewarded with frequent promotions into the deep (and rather puzzling) hierarchy of the company, further perpetuating the cult of conspicuous display.

3. Sever Along the Joints

PROBLEM: How do you preserve working relationships and synergy as a team is being disbanded?

CONTEXT: The team being disbanded is successful but does have some natural divisions within it, technically, temperamentally, ideologically, or otherwise.

FORCES:

- The team has done good work and, as a team, does very well.
- There are, however, certain breaks that act to form natural affinity groups within the team, aligned along temperament, technical ideology, or other, unknowable borders.
- You want to preserve those deep interdependencies that will be diminished severely if they are split.
- Business needs dictate redistribution of personnel.

SOLUTION: Divide the team along these natural boundaries.

RESULTING CONTEXT: The best parts of the team are preserved while perhaps eliminating obstacles to better realize their capabilities. This should be used in conjunction with [6] *Let 'Em Choose*, which looks at where to send team members.

RATIONALE: As in the dismemberment of a chicken for the fryer, it's always easier to sever the parts along the natural joints. There will be certain natural groupings and working relationships that develop within a team over time. Sometimes these are created by happenstance, that is, Larry and Moe and Curly all happen to be working on components with common interfaces and so develop a sound and comfortable working relationship. But it may be that Larry and Moe and Curly also happen to form a subteam easily by virtue of complementary temperament, and so an opportunity to put them together on another project might help create a positive force on the new team. To take advantage of this, the team should be severed along these natural joints to deepen the bond between members remaining together in the new endeavor. It is important to recognize and honor these natural groupings, because, let's face it, people like working with people they like to work with, a fact which seems perfectly obvious but is almost universally ignored. Norm Kerth has commented that, "There are many cases where particularly effective natural affinity teams, when broken up, find ways to come back together - often at a different firm. We can not afford to lose such a valuable resource."

4. Pairs or Better to Open

PROBLEM: How can you preserve successful practices and knowledge as much as possible when people are transferred to other teams?

CONTEXT: A successful team is being disbanded for business reasons and its members sent to other teams, some of which are new, others of which are established but taking on some of the organizational and/or technical artifacts from the disbanding team.

FORCES:

- New team members sometimes have problems establishing credibility.
- Knowledge can lose something in the translation.
- People work bolder and better in pairs.

- Two views are better than one.

SOLUTION: Send new members to a team in twos, threes, or better, whenever possible.

RESULTING CONTEXT: Two or more people with shared experience brought into a team are able to communicate in the same language, support each other in team meetings, validate each other's experience, and, as in the pattern *Developing in Pairs* (Coplien) they tend to be willing to take chances and risks in the new environment. Additionally, the new team benefits from having two or more views into the architecture, framework, plan, or whatever seeds were sown prior to the arrival of the members from the disbanded team. Another dimension to this is to pair each experienced person with someone new to the practice, which is an excellent way to spread expertise, as detailed in Ward Cunningham's *Programming in Pairs* pattern (http://c2.com/cgi/wiki?ProgrammingInPairs) from the *Episodes* pattern language.

RATIONALE: Human nature being what it is, there is always credibility in numbers. Two or more people from the same team telling the same story, albeit from different angles, give a more credible and rounded view of things. In team meetings they also work as checks and balances on each other's more hyperbolic contentions, which, in the author's particular case, one may have an overly optimistic bent toward performing. This further gives credibility to the concepts the members bring. Additionally, they bring with them some of the synergy that existed in their previous team. When [5] *Members Follow Model*, this is especially important.

5. Members Follow Model

PROBLEM: How do you best preserve expertise in an architecture, a framework, a process model, or an organizational model?

CONTEXT: A team which originated or used a particular architecture, process model, framework, and/or organizational plan has been successful is disbanding for business reasons. Its members are being reassigned to other teams, some of which may use the same artifacts or processes.

FORCES:

- Many teams need personnel.
- These particular individuals have recent and specific hands-on experience with an architecture adopted by other teams, or in the particular practices that have lead to success.
- Mapping the organization to the architecture is not always clear, obvious, or intuitive.
- Business needs dictate moving personnel.
- You can't afford to have teams relearning or reinventing what is already proving successful.

SOLUTION: Members should, whenever possible, go to teams who are reusing frameworks, practices, or architectures pioneered or used by the original team.

RESULTING CONTEXT: The newly transplanted team members can be of great help in shaping the new team and its distribution of project responsibilities to use the architecture and/or practices most effectively. If you [6] *Let 'Em Choose*, and they choose to follow the model, the evangelical zeal they may bring will be invaluable.

RATIONALE: As a corollary to Conway's Law (*Pattern Languages of Program Design*, Coplien, Schmidt, 1995) which ties organization to architecture, people too are tied to architecture and to specific components through *Code Ownership* (Coplien, 1995). This practice reduces or eliminates kinks in the learning curve, particularly with respect to the pitfalls that may bushwhack a team entirely new to the architecture or practices they have adopted but not used to completion of a product. In our experience, having two team members seeded into another team even after they had adopted our architecture and begun design work gave much benefit in unexpected ways. For example, the new team had organized their work among team members in a way that was less effective for the given architecture. The transplanted team members discussed a redistribution of labor based on their direct and recent experience which made interfaces between components and their developers much cleaner and reduced coupling and improved system integration considerably.

In regard to transferral of architectural expertise, a danger exists when this pattern is applied repeatedly in that people will get trapped as the gurus for an architecture and be unable to free themselves to work creatively again. In fact, it is important to enlist new gurus with every reuse of the architecture in order to spread expertise and free the original promulgators to make new advances. In our experience in the Intelligent Networks group at AG Communication Systems, we have not yet run into this phenomenon. This is probably due to the relative youth of the organization and the still-evolving nature of the architectures with which we work. We are concerned that we often bury our best architects in development work, but as we believe strongly that *Architect Also Implements*, (Coplien, '95) this trade-off must be carefully managed. The primary danger is that the architects are not sufficiently recognized by management.

6. Let 'Em Choose

PROBLEM: How do you maximize the beneficial effect of new members sent to a team?

CONTEXT: An experienced, successful team is being disbanded at a project's end. It is desired to maximize their beneficial effect on the new teams that they will be joining. Several teams need people. Their needs are roughly equivalent, and other sources of personnel are available.

FORCES:

- Several teams may be needing members.
- Some of these teams are reusing the architecture.
- Some of these teams are adopting similar organizational guidelines.
- Personal preferences can affect outcome.

SOLUTION: Let team members, as much as possible, choose their own assignments and with whom they transfer.

RESULTING CONTEXT: The new team members are paired by their own choosing ([4] *Pairs or Better to Open*) and thus would be expected to have a solid and complementary working relationship already. However, the receiving team must be willing to accept these new members; hostile invasions or takeovers are not recommended. Additionally, homogeneity, though comfortable, can magnify common weaknesses, as Norm Kerth has pointed out, and may require some oversight, intervention, or simply, honest self-appraisal.

RATIONALE: This is supported by the notion of *Self-Selecting Teams* (Coplien, 1995). Our own experience also has supported this notion by empowering team members to choose on what and with whom they throw in their lot.

7. Jumper Cables

PROBLEM: When members from a disbanded team join a new team using the same architecture, how can you take the greatest advantage of their expertise in the new team?

CONTEXT: Members have been assigned to new or existing teams which are reusing the architecture of their previous team.

FORCES:

- These new members have the greatest actual hands-on experience.
- Leadership sometimes falls to those who appear as gurus.
- Team leadership is a black hole of energy-drain.

SOLUTION: Transplanted team members who primarily hold technical knowledge should not become team leaders immediately upon arrival to the new team, but rather be regular team members with the special capability to jump start the team with their specialized knowledge and experience. Process leaders or team leaders who are transplanted from successful teams are better candidates, and the team leader role probably supplies the greatest leverage for applying their knowledge. This is not to deny leadership to technical experts, but team leadership can be a full-time task which leaves little time for technical leadership, and would suggest moving technical leaders to team lead positions after they have weathered some cycle time on the new team.

RESULTING CONTEXT: These new members are available in the greatest possible way to get things moving in the project. Because they have been through an entire development cycle using the architecture and perhaps the organizational structure and plan, they can steer a team around the very obstacles they themselves encountered in their previous experience.

RATIONALE: Team leadership very rarely, in our experience, is not a full time job, and does not leave sufficient availability for technical support. Technical expertise is diluted by project responsibilities, although in our teams all team members handle starpoint responsibilities and roles for project support (training, schedule, budget, etc.). Team leadership, however, is too much interrupt-driven by outside forces to permit time for the prototyping, research, and technical involvement required of those members whose technical knowledge is greatest. Additionally, it frees the team leader to concentrate on organizational issues. If the transferral of new team members was the result of the application of [6] *Let ’Em Choose* and [5] *Members Follow Model*, then they will probably prefer to remain in the roles they had previously and playing to this preference maximizes their effectiveness, particularly when schedules are tight.

Epilogue

These patterns have been observed over the past several years across two companies: one largely successful, the other successful at the expense of stability, comfort, morale, and general concern for the well-being of the staff. This suggests a treasure trove of anti-patterns, and in fact many of these have been mined and are scheduled for publication elsewhere. Many others are yet to be written, and my suspicion is that this will be a rich vein for generations to come.

This raises an interesting observation. It has been said that success has many parents, while failure is an orphan. To this we can add, failure has many authors, while success lives in the bones as an unspoken language. Successful teams often contain members who really can't articulate why they are successful, and it often *seems* to depend more on uncontrollable characteristics like personality, interpersonal dynamics, and dumb luck. It is very distressing to think that these factors count for more than those under our control, but perhaps our distress stems from the view that what is not under control is chaotic and destructive. The patterns community, however, seems to have recognized that absolute control is not the goal, but rather seeks the creation of a harmony of the natural forces of human organization with the needs of business to make both profits and promise to those who choose the engineering career.