The 7 C's for Multi-site Reuse Success

Gary Jackoway

Analytical Products Group
Hewlett-Packard Company
2850 Centerville Road
Wilmington, DE 19808
Tel: (302) 633-8762, FAX: (302) 633-8924
Email: gary@lf.hp.com

Abstract

This document describes briefly the key success factors for a multi-site reuse program. The factors are in priority order and have been culled from the results of several years of work on multi-site reuse at Hewlett-Packard's Analytical Products Group.

1. Communication
2. Commitment
3. Control
4. Commonality
5. Compensation
6. Compromise/Consensus
7. Changing mindset

The primary conclusion that can be drawn from this list is that reuse in general, and multi-site reuse in particular is not so much a **technical** issue as it is a managerial, political and sociological issue.

Keywords: multi-site, communication, teamwork

Workshop Goals: learning new techniques; comparing results with those of others in the field; finding potential areas of improvement.

Workshop Groups: reusable component definition and certification; tools and environments; design guidelines for reuse in C/C++; reuse management and organization; reuse process models.

1 Background

Hewlett-Packard's Analytical Products Group (APG) develops applications which acquire and analyze data from a variety of chemical analyzers. The software for these instruments is developed at three different sites around the world; but much of the software is shared among the product lines. More than ten years of work has been done involving reuse at a variety of levels, from leveraging entire applications to developing true reusable components which are shared without change across sites.

2 Position

In our quest to reuse software, we have learned many lessons concerning successful and unsuccessful techniques. The prioritized list that follows details the key factors for our success in multi-site reuse.

2.1 Communication

The key consideration for a multi-site reuse program is communication. If individuals cannot communicate effectively a multi-site reuse program will surely fail. The best way to communicate is person-to-person; second best is video teleconferencing; third is voice teleconferencing; and last is text (electronic mail, FAX, etc.). Person-to-person contacts are the key to a good working relationship. The best way to initiate person-to-person contacts is to first get the full teams together, then allow time for one-to-one associations to grow. It is important that each group see the other group not as "we" versus "they" but as a group of individuals. Another key communication issue is differing cultures, both in the narrow sense of the working methodology at each site and also in the broader sense of the social culture of the location. This is especially true for international reuse programs. One successful international reuse project started with the entire team visiting for one week at one site, then a few months later for one week at the other site; the design was completed during a one month visit to the first site. After these visits, the team was able to operate effectively throughout the implementation phase without further visits.

In dealing with international reuse, the ordering of communication techniques has to be revisited, however. In many cases where some team members are speaking in a foreign language, text moves up to the best way to communicate. Textual communication gives the recipient time to read and translate the information; and it gives the sender an opportunity to make sure the message is clear. Textual communication also allows for archiving of information; in APG, archiving has become more important with the advent of ISO 9000 standards for product development.

2.2 Commitment

Commitment to reuse is the second key point. This commitment must start at the highest level of the organization, but, equally important, must be embraced by all levels within the organization. This is a case of a chain being as strong as the weakest link. If the first-level managers are the weakest link, for example, they will make decisions based on what is best for their project and ignore the effect these decisions have on the other projects. Further, commitment must be equally strong among all sites involved in the reuse program.

2.3 Control

Success in reuse requires controlling the process and products of software development at a level not generally required for single site / single project development. This control takes many forms: documentation standards, interface standards, well-defined component development and maintenance processes, etc. Another key consideration is level of control. It is not acceptable, for example, to set up a scenario where engineering decisions must be made by a third-level manager or higher because that is the only level that is shared between sites. Managers at this level have strategic not tactical responsibilities. For a multi-site reuse program to succeed, technical decision-making authority for the full reuse program must be made by a lead engineer or first-level manager. For the first level manager especially, there is often a sense of loss of control. Before reuse, this manager might have had full control over the product being developed -- all the code going into the product was developed by this manager's group -- but now the manager must depend on others to deliver critical components for the product. This loss of control can be very difficult to accept.

2.5 Commonality

Multi-site reuse is both easier to motivate and easier to maintain if the sites see great commonality among the products being developed. An upper-level manager will look at the organization and wonder "Why am I paying to do the same thing many times over? Can't the organization develop components which can be used by all sites doing this kind of work?". Also, sharing of technology will likely have begun long before the reuse program begins, so there are likely to be informal contacts made among sites. All of this helps start the program.

Beyond just internal commonality, it helps to have external commonality. In APG, for example, our customers often have several of our instruments in their labs. It makes sense that the software looks the same and performs the same across the full family of instrumentation.

It is not sufficient that the organizations have commonality at one moment in time. It is also important that the organizations are headed in the same direction. If one site is, say, moving toward multi-user systems while another site plans to stay with single-user systems, the ability to share low-level components may be hampered by these differences. The single-user organization is likely to find components built by the multi-user organization to be inefficient because the components are attempting to handle situations that simply don't occur in a single-user scenario. The multi-user organization may find components built by the single-user organization to not be sufficiently flexible to handle their more complex needs.

2.6 Compensation

Developing a fair compensation plan is a common problem in reuse; and it is often exacerbated by multi-site programs. It is difficult to spend the extra time and resources to develop reusable components, when this extra effort has value primarily to some far away team. Often no tangible rewards are given. And, having spent the extra effort to develop the component, one is often saddled with the support of that component for years hence. One solution to compensation problems that we have had success with is to have some engineers report directly to the group-level reuse program. For

these engineers servicing the entire organization is their primary job. But perhaps a better solution is to develop reusable components at all sites; while a manager at one site may be paying extra to develop reusable components, that same manager is benefiting from the components developed off-site.

2.7 Compromise/Consensus

To work together successfully, teams must recognize when it is time to compromise on an issue, when consensus has been reached, and when an impasse requires escalation to a higher authority to resolve. The key issue here is efficiency: some areas need little consensus and can be determined independently by one engineer; other areas need careful evaluation and clear communication to make sure the correct decision is made. The challenge is to know what level of interaction is necessary for each problem.

2.8 Changing Mindset

A key thread that runs through all of the above points is that succeeding in multi-site reuse requires changing mindsets. Engineers must become mindful of opportunities not just to reuse existing components but to develop new reusable components. First-level managers must reset their thinking so that, instead of fully controlling the software that goes into their product, they incorporate components into their code stream. Higher-level managers need to support the process of multi-site development, which includes converting from short-term product orientation toward long-term component development, and ensure that the rewards are commensurate with the effort undertaken.

In conclusion, it is clear that achieving success in multi-site reuse requires much more than a good multi-site library tool. Indeed, most of the changes necessary are to the way people behave and interact. Addressing these concerns should be a primary activity in developing a multi-site reuse program.

3 Comparison

In [1], an evaluation of multi-site reuse was done, but more from a technical/network perspective. [2] describes a tool for managing software across a network. Standard reuse references such as [3] which generally spend a good deal of time on management issues, often spend little or no time discussing the extra challenges involved in multi-site reuse.

References

[1] Lewis, J.A. and M.C. Pfeifer, "Multi-Site Software Development," *Proceedings of the International Symposium on Engineered Software Systems*, Malvern, Pennsylvania, May 1993.

[2] Kramer, Scott, 1991. "History Management System", *Proceedings of the 3rd International Workshop on Software Configruation Management (SCM3)*, June 1991, pp. 140-4.

[3] Hooper, James W., 1991. *Software reuse: guidelines and methods*. Plenum Press, New York, New York.

4 Biography

Gary Jackoway is a lead software engineer with Hewlett-Packard's Analytical Products Group. He has a Master's in Computer Science from Duke University. His previous work has included Computer Aided Electronics and Natural Language research.