

Organizational Factors and Reuse

DANIELLE FAFCHAMPS, *Hewlett-Packard Laboratories*

◆ *It is now obvious that reuse is not just a technical issue, it's also a people issue. Hewlett-Packard studied why people sometimes resist reuse and which organizational models appear to encourage reuse more than others. The study found that successful reuse programs must be integrated within the culture of a company's existing organizational structure.*

Over the last 20 years, reuse programs have assumed that technical solutions would overcome barriers to effective reuse. However, recent retrospectives of reuse programs show that organizational factors can greatly affect the implementation of reuse programs. One crucial organizational factor is the relationship between *producers* and *consumers* of reuse components and services. What are these relationships and how well do they work?

To answer this question, I conducted an empirical study of 10 engineering sites at Hewlett-Packard engaged in systematic reuse. My study is described in the box on page 35.

From this reuse experience, I identified four models of producer-consumer relationships; evaluated the

models in terms of their organizational structures, advantages, and disadvantages; and identified goals for management to enable a successful implementation.

PRODUCERS AND CONSUMERS

Reuse is about the production and consumption of reusable components. Over the years, practitioners and researchers have advocated distinct organizational roles for producers — who design and develop reusable components — and consumers — who design and develop products with reusable components.¹ This role distinction has no precedent in a traditional product-focused engineering structure, and we have much to learn

about how to make these new relationships work effectively.

Process analysis is a systematic way to analyze task requirements,² but this technique does not answer managers' questions on how to assign roles to individuals. Decisions about roles may be made at various levels of granularity. For example, questions about the tasks of a reuse librarian or the skills necessary to develop a product with reuse are addressed in the literature,³ but we know little about organizational structures that support effective producer-consumer relationships.

Software researchers have called for an empirical approach to discover the real problems that confront practitioners and have advocated that software engineers become more systematic in reporting what they learn.⁴ Both managers and engineers have stated that many HP sites have accumulated valuable reuse experience, but they agree there is no mechanism for disseminating lessons learned. Thus, individual sites spend precious resources exploring paths others have already mapped and making mistakes others have already made.

In this article, I offer four models that capture HP's experience in estab-

lishing producer-consumer relationships. These models should make it easier to understand the pros and cons associated with each relationship. In the end, however, which model you select will depend on the scope of the reuse program and on the characteristics of your organization, including size, talent pool, and structure. In a *functional* structure, organizational entities group diverse technical expertise. In a *divisional* structure, specialists are grouped into teams dedicated to the delivery of a product. In a *matrix* structure, multiple interdependencies exist among different organizational entities.⁵

As is true of many large electronics companies, HP has a divisional structure. In this hierarchical structure, direct, official reporting relationships are represented with solid lines on an organizational chart. Transitory, temporary, or secondary reporting relationships are represented with dotted lines.

FOUR MODELS

I abstracted four models from data gathered at 10 HP sites: *lone producer*, *nested producer*, *pool producer* and *team*

producer. Table 1 summarizes the characteristics of each model.

The order of presentation does not imply any preference or level of maturity. Moreover, two or more models may occur within a given reuse program. The recommendations to management are based on both current successful practices and interviewees' suggestions. I also include some tentative guidelines on what environments are best suited for each model.

Lone producer. Figure 1 depicts this model. The lone producer provides reuse services to at least two consumer teams. This position has no natural niche on the organizational chart of a division (in which an individual is represented as either a manager or a member of a team, and each individual has only one boss). Depending on the stage and scope of the reuse effort, the primary role of a lone producer is to design, develop, or maintain reusable components. The involvement of lone producers with customer teams also varies from implementing change requests to working with a team in all phases of product design.

Advantages. Lone producers who *maintain* components are typically newly hired, junior programmers who view their involvement with multiple teams as an opportunity both to network and build knowledge. Those interviewed felt they were able to gain more knowledge in this position than if they had been assigned to a specific team.

Lone producers who *design and develop* large components are experienced programmers. They derive satisfaction from the delivery of a complete product—an achievement highly valued in the HP culture.

Disadvantages. This level of satisfaction has definite limits; lone producers tend to view their position as paying dues on the way to a "real" job. The data suggest that three factors contribute to this view:

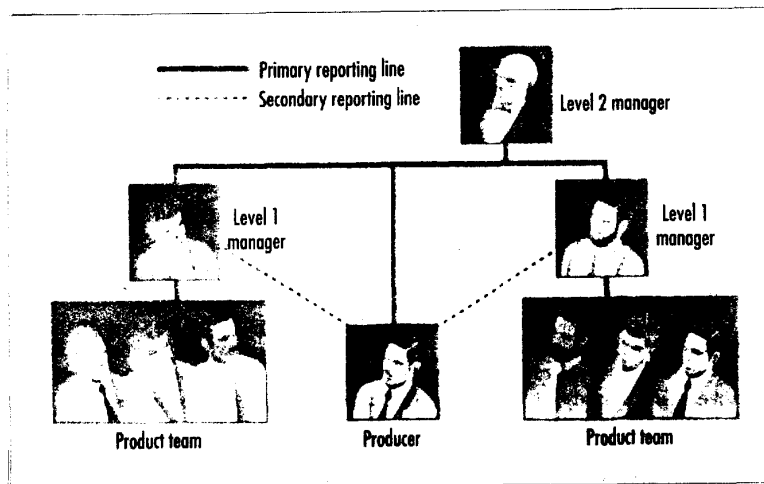


Figure 1. The lone-producer model, in which a single individual handles several product teams' reuse needs. The dotted line indicates a secondary reporting relationship that may involve written status reports without face-to-face interaction.

♦ *Informality and change requests.* It is well documented that the engineering culture has a fondness for informal communication — personal persuasion is commonly used to convince a colleague to modify a design feature or function. For example, one engineer stated, “No, we don’t really have a formal communication setup, or at least I’m not aware of it. Someone just gets an idea about how some feature should be done and then tries to convince others.”

This informality may work well for exchanges within the confines of a product team, but such persuasive tactics impede the effective management of change requests within a reuse program. A major source of frustration for the lone producer is how to answer the steady stream of uncoordinated, often conflicting, change requests from individual members of consumer teams. Lone producers do not have the authority to arbitrate the conflicts, and they end up taking the requests back to each consumer-team manager for approval. Meanwhile, their consumer colleagues perceive them as uncooperative because they do not play by the “informality rules.”

Informality also leads to considerable variance in the consumers’ cooperation with a lone producer. At one extreme, consumers readily blame reuse for any problems encountered during testing and expect the lone producer to produce proof to the contrary. To establish this proof, a lone producer must be knowledgeable about the software involved, have access to the consumers’ hardware, and be provided with the steps that led to the problem.

At the other extreme, highly cooperative consumers notify the lone producer after they ascertain that the failure is related to the reuse component; sometimes even taking the extra step to identify the faulty line of code. With this much variance in communication and cooperation, it is not surprising that some teams unable to meet their deadlines attribute their

**TABLE 1
SUMMARY OF FINDINGS**

Models	Advantages	Disadvantages	Recommendations
Lone producer	<ul style="list-style-type: none"> • Networking opportunities • Broad knowledge base • Product delivery 	<ul style="list-style-type: none"> • Communication overload • Priority setting • Work overload • Communication overload • Multiple reporting lines • No home team 	<ul style="list-style-type: none"> • Mandate reuse • Specify process for change requests • Plan fair reporting relationships • Legitimize the producer role • Define career path of the producer • Clarify evaluation criteria • Teach the consumers the features of the reuse component(s) • Hire the right person
Nested producer	<ul style="list-style-type: none"> • A home team • Product-delivery 	<ul style="list-style-type: none"> • No specialization • Isolation from reuse peers • Double reporting • Negotiation with hardware manager • Dispatched to other task • Resources diverted 	<ul style="list-style-type: none"> • Model does not work well at HP
Pool producer	<ul style="list-style-type: none"> • No changes to the organizational structure 	<ul style="list-style-type: none"> • Communication overload • Unresolved conflicts • Priorities of the different marketing lines • Different time-to-market • Different priorities for R&D and marketing • Documentation 	<ul style="list-style-type: none"> • Have a maximum of three to four teams • Resolve old conflicts • Formalize communication structure • Ensure rapid feedback to change requests • Balance workload among teams • Define clear ownership • Reward team behavior • Select the right people
Team producer	<ul style="list-style-type: none"> • One manager • Team membership • Control over resources • Specialization opportunities • Flexibility • Ownership • Knowledge about other projects 	<ul style="list-style-type: none"> • On call to consumers • Work with too many projects • No complete projects • Communication overhead • Maintenance overhead 	<ul style="list-style-type: none"> • Have a consumer-driven mission • Communication strategies • Support flexible ownership • Select the right people



delay to the inadequacy of the lone producer's contribution.

◆ *Work overload.* The lone producer's involvement can vary extensively from brief, purposeful interaction needed to handle a single change request to full involvement during all product phases (investigation, design, implementation, and delivery). The latter situation has a high potential for work overload. One of our interviewees was expected to be closely involved with four teams!

◆ *Isolation and communication overload.* Organizationally, both product-team managers and the lone producer report to the same level 2 manager. This arrangement lessens team conflicts when priorities are assigned to change requests, but it does restrict the lone producer's access to needed managerial support; dedicated level 2 managers have neither the time nor probably the interest to deal with the needs of a lone producer; level 1 managers must watch out for their own team members and are under tremendous pressure to deliver their product in the shortest time possible. In effect, lone producers are involved in numerous de facto interactions with consumers, and as the number of consumers increases, lone producers quickly become entangled in a web of written reports to numerous level 1 managers (who don't have time for informal meetings), thus increasing their workload even more.

Management recommendations. The successful implementation of this model requires specific interventions at different levels of the hierarchy, depending on the scope of the reuse effort. Managers at the higher levels should

◆ *Mandate reuse.* This applies to all four models. The reuse mandate must come from the correct level of man-

agement: the manager at the apex of the hierarchy of entities involved. For example, if a reuse program involves only teams with level 1 managers, then the reuse mandate should have the *active support* of at least the level 2 manager they have in common. If the level 1 managers report to different level 2 managers, then their common level 3 manager should be involved. Active support means more than paying lip service. For example, one level 3 manager requires that new-product proposals address reuse or they are not allowed to proceed beyond the investigation phase, unless there are convincing overriding reasons.

◆ *Ensure careful planning and shared understanding about goals, costs, and benefits.* Managers and engineers both stress the need for everyone to have a clear understanding of the reuse program.

One engineer described how the code of a well-received product became the focus of a reuse effort: "...even though the code was not written to be reused.... I don't know whose idea it was. All I know is that it was forced down the other sites' throat. We had it, then the other site had it. They added some stuff for their own needs and meanwhile broke some of ours and then gave it to the third site who added some stuff,

broke some of the second site's and some of ours. And so the code was shuffled, round-robin, between the sites."

The reuse plan should include training. Lone producers reported that training consumers about the functionality of the reusable components should be a prerequisite to implementing this model because there often is no time to do so once the reuse program begins.

To support this model, level 1 managers should

◆ *Agree on the lone producer's short-*

and long-term responsibilities, outline a development plan (preferably in consultation with the lone producer), and explain the lone producer's role to all consumers.

When the lone producer is a new-hire, it is imperative that management be very specific about his or her responsibilities and communicate them clearly to everyone. Studies have shown that positions located at the junction of teams are especially difficult because people in these positions have little or no authority to influence the behavior of the members of those teams. New-hires are particularly vulnerable: They must fend off persuasive tactics as they try to become a member of more than one team.

◆ *Manage change requests.* Junior lone producers stressed that they should not be responsible for orchestrating the priorities of the consumer teams. Level 1 managers should negotiate change requests first within their own team, then across teams. This process facilitates cross-team negotiation, decision-making, and alignment. To this end, managers find it useful to exchange their own schedules, plans about specific changes, and individual team priorities.

◆ *Establish a formal communication structure.* This applies more to new employees than experienced programmers, who prefer a more active role in establishing communication among teams. One experienced lone producer, for example, developed a process for helping his consumers align their priorities: He organizes brainstorming sessions and compiles a list of the inputs for domain-application experts, who assess the feasibility of the requests. Then he sorts the requests into broader categories and briefly explains their technical feasibility, necessary resources, schedules, and alternatives. The consumer-team managers use this annotated list to align their priorities.

◆ *Encourage informal interaction.* A lone producer can become isolated. Lunch meetings of the lone producer and consumers are excellent informal venues that facilitate relationships.

THE LONE PRODUCER CAN QUICKLY BECOME ENTANGLED IN A WEB OF REPORTS TO VARIOUS MANAGERS.

ACTION-RESEARCH: BASIS FOR RESULTS

This study is based on *action research*, an approach that relies both on data from real experience to focus data collection and on theoretical models to formulate strategies for action. Working hypotheses guide the research, and findings are validated iteratively. In this collaborative enterprise, both practitioners and researchers contribute to and learn from the experience.¹

The data for this study were collected between December 1992 and October 1993 in divisions that had practiced systematic reuse for at least one year and as much as seven years. Table A shows the interview data. Formal interviews lasted between one and two hours and some individuals participated in multiple interviews over 10 months. In addition to transcripts of interviews, the data includes organizational charts, documents produced in the divisions, and reports that members of the Software Reuse Department at HP Labs² wrote following visits to the divisions, phone conversa-

tions, and e-mail exchanges with division members.

I used *theme analysis* to organize the qualitative data. In the tradition of anthropological research,³ theme analysis refers to the tracing of recurrent patterns of human activities across space and time. By design, qualitative data is rich and varied, and here *role theory*⁴ provided the overarching theoretical framework that guided

and focused the selection of themes and patterns.

Role theory is a field of inquiry that has contributed to the understanding of roles in a wide variety of professional environments over the last 30 years. The results of the analyses were presented to HP and non-HP audiences to validate the findings with professionals who had not been included in the pilot groups.

REFERENCES

1. W. Whyte, *Participatory Action Research*, Sage Focus Edition, Sage Publications, London, 1991.
2. M. Griss, "Software Reuse: From Library to Factory," *IBM Systems J.*, No. 4 1993, pp. 1-23.
3. O. Werner and M. Schoepfle, *Ethnographic Analysis and Data Management: Volume 2*, Sage Publications, Beverly Hills, Calif., 1987.
4. R. Katz and R. Kahn, *The Social Psychology of Organizations*, 2nd ed., Wiley and Sons, New York, 1978.

**TABLE A
FORMAL INTERVIEWS**

Divisions: Pilot groups	Sites	Formal interviews
Instruments: A	US	6
	US	3
	Overseas	10
Instruments: B	US	6
	Supports	1
Peripherals	US	No formal interviews but informal interactions and artifacts
	Overseas	1
	US	1
Communication	Overseas	1
	Overseas	1

◆ *Choose the right person.* A lone producer must survive at the junction of many teams and so must possess excellent communication skills, an ability to handle conflict, and technical competence.

The easiest solution is to assign the lone producer to one team, but this will work only if level 1 managers trust each other to make selfless decisions, and if the criteria for prioritizing change requests are clear.

When to use the model. This model works best when a few large reusable components are available for a division's different products and one lone producer maintains the components for each consumer site. Depending on the stage and scope of the reuse program, decision-makers should carefully consider the types of skills needed

to fill a position that at one extreme may require the design of new reuse components and at the other may simply require the management of a component library. For example, a trusted, experienced programmer with effective communication skills would be best as the *designer* of reuse components, yet few programmers would be satisfied as librarians. One HP site trained a secretary, who was exploring ways to diversify her responsibilities, to be the reuse librarian. By all accounts she did an excellent job.

Nested producer. Figure 2 shows this model, in which each product team has a member dedicated to providing reuse services and expertise. In addition to directly reporting to a product team manager, nested producers may have a secondary reporting relation-

ship to a reuse manager located on site or at another site, if the reuse program spans multiple sites. The nested-producer model is inspired by the firmware-team structure, in which each hardware team has its own dedicated software engineer.

Advantages. Nested producers value their membership in a home team and the opportunity to contribute to the completion of a product.

Disadvantages. Various problems may arise from this team membership:

◆ *Negotiations.* Reuse firmware engineers may get caught in tough, time-consuming negotiations with hardware managers who sometimes lack adequate software knowledge.

◆ *Resource diversion.* During critical phases, level 1 managers may divert



reuse efforts and resources toward the solution of critical software needs.

♦ *Unsatisfactory reporting relationships.* The reporting structure in a nested model is unsatisfactory to both nested engineers and to their managers. Nested producers report they feel isolated from each other, despite a reporting relationship to a reuse manager, and must pursue intensive interpersonal communication to link with the reuse community. For example, at one site, nested reuse engineers have a reporting relationship to a part-time reuse manager who coordinates the reuse work but who has other responsibilities as well. The nested producers have regular meetings with their reuse manager but they do not feel their interests are well served because the manager is pulled in too many directions and does not have time to adequately pursue interactions with other relevant reuse efforts.

On the other hand, reuse managers feel that there is not much they can do

to alleviate the situation. As one remarked, "In several situations, I was jerked on the dotted reporting lines until they broke." The position of reuse manager is ambiguous with regard to other divisional managers. For example, a product team manager counteracted the assignment a reuse manager had given to a nested producer. As the nested producer of a multi-site reuse program noted, "We had problems because we were still owned by our site and the reuse manager did not have ultimate authority to say 'You work on this today and you're not going to do anything else but the reuse component.' My primary reporting relationship was to my team manager. It was an interesting situation: I was kind of reporting to the reuse manager and working for the team manager and sometimes I didn't know which hat to wear."

Management recommendations. The engineers and managers did not have any

suggestions as to how to improve this model except by requesting that all commitments for reuse work be made in writing. At one site, the problems were solved by pulling out the nested engineers and forming a team-producer model.

When to use the model. We did not find this model very beneficial.

Pool producer. Figure 3 shows this model, in which two or more teams collaborate to both produce and share components. This collaboration takes place without disrupting the organization's structure.

Advantages and disadvantages. Pool producers value their ownership in the product, the clear reporting structure, membership in a stable, identifiable group, and a recognizable organizational niche and identity. However, the communication overhead is high, especially in the early stages of the collaboration, when the potential for conflict is greatest.

Communication intensity increases as you increase the number of teams and product lines involved and the number of marketing representatives, especially when there is a distinct marketing function for each product line. This model increases the need for complete documentation, especially if producers and consumers are geographically remote.

Finally, unresolved conflicts among consumer teams can seriously impede the effectiveness of this model, especially if reuse is mandated instead of being a grass-roots effort. In one case, teams located across multiple sites reasoned that collaboration was technically infeasible, formed alliances along lines of friendship, and fragmented the division's reuse effort. As one reuse engineer noted, "A battle has been going on for the last two years. Should their site follow the rules of the other sites or should we follow their rules? Everybody is saying, 'I do it better than you, so you better do what I do.'"

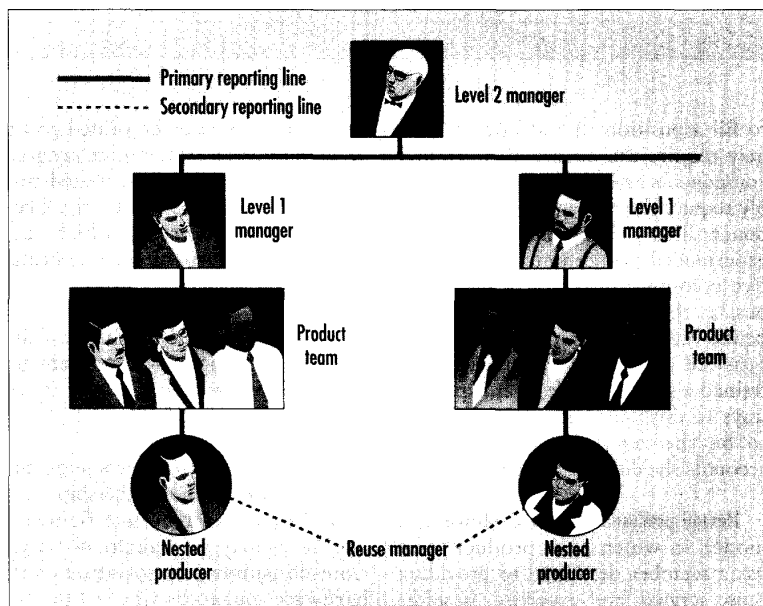


Figure 2. The nested-producer model, in which each product team has a member dedicated to providing reuse services and expertise. The dotted line indicates a secondary reporting relationship.

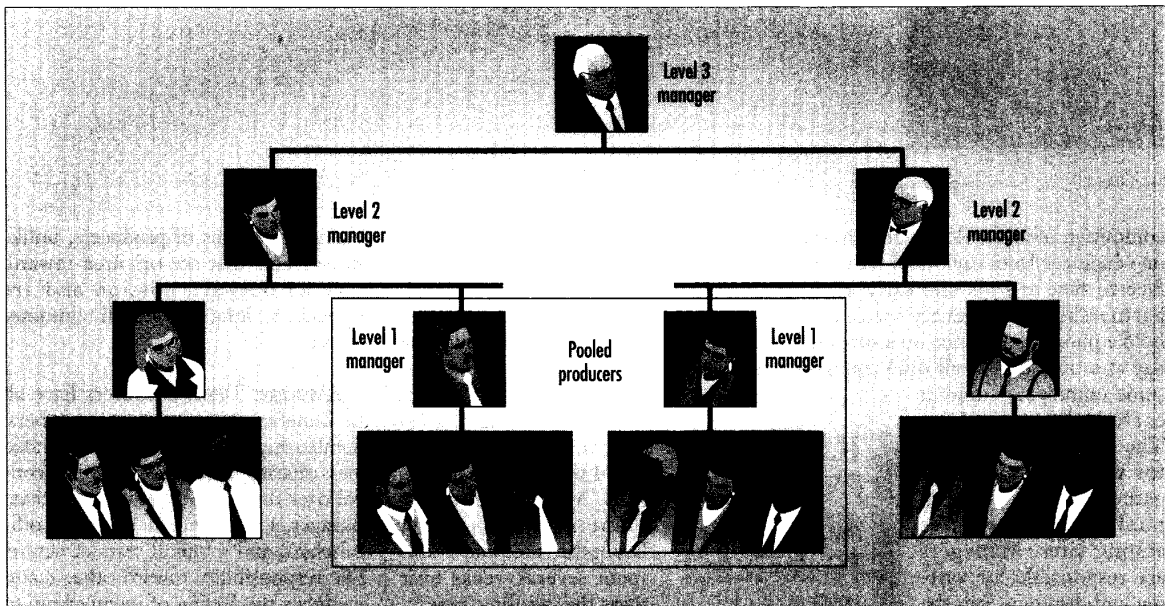


Figure 3. The pool-producer model, in which two or more teams collaborate to both produce and share reusable components.

Management recommendations. The commitment of the right high level of management is especially important for the pool model because the cooperating teams may be located in different sites across different geographical areas.

Managers in charge of this kind of model should

- ◆ *Articulate a vision.* Pool producers work best when they have a common vision that articulates their synergy. Vision is not just a reuse plan. It is a clear articulation of how reuse will help them better address customer needs. Customers can help, as one engineer described: "Reuse got off the ground when we went to our biggest customer. He drew a picture of what we were trying to do on the blackboard and said, 'This is what you ought to be doing!' That did it! [Your customer needs] will keep you together even when the going gets rough."

- ◆ *Negotiate to align priorities.* The asynchrony of product teams' schedules is a major hurdle in all reuse programs. Teams working within a pool-producer model, however, are much more interdependent than teams working in other models. Aligning priorities is a real challenge — managers must be willing to spend much time in negotiations. As a level 1 manager said, "The collaboration required much detailed work and many compromises

between the other managers and myself to let one product go to market without hindering the other teams. The 'ABC' project had to develop its own codes to some extent, keeping the interface and the code stable while we were making the release. And after their release we began to talk about how to remerge the asynchronous parts of the components. Now our product will be released by October of next year, 'ABC' wants to release in the fall of next year, and 'DEF' wants to release by December of next year. So we are in a very synchronous development phase for the component."

Managers reported that sharing project plans, documentation, and to-do lists facilitated understanding of each other's constraints, and hence improved negotiations and synchronization.

- ◆ *Clarify ownership and responsibilities.* The pool producers experimented with different ways to structure interactions. They found the best solution is to attribute ownership of components. Shared ownership among members of different pool teams facilitates buy-in and ensures that all teams feel their interests are well represented. Component owners are responsible for designing reusable components, managing change requests, and informing others. Owners do not have to consult all teams for every decision;

they have leeway, as long as they reach a consensus for their own area of ownership.

Component owners have found it good practice to submit their work for review by other experienced programmers on their respective teams, as described by an engineer: "We had two documents ... and we asked John to review them. He knew both documents but he was not biased by authorship. We included his suggestions then submitted the proposal officially, and it was adopted."

- ◆ *Manage change requests.* One condition cited by level 1 managers is "a very fast understanding of what the priorities are and a very fast feedback from the component owners about what could be done or could not be done.... What has really jumped out in the last two or three years in our development process is that a good communication process is one of the most important things not only within each team but across teams — especially when you work over the ocean."

Pool producers manage change requests formally, via a status document that is checked into a revision-control system accessible to all team members through e-mail. Managing change requests works best when communication across teams is structured. For example, in one division with sites on different continents, a



committee of all level 1 managers and one engineer from each pool team met face-to-face at each site early in the partnership. Then every week they hold a phone conference on a specified day at a time that fit all time zones — some employees come in early; others stay late. The engineer responsible for the agenda and for gathering input from all pool teams alternates among sites. Managers are responsible for writing and disseminating the minutes. Topics include changes to existing products, reasons for the changes, and the effect on schedule and resources.

Finally, the committee meets at alternating sites twice or three times a year to negotiate priorities face to face.

◆ *Establish a conflict-resolution strategy in advance.* Conflicts among teams are inevitable and can seriously impede the cross-team collaboration necessary in the pool-producer model. Before mandating reuse, management should set up a conflict-resolution strategy if the potential pool producers have even the slightest history of conflict.

◆ *Invest in face-to-face meetings.* Even when there is no history of conflict, future pool producers should invest in an initial face-to-face encounter. The benefits are well worth the upfront costs. As one engineer said, "I tend to have much better communication with the persons I have met and perhaps been out to dinner with. I think it has to do with my mental image of them and what they're like and how they like to communicate. It is difficult to build a relationship over e-mail and for people to express more than simple questions — to actually express what they need — their difficulties."

Engineers seem to want to be involved only after the relationship has been worked out at the managerial level, but they do like to be kept informed of progress, through trip reports or updates given at regular

project meetings.

◆ *Plan for informal interactions.* Carefully planned and structured interactions are important, but an effective relationship also involves informal interactions. Letting engineers get to

know each other pays off, they said: "We have good interactions because we spend time at each other's sites. My colleague Bob spent a month at the European site, and I also spent several weeks over there the last three years.... We recently had an engineer from Europe. He gave a few presentations, went to a meeting, and he had the time to just work.

They work just as they would at their home site. It is that level of feeling that you are one team that is essential for success."

◆ *Select component owners carefully.* The judicious selection of component owners is an important factor. Choose experienced programmers from each pool team that other engineers trust to make the right decisions.

When to use the model. This model works best when the reuse program has a limited scope, such as the sharing of a few selected components common to the products of a small number of teams. The model seems ideal as a grass-roots effort to minimize work redundancy among teams at the same division. It also appears to work well for a pool of two or three teams. The communication and coordination overhead become much too taxing for more than three teams. The model is *not* suited to the implementation of a long-term designing-with-reuse strategy.

Team producer. Figure 4 shows this model, which also fits smoothly into a divisional organization's structure: the producer team occupies the same organizational niche as other teams; producers have a level 1 dedicated software manager who is equal to other level 1

managers. Teams of producers, unlike other teams that are oriented inward, have an outward mission and are expected to interact with all consumer teams.

Advantages. Team producers have all the benefits of typical team members: membership in a group of like-minded peers sharing a common culture, control over their own budget, a dedicated manager, a legitimate niche on the official chart, and a typical team structure. The responsibility toward other teams heightens the feeling of membership to the whole division.

Team producers value the opportunity to specialize, believing that specialization increases productivity and helps in timely, competent responses to consumers' requests. Ownership is gratifying but not confining because the internal dynamics of the group allow both flexibility and the amiable negotiation of roles over time to accommodate individual preferences.

Disadvantages. There are, however, two categories of risk that could alienate producers from consumers, ultimately causing the reuse effort to fail. First, producers might withdraw to create the perfect solution, but consumers will reject a perfect solution that does not answer their needs.

Second, experienced programmers are unhappy with some consumers' perception that the producer's role is to solve problems on demand rather than contribute to product definition, which is perceived to be more fun, more challenging, and carrying a higher status.

If producers work on many components, maintenance can become overwhelmingly time-consuming and further restrict individual producers from understanding the work of their respective consumer teams. A team producer remarked that during the investigative phase of a new team, people constantly meet and talk. "You can't come in once a week and say, 'Hey, how's it going?' You must participate; it's a full-time job, even though you're

TEAMS OF PRODUCERS ARE EXPECTED TO INTERACT WITH ALL CONSUMER TEAMS.

not as efficient; you're not writing code."

As with the pool model, the asynchrony of consumer teams' development stage is a real problem for the producers, who must manage and track the different versions of reuse components. One engineer said this adds complexity: "When you develop a component for only one product, you just keep track of goals and timing for that product, but when you have more customers you have to compromise. It's a new type of problem."

Management recommendations. Reuse will not happen simply because a terrific component is available — management must demonstrate and demand commitment. The success of this model depends on the commitment of high-level management. A team producer explained it this way. "There were no technical reasons why they should not reuse. They produced an instrument that looked surprisingly similar to ours. At that point managers were neither supporting nor impeding. They just let things evolve on their own."

Managers in charge of this model must

- ◆ *Demonstrate their full commitment to reuse.* Again, this is important in every model. Here, managers must advertise the creation of the new team, describe its responsibilities, and emphasize its consumer-driven mission. As with the lone producer model, managers should facilitate the alignment of priorities within and across consumer teams and with marketing.

- ◆ *Choose members carefully.* Team members should be respected programmers who are highly skilled in communication and interpersonal relationships, experienced in the functional area, well-versed in software techniques (because reusable solutions require advanced techniques), and committed to industrial product requirements (not research).

The level 1 manager in charge of a producer team should

- ◆ *Keep the team focused on consumer-*

driven solutions. A product-focused team environment fosters inward interaction rather than cross-team collaboration. More so than in other models, team producers must be diligent to keep active communication links with the consumers.

- ◆ *Let consumer teams set priorities.* This is the golden rule of team producers, and it is a highly effective strategy that will ensure continuous interactions among consumer teams and minimize conflict. As one team producer put it, "The consumers often disagreed as to whose problems should be worked on first, but the producers did not take sides. The resolution had to work its way up the management tree."

Team producers may not set priorities but they do manage priorities. For example, at one site the team asked consumers to rate the importance of reusable components on a scale from 1 to 10, as well as the quality of the service they were providing on a scale from 1 to 10. The results, one team producer said, "were quite educational. They changed our priority pegging. We hadn't realized that the performance of the compiler was a big issue for our consumers. Well, we knew it

mattered, but hadn't realized that it was something that bothered them quite as much as it did. [In] the survey, they'd ... said, 'Yes, this is really bad.' So we addressed it. We had not addressed this before because it was never regarded as more important than adding new functionality." To minimize this type of risk, one team producer keeps two lists for different types of change requests: enhancements and defects.

- ◆ *Provide sample solutions.* Consumers are more receptive to reusable solutions when they are supplied with an example solution that is relevant to their domain. A useful sample covers a set of possibilities that might occur in real life. A good sample "jump starts" consumers to adapt the scenario for their own solutions. They do not start from scratch; they do not have to read manuals. Instead, they begin with a running system and improve it incrementally. Some producers have learned painfully that domain relevance is key. One noted that another engineer "created a game as a demo. Management looked and said, 'that's a nice game!' and that was it. They didn't appreciate what they were seeing. Now I have to

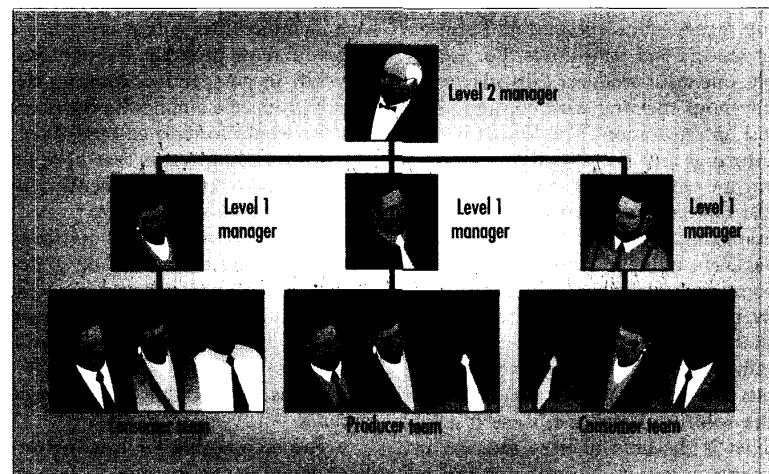


Figure 4. The team-producer model, in which the producer team occupies the same organizational niche as other teams. That is, producers have a level 1 dedicated software manager who has a rank equal to other level 1 managers.



say that he picked a very bad example. It looked like a game, so that's what the managers saw."

- ◆ *Provide documentation.* Consumers should have access to a complete up-to-date documentation of a component's functionality and how to use it. One team uses manuals very effectively. They release selected sections systematically to those who need them and make the entire document available to everyone on demand. Documentation can be provided electronically — Unix users have adapted man-page retrieval capabilities. One team encourages consumers to provide feedback on the quality, clarity, and coverage of all the documentation.

- ◆ *Encourage informal feedback.* Persuasive tactics can really make life difficult for producers. Team producers are perhaps less vulnerable to these tactics than lone producers, nevertheless, experience suggests that a formal defect-tracking system is very important to handle specific problems.

But problems that cannot be clearly articulated should not be ignored. These problems are best handled during face-to-face encounters between producers and consumers. For example, one team producer calls a biweekly one-hour meeting with consumers to just air frustration. The format is very informal and the agenda is open to anyone who wants to contribute. "We just talk and see what comes up. For example, people may be annoyed with variable names, awkward language constructs, functions they have to call over and over again in difficult ways or that require a lot of typing. These things perhaps aren't intuitively obvious to the programmer who engineers the solution but doesn't use it a lot. That is the kind of issue that we are trying to get at. No minutes and no note-taking. But if an issue is recognized as important, there is verbal agreement that we will change the sys-

tem, and it will be recorded in the defect-tracking system later."

- ◆ *Reuse by wandering around.* Team producers should actively identify and seek out individual consumers who are perhaps less vocal at meetings. The goal is to just sit down and talk with them, to discuss how they are getting along and what their problems are. This is a good way to deal with consumers who are not comfortable expressing their opinions in group situations.

AT HP, WE FOUND THAT THE TEAM PRODUCER MODEL SEEMS TO BE BEST.

When to use the model. For us, this was the most successful model and one that appears to have the most potential for the successful implementation of a long-term reuse strategy. This model scores well on the three dimensions of an organizationally defined role, as

described by John Van Maanen and Edgar Schein:⁶ membership in a legitimate organizational niche, a clear career path, and a home functional area.

A valuable side-effect of this model is the transition from a project to an organizational frame of mind. Our division partners and the literature⁷ have identified this shift as crucial to a successful reuse program. To reinforce this shift in perspective, rotating the role of producers among the members of a site would foster a culture of collaboration among teams that have no previous experience or incentive in sharing and working together.

One of our instrument divisions used this model to successfully redefine the producer-consumer roles; seven products now on the market use the same large reusable solutions. Producers

- ◆ Focus on providing reusable code modules.
- ◆ Are responsible for total instrument design or architecture.
- ◆ Conduct interviews with end-users.
- ◆ Are the official marketing contact for software that spans the product line.

- ◆ Achieve job satisfaction by "seeing technical solutions applied across multiple instruments."

Consumers

- ◆ Take marketing inputs from the producers, and thus have more time for examining design issues.

- ◆ Use the code and solutions from the producers.

- ◆ Achieve job satisfaction by producing higher quality instruments.

The inability to effect product definition is a major source of frustration for producers. This role redefinition successfully attributes responsibilities for product definition to both producers and customers, although this instrument division had the advantage of dealing with only one marketing line.

BEYOND NIH

Reuse practitioners and researchers emphasize the judicious separation of producer-consumer roles as a key factor in the success of a reuse program. Information about this specific aspect of a reuse strategy, however, is scattered across retrospectives of reuse programs and has never been addressed systematically. Without labels and a common understanding of the available models, it is difficult to identify and compare reuse programs along the structure of producer-consumer relationships and to articulate why resistance appears more acute in some settings than in others. It is even difficult for HP product teams to learn from each other.

However, HP divisions constitute a good empirical research base because they practiced reuse for many years with various models and because their experience includes reuse programs across geographically dispersed sites.

This study provides new insights about resistance to reuse, which is too often summarily captured in the expression, "not invented here." This expression attributes resistance to an individual's reluctance to change. And when the individual is perceived as the

problem, individual solutions are sought. This study shows that NIH inadequately captures the phenomena of resistance to reuse and suggests three alternative explanations. Managers and engineers should plan how to address each of these when deciding which model to implement.

Collaborative praxes. Resistance to reuse is a symptom of weak collaborative praxes inherent in the divisional structure, in which specialists are grouped into teams dedicated to delivery of a product. This structure effectively promotes intrateam collaboration but hinders the cross-team collaboration necessary for successful organization-wide improvement programs. Within this structure, each team is a fiefdom with its own turf and control over its own fixed resources. The team is rewarded only for the production and delivery of a product. Hence, activities that do not contribute to the immediate goal of delivering a product or that divert the team's fixed resources are threatening.

Expectations. Professionals working within a divisional structure have basic

expectations about the nature of their work, including membership on a team, one boss, rewards for contributing to a product, a clear notion of acceptable communication paths, and a career plan. The findings suggest that increased resistance is related to the disruption the reuse program causes to the preestablished organizational structure.

That is, the more a reuse program disrupts professionals' expectations, the more resistance it will encounter. For example, within HP resistance occurred when producer-consumer relationships involved new types of membership (the lone producer), dual reporting relationships (the nested producer), or numerous interactions among teams who do not normally need to communicate (the pool producer).

Reuse scope. Reuse is often a key business strategy but decision-makers' eagerness to implement reuse should follow a careful analysis of the reuse scope. As one engineer said, "Group management had a very worthwhile goal. The product had been well received on the market and group

management decided we were going to reuse the code even though it had not been written to be reused. So the code was shuffled from site to site, each site destroying the other's changes."

The four models I have described can serve as a framework for comparing our experiences, with accounts of reuse programs elsewhere. They have provided us with a terminology to frame, label, talk about, compare, and learn from our reuse experiences.

I do not claim that these models exhaustively cover every producer-consumer relationship available in a divisional structure. I do claim, however, that adopting the latest technology will not guarantee the success of a reuse program.

A successful reuse program begins with a careful understanding of an organization's structure, the culture that structure fosters, and the strain the reuse program may impose on the culture. Whatever consumer-producer model you choose to implement a reuse program, you must first understand your own environment and address the relevant people issues. ♦



Danielle Fafchamps is a member of the technical staff at Hewlett-Packard Laboratories. Her research interests include work-flow analysis and the study of professional cultures to support both system design and organizational change.

Fafchamps received a licence (MA) in psychology and education from the University of Liège, Belgium and an MA in curriculum design and evaluation and a multidisciplinary, self-designed PhD in socio-technological studies, both from Stanford University.

Address questions about this article to Fafchamps at HP Labs, 1501 Page Mill Rd., Palo Alto, CA 94304; fafchamps@hpl.hp.com.

ACKNOWLEDGMENTS

I am grateful to the HP engineers and managers who found time in their crowded schedules to describe and reflect on their experience and candidly express what works and what should be improved. I thank Bill Frakes and the anonymous referees, Derek Coleman, Patricia Collins, Pankaj Garg, Martin Griss, Reed Letsinger, James Navarro, and Mark Simos for comments on a draft of this article, and Kathe Gust for library support.

REFERENCES

1. M. Griss, J. Favaro, and P. Walton, "Managerial and Organizational Issues" in *Starting and Running a Software Reuse Program*, W. Schäfer, R. Prieto-Díaz, and M. Matsumoto, eds., Ellis Horwood, New York, 1994, pp. 51-78.
2. B. Boehm, *Software Engineering Economics*, Prentice-Hall, Englewood Cliffs, N.J., 1981.
3. R. Prieto-Díaz, "Making Software Reuse Work: An Implementation Model," *Software Eng. Notes*, July 1991, pp. 61-68.
4. B. Frakes, *An Empirical Framework for Software Reuse Research*, Case Center, Syracuse University, N.Y., 1990.
5. R. Jones and R. Deckro, "The Social Psychology of Project Management Conflict," *European J. Operational Research*, No. 64, 1993, pp. 216-228.
6. J. Van Maanen and E.H. Schein, "Toward a Theory of Organizational Socialization," *Research in Organizational Behavior*, Vol. 1, 1979, pp. 209-264.
7. V. Basili and G. Caldiera, "A Reference Architecture for the Component Factory," *ACM Trans. Software Eng. and Methodology*, Jan. 1992, p. 53-80.