Editor: Suzanne Robertson ■ The Atlantic Systems Guild ■ suzanne@systemsguild.com

Requirements by Collaboration: Getting It Right the First Time

Ellen Gottesdiener

We know that we must involve all the stakeholders if we want to discover a project's requirements. But we need some guidelines on how to involve the right people and, given how busy everyone is, how to minimize the time and maximize the result. In this column, Ellen shares her considerable experience running requirements workshops. What are your experiences in running collaborative sessions? —Suzanne Robertson

There is plenty of hard and anecdotal evidence that the most critical factor for your product's success is your stakeholders' involvement in defining requirements.[1,2] Put another way, insufficient stakeholder involvement is often cited as causing erroneous requirements and project failure.

Why do we often fail to exploit this rich resource? As a longtime requirements facilitator and consultant, I think I know. Project stakeholders—those who affect or are affected by the software—have diverse and even conflicting needs and viewpoints that complicate the already slippery slope of discovery and ambiguity that often characterizes requirements development. In other words, getting stakeholder input is hard, and we often don't know how to do it effectively.

A technique I've used many times in various industries and types of organizations is to hold collaborative workshops. You can use workshops for many purposes—for example, to outline the project's vision and scope, to create a release strategy or iteration plan, or to define requirements in varying levels of detail. The beauty of this technique is that it creates an efficient, controlled, and dynamic setting where you can quickly elicit, prioritize, and agree on a set of high-quality project requirements.[3,4] This column explains a key aspect of collaborative requirements workshops: identifying who should participate and outlining their roles.

Calling all stakeholders

The first step is to make sure you've identified everyone who should participate in defining your requirements. It's helpful to think in terms of broad categories based on how people relate to the software. Each group's members have different but valid perspectives on both functional and nonfunctional requirements. Armed with the following list, you can plan ways to deal with the strengths, weaknesses, needs, and questions your stakeholders will bring to the process.

Project sponsor

The project sponsor funds the project and ensures that the resources are allocated. The sponsor should ensure that the product meets business objectives and that the project is fin-

0740-7459/03/$17.00 © 2003 IEEE

ished as quickly and cheaply as possible. Typically, the sponsor understands the larger business goals but is not close to the software's details, nor does he or she fully understand technical issues and user concerns.

Product champion

The product champion acts as an ambassador user, ensuring that the needs of various direct user communities are met (you can have multiple champions). His or her title might be business manager, project manager, or product manager. The champion wants his or her constituent group's real needs to be met and might not fully understand other users' needs. Also, the champion might not have authority to sign off on the final requirements.

Direct users

Direct users are the people (or even other software or hardware) who come into direct contact with the software. They want the product to meet their real needs, such as simplifying or facilitating their jobs. They might not know or care about the "big picture"—how your project fits into the organization's business strategy—but they have abundant interest in (and information about) the details. At the same time, they might resist the change the project implies, including loss of their jobs.

Indirect users

Indirect users come in contact with the system's products or byproducts. They receive reports or files that the software generates, or the software's decisions affect them. Like direct users, indirect users might not understand the big picture nor details such as how data must be loaded and cleaned. They might not know how direct users employ your system.

Advisors

Advisors don't see or use the system but might have relevant information, such as business rules and policies. They might be involved in training, support, or installation. Advisors are often overlooked in requirements efforts.

Table 1

Requirements workshop roles

Role	Responsibilities
Workshop sponsor	Authorizes and legitimizes workshop
Facilitator	Plans and designs workshop Recommends requirements deliverables Leads process
Content participant	Creates workshop products
Recorder	Records the group's work
Observer	Listens and learns
On-call subject matter expert	Is available to answer questions

Suppliers

Suppliers design and create the software. They include developers, architects, engineers, testers, QA specialists, and vendors. They understand existing systems and the technical constraints and possibilities, and they must understand the requirements early so they can move quickly to design, test, code, and install. They might be blind to business needs or enthralled by technology. They might not understand users' needs.

How collaboration works

One key to an effective collaborative workshop is to systematically, resolutely—dare I say stubbornly—elicit and then resolve everyone's concerns.

Your requirements workshop will need a facilitator to plan and lead the work and a recorder to record the workshop contents. Table 1 shows a list of specific workshop roles that stakeholders might represent.

The workshop's sponsor might be the project sponsor or a product champion. He or she might kick off the workshop and return afterward for a "show-and-tell" (when the team presents the deliverables). If the sponsor has rich domain knowledge, he or she might even participate, adding a lot of expertise if you're, say, developing the requirements' scope and creating an iteration plan for a new product line.

Some subset of direct users will also be workshop participants. Those new to their roles might be workshop observers. In one workshop I facilitated, the newly assigned business analyst sat in to learn about the new functionality the group was defining for an application she would soon use daily. Advisors with experience in the business domain might also participate. In other cases, an advisor benefits from observing. For example, a product trainer observed one requirements workshop, which gave him an early heads-up about the new and revised requirements for the company's flagship product.

Indirect users might also participate. Business analysts often attend my workshops because they need data feeds from the application under development. Suppliers might observe and even help plan the workshop. In most workshops, we include developers and IT analysts with deep applica-

One key to an effective collaborative workshop is to systematically, resolutely—dare I say stubbornly—elicit and then resolve everyone's concerns.

Table 2

Forming, storming, norming, performing, and adjourning

Stage	Workshop facilitator actions
Forming	Kick off workshop with sponsor
	Use ice-breakers
	Focus on ground rules and deliverables
	Model desired behavior
	Provide direction for group activities
	Give participants simple tasks to complete
	Build in awareness mini-training or awareness presentations
Storming	Stay calm and neutral
	Capture issues and concerns
	Apply and enforce the agreed-upon ground rules
	Acknowledge and address conflict
	Invite input and feedback
	Mirror or paraphrase disputes
	Provide observations of "storming" to the group
Norming	Back off on structure and explicit directions
	Allow adjustments to the process
	Ensure even distribution of work, contribution, and discussion
	Actively seek feedback on the process
	Give participants more complex tasks
	Reinforce positive behavior
Performing	Provide all information requested
	Inject celebration frequently
	Rotate roles in small group activities
	Allow members to volunteer for workshop tasks and follow-up
	Back off and challenge the group to figure things out
Adjourning	Explicitly conduct a workshop debrief
	Allow sufficient time to adjourn
	Promote appreciative inquiry, allowing members to review effective behaviors and actions
	Use rituals from project retrospectives[6]

tion knowledge as full participants. Some organizations train and mentor analysts or project managers to be workshop facilitators if they can be neutral to the workshop outcome.

Working the workshop

Structure the workshop around a set of agreed-upon deliverables. For example, if your workshop's goal is to create a document describing the project vision or charter, you might plan to deliver a list of features, a proposed sponsorship organization chart, or a product vision statement. If you're looking for scope or high-level requirements, you'll deliver a context diagram, a conceptual domain model, or a set of high-level use cases. If you're defining detailed requirements, you'll deliver statecharts, business rules, detailed use case steps, prototype screens, and so on. A workshop planning team participant or the facilitator briefly explains each deliverable; the facilitator leads the group through the process and sees that everyone agrees on the final product.

It's crucial to work iteratively. You explain everyone's role ahead of time and assign "prework" based on individual roles; you also plan to complete certain tasks ("post-work") after the workshop. That way, everyone is focused on collaboration, which accelerates the time the group is together. During the workshop, you create your deliverables step by step. You can use various techniques based on well-known rules of group dynamics, such as the iterative process referred to as "forming, storming, norming, performing, and adjourning."[5] Table 2 describes how the workshop facilitator makes the process work smoothly for each stage.

Forming

Forming involves orienting the group and explaining various options. The group is just coming together and feeling each other out, so the members produce few measurable results. However, they do identify tasks, establish ground rules, socially relate, give advice, and evaluate each other.

Storming

During storming, group members challenge each other's roles and dispute and debate the process. Conflict and diverse opinions emerge along with role confusion, rebelling, and concerns about team and individual responsibilities. Overt and covert hostility, power struggles, rebelliousness, and attacks on participants or the facilitator can occur.

Norming

Norming involves group members understanding and supporting each other's viewpoints and dialoging about the project. Conflict subsides, and the group becomes cohesive; members begin to interact and create products. Tolerance increases as harmony develops. Conflict is avoided and mutual support, trust, and communication increase.

Performing

During performing, acting as a team becomes the norm, and members quickly define and solve problems. Solutions emerge; requirements become more detailed and easier to prioritize. Creative problem solving and collaboration occur as members gain insight into their perspectives and effective group interactions.

Adjourning

The group ends its work during

adjourning, wrapping up and reaching closure. This transition phase should not be ignored or glossed over because individuals can harvest their most important lessons from this stage, learning what they can bring to other projects and groups.

Workshop timing

How long should a collaborative workshop take? It can be as short as two or three hours if the group has gathered before and had time to "form" and "storm." Or you can hold workshops over several days. The key is to allow enough time for the team to reach closure on a subset of important deliverables.

The best plan is to schedule contiguous hours in a short time (days, or at most several weeks). One group I worked with arrived at their scope in a one-day workshop, then held four more short sessions to complete their requirements in almost four weeks. Another team was more efficient. They drafted some requirements as prework and then allotted two full days for the workshop. They reached closure on all their high-priority use cases, including details around business rules and interfaces. It took less than two weeks to complete their requirements.

The case for collaboration

Requirements workshops appeal to business people, developers, and managers alike. Business people want to be heard by IT, and they want enough technical perspective to make good techno-business decisions. Developers want stable, clear, and correct requirements that reflect users' real needs, and they also need to better understand and appreciate the business domain and the big picture. Management wants to minimize conflict, speed up the project, and get a quality product delivered. All these audiences can find workshops useful in the search for excellent requirements.

Having the right participants—the right stakeholders present—is key to your workshop's success. Little things also go a long way, such as participants sharing the same physical space, working hands-on with a variety of tools (text and visual), and displaying the group's work on walls (see Figure 1). A skilled facilitator stacks the deck for success by

- Helping plan prework that jump-starts group activities
- Making sure activities follow a logical structure
- Promoting high group productivity by using "serious play" throughout the workshop

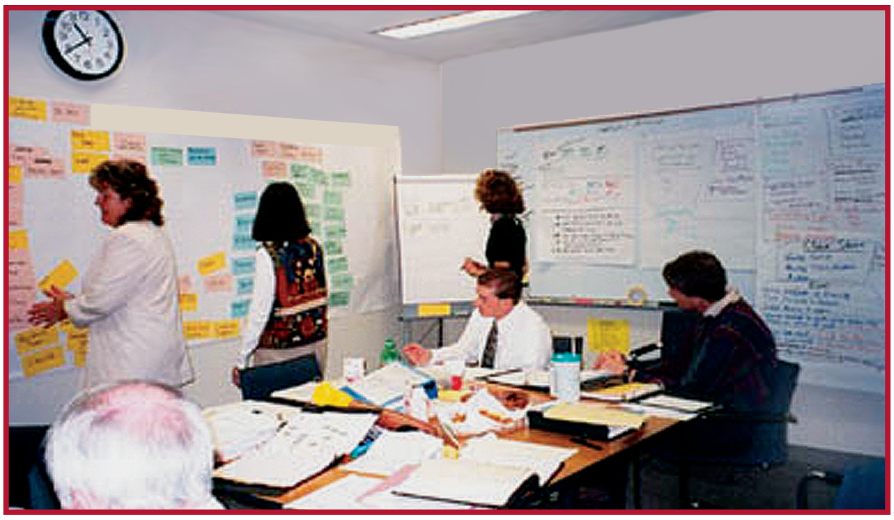
Figure 1. Participants in a requirements workshop working on walls in shared space. They are using various modes (individual, small group, and whole group) and media.

A collaborative workshop builds and nurtures a healthy project community, not only for requirements but also for the whole development process.

Well-run workshops deliver high-quality requirements, fast. Just as important, a collaborative workshop builds and nurtures a healthy project community, not only for requirements but also for the whole development process.

References

1. S. McConnell, *Rapid Development: Taming Wild Software Schedules*, Microsoft Press, Redmond, Wash., 1996.
2. Standish Group Int'l, "Chaos: A Recipe for Success," https://secure.standishgroup.com/reports/reports.php?rid=1.
3. E. Gottesdiener, *Requirements by Collaboration: Workshops for Defining Needs*, Addison-Wesley, Boston, 2002.
4. J. Wood and D. Silver, *Joint Application Development*, John Wiley & Sons, New York, 1995.
5. W. Tuckman and M.A. Jensen, "Stages of Small Group Development Revisited," *Group and Organizational Studies*, vol. 2, no. 4, 1977, pp. 419–427.
6. N. Kerth, *Project Retrospectives: A Handbook for Team Reviews*, Dorset House Publishing, New York, 2001.

Ellen Gottesdiener is a Certified Professional Facilitator and the principal consultant at EBG Consulting, where she works with business and IT project teams to help them explore requirements, shape their development processes, and collaboratively plan their work. Contact her at ellen@ebgconsulting.com; www.ebgconsulting.com.