

Requirements Workshops: What You Need to Know Now

BeWell Medical Corp. assigned a group of people to define the requirements for a software system to be used by its sales agents and customers. The system was to be called “RegTrak.”

Adam, RegTrak’s project manager, called me in. “They’re all over the place with what they want,” he said.

I asked him to explain. Who was all over the place? What did they want?

“Product Development wants our global sales agents to use RegTrak to register and order products for their regions or countries,” he explained.

He looked at the sheet of paper in his hand. “Distribution wants hospital purchasing agents to be able to place orders and check on them. Oh, and Marketing wants health care workers in the hospitals to have access, too.”

It was clear that the RegTrak team needed to nail down the scope and high-level requirements of the proposed project. Because Adam was newly assigned to RegTrak, he was especially concerned about getting the project off to a good start.

His comment confirmed my typical experience: Each of the various stakeholders had his or her own understanding of what the scope should be, based on personal experience and knowledge.

I suggested that we bring together all of the stakeholders to define the scope of the project, and Adam agreed. We formed a workshop planning team and got to work.

Smart Starts

How important is it to fully develop your project’s requirements? After all, nailing down your requirements usually takes only 8% to 15% of your overall project effort. Yet this work significantly affects the quality of your product, your customers’ level of satisfaction, and the amount of repair work you’ll have to take care of after implementation.

Let’s look at product quality. You’re likely to introduce more defects into your software product in the requirements phase than in any other phase—and these defects account for as much as half of the product’s total defects (Nelson et al, 1999; Schwaber, 2006). What’s

more, defects originating in requirements are harder to remove than defects originating in any other phase. And that's not all. Fixing them later in the project will *cost* you more, too, by as much as 100 times (or more!) after implementation than if you detected and corrected them in the requirements phase (Reifer, 2007; Dabney and Barber, 2003; Nelson et al., 1999; McConnell, 1996; Boehm and Papaccio, 1998). It's no wonder that rework due to requirements defects can eat up roughly one-third of your overall budget (Hooks and Farry, 2001).

Another aspect of low-quality requirements is harder to measure but just as treacherous. It's called "scope creep," and it's cited as the most vexing problem in software development. Unrestrained by carefully developed requirements and mutual agreement between your IT team and your customer or product development team, the scope of the project keeps creeping—expanding as the work proceeds. In the end, teams spend time and money defining many requirements that aren't even implemented. Some scope creep stems from customer issues. Customer involvement is one of the most important factors—if not the most important factor—in a project's success (Standish Group International, 2003).

One way to develop requirements as defect-free as possible, reduce the risk of scope creep, and define requirements quickly—while also building productive business-IT relationships—is to use collaborative workshops.

What Are Workshops?

In collaborative workshops, participants share a common goal, and they agree to join together to create work products in the pursuit of their goal. Members of the group act interdependently, relying on one another's knowledge, experience, skills, and perspectives. They are cohesive, meaning that they are motivated to act together. With appropriate direction, such a group can be highly productive. That's because collectively it has the right mix of skills and knowledge to create the work products.

If you're familiar with the acronym JAD (Joint Application Design or Development), then you're already familiar with one type of *collaborative workshop*. JAD-like workshops, which I generically refer to as *requirements workshops*, are a facilitated group process in which a carefully selected group of stakeholders and content experts work together to create, correct, and document a set of predefined requirements-focused *deliverables*, or *work products* (Gottesdiener, 2002).

For this kind of workshop, you design group processes for the specific team that is delivering requirements for a specific project. The interactions in the session enable the team to discover, elaborate, clarify, and close a set of requirements.

Some of the participants often produce a subset of the deliverables before the workshop to jump-start the process. This approach enables the group to focus on what's important.

A key element of successful workshops is that the participants agree to the workshop's purpose, ground rules, and products ahead of time. They also determine a decision-making process ahead of time; in this way, they can reach closure on their deliverables

and the actions they will take after the workshop.

You can use workshops for many purposes—to charter a project, to articulate the product’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.

Workshops Are Not Meetings

If you’ve ever sat through a poorly run meeting, you know how unproductive and negative an experience it can be. How are workshops different from other kinds of business meetings?

On the surface, collaborative workshops are like “meetings” in that both types of gatherings involve people meeting together at the same time, and both (presumably) follow a logical flow. But there are significant differences. Among them is the use of predefined roles, as summarized in Table 1.

Table 1: Workshop Roles

Role	Responsibilities
Workshop sponsor	Authorizes and legitimizes the workshop
Facilitator	Plans and designs the workshop Recommends the requirements deliverables Leads the process
Content participant	Creates the workshop’s products
Planning team member (a subset of the participants)	Works with the facilitator to plan the workshop Creates the workshop’s inputs As participant, creates workshop products
Recorder	Records the group’s work
Observer	Listens and learns
On-call subject matter expert	Is available to answer questions

Workshops include two people who fulfill specific process roles: a facilitator and a recorder. The *facilitator* is responsible for managing the group’s activities, dynamics, and work products. The *recorder* documents the group’s work as it proceeds. Neither person operates as a content expert, nor do they collaborate in product creation. As a result, they are free to focus on the process. As the group becomes more familiar with the process, the facilitator’s role in controlling the process can be relaxed.

Unlike a classic meeting, a workshop is authorized by a sponsor, who ensures that the right participants are present, verifies the workshop’s purpose, and monitors implementation of the workshop’s outcomes. 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). Sponsors who have rich domain knowledge might even participate. This adds a lot of expertise if you’re, say, developing the requirements’ scope and creating an iteration plan for a new product line.

Workshop participants will also include some subset of direct users—those who will operate the product after it is implemented.

Those who are 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.

Advisers with experience in the business domain might also participate. In other cases, an adviser might benefit from observing. For example, a product trainer observed one requirements workshop, gaining an early heads-up about the new and revised requirements for the company’s flagship product.

Indirect users—those who will receive products such as files or reports from the application—might also participate. For example, database administrators often attend workshops because they receive data feeds from the new application.

Product providers, such as testers and developers, might observe and even help plan the workshop. People in these roles have unique perspectives that can be great assets to workshops. If testers and developers they have deep domain knowledge, they may be participants.

Many organizations train and mentor business analysts and project managers to be workshop facilitators. They can be excellent choices as facilitators, as long as they can be neutral to the workshop’s outcome and the workshop participants also view them as substantially neutral.

Working the Workshop

In effective collaborative groups, energy is high, individuals respect one another, skills are complementary, and responsibilities and roles are clear both inside and outside the group setting. To maintain energy, creativity, and motivation, the facilitator uses interactive as well as parallel group activities.

For example, in a requirements workshop, subgroups can be formed to work on portions of a single deliverable, such as business rules. Alternatively, subgroups might be assigned to work on entire work products—one group might work on use cases while another drafts a prototype and yet another creates a conceptual data model. The subgroups then reconvene in a plenary (whole group) activity to share and critique their work.

In many of the workshops I facilitate, I also integrate verification activities, such as scenario and prototype walkthroughs, to detect requirements errors. Other times I have

subgroups check requirements models against a short checklist to find deficiencies quickly—early in the elicitation process. These activities vary the flow, increase productivity, and improve the quality of requirements.

It is the facilitator's job is to balance the needs of content, process, and people. Balancing *content* involves ensuring that the necessary requirements are delivered at the right level of detail and with the needed degree of quality. *Process* balancing requires the facilitator to design a sequence of activities that follows a logical progression within the specified time constraints; the goal is to promote participation and keep team members energized and engaged.

Balancing *people* needs is also a key responsibility for the facilitator. This involves helping participants build their relationships, exploiting the strengths of different styles of thinking, learning, and interacting, and helping participants become a high-performing group.

Workshops Deliver

A key difference between workshops and meetings is that workshops deliver specific, predefined products. These include tangible products—the requirements models. Workshops also deliver important intangible products, such as mutual learning, increased understanding, decisions, motivation, and teamwork. These products are specifically planned ahead of time.

The following are examples of the work products created in actual facilitated workshops (Gottesdiener, 1999; Gottesdiener, 2007):

- In a half-day midpoint debrief (retrospective), a beleaguered project team created an action plan for correcting critical project problems, reestablished how and when team communication would occur, and redefined several key roles, including that of the project sponsor.
- Workshop participants generated 119 events, classified them, and then removed those not in scope during a two-hour requirements scoping workshop.
- In a use-case modeling workshop, business participants were able to test their use case model and structural business rules using 55 scenarios in less than 90 minutes.
- In three hours, a team validated a medium-complexity data model and made decisions about the scope of the data and which business performance metrics would be applied.
- In four hours, a project team defined the deliverables needed for the upcoming design phase, decided who would own, review, and create each deliverable, defined their quality criteria, and mapped them into dependencies along a time line.
- In two hours, a team of architects and designers created a high-level architecture for their next-generation product line.
- In less than two hours during a chartering workshop, a team of business stakeholders generated 28 risk mitigation strategies for their soon-to-launch project.
- In four hours, a team of business product managers created five state diagrams for each major state of the key domain in their global supply chain.

Workshop planners need time to define what to deliver, what level of detail the materials should contain, and how the intangibles will be achieved. A useful guideline is that you need *at least* one hour of workshop planning and design time for each workshop hour. For new facilitators or complex projects, a 2:1 ratio is more apt: two hours of preparation for each workshop hour. This is not insubstantial, but the benefits can be vast.

The Business Case for Workshops

Some organizations shy away from workshops, shunning the investment in time it takes to plan and execute them. Others claim that the cost of gathering people together in the same place at the same time is prohibitive. But requirements workshops can provide business value by reducing the time it takes to gather requirements, boosting team productivity, and reducing the risks associated with projects.

Here are examples of the effects of well-designed and well-run requirements workshops:

- Reducing the risk of scope creep from 80% to 10% or less when combined with prototyping (Jones, 1996).
- Cutting requirements creep in half (Jones, 2000).
- Providing a 5% to 15% savings in time and effort for the project as a whole (Jones, 1996).
- Reducing defects delivered in software by 20% (Jones, 1996).
- Reducing project failure and cancellation rates by about 50% (Jones, 1996).
- Increasing productivity by 10% to 50% (Vosburgh et al., 1984).
- Providing a 10-to-1 return on investment (\$10 for every \$1 invested) (Jones, 1996).
- Cutting elapsed time for requirements specification by 20% to 60%, and reducing total project effort by 20% to 60% (August, 1991).

Aside from the hard data, requirements workshops contribute to customer and user satisfaction because they allow those with the needs to participate directly in defining those needs. On one project, in which we built multiple requirements models (use cases, business rules, user interface prototypes, a data model) during requirements workshops, not one high- or medium-priority defect originated with requirements.

The long-term prognosis for a project that uses requirements workshops is better, because the workshops help build a healthy project community early in the project's life. After all, people support what they help to create. In addition to delivering requirements, workshops help your team nurture healthy interactions, trust, and collaboration. They also show by example the value of maintaining active customer involvement in the software development process.

When *Not* to Use Workshops

You need to use requirements workshops under appropriate project conditions. If your project is small and low-risk, I don't recommend using workshops. However, if your requirements are complex and you value the side benefits of collaboration—trust, mutual

understanding, and participation—then requirements workshops may be for you.

Requirements workshops are most appropriate for business or commercial software when you can get knowledgeable customers, users, or surrogates. Requirements for real-time systems are less visible to customers, so requirements workshops are probably not suitable for that type of development.

Size matters, so be careful not to overload the workshops with too many people. The number of participants should not exceed sixteen unless you're using two facilitators. Most workshops have seven to twelve participants. Each person you add to the workshop can require additional time or reduce the number of deliverables. Strive for the minimum number of participants who can represent the collective needs of customers, users, testers, designers, and analysts. My own rule of thumb is "as many as necessary, as few as possible."

If you can't identify a workshop sponsor, you won't have the necessary commitment to ensure that the right players participate and prepare for a successful session. For some organizations it isn't feasible to gather users or their surrogates at the same time and the same place. In that case, you might consider using collaborative software tools that allow people to gather in a virtual environment. Be aware that you'll need a skilled facilitator who can lead the process and also use the tool.

It takes time to plan a requirements workshop. If management won't allow time for planning and pre-work, your workshop results are likely to be mediocre.

Finally, don't use workshops unless you have a facilitator who is neutral to the outcome, experienced with the group process, and knowledgeable about the deliverables you need to create. These skills can be developed within your organization and shared across projects.

In lieu of workshops, you can use techniques such as observation, interviews, prototypes, reuse of existing documentation, and research-based approaches, including surveys, competitive analysis, and product complaint data. Of course, these techniques can be powerful *in combination with workshops*, if workshops seem to be a fit for you.

BeWell Medical Corp. Starts Smart

Recall that Adam was the project manager for RegTrak, a software system to be used by sales agents and customers. Adam needed to define the requirements for the proposed project, and he decided to use workshops to define those needs. Over the course of the fifteen-month project, the team used multiple workshops to deliver what was needed.

They started with a chartering workshop, facilitated by the RegTrak business analyst, to define the business requirements—goals, objectives, product vision, stakeholders, and scope-level requirements models. This workshop and the follow-up activities helped the business clarify what it really needed, where software might fit, and which business processes would be affected.

In several business modeling workshops, the stakeholders dug into the business processes and policies that needed improvement as part of implementation. These business modeling workshops helped the company's high-level executives understand how software solutions would help achieve their business goals. Because of the size and complexity of the potential software solution, they agreed that an agile approach—incremental delivery with heavy reliance on just-in-time requirements modeling, prototypes, and user acceptance testing—would be an excellent way to reduce the risk of the project.

Before jumping into project iterations, the team gathered for a product roadmap workshop to define the releases and develop an iteration plan for the first release. After that, the team conducted short, focused requirements workshops for each iteration. In these workshops they defined enough user requirements and quality attributes to enable them to deliver a working demonstration of the software.

“It was hard work, but worth it,” Adam declared in the final project retrospective. “I can't imagine not using this workshop approach and going back to the 'old' way of back-and-forth tug-of-war between the stakeholders. “

Summing Up

Requirements workshops appeal to businesspeople, developers, managers, and business analysts alike.

Executives 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. Business analysts want to help all the parties get the right stuff, fast.

All these audiences can find workshops invaluable in the search for excellent requirements.

About the author

Ellen Gottesdiener, Principal Consultant, [EBG Consulting](#), helps teams to collaboratively explore requirements, shape their development processes, and plan and review their work. Her book [Requirements by Collaboration: Workshops for Defining Needs](#) (Addison-Wesley, 2002) describes how to use multiple models to elicit requirements in collaborative workshops. Ellen's most recent book, [The Software Requirements Memory Jogger: A Pocket Guide to Help Software and Business Teams Develop and Manage Requirements](#) (GOAL/QPC, 2005), describes essentials for requirements development and management. Both are available on Amazon.com and via other quality booksellers.

Ellen presents seminars, speaks at conferences, and serves as adviser for numerous industry events, and she has authored numerous articles. She is a Certified Professional Facilitator (CPF) and a member of IEEE, IAF, ACM, DAMA, and IIBA. Ellen serves on the Expert Board of the International Institute of Business Analysis (IIBA) Business Analysis Body of Knowledge™ (BABOK™).

You can subscribe to *Success with Requirements*, EBG [Consulting's free monthly eNewsletter](#). When you sign up, you'll receive a free .pdf article by our Senior Associate, Mary Gorman, on an essential requirements modeling technique. We created *Success with Requirements* to help you get the right requirements so your projects start smart and deliver the right product at the right time.

References

August, Judy H. 1991. *Joint Application Design: The Group Session Approach to System Design*. Yourdon Press/Prentice Hall.

Boehm, Barry, and Philip N. Papaccio. 1998. "Understanding and Controlling Software Costs." *IEEE Transactions on Software Engineering*, October.

Dabney, James B., and Gary Barber. 2003. "Direct Return on Investment of Software Independent Verification and Validation: Methodology and Initial Case Studies." Assurance Technology Symposium, June. See <http://sarpreresults.ivv.nasa.gov/ViewResearch/24.jsp>.

Gottesdiener, Ellen. 2007. "Multi-Modeling: Exploring Requirements to Define User Needs." TechTarget's SearchSoftwareQuality.com. February.

Gottesdiener, Ellen. 2005. *The Software Requirements Memory Jogger: A Pocket Guide to Help Software and Business Teams Develop and Manage Requirements*. GOAL/QPC.

Gottesdiener, Ellen. 2002. *Requirements by Collaboration: Workshops for Defining Needs*. Addison-Wesley.

Gottesdiener, Ellen. 1999. "Decoding Business Needs with Workshops." *Software Development Magazine*. December.

Hooks, Ivy F., and Kristin A. Farry. 2001. *Customer-Centered Products: Creating Successful Products through Requirements Management*. Amacom.

Jones, Capers. 2000. *Software Assessments, Benchmarks and Best Practices*. Addison-Wesley.

Jones, Capers. 1996. *Patterns of Software Systems Failure and Success*. Thomson Computer Press.

McConnell, Steve. 1996. *Rapid Development: Taming Wild Software Schedules*. Microsoft Press.

Nelson, Mike, James Clark, and Martha Ann Spurlock. 1999. "Curing the Software Requirements and Cost Estimating Blues." *The Defense Acquisition University Program Manager Magazine*, November–December.

Reifer, Donald J. 2007. "Profiles of Level 5 CMMI Organizations." *Crosstalk: The Journal of Defense Software Engineering*, January.

Schwaber, Carey. 2006. "The Root of the Problem: Poor Requirements." IT View Research Document, Forrester Research, September.

Standish Group International, Inc. 2003. "Chaos Chronicles."

Vosburgh, J. B., et al. 1984. "Productivity Factors and Programming Environments." In *Proceedings of the 7th International Conference on Software Engineering*, IEEE Computer Society, pp. 143-152.