

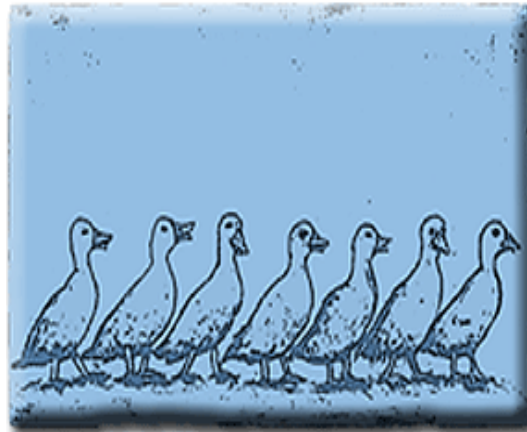
► **Requirements Management: Your Insurance Policy on Large-Scale Projects**

by **Dan Gonos**

Chief Technologist

Electronic Data Systems (EDS)

Most people think requirements definition starts after the contract is signed. But the bigger the project, the more important it is to begin defining and tracking requirements immediately upon getting involved with the project -- when you're evaluating the Request For Proposal (RFP). On projects for which development timeframes are measured in years, or even decades, the ability to track requirements is properly seen as a mission critical component of your business strategy, not just a first step in the software development lifecycle.



- [subscribe](#)
- [contact us](#)
- [submit an article](#)
- [rational.com](#)
- [issue contents](#)
- [archives](#)
- [mission statement](#)
- [editorial staff](#)

The Challenges of Big

It's a truism in the software business that the bigger the project, the later and further over budget it goes. One reason, surely, is that whatever requirements are submitted up front, even if they are comprehensive and well-defined, often tend to fade out of the picture as the job grinds on. Many of us have experienced first-hand how big projects frequently seem to lose their focus, especially when distributed teams are each working on their own bits and pieces. The reference point, especially for developers, tends to become "the last deliverable," not the ultimate goal as defined contractually. When this happens, business objectives, too, can gradually become unclear. And if you don't know where you're going, then where are you going to get to?

"Finding out when you get there" is not the way to make money and satisfy clients as a systems integrator or software consultant. You need to

know exactly where you're going, and you had better be able to point to the map once in a while. That's why it's absolutely essential to define requirements clearly, track them religiously, and stay focused on them steadfastly, before you even respond to that RFP.

When managed in a disciplined way, your project's requirements become a vital tool for maintaining communication among all participants, rolling with changes, adjusting schedules, and controlling costs. Leave requirements to chance, and your project will quickly drift off course.

My Project

I'm a Chief Technologist at EDS. We're a large company, and we take on some very big projects. But even by our standards, my project is truly immense. Basically, it entails building a new computing infrastructure from the ground up to support one of the largest social services environments in the US, encompassing eighteen California counties. When completed, our system will help these counties deliver public assistance programs like cash assistance, food stamps, and employment services benefits to more than three million clients. The system will be accessed by over 30,000 users, ranging from clerks to directors, at 700-plus geographically dispersed sites throughout the State of California.

Taking Requirements Seriously

Needless to say, this project entails custom software development on a level that's highly unusual -- even for EDS. But that's literally not the half of it. Each of those eighteen counties administers its programs separately, and has its own unique management practices. From the user's perspective, we will in effect build eighteen different implementations of a single system, each with its own set of requirements.

Now that's a major requirements management challenge. But it's not the only one on this project. Public policy changes that impact our system happen routinely at all levels of government in response to court cases and legislation; over the course of this project, that will necessitate thousands of changes to requirements.

Obviously, if we don't do a good job defining and managing requirements, this project might never get done! (How would we know when to stop?) But our motivation to pay attention to requirements goes much deeper than that. Our contract with the CalWIN Consortium, a consortium of 18 California counties, is fixed price. Therefore, if we build something that isn't what the client asked for, then *we have to fix it on our nickel*. For this reason alone, we would be serious about requirements management around here, I can assure you. We're also confident that we have the methodology, the tools, and the discipline it takes to make that kind of contractual obligation work for us.

Starting Smart

When EDS decided to bid on this project, we knew that effective requirements management would be pivotal to our success. We began

tracking requirements right from the RFP, both to enable us to identify high-risk or unclear requirements, and to give us a head start once we got the job. (And it probably didn't hurt our chances for the client to know we were already thinking proactively about project management.)

We chose Rational RequisitePro as our requirements management tool for several reasons: myself and other key staff were familiar with it and knew it could do the job; it's flexible enough to be adapted to our EDS project management methodology; and it can scale well enough to handle the deluge of requirements and users this project entails.

Starting with a Methodology

At EDS, every project starts with our company methodology: the project management processes that form the core of how we get the job done. We put these processes in place first, and then worry about implementing tools to facilitate them.

A business process methodology should always be the starting point for choosing and configuring project management tools -- not the other way around. You need to know what you want to do first, before you tell some tool to go do it for you. That means you need to select tools you can readily configure, so you can integrate them with how you plan to do things.

For example, our EDS methodology includes an explicit operational definition of what constitutes a requirement. To wit:

A requirement is a condition that must be met for the client to find our products or services acceptable.

The ideal set of requirements is solution-independent; that is, it is stated completely in terms of the client's business needs. This gives EDS the greatest flexibility to satisfy the requirements.

Our methodology also defines activities for managing requirements. These include:

- Planning for the requirements management process.
- Obtaining source information from which to derive requirements.
- Development of requirements.
- Validation of requirements.
- Incorporation of requirements into the requirements baseline.
- Evaluation and refinement of the requirements management process.

By following this methodology, we can manage requirements effectively. The benefit our client derives from our efforts is the receipt of a product or service that adds value to their organization. This results from a clear understanding of the client's needs and expectations. It also leads to fewer

product or service design and construction errors, thereby lowering or eliminating costs associated with rework. (This is critical because project estimation often does not factor in "rework time.")

In short: the effectiveness of your requirements management process is inversely proportional to the need for rework associated with faulty requirements, and directly proportional to the accuracy of your estimates pertaining to total lifecycle cost and schedule.

The Role of Tools

At EDS, our methodologies drive the project, and the tools we choose support our methodologies. Rational Requisite Pro turns out to be a good fit with how we need to manage requirements. Likewise, other Rational tools are a good match for our change and issue management procedures, and for the way we do modeling, planning, and testing. But again, we never think "automation first, process second." We focus on defining a solid change management process, a solid deliverables approval process, a solid software validation process, and so on. On different projects, different tools might be appropriate.

In the current stage of our project, we're tracking roughly 12,600 functional and technical requirements. By the time we're close to completion that number will be nearly 16,000. All of the 200-plus staff members on the project interact with the requirements management system in one way or another as part of their jobs.

After we put tools in place to support our business processes, we continuously strive for improvement by refining those processes and reconfiguring the tools. This is another reason why we invariably choose tools that are flexible and offer a comprehensive, easy-to-use feature set. One thing I like about ReqPro, for instance, is that it's simple to add and delete attributes as the project evolves. We do this frequently, because we strongly believe in incorporating lessons learned from previous phases of the project. If it's broke, fix it -- don't perpetuate foolishness!

The Value of Requirements Management

Since my project began 18 months ago we've hit every major project milestone. All our deliverables have been submitted and approved on time, and we're well within our budget.

Calculating a Return on Investment (ROI) to illustrate the value of our requirements management process wouldn't demonstrate its true worth to the project as a whole. And good requirements management is obviously not the *only* thing that's gotten us to this happy place. But our focus on requirements has been critical to our success in tremendously important ways. In particular:

- **Requirements management facilitates the software lifecycle.** Our close management of requirements is a form of "adult supervision" to help all developers on all

teams stay focused on business objectives from deliverable to deliverable. Whether they work for the client, EDS, or one of our subcontractors, our developers consider the client's requirements and the software design specifications together as they work. They have to, or they'd *never* hit the mark.

- **Requirements management enables us to cope with both rapid and gradual changes over a long period of time.** As I mentioned above, our requirements change constantly in response to court cases and legislation. Faithfully tracking and categorizing requirements enables us to much more easily identify the specific requirements that are impacted by a legislative maneuver. Otherwise, we'd be continually digging through the program specifications, or even the code! And that holds equally true for changes driven by our defects management process, through client feedback, new technical input, etc. Because we have requirements under control, we're much better equipped to efficiently and appropriately revise components in response to *all* forms of change. Which, I don't need to remind you, can make or break your bottom line on even the smallest project.
- **Requirements management helps you relate deliverables to contractual obligations.** On a behemoth project like mine, for which requirements change constantly and the development lifecycle will take approximately four and one-half years, our ability to track requirements relative to deliverables is paramount. We must specify precisely what requirements a deliverable will embody. That's how we can verify that we built what the client *really* asked for -- not what they thought they asked for, or what we thought they asked for. This allows us to cleanly settle issues of accountability should a dispute arise.

A Must for Project Management

I can only imagine (fortunately...) what my project would look like if we weren't focused on managing requirements. For one thing, we'd find it next to impossible to plan and determine the features of a deliverable, because we'd have no way to ascertain what the requirements were, or how they could be measured and quantified. And if we weren't able to communicate the latest requirements effectively to our development teams in the first place, then we'd find it much more difficult to manage the development cycle. Furthermore, if we weren't able to track test cases relative to the latest requirements, we'd never know whether we tested all the functionality. Without the ability to quickly ascertain the impact of a court case or legislative change, we'd waste countless person-days

creating functionality that might or might not meet the client's needs. Plus, without a valid set of requirements, there'd be no way to monitor and control "feature creep," so we'd likely miss delivery dates and run over budget. Who knows, we might never complete the job! In other words: No requirements management equals a project out of control.

Back to Basics

Another way to think about requirements management is that a shared understanding of requirements keeps the lines of communication open, so everyone can work together to build the system that best meets the client's objectives -- on time and within budget.

If, on the other hand, you overlook requirements management, the scope of your project will slowly transform itself, through lack of attention and guidance, into something the client didn't ask for. Hoping they won't notice, or that they'll like your *ad lib* better than the script, are risky bets.

As the Chief Technologist, I'm squarely in the line of fire if deliverables don't meet requirements and EDS has to start fixing things for free. That's why I view our requirements management as "life insurance." Because when it comes to requirements, why take unnecessary risks?



For more information on the products or services discussed in this article, please click [here](#) and follow the instructions provided. Thank you!