

**How to Create and Communicate Solid  
Requirements for a Forms Management Database:**

**Laurie Weaver  
Forms Coordinator/Developer  
Capital Group Companies**

## How to Create *and* Communicate Solid Requirements for a Forms Management Database:

*You know you need help managing your forms process, and you suspect technology might be the answer – but what technology? What are the critical items you need to communicate to technical developers, IT departments and vendors to help answer that question? Further, how can you, as a forms professional, feel confident that you have successfully translated those critical business needs into the proper technical communication once you've figured them out?*

Busy forms professionals looking to acquire or enhance a forms management database face questions like this all the time. Consumed with the pressing needs of the day, they can feel intimidated when it comes to identifying and communicating high priority business problems that might be solved by technology.

We know, because we've been there.

The following tips, recommendations, and methods are based on our experience as a forms team manager and a forms team developer who have waded through the process to create an effective database for our team. We definitely had success, as well as learned some important lessons.

### **Lesson one: What is a requirement and how do I write one?**

The resulting database will only ever be as good as the requirements used to build it – even if, like us, you are thinking of building the database yourself in-house. Targeting an effective solution requires taking the time to define exactly what your needs are. Moreover, even an effective solution can be rendered useless by poor architecture. Good architecture fits both your business context and technical standards. To that end, let's explore how to go about it.

Requirements begin with business problems. Good requirements then go through a process whereby they are clarified into desired outcomes. It is critical that the desired outcomes be broadly stated vs. confined into design specifications – but more on that point later.

Let's examine a typical forms management problem and polish it into an effective requirement using four steps.

#### **Step 1: Briefly state the problem.**

- *"I don't know, without physically auditing, which version of my forms are in which locations."*

This is a good summary of what's wrong, but doesn't yet lend itself to a desired outcome. For that, we need to take a crack at turning the negative problem into a positive. Let's take a look at what we *wish* we had in place today.

**Step 2: Briefly describe the solution(s) in broad terms.**

- *"I need to track all forms that reside in a location."*
- *"I need to track all locations in which an individual form resides."*
- *"I need to track form versions."*

Be cautious at this point not to delve into specific design solutions, such as, *"I need to have a drop-down menu in my database listing all possible form locations."*

If you are working with a developer or vendor, you want to leverage their expertise – that person or firm may have a better design solution than you outlined – yet they assume you mean exactly what you have stated. They will then build what you have requested vs. the better solution. This applies even if you are going to build your own solution. Thinking in specifics at this point is premature and limits your architecture possibilities.

**Step 3: Ask yourself what terms need to be clarified.**

- *"Will everyone understand what I mean by location?"*
- *"Do I need to track by a specific method?" (e.g., by a form's serial number or name?)*
- *"How are different versions indicated?" (e.g., part of the serial number?)*

The key to this step is clarity and succinctness. This helps to develop robust design solutions. For example, when you are writing your requirements, you refer to office location. To you, office location means a three-digit code representing an individual office building from among several hundred office buildings. To a developer, tracking that specific information brings to mind significantly different design ideas than the notion of tracking office locations by street address.

**Step 4: Rewrite your requirements to include important specifics.**

- *"I need to track all form versions by serial number. Serial numbers are mixed alphanumeric codes of five characters. The number of characters may increase."*
- *"I need to track all form versions by location. Locations are offices that distribute the forms. Office locations are referred to by a three-digit code. Offices may be added or removed."*

Be careful not to assume that the person with whom you are communicating knows what you're taking about. Save time by taking time to spell out your thoughts.

Before moving on to the next lesson, go back and compare the difference between what we wrote in step 1 and step 4. Imagine what the developer would take away from each case. If each was an example of your written requirement, which would likely provide the best result? This should give you the flavor of how taking your business needs through a requirements process can, in the long run, significantly save business time and aid in communication.

## **Lesson two: What should I write up as a business requirement?**

Like any other business process, requirement gathering can generate a “kitchen sink” level of minutia that can overwhelm you. On the other hand, filtering out business needs too soon can limit the true big-picture view needed for your solution.

Here are three questions you can ask yourself to help identify issues that are “requirements” worthy. The answers can then be used to develop an action plan to capture and develop the requirements that will provide the most value.

### **Question 1: How do I become aware of business problems?**

- *“I notice when I’m annoyed.”*
- *“My team or supervisor brings various issues to my attention.”*
- *“I review resource time spent and note excesses in time or effort.”*

Above are just a few examples of what might cause a problem to hit your radar. The reason to consider all of these processes consciously is that it helps you determine the following:

### **Question 2: How do I keep track of business problems?**

- *“I write them on sticky notes.”*
- *“I add them to a spreadsheet or written document log as they come up.”*
- *“I write them in my day planner or PDA.”*

If you don’t have an answer to question 2, now’s the time to pick one and go for it! Decide how you are going to log issues as they arise, and follow through. At this point, don’t worry about how big or small the issue is or if you’ve captured it before, just get a process going and capture your issues.

### **Question 3: How do I prioritize my business problems?**

- *“Weekly, I review my log and highlight important issues.”*
- *“I add priority numbers to my spreadsheet log so I can sort issues by it.”*
- *“I discuss the issues every other week with my team or supervisor, and together we flag the important issues according to the criteria we determine.”*

The key to this step is to detect what's most important and pressing for your team or group, and then refine the critical business needs from the "nice to haves." Is it cost savings that's most important? Inventory issues? Workflow? Determine your themes and issues. Then you can spend time categorizing your requirements accordingly.

Before continuing with the last lesson in our series, take a moment to decide how you are going to log your business issues. Try it for a set period of time. Once you have your requirements roughed in, we come to the last challenge.

### **Lesson three: How do I make sure the developer or vendor understands my business requirement?**

All business areas have their own vocabulary and terms. Most of us have experienced times where we see "heads nodding" when we speak, but the resulting communication doesn't match up. This is particularly true with communication between technical and business folk. Some forms professionals are lucky enough to have a Business Analyst to help with this translation. Even so, it is helpful to clarify for yourself exactly what you mean and to discern if your carefully crafted message has been received.

To that end, we'll wrap up with three key items to check for good technical communication:

#### **Key 1: Watch out for techno translation mishaps. Clarify your terms.**

- *Example: You use the word "sites" meaning office locations. The developer thinks "websites."*

Instead of using your office or business jargon in requirements, write using plain language or use clarifying examples. Also, don't be afraid to ask questions if you don't understand what the technical person is saying to you. Ask that person for examples until you do understand. It is critical that both you and the developer have a meeting of the minds about the job at hand.

#### **Key 2: In verbal discussion, have the developer restate what he or she heard. Clarify if needed.**

- *You: "Just to clarify our terms, could you restate for me your understanding of the last point?"*
- *Developer: "I heard that you wish to track which websites carry your forms."*
- *You: "Actually, I meant office locations. We refer to office locations as sites."*

Don't rely on head nodding, or hearing "I understand." Take a moment and verify verbally or in writing that your business needs are understood.

**Key 3: Put your technical requirements into business context where possible. Compare the following:**

- *"We need to track the form version and sub-version number."*
- *"We use various versions of forms across the country, and particular states require particular legal language. When I say I need to track the sub-version of a form, I mean which state-specific version is in use."*

Technical folk may, or may not, "get" the context for your requirements or requests from the requirements document, particularly if you don't have a business analyst on the project. Providing your business context may point to a needed feature, clarification or discussion about specific alternatives with your vendor or developer.

In closing, sometimes we forget that any database or other technical marvel is really the result of people formulating and implementing a plan. Taking the simple steps outlined above will help make sure your plan is focused, solid and targeted toward the result that will benefit your business.