

What's the Latest on Business Rules and Decisioning?

A Q&A with ...

Ronald G. Ross

Chair, Business Rules Forum Conference
Executive Editor, www.BRCommunity.com

The 12th International Business Rules Forum (www.businessrulesforum.com) is happening November 1-5 in Las Vegas (Bellagio). Find out about the very latest in this important space in this special interview with the Forum Chair and author of the newly released 3rd Edition of *Business Rule Concepts*.

~~~~~

**\*\* The Requirements Networking Group has arranged a special 10% discount on registrations to this event. Use code "RNG09" when you register. \*\***

~~~~~

RQNG: You are widely known as a critic of the way most systems are developed and built today. What do you see as the fundamental problem?

Ross: To put it in a nutshell, systems today aren't agile – even when you use agile software methods to develop them. Companies need *business* agility, and in most cases we simply aren't delivering it.

RQNG: Can you give an example?

Ross: I recently visited a very large health care organization and had conversations there with a variety of people. One manager confided to me that making a change to business rules of even moderate complexity took them 400 person-days over a 4-month period. That's staggering. It's simply not sustainable. Unfortunately, their experience is not really that far from the norm. It's safe to say that this organization, like many today, is living in what I call *change deployment hell*.

This same manager went on to observe that a subtle stagnation had crept into the staff's very way of thinking about the business. He noted that they often don't even consider business innovations they know from experience to be difficult for the existing systems to handle. He wondered out loud whether they could even think through any real innovation effectively any more.

That situation is simply not acceptable. I say we can do better than that, *smarter* than that – that we can and that we should. By the way, the people at this organization were hard-working and very engaged in their activities – they did want to deliver good quality. So that was not the problem. Indeed, I find most people in our field to be very professional and astute and to have the best of motivations.

RQNG: So where does the problem lie?

Ross: You might say that their systems were poorly engineered. I mean they were unwieldy, expensive to maintain, and unfriendly to the business. But I'm not sure that's accurate or even fair. Rather than poorly engineered, I think it's more accurate to say the systems had been *over-engineered*. What happens when you over-engineer something? The solution you produce is usually too stiff or rigid or cumbersome for the real-world problem. Think tree that doesn't bend with the wind. That's exactly the problem with legacy systems. The speed of business itself is accelerating rapidly, but the architecture of traditional systems is rigid and static.

RQNG: What exactly is the problem?

Ross: The problem lies with the embedding of criteria used to make day-to-day, minute-to-minute business decisions within the systems themselves.

Building systems that way is actually quite hard. Indeed, traditional development methodologies require IT developers to have a high level of expertise both in the business and in how to implement functional requirements under the given platform(s) or language(s) most effectively. That's a lot to ask. Even if you get the business rules right (doubtful in itself), the rules are now hard-coded in the application logic where they are hard to find, hard to understand, and even harder to change.

And that's just it – the business rules *will* change. So you *will* be revisiting the code. There's a certain mindset in traditional IT departments that revisiting code with any frequency means the code must be fundamentally flawed. But with respect to business rules, that's *way* off-target.

The obvious solution is a separation of concerns. Quite simply, the business rules should be engineered separately from the functional requirements. Can you really do that cleanly and effectively? Absolutely. It's a proven fact. If you come to the Business Rules Forum Conference next month, you'll hear from some 50 companies about how they did exactly that.

Over-engineering resulting in rigidity is not a good thing. It results in buildings that tumble in earthquakes and bridges that collapse in windstorms. In IT terms, it produces a world where some 75% or more of all IT resources go toward system 'maintenance'. It's time to move on to the next engineering paradigm.

RQNG: What exactly is a business rule?

Ross: I define a business rule as criteria used to make a decision in day-to-day operation of the business. Some people think of business rules as loosely formed, very general requirements. That is not the case at all. Business rules have definite form, and are very specific.

RQNG: Can you give some examples?

Ross: Sure, here a few simple ones. A customer that has ordered a product must have an assigned agent. The sales tax for a purchase must be 6.25% if the purchase is made in Texas. A customer may be

considered preferred only if the customer has placed more than \$10,000 worth of orders during the most recent calendar year.

Each example gives well-formed guidance focused on making some specific decision. Each uses terms and facts about business things, which should be well defined. Each is declarative, rather than procedural. You should think of your company's business rules as a resource that needs to be managed, so we encourage people to think in terms of rule management – or *rulebook* management as I call it in the recently released newest edition of *Business Rule Concepts*.

RQNG: How do business rules relate to requirements?

Ross: Professionals need to stop thinking of business rules as simply another form of software requirement. There's a big difference. When a project is over, software requirements, in theory, are satisfied and go away. For business rules, in contrast, that's just the beginning of life.

The business rules, and the vocabulary on which they are based, become central to the problem of affecting continuous change. They need to remain right at the fingertips of both business people and business analysts. They must be accessible and well-managed. Above all, you want traceability from original sources (business policies, agreements, contracts, laws, regulations and so on) into the points of operational deployment. You want to know who created what rules for what purpose at what time. I call all that *corporate memory*. Without traceability, you can have no accountability, and without accountability, you can have no transparency. And you can forget about rapid change and business agility altogether.

RQNG: Can business rules be managed using tools and repositories aimed at software development and IT developers?

Ross: I'm doubtful. I believe you need a new breed of tool, which I call a *general rulebook system* (GRBS). The point is that the life cycle of business rules and the life cycle of software releases are different. They serve audiences with very different agendas, and have a very different natural pace. They need to be *radically* decoupled.

RQNG: Do you think business rules are going mainstream?

Ross: I'm getting constant webinar announcements from IBM these days about Ilog, a business rule management system (BRMS) it acquired in the last year or so. When a company like IBM jumps into this space, you know it's gone mainstream.

RQNG: What makes BRMS different?

Ross: In traditional implementations, it's very hard to get at the business rules. They're hidden from view – really the implementations are black-box with respect to the rules. BRMS-based solutions, in contrast, are white-box. It's far easier to get your hands directly on the actual rules – and to change them.

Procedural languages are simply not good for run-time evaluation of rules, especially at the scale we now require. Instead, you need business rule management systems (BRMS) – often called *rule engines*. Think of a BRMS as being to rules more or less like a DBMS is to data. Would you even consider writing your own DBMS these days? *No!* Yet many organizations today are still doing the equivalent for run-time evaluation of business rules. It's simply inappropriate for IT organizations to continue doing that any longer.

Growing numbers of organizations have adopted commercial or open-sourced BRMS. Five or ten years from now, we'll look back and wonder why the heck it took so long!