Practical Business Rules Development and Use

Mike Cook

Senior S1000D Business Analyst

S1000D User Forum 2013
Vienna, September 16 - 19

Key points

- **Benefits of business rules**
  - They make you stop, think, plan, test, and then implement
  - Gives you a chance to create sample data and play “what if?” games
  - Forces you to communicate ideas to a broader team
- **Benefits of analysis and design**
  - Saves money by finding design issues early
  - Reduces costs by keeping the project team small at the beginning
  - Accelerates review of good design principles early in the design phase
- **There’s more to business rules than the BRDPs in the spec**
  - The spec introduces important business rules needing early decisions
  - Custom rules are ***not*** in the spec and need to be captured – but it’s up to the analysis team to identify and document them
- **Catching errors early and correcting them**
  - Identifying missing capabilities, understanding how to use elements and attributes early
- **Saving money**
  - Additive effects of good analysis practices (business rules), leveraging S1000D features and capabilities, and building a solution you can support long term is the key to success

History of Business Rules

- Where Business Rules came from
  - Business rules are an outgrowth of systems analysis and design
  - Look in early issues of S1000D and see what the definition of a business rule is
  - The concept of business rules has morphed since its first definition in the specification (read up on it in Issues 2.1 and 2.2 – compare that with today)
  - The original intent of the BREX DM did not include the testing of element or attribute values – only the appearance or lack of an element or attribute was tested
  - Adding in the ability to test the ***value*** of an element or attribute using XPath and XQuery is great but increases complexity and time to evaluate the value
- Why Business Rules were created
  - Rules keep projects honest and conformant to the specification by informing everyone HOW, WHEN, and WHY elements and attributes are used in the various schemas
- Why we have BRDPs today – (hint: been there, done that)
  - Good business rules come from the experience of others who have been there before and know the question is important to your project's success
  - Properly documenting a rule so everyone understands what to do is more important than answering all the rules. Good results come from good communication.

    Note: Not every rule is important to your project. Just because a rule is in the spec doesn't mean you have to answer it. Know what to ignore.

    Also, not all the Business Rules you need to answer are in the spec (custom rules). Many rules should be answered as part of any project and are "custom" rules. For example, not every project needs to use the ***skillLevelCode*** attribute the same way

Keeping technical writers honest

- Minimizing errors early
  - Authors are going to make mistakes. They'll make more mistakes if you don't tell them how to use the various elements and attributes correctly
  - Documenting how to use an element, attribute, or S1000D capability makes it easier for everyone to "get it right"
- Indirect training
  - The BREX DM can indirectly train authors how not to use an element or attribute, and vice/versa
  - Being able to read a Business Rule Design Document (BRDD) instead of guessing how to use an element or attribute makes a big difference
- Improving throughput
  - Getting more out of a writer, editor, or IT employee makes a big difference in the bottom line – getting there takes education
- Increasing reliability
  - Reducing mistakes saves time, increases throughput, and ultimately increases quality of the final product. It's easier to build it in up front than it is to add it in half way through the implementation of the project

Leveraging the simple checks

- Testing use of element and attribute values
  - Testing the value of an attribute or element is a great way to make sure information is being added correctly – however this can add significant time to a batch related QA pass
- Validating proper use of data structures
  - Confirming the combined use of required elements and attributes (regardless of the data within them) is an important QA pass – it let's you know how the writers are doing
- Validating related data
  - Evaluating the use of data in one element or attribute and comparing it with values in another can be a significant benefit, but do you want to do this in a BREX DM or from within an executable that has the horsepower to do it better and faster?

Note: Resolving a BREX "rule check" to a data module usually uses ***interpreted*** languages – not compiled programs (the equivalent of Javascript). This is very slow. An interpreted language is not the same as a compiled executable. Recommendation, use JAVA, or a .NET enabled programming language instead and create an EXE program instead of a BREX DM to perform validations.

What's up with Layered Business Rules?

- Disappointment in tech pub land
  - The layered business rules paradigm does not work in the real world (if you take the blue pill it does work – if you take the red pill it doesn't)
  - Double-binds whenever there is more than one customer per project
  - Deviations (or waivers) from a "customer" based layered rule are nearly impossible to obtain – therefore businesses are choosing to ignore the layered hierarchy in order to deliver product in a sensible manner
- What's working and what's not
  - "Customers" don't really understand what business rules are all about
  - "Customers" don't think content providers deliver publications to more than one customer (the egocentric attitude of we're the only ones out here)
- What we're seeing in the real world
  - Costs are escalating to support layered rules when multiple customers exist
  - Overlapping "customer requirements" defeat the ability to create content consistently and at a reasonable cost
  - Content providers must ignore the rules of some customers but accept the rules of others – potentially alienating smaller customers
  - This creates a situation of a two class system of customers; if you're not big enough, the content provider may ignore a smaller customer's rules in favor of making a bigger customer happy

Layered Business Rules (cont'd)

- Downside of layered business rules to a technical publications development team
  - Customers are getting into the business of "knowing" S1000D. Should they?
  - Why does the recipient of content need to know the inner workings of S1000D?

    If customers are going to be in the business of tweaking the content/data modules then they might as well take on the responsibility of aligning the content to suite their needs and take the pressure off the technical publications groups developing the original content (thereby reducing the initial cost of content development).
  - Multiple customers with divergent requirements are escalating the cost of development
  - If customers create BREX data modules and you have multiple BREX DMs to validate against, how can you possibly resolve all the development issues?
  - Customers who think they know S1000D are coming up with rules like the following:

```
<structureObjectRule>
  <objectPath allowedObjectFlag="1">//dmAddress/dmIdent/dmCode/@infoCode="00S"</objectPath>
  <objectUse>Prohibited exclusion of the LOEDM information code "00S" </objectUse>
</structureObjectRule>
```

    The above states all data modules must use the information code "00S". Not a good idea.

```
<structureObjectRule>
  <objectPath allowedObjectFlag="1">//refs</objectPath>
  <objectUse>Prohibited exclusion of the required element /refs. /refs must be used.</objectUse>
</structureObjectRule>
```

    The above indicates ALL instances of <refs> available within any schema MUST be used. Again, not a good idea. Also, do you have any idea how many times you'll see this message?

Setting the stage for success

- Proper analysis and design
  - Layered business rules are not working, therefore, don't respect the layered business rule concept – but do document your rules
  - Take into account the "desires" of customers, but only use what makes sense across all customers
- Data defense is king
  - Using business rules for getting everyone on the content development team on the same page is the best use and solution overall
  - Using the BREX DM for QA is important, but it may be more important to create a faster more flexible method of performing QA using JAVA or a .NET enabled program
- Streamlining the workflow
  - Workflow is a business' best friend, it can make the difference between success and failure
  - Leveraging workflow as part of the business rules can save money and time

Reeling in practical testing

- What makes sense to test using a BREX Data Module?
  - Performing tests of “values” in an element or attribute is probably not the most practical use of a BREX DM
  - Custom JAVA or .NET enabled applications can do a much better and faster job of testing content in an XML file than a BREX DM
  - A BREX DM is great for simple feedback to a writer about what they did right and what they did wrong (did you use the correct element or attribute?)
- What makes sense to test using something OTHER than a BREX Data Module?
  - Testing values of an element or attribute where complex associations or conditions need to be evaluated are the domain of executable programs with JAVA or .NET like capabilities
  - You can’t test the data module filename of a data module against the <dmCode> element within the <identAndStatusSection> of a data module using a BREX DM – however a .NET or JAVA application can

It’s about being practical

- Business Rules are about evaluating what’s important
- Don’t just go with your first blush answer, think about the overall ramifications across all schemas and publications
- Success is in knowing how much of the BREX DM your project should try to implement and how much should be deferred to another method (Java or .Net)
- Communicate with the project team as much as is practical and use the BREX DM to enforce the important points
- Communicating also includes documenting decisions so everyone knows where to get an answer and how to work
- Avoid the mad dash to implement. Think about how you want to do business and use those features of the specification that make sense using a phased approach (don’t try to do it all in one lump)

Thank you for your time and I hope you gained extra understanding of how to use and implement BREX.