



how to

mine, manage, document and extract

business rules

from

existing IT systems

a blackboxIT white paper

This article is part one of a two part publication in the November Business Rules Journal – www.BRCommunity.com . It explores the benefits of finding and re-using those business rules in your existing systems that are actually running your business today. Part two will expand on the proposed methodology and examine a tool based approach.

The business rules in your existing systems

With "Business Rules" having been a topic for discussion for several years, with many tens of published success stories, and with an upsurge of standards activity, it is worth stepping back and asking "what has really been achieved?" Rules-driven architectures & technologies are mature, and Business Rules approaches to requirements & design are well-understood, yet it appears that Business Rules projects still tend to be peripheral to the Business; considered suitable for new "user facing" functionality, to support Business Process extensions, and when, as a last resort, a "legacy system" must be retired in favour of some new, modern, technology implementation. Business Rules are still considered as just one more technology option, rather than being considered as a fundamental component of Business requirements and a day-to-day interface between Business and IT.

At the root of this is the fact that most core Business Systems are still implemented on the mainframe, using "legacy" architectures and languages. They were not built with a conscious awareness of Business Rules, and the applicability of using a Business Rules approach in relation to them has generally either been overlooked or consciously dismissed.

It remains the case that the same rationale for using Business Rules in relation to "green field" developments would apply for core systems. If ways can be found of introducing Business Rules into the development, maintenance, and evolution of these systems, then perhaps the "holy grail" of flexible, compliant, business systems supporting agile businesses can be made more achievable.

"Mining" of Business Rules from so-called "legacy" applications has been proposed as a desirable activity for several years, but has never become mainstream. This is largely due to mismatching of expectations between tools vendors and Business Rules practitioners, and to the potential end-users' perception of an extreme level of complexity for the task. In this article I want to argue that gaining an understanding of the Business Rules that are **actually** enforced by core Business Systems, rather than simply those which one might be led to expect based on the understanding and expectations of current users, owners, and developers, is not only of significant advantage to businesses seeking agility and compliance, but is also, in fact, a feasible thing to do.

The importance of Business Rules

The significance of Business Rules for today's businesses derives from three of their primary drivers of business change:

- The quest for Agility
- The demands of Regulatory Compliance
- The perennial requirement to drive down cost

The Quest for Agility

In a very real sense, an organisation's Business Rules **are** its business – they determine how it responds to external and internal stimuli, and what outcomes it strives to achieve. The combination of suitably chosen Business Rules (there is probably no such thing as a "right" or

"wrong" Business Rule), together with appropriate enforcement policies, result in a successful business. Business Processes define the steps which have to be taken to apply the Business Rules.

When an organisation seeks to become more agile by re-structuring the way it does business it must, of necessity, re-structure its Business Processes. The objective, however, in such an activity is generally **not** to change the outcomes or responses, only to produce them more effectively. That is to say that whilst the **process** must change the **rules** that the process enforces must **not** change – the Business Rules are invariant.

As part of such a Business Process restructuring it is inevitable that the implementation of many core Business Systems will have to change. Reducing process latency typically implies a move from batch to straight-through processing; Business Rules which have been enforced in a distributed manner - in "time" (across batch runs), and in "space" (across different programs within the runs) - must be enforced "on demand". Existing enforcement algorithms and architectures will have to change.

In such a situation it cannot be an acceptable risk to rely for an understanding of the Business Rules upon the memories, experiences, and expectations of today's owners and users, however well these may be harvested. Today's operational systems have evolved over long periods of time, under the guidance of disparate individuals, many of whose expertise is no longer accessible. Many scenarios coded into the systems may have never, or only rarely, been encountered, or are obsolete and potentially forgotten. Experience shows that it is more than likely that Business Rules are enforced within those systems whose details have been forgotten. There also seems a high likelihood that omission of rarely used enforcement policies and "out of the ordinary" business events may lead to inaccurate or incomplete rules capture. The Business Rules that are **actually** enforced are those which are coded into the operational systems. The risks inherent in incomplete harvesting of the operational Business Rules are significant, and may well provide a major inhibitor to successful process re-organisation.

It is also the case that knowing **where** and **how** Business Rules are enforced within operational systems must significantly reduce the development risk, and potentially the development effort, associated with re-implementing the enforcement within a new, more agile, architecture.

The Demands of Regulatory Compliance

Prompted by a number of significant business failures, a number of industry sectors are placing upon themselves, or are having placed upon them by governments, a significant burden of Regulatory Compliance. Increasingly this means that an organisation must be able to "prove" that its operational processes produce outcomes whose characteristics match a number of mandatory criteria. Legal sanctions for violation can be onerous, often placing personal liability on individual officers of the company.

However strenuous an effort is made by an organisation to state a complete and consistent set of Business Rules and to demonstrate or prove that they are complete and consistent with respect to the external regulation, compliance remains purely theoretical unless it can be reliably related to the operational systems that enact the processes. In the end it must be necessary to prove that the operational systems **do** enforce all of the Business Rules necessary to ensure compliance, **and** that no Business Rules that are enforced in the operational systems are in conflict with that compliance.

To achieve such proof requires, first of all, that the expression of Business Rules is adequately formal – otherwise no convincing proof can be achieved. Formal proof of a theoretical model, however means nothing in relation to the operational business. For the proof to be meaningful it

must be possible to present **all** of the Business Rules which are operationally enforced, and to be able to show exactly **how** and **where** the Rules are enforced – what combinations of machine and human activity cooperate to perform it.

The Cost Imperative

Every organisation in today's business climate faces the need to do more with less – especially within IT. "IT latency" – the cost of change within operational systems – is seen as a significant overhead. The archetypal approach involves some or all of Outsourcing, Package replacement, and IT Infrastructure change. In all these cases, visibility of Business Rules will provide considerable benefit.

Outsourcing

Typically outsourcing organisations support a wide range of applications suites outsourced from businesses operating within the same or similar market sectors. The outsourcer will have a deep understanding of the generic domain, but must somehow take on board the specific differentiators, and the specific development history, of each individual client – hence the necessity of transferring staff, with their specific knowledge, as well as applications. This is potentially both a risk and a burden; if, on the other hand, detailed knowledge of Business Rules enforced, with information as to **where** and **how**, can be transferred, or acquired, the risks are greatly reduced.

Package Replacement

It is very common for a business to seek to divest itself of IT development risk by seeking to replace core business components with packaged solutions. This can be a very effective solution, but packages, typically, are written to handle generic capabilities, and will enforce generic Business Rules, with generic policies, based on a generic understanding of the business domain. The systems that they replace, however, have evolved over time to suite the specific differentiators and preferences of the owning organisation. The risk in adopting a package comes with the difference between the generic capability of the package and the specific needs of the current operational business, and the cost of configuring and/or extending the package to suite.

Once again the key issue becomes that of comparing the Business Rules enforced by the package, with the Business Rules enforced in the current systems. Hopefully a package supplier is able to define what their rules are, and how they can be changed. It is, however, necessary to have a complete picture of exactly which Business Rules are enforced by the operational systems – in order to ensure completeness & consistency of the new package solution – and where they are enforced – in order to ensure that the correct scope for the replacement is understood, together with its implications for other, interacting, systems which are not being replaced.

IT Infrastructure change

Perhaps the only kind of change, in this area, which does not necessitate awareness of operational Business Rules is the so-called "lift and shift" approach to infrastructure change. In this scenario the objective is simply to transfer unchanged functionality and behaviour between one execution platform (typically a hardware architecture, such as the mainframe) and another (typically a lower cost, "open" platform such Windows or Linux). Whilst this can have very significant cost impact in both the long and the short term, it fails to deal with the costs and risks of the IT "people" infrastructure – that is, the perceived cost and time delay associated with achieving change within the implementation of the operational systems.

Much of this cost arises from the difficulty of accurately communicating Business needs and translating them into implemented behaviour in such a way as to ensure that the desired outcomes are effected, and that other outcomes are not affected. This comes down to ensuring that only the Business Rules that must be changed are actually changed. It should be clear that

an explicit awareness of the Business Rules enforced, and **where** and **how** the enforcement occurs will have a significant impact – even if the Business Rules are not explicitly externalised as they would be in a fully rules-based architecture.

Business Rules within Operational Systems

What are Business Rules?

After so much discussion of “Business Rules” in relation to Operational Systems and Business change, it is perhaps advisable to make clear exactly what is meant by “Business Rule”

The original GUIDE report defined a Business Rule thus: “a statement that defines some aspect of the business. It is intended to assert business structure or to control or influence some aspect of the business”. Implicit in this are a number of important points

- Business Rules are about the way that a business works, not about its Information Systems
- Business Rules are about the controls or guidance applicable to Business Processes, they are not the processes themselves – “Business Rules” are not the same as so-called “Business Logic”.
- The enforcement of a Business Rule is separate from the Rule itself. Different business events may necessitate differing responses to the application of the same Business Rule.

Herein lies the issue. Every Business Rule is applied by some combination of human and IS activity – embodied in operational procedures and executing applications code. The processes and procedures are algorithmic statements of a strategy chosen (generally from a number of possible alternatives) to perform the enforcement and effect the resulting outcomes. The benefits we have discussed above derive from having clear visibility of the rules, accessible to, and comprehensible by, all the relevant stakeholders, from business owners through to implementers. Whilst this may often exist for manual procedures (most business will have a “Company Procedures” or “Standards” manual) most “legacy” application architectures will fail to separate rules from enforcement policies, and will distribute the implementations throughout the code, in line with the dictates of efficiency, the preferences of the implementer, and the architecture of the implementation.

From a Zachmann perspective, we may say that the Business Rules are “Row 2” Models, whereas the code of the operational systems and data which enforce the rules are “Row 4” Models. The difficulty of bridging this gap, with code as the starting point – which is clearly not a deterministic or automatable process – is at the heart of the problem for those who would seek to bring the benefits of a Rules-based approach to core Business Systems.

What does the enforcement of a Business Rule look like?

Business Rules, like any other rules, would be most accurately described using mathematical notations, however it would be unreasonable to expect business users (or even IT professionals) to use such techniques. Instead a variety of more or less “formal” modes of expression have been developed. Often these “languages” provide a range of template sentences which can be used to express Business Rules (and for which well-defined mathematical models exist – to allow for the possibility of formal rule verification). More recently the OMG’s SVBR standard allows for well-defined, generic, formalised expression of rules in a near-natural language form.

Leaving aside the details of the different models, it is convenient to take as a general form for Business Rules expression something along the lines of

<subject> <conclusion> <conditions>

for example

An urgent order	... <subject>
must not be accepted	... <conclusion>
if the order value is less than \$30	... <conditions>

A customer	... <subject>
may be raised from 'bronze' to 'silver' status	... <conclusion>
only if at least one of the following is true	... <conditions>
- the customer's account balance has	
been positive for the last 12 months	
- the customer's average account balance	
over the past 12 months exceeds \$500	

A customer	... <subject>
must be considered as cash-only	... <conclusion>
if they have no credit clearance	... <conditions>

In terms of a code implementation we might therefore expect to see Business Rules enforced by

1. A set of computations performed on some inter-related program data (the <subject> & its "properties") ...
2. To determine a value which is tested in expressions and/or stored for later testing (the <conclusion>) ...
3. Occurring as a result of some set of conditions evaluated within the application control flow (the <conditions>)

The problems here lie in the fact that it is unlikely, in general, in a conventional application architecture that

1. the computations which determine the conclusion or the conditions which apply to it would be represented by single, or even contiguous sequences of, statements, or
2. The data items which represent the business meaningful concepts referenced in the rule are directly identifiable for what they are – a specific database or file field may be recognisable, but at the point at which a Business Rule is applied, the data used is likely to be local program data that is derived from the original business data through assignments, parameter passing and file or database writes and reads.

As a simple example, consider the following COBOL code fragment

```
EXTPRICE.  
0004B4    MULTIPLY L-QTY-SHIPPED BY L-LIST-PRICE GIVING L-EXT-PRICE.  
  
DISCOUNT.  
0004BD    MOVE L-CUSTNO TO L-DISC-CUSTNO.  
0004C3    MOVE L-PRODCODE TO L-DISC-PRODCODE.  
0004C9    MOVE L-QTY-SHIPPED TO L-DISC-QTY.  
          MOVE L-DISCOUNT-CALC TO W-DISCOUNT-CALC.
```

```
0004CE    CALL "CALDSC" USING W-DISCOUNT-CALC.
0004D1    COMPUTE L-DISC-PRICE ROUNDED =
           L-EXT-PRICE * (100.0 - W-PERCENT-DISC) / 100.0.
           MOVE W-PERCENT-DISC TO L-DISC-PCNT.
```

We can see (part of) the enforcement of a Business Rule

The nett price of an order line must be computed as the list price multiplied by the quantity shipped multiplied by the applicable discount provided that a special deal has not been offered

("applicable discount", and "special deal" are, of course, themselves defined by separate Business Rules)

The conclusion of the rule is applied by two, non-contiguous, statements: the MULTIPLY at 0004BD, and the COMPUTE at 0004D1, and the conditions for the rule are not visible within the fragment at all, because they are embodied in the conditions under which the fragment is executed within its containing program, and the conditions under which the program itself is executed, within the running application.

Finding and Recognising Business Rule enforcement

In conventional mainframe batch architectures this distribution of Business Rules can be even more extreme. Daily batch runs will collect information and evaluate conditions applying to one or more Business Rules, or apply classification rules, storing interim (or even tentative) conclusions as "markers" in database fields, or in files. Subsequent runs may modify or override these conclusions, and the final conditions necessary to establish the conclusions of the actual Business Rules are evaluated during a consolidating weekly or monthly run.

It is this "temporal" and "spatial" distribution of rule enforcement that makes the identification of the Business Rules based on operational code such a challenge. Simply looking at a fragment of algorithm in isolation cannot possibly provide any information as to its Business relevance, let alone permit its interpretation as a "Business Rule".

The recognition of the Business Rule behind the code fragments discussed above depends, fundamentally, on the recognition that the COBOL data field L-DISC-PRICE represents something that the business refers to as a "nett price", that a nett price is a property (a fact about) an entry in an order, and that its calculation involves two disjoint steps, linked by data (and indirect control) flow.

In order to recognise Business Rule enforcement, three considerations are necessary

- The Business Vocabulary, and its manifestation at the "boundaries" of the system
- The data dependencies over time and space which determine how business meaningful results are derived
- The control flow dependencies, over time and space, which determine the conditions under which the results are determined.

Business Vocabulary

In our example above the field name 'L-DISC-PRICE' carries implications (at least for an English speaker) of its purpose. Such informal assumptions provide some clues, but given, for example

```
ADD 1 TO OUT-PARAM.
```

As a fragment of code, its relevance to Business Rules is far from clear. It is a computation, but the purpose of OUT-PARAM determines its relevance: an iteration counter may not have Business relevance (but the boundary conditions of the iteration may well embody some condition in a Business Rule); on the other hand, its use in a credit scoring subroutine may render it extremely significant.

Critical to any Business Rules "mining" is the development of a Business Vocabulary, and the mapping of that vocabulary to operational program data.

Vocabulary (or "Fact") models are well documented in the Business Rules literature, and a recent OMG standard "Semantic Vocabulary of Business Rules" is targeted as bringing standardisation to this area. In its simplest form a Business Vocabulary defines an agreed the terminology which an organisation can use to describe its "business". Typically this involves "terms" whose meaning can be agreed - "Customer", "Order", "List Price" - and "facts" which can be known or determined; sentence templates which can be used to make statements about the terms - "Order is placed by Customer", "Order contains Order Line", "Customer may be a VIP".

It is important to distinguish a "Business Vocabulary" from a "(Business) Object Model". Typically "object models" are implementations which mix Business Vocabulary and Rules - cardinality, for example should not be considered in the Vocabulary model. For example a 1:N cardinality of a relationship "customer places order" relates, in practice, to a variety of Business Rules and partial Rules, which might be things like:

- A Customer may place an Order provided that they have no more than 3 outstanding Unpaid Orders
- A Customer is any client organisation which has placed one or more Orders in the past 5 years.

An implementation choice may be made to embody some of the conditions of the Business Rule implicitly in the data structure (the possibility of a Customer placing multiple orders). The chosen cardinality may represent a clue as to the enforced Business Rule - changing the cardinality will effect the Rule, but not the Vocabulary.

A Practical Approach to Business Rules Mining

Once a Business Vocabulary is established, a basis exists for discussion between a Business Rules "miner", and the business users and owners of the system under consideration. The system will have a variety of "surface artefacts", which are visible products of its operation: screens are displayed, reports are printed, databases are updated. Much (but probably not all) of the information carried by these artefacts serves to convey the Business Outcomes of the operational process. Indeed these surface artefacts are the only data elements of a system that can readily have such a meaning attached. Because these are the aspects of the system which are visible to the system's owners and users, the analyst can use the content, layout, and structure of the artefacts to associate technical artefacts - typically program, file, or database elements with elements of the Vocabulary: for example

"that field is the available funds we have determined for the applicant, and it is red because the value is insufficient for the application"

Enables the analyst to "tag" the variable displayed in the field as representing an "applicant has available funds" Fact, and the data element determining the screen colour at the point at which it is displayed with an "applicant is entitled to loan" Fact.

This tagging represents a “hard” link between “IT knowledge” and “Business knowledge”, and provides a access point into the internal structure of the applications. If a practical and cost-effective combination of tools and methodology for supporting the analysis of data and control flow within them can be established, then it becomes feasible to consider using the operational code to drive a discovery of operational Business Rules.

In the final part of this paper, I will examine the necessary capabilities of such tools, and outline a potential methodology for their use.

About the author

John Pocock is CTO of blackboxIT Ltd, a specialist Business Rules tools and consultancy company, based in Bournemouth, England. With a 30 year background in software tools and methodologies and 7 years specialising in legacy understanding and modernisation, John is a leading expert in the field of Business Rules in relation to legacy systems.

He may be contacted as john.pocock@blackboxIT.com, or on +44 (0)1202 233210.

blackboxIT products

With extensive experience of mainframe applications blackboxIT have created **emergeIT**, a dedicated business rules mining product.

In conjunction with our evolveIT product and our unique methodology and services, **emergeIT** enables you to identify where and how business rules are enforced within your operational code and build a fully cross-referenced inventory of business rules and vocabulary alongside the application code that supports them.

From our business rules inventory, our **exportIT** products can be used to assist in re-hosting your business rules into alternative technologies including rules engines and web services.

The foundation for these capabilities is provided by **evolveIT**, which analyses your code to provide a detailed structural understanding of your applications together with comprehensive models of their data and control flows.

For more information, visit our stand in the exhibition or visit

www.blackboxIT.com

info@blackboxIT.com

+44 1202 233 215