The Decision Model #2: Improving Process Models and the Requirements Process

Barbara von Halle & Larry Goldberg

Abstract

This is the second in a three-part series introducing the Decision Model. The first of these three articles reveals how the Decision Model, as a model for business logic, becomes a strategic instrument of business agility for business leaders. This second article explains how the Decision Model revolutionizes today's business processes and requirements management. The third article reveals how the Decision Model revolutionizes the testing process. The audience for this article includes business architects, requirements analysts, use case modelers, business process modelers, and agile developers. This article establishes the Decision Model, not only as a new business asset, but also as a new requirements artifact and an imperative element in decision-aware business processes.

Background

The Relational Model changed the way we manage, leverage, and store data. The discovery of this technology independent-model of data brought about the information revolution by enabling us to separate data and realize and leverage its considerable value as an organizational asset. Today, the Decision Model, as a technology-independent model of business logic, promises to release perhaps even greater value from our technology assets and business processes.

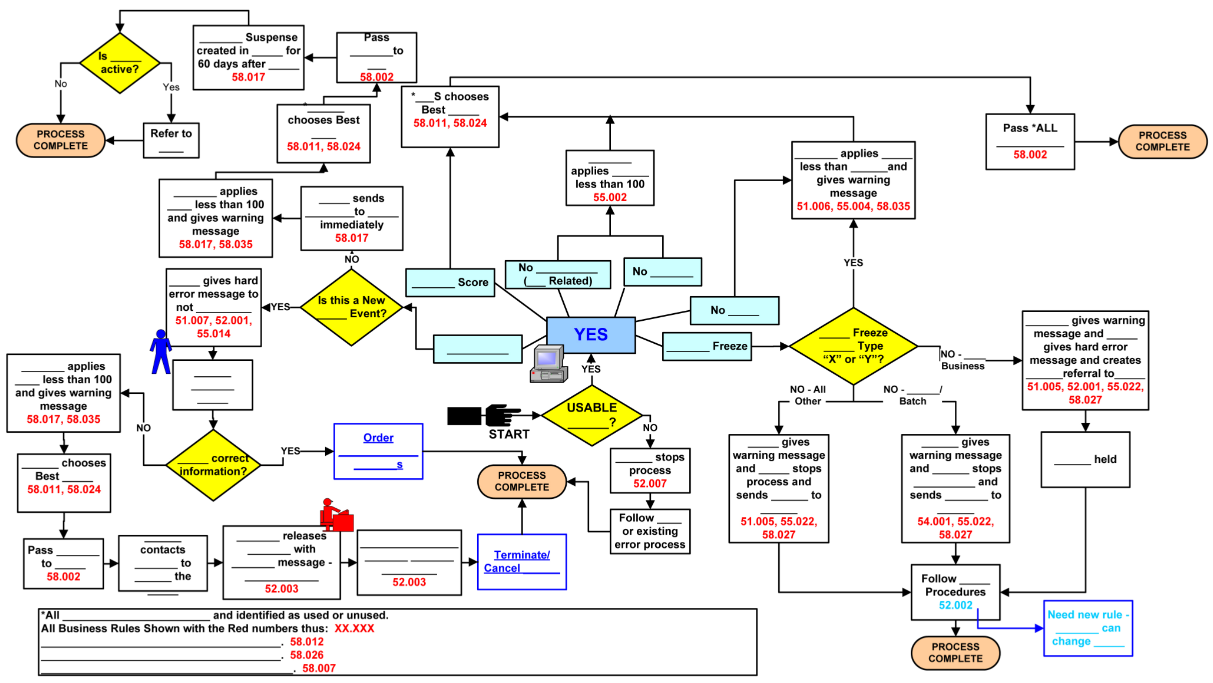
The Decision Model organizes business rules and business logic in a similarly rigorous manner to the way the Relational Model organizes data. It enables innovation in a wide range of business endeavors, including business planning and management, business architecture, business transformation and re-engineering, and business process management. The Decision Model also brings significant opportunity and agility in information technology practices, including requirements capturing, application development and testing, and application maintenance.

The Decision Model recognizes that business logic has its own existence, independent of how executed, where executed, and whether or not it's implemented in automated systems. It has a recognizable structure different from other model structures. Decision Model normalization principles minimize redundancies and anomalies in the business logic. Details of the Decision Model and commentaries on its place in the world of business and IT are provided in the book The Decision Model: A Business Logic Framework Linking Business and Technology (von Halle and Goldberg 2009). This article consists, in part, of abstracts from the book; directly quoted passages, diagrams and tables are cited, and are copyright © 2009 Auerbach Publications/Taylor & Francis LLC, and Reprinted with the permission of the Publisher.

To gain a rapid understanding of the Decision Model as insight into the ideas in this article, please read or refer to the Decision Model Primer.

How the Decision Model Modernizes Today's Business Processes

The most visible and dramatic impact of the Decision Model is how it simplifies business processes and renders them truly agile. This impact is best understood by looking at a real-world example.



Is _____ active?
No
Yes
PROCESS COMPLETE
Refer to _____
_______ Suspense created in _____ for 60 days after _____ 58.017
Pass ________to ____ 58.002
chooses Best _____ 58.011, 58.024
*___S chooses Best _____ 58.011, 58.024
Pass *ALL __________ 58.002
PROCESS COMPLETE
______ applies _____ less than 100 55.002
_______ applies _____ less than ______and gives warning message 51.006, 55.004, 58.035
_______ applies ____ less than 100 and gives warning message 58.017, 58.035
_____ sends ________to _____ immediately 58.017
NO
_____ gives hard error message to not _________ 51.007, 52.001, 55.014
YES
Is this a New _____ Event?
_______ Score
No _________ (___ Related)
No _______
No _____
YES
_______ Freeze
_____ Freeze _____ Type "X" or "Y"?
YES
NO - _____ Business
_______ gives warning message and _____ gives hard error message and creates ______referral to_____ 51.005, 52.001, 55.022, 58.027
NO - All Other
NO - ______/ Batch
USABLE _______?
START
NO
_______ stops process 52.007
Follow ____ or existing error process
_____ gives warning message and _____ stops process and sends ______ to ______ 51.005, 55.022, 58.027
______ gives warning message and ______ stops __________ and sends _______ to ______ 54.001, 55.022, 58.027
______ held
Follow _____ Procedures 52.002
Need new rule - _______ can change _____
PROCESS COMPLETE
_______ applies ____ less than 100 and gives warning message 58.017, 58.035
NO
_____ correct information?
YES
Order __________ ________s
PROCESS COMPLETE
______ chooses Best _____ 58.011, 58.024
Pass _______ to ______ 58.002
______ contacts _______ to ______ the ______
_______ releases _______ with _______ message - __________ 52.003
__________ ______ ______ __________ 52.003
Terminate/ Cancel ______
*All ______________________ and identified as used or unused.
All Business Rules Shown with the Red numbers thus: XX.XXX
__________________________________. 58.012
__________________________________. 58.026
__________________________________. 58.007

Figure 1 Business Process Model without Regard for Business Decisions

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

Figure 1 depicts a business process model created by a project team aiming to make business rules and logic visible but separate from the process model. The team worked with experienced consultants in the Business Rules Approach and had spent two years creating this model. The red notations in the model are some of the business rules and logic, but closer inspection reveals that additional business logic is built into the process flow itself and some of it is not represented at all. The result was that the business rules, captured as lists, annotations and process tasks, became unmanageable. There were overlaps, redundancies, and inaccuracies; some visible and some hidden.

At this point, the team introduced the notion of the Decision Model. After just a few weeks, the business process model of Figure 2 emerged. You can see that it is much simpler because we removed from the business process model the business decisions (and their complete business logic) and recast each in their own Decision Model. More importantly, the business process flow now can be changed without changing the business logic and vice versa.

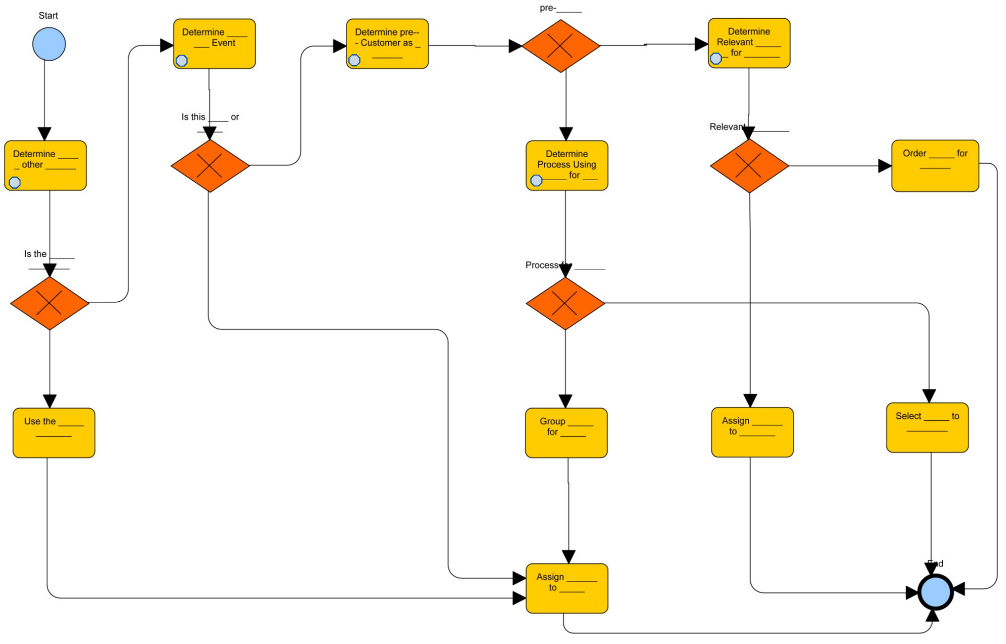


Figure 2 Business Process Model with Business Decisions Separated

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

So, this deliberate separation of business process and business decision into their own models proved crucial to the ability to manage business logic and also transformed the business process into one that is decision-aware. You may be asking: what does it mean to separate business processes and business decisions?

“To separate business processes and business decisions, they must somehow be different from each other in a recognizable way. It turns out that they are truly different in a very significant way. In fact, the inherent nature of a business process is very different from that of a business decision. To date, however, this difference has not been well understood, but the advent of the Decision Model brings this difference to the forefront. Essentially, a business process is procedural in nature, but a business decision is declarative in nature. However, without a clear understanding of declarative versus procedural nature, common practice involves creating business process models in which business decisions are loosely represented as just another part of the business process. In other words, it is common practice to model business processes and business decisions in a procedural manner rather than modeling the latter in a declarative manner. This common practice not only constrains the business decision unnecessarily, it seriously hinders agility for both the business process and the business decision. Understanding the difference between a business process and a business decision means distinguishing and preserving the difference between a procedural versus declarative solution.

A procedural solution specifies how, in a step-by-step manner, something is to be done. So a business process model is a procedural solution because it prescribes a set of tasks that are carried out in a particular sequence. The business process model is the "How" of a unit of work. A declarative solution, on the other hand, only specifies what needs to be done, with no details as to how, in a step-by-step manner, it is to be carried out, because sequence is irrelevant to arriving at the correct result. A Decision Model is a declarative solution because it is a set of unordered business logic, not a set of ordered tasks. A Decision Model is the "What" of a special kind of unit of work. "HOW means saying how, step by step, the work is to be done; WHAT just means saying what the work to be done is" (Date, 2000). The declarative Decision Model for a business decision should be removed from the procedural business process so that it can be managed separately in a declarative form. Separating business decisions from business process tasks simplifies the business process model, offers more creativity in organizing the business logic, and delivers the business logic in a form that transcends technology options." (von Halle and Goldberg 2009)

Let's take a closer look at distinguishing a procedural solution from a declarative one.

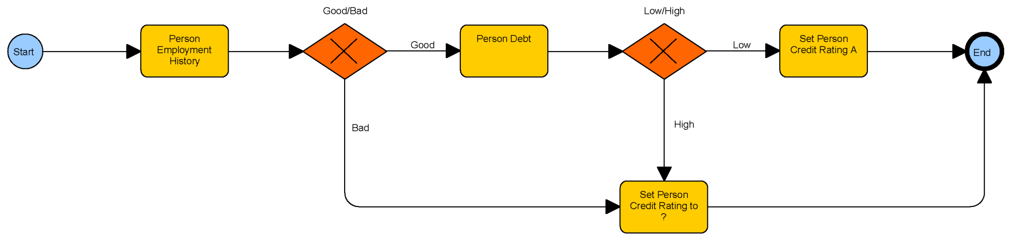
How to Distinguish a Procedural Task from Declarative Business Rules or Logic

"Option 1 in the business process models in Figure 3 prescribes that first the process determines a person's employment history, and then if the result is good, the process next determines the person's debt. If the debt is low, the process sets the person's credit rating to "A." However, if the results are bad and high, respectively, the process branches elsewhere. In this business process flow, the sequence of evaluating the business criteria or conditions is set in a specific order. The process flow is rigidly specified, not allowing for alternative sequencing.

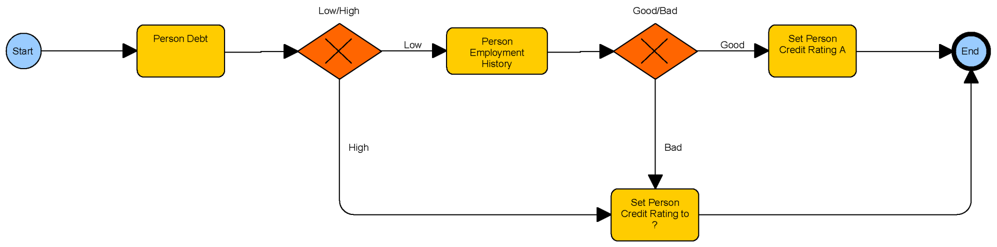
Yet Option 2 prescribes a different sequence that also works, although this process flow likewise does not allow for alternative sequencing.

Option 3 offers a significant improvement simply by removing the declarative business decision from the procedural process flow. It represents a simpler process flow consisting of only one task. That task combines the whole previously sequential set of tasks into one task, denoted as a decision task, behind which a business decision executes in a declarative fashion. Within the process flow, the decision task looks like any other task but contains a decision shape within the task box. Option 3 also includes the Decision Model diagram, which puts the Decision rule Family in context with its related Rule Families, and Option 3 includes a Rule Family table for the Decision Rule Family. Neither the Rule Family table nor the Decision Model diagram is embedded in the process flow. They are separate deliverables, anchored to the process flow by the decision shape.

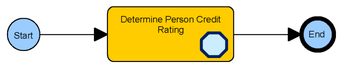
Option 1


Start
Person Employment History
Good/Bad
Good
Bad
Person Debt
Low/High
Low
High
Set Person Credit Rating A
Set Person Credit Rating to ?
End

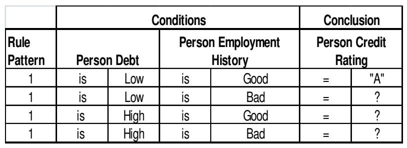
Option 2


Start
Person Debt
Low/High
Low
High
Person Employment History
Good/Bad
Good
Bad
Set Person Credit Rating A
Set Person Credit Rating to ?
End

Option 3

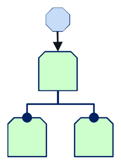


Process Model



	Conditions				Conclusion	
Rule Pattern	Person Debt		Person Employment History		Person Credit Rating	
1	is	Low	is	Good	=	"A"
1	is	Low	is	Bad	=	?
1	is	High	is	Good	=	?
1	is	High	is	Bad	=	?

Rule Family Table



Decision Model Diagram

Figure 3 Procedural vs. Declarative Solutions

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

The Rule Family, by definition, implies no particular sequence among the conditions to be tested. The Rule Family in Figure 3 also indicates via the "?" that there are other possible combinations of conditions to consider. The Rule Family can contain as many rows as are needed to reach the correct conclusion. For that matter, it can contain additional columns if other conditions are needed to determine a person's credit rating. The Rule Family table also contains business logic for the logic not modeled in the business process models of option 1 and option 2. These include the adjudication of the credit rating for all values of person's debt and employment history other than "low" and "good." Incorporating these into the business process model rather than in the Decision Model would have enlarged and added unnecessary complexity and unnecessary sequence to the business process model. To change or add conditions in such a business process model is far more cumbersome than doing so in the corresponding Rule Family.

Immediate observations are that Option 3 is an improvement over Options 1 and 2 because it:

- Allows a much simpler business process model

- Easily highlights all possible combinations of conditions
- Permits changes in the Decision Model without changing the business process model
- Permits changes in the business process model without changing the Decision Model" (von Halle and Goldberg 2009)

Less obvious is that you will most likely never know all of the business logic when you create business process models similar to Options 1 and 2. Let's investigate this further.

How a Business Process Model Never Reveals All Business Logic

"Figure 4 shows two business process models for determining Food Stamp (FS) eligibility. This is an entirely imaginary process, and one that has no bearing on an actual food stamp eligibility process. It is, in fact, drawn from an eligibility process for a completely different federal program, but has been changed to preserve the anonymity of the real process.

Option 2 depicts a much simpler business process model than does Option 1. That's because Option 1 depicts a sequence of process tasks that are forced to occur in a particular sequence but for which such sequence is actually not required. The business process model is simplified by removing parts of it that can be represented in a declarative Decision Model. So, the high-level Decision Model in Figure 4 represents the business decision "Determine FS Eligibility." Although the Decision Model contains business logic from several of the tasks in Option 1, namely, FS Eligibility and Children Qualification, it also contains several Rule Families that are not represented by process tasks in the business process model: Citizenship Status, SSN Validation, Employment Status, and Income Qualification. So, mingling business decision logic with process flow as in Option 1 does not necessarily expose all of the business logic in that process flow.

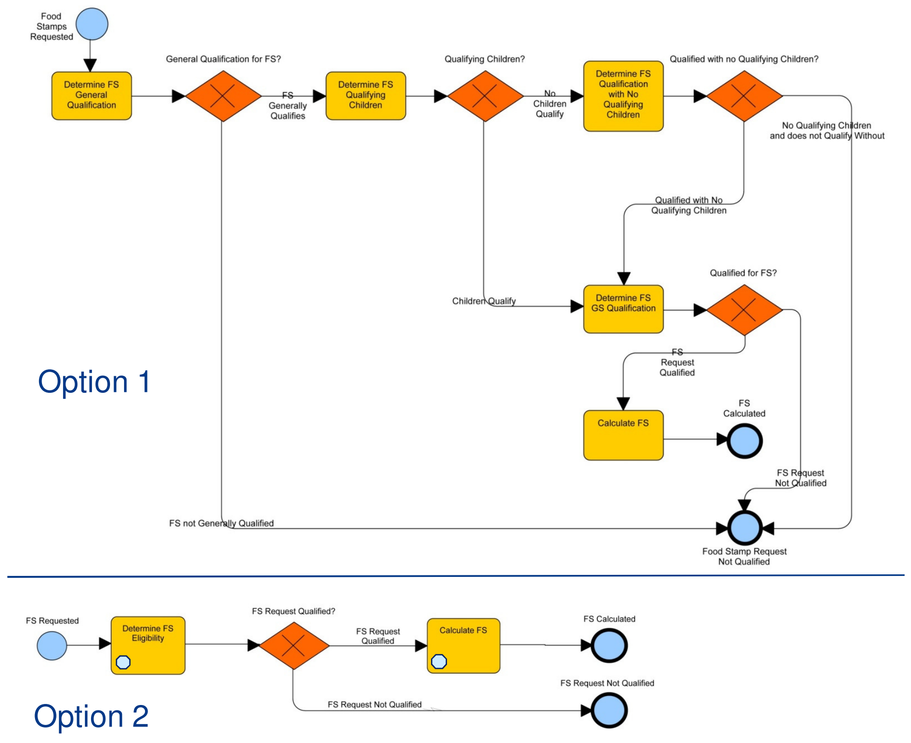

Food Stamps Requested
Determine FS General Qualification
General Qualification for FS?
FS Generally Qualifies
Determine FS Qualifying Children
Qualifying Children?
No Children Qualify
Determine FS Qualification with No Qualifying Children
Qualified with no Qualifying Children?
No Qualifying Children and does not Qualify Without
Qualified with No Qualifying Children
Children Qualify
Determine FS GS Qualification
Qualified for FS?
FS Request Qualified
Calculate FS
FS Calculated
FS Request Not Qualified
FS not Generally Qualified
Food Stamp Request Not Qualified
Option 1
FS Requested
Determine FS Eligibility
FS Request Qualified?
FS Request Qualified
Calculate FS
FS Calculated
FS Request Not Qualified
FS Request Not Qualified
Option 2

Figure 4 Food Stamp Processes

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

But then where is the missing business logic in business process models like those in Option 1 of Figure 4? Apparently, because all of the business logic is not directly visible in the business process model, some of it must be buried in one or other of the tasks. In fact, it probably is buried in many places, because some of it may be used in several of the tasks (which are the case in this business process model).

So, it is likely that some of the business logic is hidden from view in a procedural business process model such as in Option 1. Further, it is difficult, if not impossible, to resurrect all of it in one visual artifact—not even by drilling into the detail behind each of the process tasks. Therefore, the business logic is rendered unmanageable.

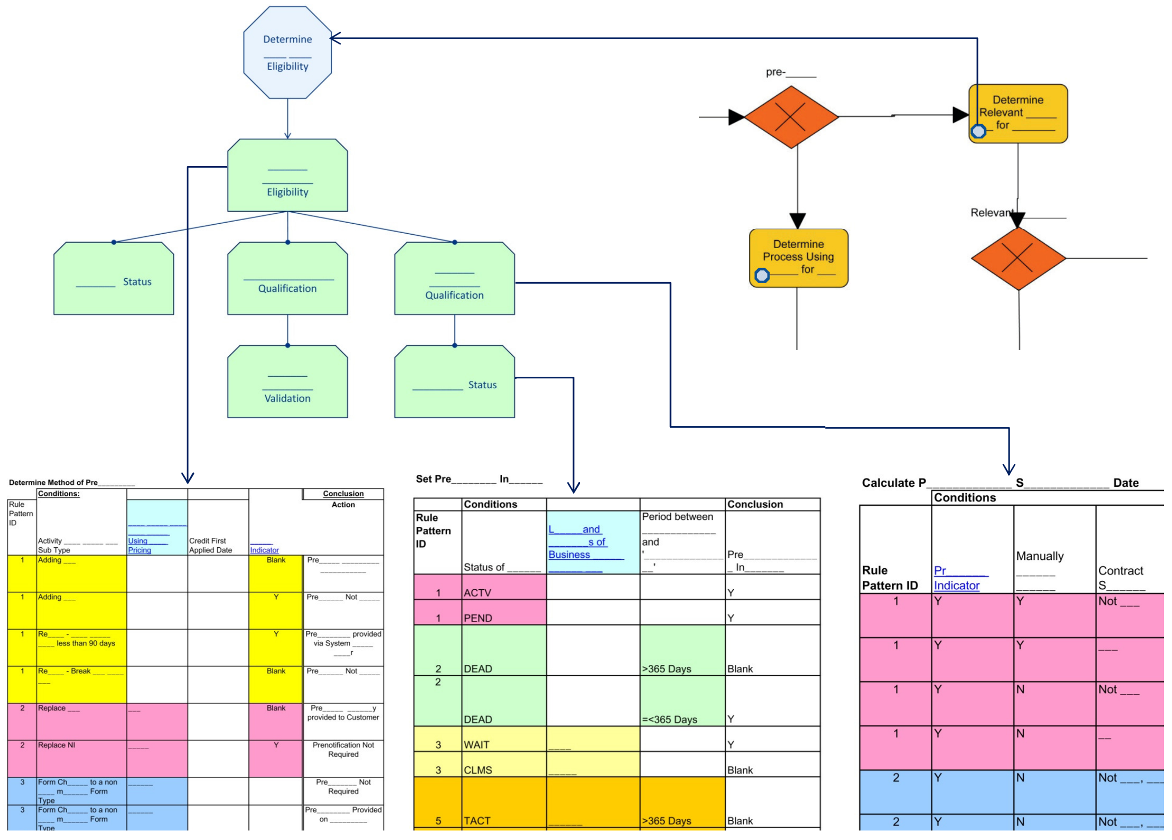
On the other hand, the Decision Model, by definition and purpose, resurrects all of the business logic in one visual artifact. In a populated Decision Model, all business logic is clearly visible in one place and assists in rapidly and accurately gauging the impact of suggested business logic changes without reviewing every task in which some portion of that business logic may reside." (von Halle and Goldberg 2009)

So, a Decision Model is declarative and contains complete business logic, but does this add complexity, time, and cost to developing business processes and their models? Quite the contrary.

How a Decision Model Delivers Simplicity, Productivity and Cost Savings to a Business Process

As you already know, “the business process model in Figure 2 is a reengineering of the one in Figure 1, but with one subtle and important difference: The business decisions are noted simply by the decision icon within decision tasks. The corresponding business logic statements or business rules are nowhere on this diagram. This solution depicts tasks in a prescribed sequence, differentiating those tasks representing a conclusion carried out through business logic. The detailed business logic is captured in corresponding Decision Models.

Simplification became obvious for this project when the quantity of business process models for the entire project was reduced from approximately 25 to 10. Additionally, at least 5 of those business process models were reused elsewhere because removing business logic resulted in more generic business process models. Most of the Decision Models also became candidates for reuse.



Determine Method of Pre________

	Conditions:				Conclusion
Rule Pattern ID	Activity ____ _____ ___ Sub Type	___ ____ ____ ____ ____ Using ____ Pricing	Credit First Applied Date	_____ Indicator	Action
1	Adding ___			Blank	Pre_____ _________ __________
1	Adding ___			Y	Pre______ Not _____
1	Re____ - ____ _____ ____ less than 90 days			Y	Pre________ provided via System _____ ____r
1	Re____ - Break ___ ____ ___			Blank	Pre______ Not _____
2	Replace ___	___		Blank	Pre_____ ______y provided to Customer
2	Replace NI	_____		Y	Prenotification Not Required
3	Form Ch_____ to a non ____ m______ Form Type	______			Pre_______ Not Required
3	Form Ch_____ to a non ____ m______ Form Type	______			Pre________ Provided on _________

Set Pre________ In______

	Conditions			Conclusion
Rule Pattern ID	Status of ______	L_____ and _______s of Business _____ ______ ___	Period between ____________ and '_____________ __'	Pre____________ _ In_______
1	ACTV			Y
1	PEND			Y
2	DEAD		>365 Days	Blank
2	DEAD		=<365 Days	Y
3	WAIT	____		Y
3	CLMS	_____		Blank
5	TACT	______	>365 Days	Blank

Calculate P_____________ S_____________ Date

	Conditions		
Rule Pattern ID	Pr______ Indicator	Manually ______ ______	Contract S______
1	Y	Y	Not ___
1	Y	Y	___
1	Y	N	Not ___
1	Y	N	__
2	Y	N	Not ___, ___
2	Y	N	Not ___, ___

Figure 5 Business Process Model, Decision Model, and Rule Families

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

The diagram in Figure 5 illustrates one of the project’s business process models and how its Decision Models represent all of the business logic alluded to in the original deliverable but more completely, accurately, and visually. The business process model, devoid of business logic in Figure 5 contains visible anchor points (e.g., decision shapes within task boxes) for five Decision Models. One of those Decision Models is shown in Figure 5, complete with its Decision Rule Family and five other Rule Families. Three of those Rule Families appear as Rule Family tables.

With these deliverables, the original set of approximately 250 random groups of business rules was reduced to approximately 20 business decisions totaling 51 Rule Families in third normal form. It took much less time to create the revised business process models, new Decision Models, and populated Rule Families, compared to the original "combined" deliverables which were ultimately abandoned as unmanageable. Further, the "pure" business process models, Decision Models, and Rule Families in Figure 5 were dramatically easier to implement in the target technology than the original deliverables." (von Halle and Goldberg 2009)

More details on how the Decision Model improves Business Process Management and models as well as other case studies appear in Chapter 4 of the book.

Now that you understand the distinction between a business process model and a Decision Model, it will now be easy to understand how this separation and integration advances the way we capture and manage requirements today.

How a Decision Model Advances Today's Requirements Process

"Requirements consist of those elements of the system over which the business user is expected to (or should) provide. Thus, models of business processes—both the "as-is" and the "to-be"—and elements such as user interface are requirements.

Consequently, different kinds of requirements are expressed in different ways, each aiming to communicate the requirement in the most understandable manner. These methods include textual statements, business use cases, a wide range of different models, prototypes, and in certain approaches, the program code itself." We explore each of these prior to introducing the Decision Model as a replacement or enhancement to requirements, in general.

Requirements Artifact #1: Textual Statements

A series of textual statements has been long used as the principal means for expressing requirements. These have the form of "The system shall …" (for mandatory or obligatory requirements), or "The system should …" (for preferred requirements). Effective textual statements are difficult to write because natural language is imprecise and does not represent rigorous communication. If textual statements are developed as the sole source of requirements, the risk of misunderstanding and rework is high. The best use of textual statements is to serve contractual purposes. In other words, the software is to be assessed against them.

Requirements Artifact #2: Business Use Cases

A business use case is a step-by-step narrative of a single business user ("Actor") interacting with the system, where the narrative yields a valuable result. An example of a business use case for an online banking system is “Prequalify a Loan Request.” A complete business use case provides a basic course of action when all goes well, as well as alternative paths when something goes wrong. There may be many alternative paths in a full business use case, as well as several additional pre- and post-conditions.

Business use cases help stakeholders better understand requirements within the context of an actor interaction. In some cases, textual requirements statements are actually developed from the business use case.

Requirements Artifact #3: Models

The dictionary defines a model as “a description or analogy used to help visualize something (as an atom) that cannot be directly observed” (Merriam-Webster Online Dictionary “model”). This means that models give rise to requirements because the models help visualize an aspect of the system to which the requirement applies.

Models represent different aspects of the real system (i.e., borrowing from the phrase in the aforementioned definition, “aspects of reality”) so that each aspect may be addressed individually, in a simplified manner. For functional requirements, the aspects of primary concerns are reporting, data, presentation, workflow, and business logic.

Models provide graphical and semantic representations of their target aspects in ways that are easier to view and understand than are textual statements or narratives. Because each model captures just one aspect of the system, several models, each orthogonal to the other, provide a sufficiently complete view of the proposed system. So, like the business use case, models are used both for shared understanding and assistance in developing related textual requirement statements. In some approaches, a model may be used as input for the automatic generation of the system. When this is so, it is called Model-Driven Architecture (MDA).

There are at least three reasons for modeling each aspect with its own model. First, each model reduces complexity. Second, each model enables independent change of its aspect from other aspects. Third, it turns out that each aspect is best represented by a model specifically tailored to the characteristics of the modeled aspect. For example, data is best modeled in diagrams that show the relationship of each data entity to another, where icons represent the data entities and lines represent data relationships. On the other hand, process flow is best modeled in diagrams that represent flow of process tasks, where icons represent the process tasks and lines represent process sequence.” (von Halle and Goldberg 2009)

So, how should business logic be captured: as textual statements, annotations to business use cases, other options, or as a model in its own right? The answer lies in understanding the nature and agility of business logic.

The Issue of Business Logic in Requirements

“Requirements for business systems change because of a wide range of circumstances. However, business logic is the aspect that changes most frequently. Not only does it change more frequently than do other aspects, but the lead time for implementing the change can be very short. Competitive pressure in the world of global business is one driver. Once a need for change is identified, the business is often under pressure to implement change immediately. Today, more than ever before, when crude oil and other commodity prices can swing over 30% in a matter of days, when the business climate can change from a fair outlook to economic recession in a matter of weeks, and when government can introduce wide-ranging regulatory legislation within months, the cycle time for change is becoming very compressed.

So, the compressed change cycle demands that business logic emerge as a critical aspect of business requirements. More importantly, business logic needs to be implemented and managed in a way that assumes it will change at any time. Today and for the foreseeable future, business logic must be designed to change, by definition." (von Halle and Goldberg 2009)

So, are we managing business logic today in a way that expects it to change at any time?

Current Methods for Capturing Business Logic (Business Rules) Fall Short

"In publications on requirements practices, there are references to business rules. However, little is said of their structure in any detail. The most common recommendation is to indicate simply, where in a business use case or business process model a business rule applies and to create a cross-reference from that point to the business rule number or name. In some cases, a business use case simply contains a

list of business rules or their numbers at the bottom of the business use case description; they are not even attached to specific steps within it.

There is also a maintenance difficulty with connecting one business rule at a time to points in a business process model or business use case. Most often, groups of business rules are to be connected to a point in business process models and use cases. When the nature of that group changes (e.g., business rules are to be added, removed, or changed), the change needs to be made in many places because the concept of a well-defined group of business rules does not exist. In other words, the new business rules must be added to, removed from, or updated at every relevant connection point in business process models and business use cases.

Until now there has been no model specifically for the representation of business rules or business logic. So, there is no concept of connecting a model of business rules or business logic to other requirements-oriented models. Without connection points and common metadata, it becomes difficult, if not impossible, to relate the business rules and business logic to the rest of the requirements. This means that developers, coders, testers, and ultimately the business users, will not be able trace the business rules and business logic to the models that reference them. This, to a large degree, explains why, despite good intentions, the business rules become "lost" in the system, even when a classical Business Rules Approach is followed.

Requirements Artifact #4: The Decision Model

The Decision Model emerges as the model of the business logic—an aspect of a system that has not previously been modeled in a universal way. The Decision Model shares metadata with, and has connection points to, the other models commonly used in requirements today, including business planning models, business process models, semantic models, organization and location models." More details on how the Decision Model connects to these models are in Chapter 6 of the book. Because the business use case is a popular requirements artifact, we discuss its connections to the Decision Model below.

How the Decision Model Connects to a Business Use Case

The connection between a business use case and the Decision Model occurs at those use case steps identified as decision steps. In practice, the use of decision-oriented words, such as determine, calculate, estimate, validate, and so on, are good clues that a business decision exists behind the step. In the case of Table 1, steps 6, 7, and 8 are denoted as decision steps. Once the decision steps in a business use case are identified, a Decision Model is needed, if one is not already in existence. The use case step needs a unique identifier for each business decision noted in it, which becomes the connection point between the use case step and the Decision Model. In practice, the connection is strengthened by using the same naming convention for the business decision and the corresponding use case step." (von Halle and Goldberg 2009)

Use case: "Pre-qualify loan request"
Step 1: The use case begins when Applicant logs on.
Step 2: Applicant selects loan type
Step 3: System determines if applicant is known (alt 1)
Step 4: System presents loan-specific screen
Step 5: Applicant reviews/changes information
Step 6: System determines if Applicant's income is sufficient **for loan request**
    Decision #1067: Determine if Applicant Income is sufficient for loan request
Step 7: System determines Applicant's probability of default
    Decision #1068: Determine Applicant probability of default
Step 8: System determines if applicant prequalifies for the loan request (alt 2)
    Decision #1070: Determine if Applicant prequalifies for loan request
Step 9: System determines other terms for the loan request
    Decision #1071: Determine other terms for loan request
Step 10: System presents terms to applicant
Step 11: The use case ends when applicant accepts loan terms.

Table 1 Steps From a Use Case Example Adding Decisions

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

Let's now summarize a full set of requirements artifacts that include Decision Models.

How the Decision Model Fits with Classical Requirements

"Figure 6 illustrates how classical requirements fit together, including the Decision Model. The central box labeled "Project Models" illustrates all of the models that comprise the project models, including the Decision Model. External to the models are relevant project textual statements representing requirements. A person can view one of these statements and discover related models, and also view a model and discover related textual requirement statements. The project requirements also trace, where appropriate, to enterprise models and enterprise textual statements. In addition, there is a relationship of test scripts to business decisions and to the business scenarios.

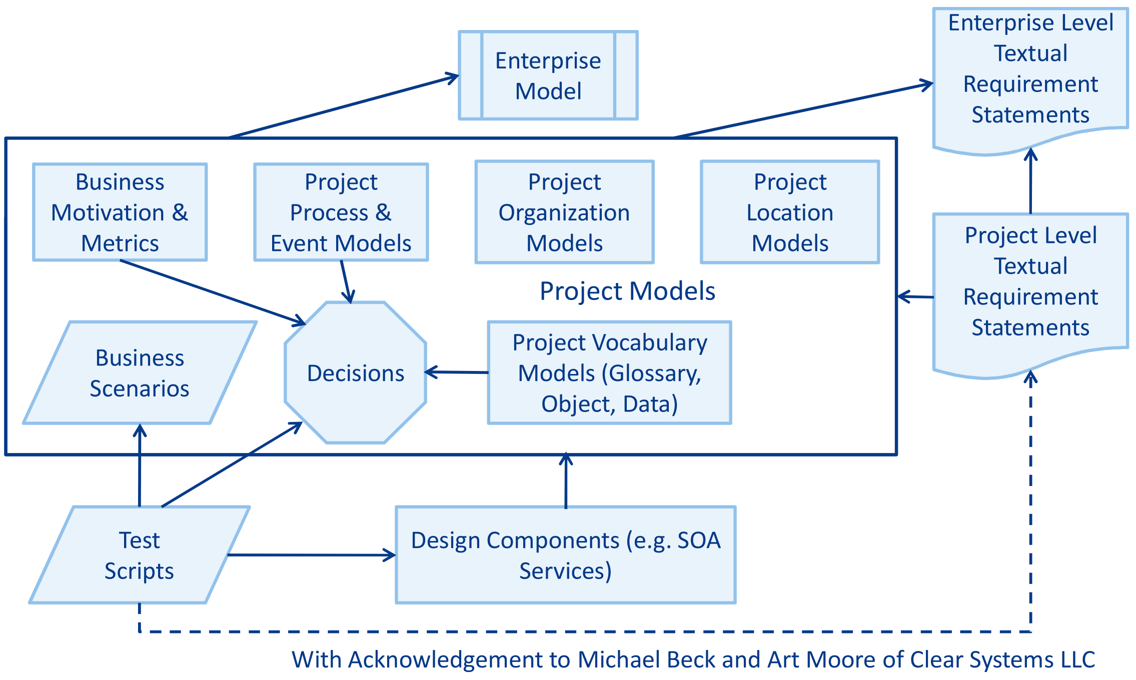

Enterprise Model
Enterprise Level Textual Requirement Statements
Business Motivation & Metrics
Project Process & Event Models
Project Organization Models
Project Location Models
Project Models
Project Level Textual Requirement Statements
Business Scenarios
Decisions
Project Vocabulary Models (Glossary, Object, Data)
Test Scripts
Design Components (e.g. SOA Services)
With Acknowledgement to Michael Beck and Art Moore of Clear Systems LLC

Figure 6 A Diagrammatic Views of Functional Requirements

Source: The Decision Model (von Halle & Goldberg) © 2009 Auerbach Publications/Taylor & Francis LLC. Reprinted with the permission of the Publisher.)

Beck and Moore explain the success they experienced in integrating business rules into the models in a large government agency:

… This … represents a validation of one final observation: Business rules were always there in our business and/or our system models and requirements statements. They may not have been separated out or completely stated, but they were there someplace, otherwise they would not, for the most part, have been implemented at all (a nod to rogue programming). It makes sense that if business rules do represent a specific, essential component of a holistic business model, then bringing more order and rigor to that component would further clarify and organize the rest of the model and its parts, and this is exactly what we have discovered to be the case. (Beck and Moore 2006).” (von Halle and Goldberg 2009).

By now you may be questioning the role of the Decision Model in agile development where formal requirements are not created or maintained.

The Agile Development and the Requirements Artifact: Program Code

"Proponents of the Agile development approach prefer a radically different approach to the classical method of gathering of requirements in advance of development. In the Agile approach, business requirements fully emerge during an iterative process between developers and stakeholders at the same time as, or very shortly before, the program code is developed. In other words, Agilists do not consider requirements to be an asset that is maintained in addition to and separate from the working system. Models and business use cases serve only as aids to shared understanding and guidance for developers and testers.

Yet, not only is the Decision Model still useful, it is ideally suited to Agile methodologies. Reasons are as follows:

- The business logic modeled in the Decision Model fits into a very well-defined scope, the business decision. Because that scope is typically a single step in a business use case or task in the business process model, it is an appropriately bounded scope for one increment that Agile development seeks for its iterations.
- The Decision Model serves as an ideal mechanism by which domain experts share understanding with programmers about the business logic for a business function.
- The Decision Model provides the business logic in a form that is compatible with its programmatic representation. This ease of transformation from the modeled business logic to program logic allows simpler, perhaps fewer and shorter, interactions between developer and business experts. An opportunity also exists for an Agile developer to maintain structures that are like Rule Families in the code such that the code truly mirrors and becomes a requirement that the user can easily understand.
- The Decision Model can easily be drawn and detailed without technology, using POW (Plain Old Whiteboard), index cards, or spreadsheets.

Summary

Experience with the Decision Model indicates it is the ideal artifact for representing business logic in requirements. This is true whether those requirements are developed in a comprehensive formal process, or developed in an informal, iterative process designed to discover the true intent of the business expert.

Because business logic is closely bound to the other models in requirements, it is natural that all models connect together, sharing metadata. These connections make possible the inclusion of the Decision Model into functional requirements as a universal model of business logic, and solve the problem of separating the business logic from other aspects.

But that is only the part of the story. The Decision Model also plays an important role in linking business and technology, the subject of the first article of this series. The Decision Model also plays a significant role in improving how systems are tested for conformance to the functional requirements. Therefore, the next and final article in this series reveals how the Decision Model revolutionizes the testing process.

To learn more about the Decision Model, readers can order the book at www.TheDecisionModel.com.

Works Cited

Beck, Micheal, and Art Moore. "The Business Rule Revolution." In *The Business Rule Revolution*, by Barbara von Halle and Larry Goldberg, 119-146. Cupertino, CA: Happy About, 2006.

von Halle, Barbara, and Larry Goldberg. *The Decision Model.* New York: Auerbach, 2009.