A Smarter Model Risk Management Discipline Will Follow from Building Smarter Models

An Abbreviated Guide for Building the Next Generation of Smart Models

Jon R. Hill, Ph.D.

Email: jonhill@optonline.net

Keywords: *model risk management, governance, validation, dynamic model inventory, model usage, transponder function, model-embedded, active intelligent agents, machine learning, SR11-7*

Introduction

What if a financial firm decided to delete its entire set of models and redevelop them from scratch. What might it do differently in the process of rebuilding its entire model eco-system in order to avoid and leverage from some of its previous mistakes? How could such a firm make Model Risk Management (MRM) platform smarter and less resource intensive than it was before?

This article will present the proposition that much of the manual efforts required by today's Model Risk Mangers (MRM) can be automated by introducing a few simple innovations, in the form of active intelligent agents, into actual model source code. Similar to the ways that tech firms have tracked the usage of their smart phones, cars, laptop computers and printers for many years, active intelligent agents embedded in model source code can support the creation of a dynamic model inventory to serve as a repository of historical data that accurately describes how, when and where a firm's models are used and to diagram firm-wide inter-dependencies between data and models. A smarter model risk management begins with the creation of smarter models. This paper will elaborate on the properties of a 'smart model'.

Initial Thoughts About Building the Next Generation of 'Smarter' Models

Quantitative models have been commonly used by banks to pursue their business interests for well over 50 years. Small financial firms such as regional banks, hedge funds, insurance companies or credit unions might have on the order of a hundred or so models in inventory while the leading investment and top tier commercial banks often register with several thousands.

For various reasons (such as increasing awareness and regulatory pressure) the number of models in inventory tends to increase year by year at nearly every financial firm. (Students of model taxonomy sometimes colloquially refer to the phenomenon of ongoing inventory growth as 'model creep'.)

For several years many leading financial firms have been pursuing the incorporation of Machine Learning (ML) as a well-publicized pathway to creating a new generation of 'smart' models. This article will examine the potential advantages of another less familiar path to smarter models.

The Crux of the Proposition

Model risk management at financial institutions is often realized through three Lines of Defense (LOD) as described in the Federal Reserve Board's signature document known as SR11-7 (https://www.federalreserve.gov/supervisionreg/srletters/sr1107.htm). The 1st LOD is comprised of model developers, owners and supervisors. The 2nd LOD is an independent Model Risk Management function that is responsible for both model governance and validation. The 3rd LOD is performed by Internal Audit (IA), which is charged with oversight over the 1st and 2nd LODs.

It will be argued in this article that Model Risk Management could greatly benefit from the introduction of a few simple innovations embedded into a model's source code. These innovations would take the form of active, intelligent agents that can enable models with a rudimentary level of 'self-awareness' that can be leveraged in various ways to reduce the manual and labor-intensive workload of Model Risk Managers. However, such intelligent agents would have to be developed and inserted into model source code by the 1st LOD developers

since they create and are responsible for model source code. But the benefits of the embedded intelligent agents would accrue mainly to the 2nd and 3rd LODs which are required by SR11-7 to be as independent of each other as possible. Therein lies the crux of the matter: model developers and model risk managers invariable operate in different silos within almost all large financial firms.

As a result, 1st LOD staff may have little or no incentive to undertake any additional efforts that only offer benefit to the 2nd and 3rd LODs. Successful implementation will almost certainly require support from management senior enough to exert authority over all LODs.

If Only We Could Start All Over and Rebuild Our Financial Models from Scratch

Although it would be unlikely to occur in the current cost-conscious environment, it can be a useful thought experiment to consider addressing the following questions: Imagine that a top tier financial firm decided to delete its entire set of quantitative and qualitative models and redevelop them from scratch? How would it leverage some of the painful lessons garnered from past mistakes in the process of rebuilding it entire model eco-system? In what ways might it transform its resource intensive MRM platform into one that is smarter and leaner, yet more effective?

Embedded Identity Tokens Can Be a First Step Towards Model Self-Awareness

Something that traditional financial models lack, with few exceptions, is any kind of rudimentary form of self-awareness. As this author has argued in a previous paper ("Shouldn't A Model 'Know' Its Own ID?", Journal of Structured Finance, Fall 2018 https://jsf.pm-research.com/content/24/3/89) model discipline at most financial firms is impeded by the unfortunate but simple fact that their models do not 'know' who they are. This is in the sense that unique model identification tokens are not embedded in the actual source code in the way that serial numbers and IP addresses are embedded in many intelligent devices such as smart phones, appliances, servers and printers.

What Form Would a Unique Model Identity Token Take?

The most basic identity token for a financial model should include a unique model ID (essential for inventory management), the model version number and/or usage ID. The version number will be important if model owners assign them in order to distinguish updates to their models; usage IDs may be assigned to distinguish specific applications of a model version. With this simple modification, requiring only a single line of source code added to the main routine of the model and perhaps a minute a developer's time, several immediate benefits can be realized that address recurring difficulties in model risk management.

What Immediate Benefits Might Model-Embedded Identity Tokens Offer?

If unique model ID tokens were to be attached to all outputs produced by a parent model, any ambiguity about which version of which model produced a particular result for a particular use will be eliminated. This is a real-world issue for many firms that store model outputs in large data repositories (sometimes called 'vertical systems') to be supplied as input to downstream models on request. A given model may have multiple variants that have been modified in different ways to satisfy the requirements of specific applications. Model variants often are not assigned unique IDs since they are derived from a parent model. Instead they may be assigned version numbers

and/or usage IDs. But unless the model ID and version number/usage ID are unambiguously associated with their corresponding outputs in a data repository, confusion can exist about the upstream origins of the data.

Such ambiguities could eventually be resolved by tracing backwards from the data repository to the upstream origins of the data, for example, to a Front Office (FO) pricing or Middle Office (MO) risk model. But why should it be a manual and resource-intensive process if instead the unique identifying information were already embedded as an attribute of every model output?

Embedded Intelligent Agents Can Be Leveraged to Capture the Dynamics of Model Usage

Model inventories as currently implemented at almost all banks, no matter how sophisticated in design and accessibility, are collections of static indicative data relevant to each model. Important and necessary indicative model data such as: model ID, model name, name(s) of model developer(s), name(s) of model validators, production date, intended uses, validation status, upstream and downstream model dependencies (from the model documentation) and the types and sources of all inputs to the model. Model inventories may include model outputs and their downstream applications, the number and risk rating of validation findings, exceptions granted, status of the remediation of each finding, validation dates and results of ongoing monitoring and other indicative data fields that model risk managers may find useful.

Static Model Inventories Do Not Typically Include Data About the Dynamics of Model Usage

However, today's standard static model inventories are characterized by an absence of data that captures the ***dynamics of model usage*****:** accurate historical data describing the *how, when and where* of model execution. For the purposes of this discussion a database of model usage information may be thought of as a dynamic model.

Few if any firms today can boast of having detailed knowledge of the dynamics of model usage based on actual execution data (as opposed to attestation by model owners and users, which is unreliable and error-prone). Such opacity is revealed by a systemic inability to answer the following types of usage questions with confidence and quantitative accuracy backed by actual historical execution data:

1) What is the exact number of different models that have been used over the last year?
2) How often has each model been executed, by day, by month, by year? Can you identify the most frequently and least frequently executed models?
3) Where are the firm's models being used? By business unit, legal entity, geographic regions?
4) Can your firm provide a complete list of the models used by each of the above entities over the last year?
5) Are there any models in your inventory with an active status that were not executed during the last year?
6) Are there any models that were executed on any of your firm's computers that do not appear in inventory? Please provide a full listing.

7) Is your firm able to provide a full list of the IDs of models that exhibit significant seasonality? If so, what are the peak and troughs of seasonal model usage?

8) Were there any instances of a retired model still being executed during the last year?

A Model-Embedded Solution to the Opacity of Model Usage is Both Portable and Scalable

The paper previously referenced by this author entitled *"Shouldn't A Model 'Know' Its Own ID?"* has proposed a portable and scalable solution for addressing these areas of opacity in model dynamics. The proposed solution relies on the introduction of unique model-embedded identity tokens operating in tandem with an embedded 'transponder' tracking function. The paper demonstrates in detail how all eight questions cited above could be answered by accurately and comprehensively tracking a few critical pieces of model-related information for each execution event: model ID combined with version number and/or usage ID, time stamp and MAC (Media Access Control) or IP (Internet Protocol) address.

The solution is portable because it is permanently embedded within the model source code and therefore travels with the model wherever it may be employed within a firm's model ecosystem. It is scalable because it does not rely on external monitoring by an execution platform such as those under the control of Information Technology (IT) staff to manage the operation of major risk, CCAR, CECL and other critical models. While such external monitoring approaches may be configured to capture model usage dynamics, they do so only for models that operate with the execution platform, which is a minority of the models found in the ecosystem of almost all financial firms. Most notably, such an external monitoring approach may not oversee the plethora of single standalone End User Computing (EUC) models (e.g. spreadsheets, R, Matlab, Java developed on individual laptops) that propagate readily at all large and small firms but are not subject to IT oversight. A solution that resides within the model code will by nature be independent of the limitations of external execution platforms and can potentially work equally well for all classes of models, including IT mainframe or EUC.

The proposed transponder function would provide model usage 'telemetry' that is similar in concept to the transponders used by all civilian aircraft that report the identity, altitude and velocity of the aircraft to civilian controllers. If this type of dynamic execution data telemetry were to be recorded for each model in inventory over time, a wealth of data about the dynamics of model usage would become available to model risk managers and amenable to sophisticated data mining techniques.

What can be accomplished by making models smart enough to report this type of usage telemetry for each execution event over significant period? The answer is "quite a lot", in combination with a little bit of imagination: historical tracking of the frequencies of model usage, comparative histograms of model usage frequency over specific intervals of time and maps of geographic usage. The graphical dashboard reproduced in Figure 1 demonstrates various ways of displaying usage tracking information for a synthetic portfolio of 100 models collected for 100,000 randomized execution events at various geographic locations:

A Graphical Dashboard for Model Transponder Simulation Outcomes

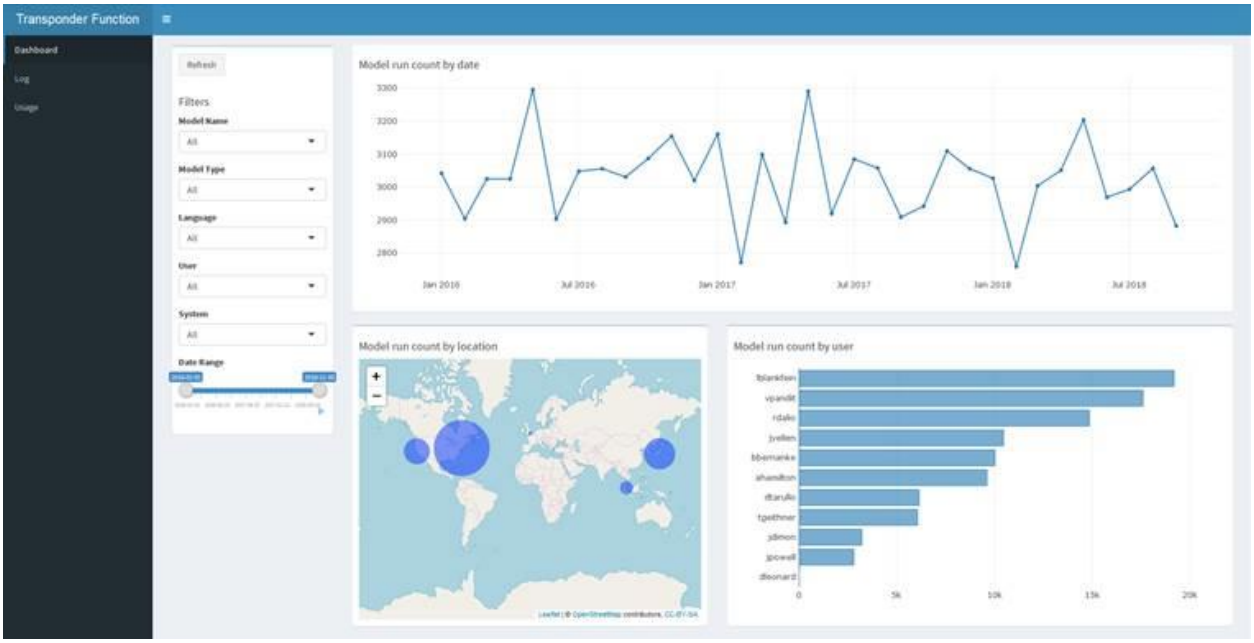

Figure 1 - *A prototype Transponder Function and dashboard display used for this simulation were developed in collaboration with the author by David Leonard at FI Consulting, Arlington, VA. The usage plots were produced by collecting 3 crucial data points for each simulated execution event: ID, timestamp and MAC/IP address (model name is optional). The graphical dashboard was implemented on an Amazon Web Services (AWS) cloud platform.*

Inherited Identity Tokens Can Capture Model Inter Dependencies Based on Execution Sequence

One of the more problematic challenges confronting today's model risk managers lies in creating a complete and accurate map of model and data inter-dependencies. This has been an area of recent focus by US regulators, as observed in previous papers by this author. Understanding model and data inter-dependencies is critical to gaining a fully holistic understanding of a firm's model ecosystem.

Model contamination refers to the collateral impact a serious flaw or error in one model or data stream may have upon an associated set of models that are dependent upon the flawed model. In order to track error propagation due to contamination it is critical for a firm to possess a complete, accurate and up-to-date map of mutual inter-dependencies within its model ecosystem. There are several ways in which firms attempt to construct such a map.

Many financial firms rely on attestations from model developers or supervisors to identify upstream and downstream model dependencies. Model developers should be able to determine upstream dependencies by tracing all input data back to its origin, be it another model or data from market or vendor sources. Other firms may employ sophisticated IT methodologies that scan through the source code of all models in its ecosystem in order to identify data and model dependencies.

Manual attestations and IT methodologies can potentially produce a complete and accurate map of model and data inter-dependencies, but only with substantial investment of resources. But even if successful, these methods will produce static snapshots taken at a single point in time. Depending on how often the mapping exercise is performed, the dependency snapshot may become stale as model and data inter-dependencies change over time with the ongoing

introduction and retirement of models (this is particularly true for models that employ Machine Learning they have the capacity for self-modification as conditions and data change).

This author has suggested an alternative, dynamic approach in a previous publication ("Shouldn't a Model 'Know' Its Own ID?") that has been further elaborated in a second paper ("The Top Fourteen Challenges for Today's Model Risk Managers", March, 2019, https://www.henrystewartpublications.com/jrm/v12). This dynamic approach relies on leveraging the embedded identity tokens described above and in greater detail the previous two papers. The basic concept is one that has been used by computer networks and email for decades: it relies on the passing of identity tokens from upstream models and data to their downstream recipients. Token-passing could create a dynamic map of inter-dependencies that is continuously updated as new models and data inputs are added or older models retired from the ecosystem.

Analogy to a Four-Person Relay Race

The dynamic approach to capturing model and data inter-dependencies can be understood most readily by analogy to four-person relay races often featured in track and field athletic events. In a relay race a single runner begins the race with a baton that he/she passes to the next runner after completing the first leg of the race. The second runner then passes the baton to the 3rd runner after completing the second leg, and so on until the 4th runner finally carries the baton past the finish line.

With one change this analogy can illustrate the token-passing concept. Imagine *instead* that each runner is equipped with their own uniquely colored baton. The first runner has a yellow baton that he/she passes to the second runner who has a red baton. The second runner passes both the red and yellow batons to the 3rd runner who has a green baton. Finally, the 3rd runner passes red, yellow and green batons to the 4th runner who has a blue baton. When the 4th runner crosses the finish line he/she will be in possession of 4 batons, yellow, red, green and blue, representing each completed leg of the race. Model and data embedded identity tokens are exactly analogous to the colored batons of this example, with some additional modifications: additional runners with unique batons may enter each for each leg of the race. This would correspond to multiple batons being handed to the downstream runner.

To implement this paradigm, an upstream model (or data input) would pass its embedded identity token to the downstream model that would collect the token from the input and add it to a token bucket list maintained by each model. The important feature is that the bucket list is dynamic in that it would be updated each time models in a chain of dependencies are executed. The final model in the chain of would have collected a complete bucket list identifying all contributing upstream models and data inputs.

After dependency bucket lists are collected for all models of interest in an ecosystem into a common data repository, it is relatively straightforward to use network graph theoretic tools to produce many-to-many network diagrams, capable of producing graphical representations of all model and data inter-dependencies for an entire model ecosystem. (An example of a network diagram demonstrating this ability to capture many-to-many relationships is reproduced in the JRMFI publication by this author.)

Such a dynamically updated ecosystem map could be produced at any point in time, replacing static snapshots for a particular point in time that can become stale. If implemented uniformly for all models in the firm's ecosystem, this approach would eliminate the need for model owner or user attestation or an exhaustive IT scan of all model source code.

Implementation Considerations for Model-Embedded Intelligent Agents

Most firms will doubtless be reluctant to undertake the effort required to create the next generation of smart models as it requires retrofitting existing models with embedded intelligent agents. This is due to the requirement for minor modifications to every model's source code (insertion of a unique identity token and a call to a transponder function), even though it would have no impact on model outcomes or performance. A brief summary of the pros and cons for implementing embedded active intelligent agents in model source code follow:

Advantages

The solution is simple, portable and readily scalable: It can be applied equally well to large IT-controlled or small standalone End User Computing (EUC) models. **The Solution Supports Accurate Mapping of Model Inter-Dependencies:** This will result in a dynamic up-to-date map of inter-dependencies within a firm's model ecosystem. **The Solution is Comprehensive:** Because it is platform independent it is a global solution that will operate on any computer with access to a firm's intranet. **The Solution is Amenable to Gradual, Incremental Implementation**: The proposed innovation can be implemented incrementally over time beginning with limited sets of models

Disadvantages:

The Solution Requires Touching Each Model: The solution requires several minor modifications to the source code of each model to be tracked, although performance and outcomes will not be affected. **Vendor models** present a special challenge since it is highly doubtful vendors would agree to install transponders in their models. **Spreadsheet (EUC) models** could present challenges as well, but they are not insurmountable. **High bandwidth** from heavily used models could bottleneck the Firm's intranet. There are straightforward alternatives such as parking results in temporary local file systems. **The Solution May Encounter Corporate Resistance:** most large, established financial firms tend to be resistant to change, especially to change that introduces unfamiliar innovations. Pushback from entrenched IT organizations should especially be expected.

Conclusion: Embedded Intelligent Agents Can Enable MRM to Reverse the Manual Attestation Paradigm

At most firms with an established model inventory database, the completeness and accuracy of the inventory, a key requirement of SR11-7, is determined through a manual process of attestation that involves polling model developers, supervisors and users (typically by email) to confirm the set models that they own or use. Model developers/owners are also queried to affirm upstream and downstream model inter-dependencies. Manual attestation is similarly employed by MRM to obtain information about how, when and where the models are used. Such resource

intensive processes often tend to be incomplete and error-prone, partly as a result of staff indifference and turnover, partly due to human error.

The error-prone MRM manual attestation paradigm employed at most financial firms has been described in greater detail in a previous publication by this author *("The Top Fourteen Challenges for Today's Model Risk Mangers"*, Journal of Risk Management in Financial Institutions, March, 2020, https://www.henrystewartpublications.com/jrm/v12).

Smart models can offer MRM a way to reverse the manual attestation paradigm. Instead of relying on model owners and users to attest about their model ownership and usage, smart models can be configured to track the how, when and where of model usage, as well as inter-dependencies within the model ecosystem, based on actual execution data through active embedded intelligent agents.

Summary

This paper has examined the proposition that Model Risk Management (MRM) can be made smarter by building smarter models, models that incorporate active, model-embedded intelligent agents.

The current paper has advocated the introduction of a unique identity token and a bespoke transponder tracking function into the actual model source code. Operating in tandem, these two innovations can create a stream of historical model telemetry that captures how, when and where a firm's models are being used. In turn, this historical model usage data can be used to populate a dynamic model inventory database that can be mined for the usage statistics as illustrated in Figure 1.

Smart models with embedded active intelligent agents can be leveraged to reverse the prevalent industry-wide MRM model attestation paradigm (described in previous publications by this author [3,4]). Instead of relying on model developers, supervisors and users to keep MRM managers updated about model usage dynamics (a manually intensive and error-prone process), smart models can be configured to continuously inform MRM management precisely about how, when and where they are being used through their embedded active intelligent agents.

Model embedded identity tokens also offer an innovative method for mapping model and data inter-dependencies within a firm's model ecosystem by passing identity tokens from upstream to downstream models through the entire chain of dependence.

Author's Bio

Jon Hill is an adjunct professor in NYU's Financial Risk Engineering Dept. where he teaches a graduate course in Advanced Model Risk Management, Governance and Validation. Jon also leads the New York Chapter of the Model Risk Managers International Association. With over twenty years of experience in diverse areas of quantitative finance at Citigroup, Morgan Stanley and Credit Suisse, Jon is recognized as a subject matter expert in model risk management, governance and validation and is the author of numerous publications on these topics.

Jon holds a Ph.D. in Biophysics from the University of Utah. He is a published author in the field of Model Risk Management and a frequent speaker and chairperson at model risk conferences throughout the US and Europe.

Email: jonhill@optonline.net