Article from

The Modeling Platform

November 2019

Issue 10

Writing Effective Model Documentation

By Dodzi Attimu, Ryan Bijur, Jeffrey Jardin, Michael Minnes and Veltcho Natchev

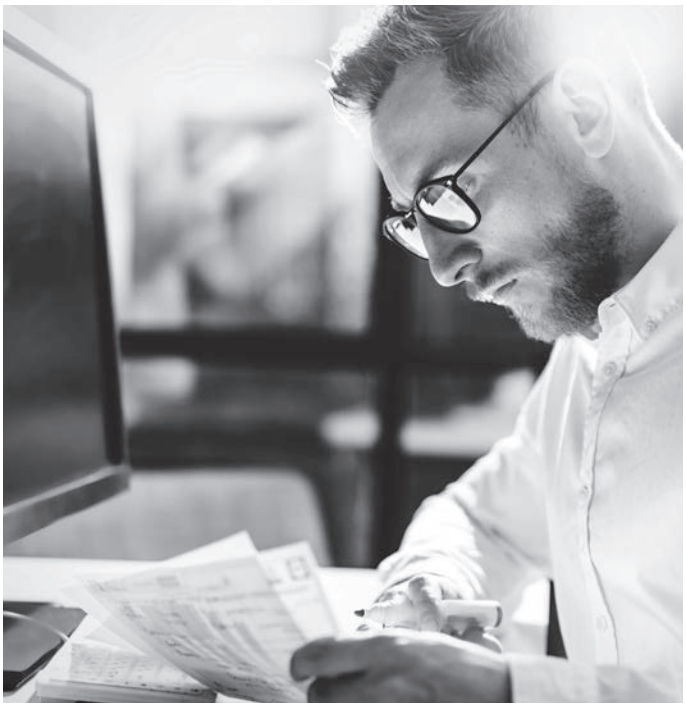
The main objective of technical documentation is to minimize key person dependency for the model users/owners and to allow any reasonably competent modeler (including a new one) to understand its methodology. The guiding principle of effective model documentation is to provide a description of the model's methodology and functionality and their implementation in support of the model's objectives and requirements.

Comprehensive documentation should contain sections on model inputs, calculations, outputs, limitations, associated business processes, governance practices, application and platform specifics. Documentation challenges arise with the operational components of a model: input structure, throughput and output structure.[1] Consequently, this article will focus on the documentation of the components:

- **Inputs.** Data, assumptions and parameters
- **Throughputs.** Model theory and calculations
- **Outputs.** Interim calculated variables and final model results

As much as possible, documentation of methodology should be platform, application and code neutral in the sense that formulas should be generic rather than reflect the programming language. This way, if there is a model conversion to an entirely different system, the documentation will not have to change (given that the methodology is the same/has not changed).

While focusing on content, it's important to remember that model documentation should be easy to maintain and update. For example, a common approach is to use lists and tables as centralized information storage units, potentially placed in an appendix, rather than creating lengthy descriptions in the body of a document. The documentation should reflect the model as it is, rather than how it should be, eliminating judgment by the documenter with the presumption that the model is working correctly.



Our approach to documentation is to use a single universal template stored in a central repository. The benefits of this include comprehensive coverage of model elements, documentation style consistency, information security and ease of use.

INPUTS

All models use inputs such as data (in-force extracts, scenarios, etc.), assumptions and parameters. It is important for the model documentation to have a detailed description of the input facility. A description of data derivation external to the model is not needed; however, there should be links to data sources.

A recommended approach would contain these steps:

- Specify the input structure and data elements (including assumptions) and sources, how they are entered or uploaded/downloaded into the system (model), for example, manually, via automated processes or combination of both
- Describe how inputs are changed and how the changes are tested
- Provide a detailed description of the input facility for the model, such as groups, tables, variables; these should include naming conventions

Inputs should be tied to existing model artifacts, such as assumption memos. This enables the user to understand how the assumptions are coded/used in the model. If the model uses

an open system (homegrown or third-party provided open-source code software), variables that read the input data should be listed.

For variables calculated using the inputs and then used for further processing within the same model, such as dynamic lapses or experience mortality, it would be appropriate to include them in the Input and Calculations sections. References to the calculations should be included where appropriate.

Input structures that support sensitivity/stress-testing should also be documented.

THROUGHPUTS (CALCULATIONS)

The calculations within a model reflect its theoretical underpinnings, and good documentation will not only present the formulas used in these calculations but also provide insight into the background of modeling simplifications and methods used. However, there is a trade-off between verbosity and accuracy: The more detailed the documentation, the higher the likelihood of unfaithfully representing its functionality.[2]

To effectively describe the model calculations, it is useful to:

- Describe how the model obtains its output. For instance, does the model aggregate cells from a seriatim projection? Does the model aggregate individual model cells? Does the model project a number of stochastic scenarios and determine the mean of a certain value?

- Outline what the model is projecting or representing along key dimensions (e.g., starting time, horizons, scenarios, elements that are being projected) in support of each of the model's uses (objectives, purposes and business processes). It is important to also clarify the order of operations, such as assessment of charges and crediting of interest.

- Decide whether to use a top-down or bottom-up approach to document the calculations, using the expertise, experience and discretionary knowledge of the documenter.

Example: Cash Flow Testing

To clarify the calculations, any aggregation used to produce the final output should be described (with formulas, if appropriate) and the core calculations should be documented for a representative "cell(s)" used in the aggregation variations in modeling for different cells should also be shown.

- It may be necessary to set the notation for the consecutive time variables $\cdots, t_{k-1}, t_k, t_{k+1}, \cdots$, for example, as follows:

  - In many instances, the beginning of period t_k is conceptually synonymous with the end of period t_{k-1}.

  - Another related consideration is the order of operations in a given time (period).

- Statutory surplus is calculated at each time t_k, for each Scenario ξ, for a policy as:[3]

  - $\text{Surplus}(t_{k+1};\xi) = \text{ProductRevenue}(t_k;\xi) + \text{NII}(t_{k+1};\xi) - \Delta\text{Reserves}(t_k,;\xi) - \text{Benefits}(t_{k+1};\xi) - \text{Expenses}(t_k;\xi)$.

- In some instances, it may help to specify the order of operations utilizing, for example, the following:

  - $t_k \equiv$ beginning of period t_k
  - $t_k^- \equiv$ end of period $t_{(k-1)}$
  - $t_k^+ \equiv$ an instant after beginning of period t_k post beginning of period deductions

  Or generically in words, for example: Fees are always deducted prior to calculation of investment income …

The next level of detail would involve documenting the individual components of the previous equation, including $\text{ProductRevenue}(t_k\xi)$, $\text{NII}(t_{k+1};\xi)$ and $\Delta\text{Reserves}(t_k;\xi)$. The components may need further documentation of their subcomponents until one gets to an appropriate level of detail, the lowest level being the input structure/variables. It is important to document the relationship between the input data structure and the variables that show up in the formulas as needed to clarify such relationships.

- Note the use of variables rather than concrete values—in the example, the time steps were denoted $\cdots, t_{k-1}, t_k, t_{k+1}, \cdots$. Of practical importance are the intervals between time steps, $\delta t_k := t_k - t_{k-1}$, as well as the last time step, t_N.

- Using the variables this way makes the document easier to maintain because, even if the current implementation consists of monthly projection time-steps for 40 years, one section of the document could mention that the current implementation has δt_k to be monthly with $N = 12 \times 40$. Other potential nuances like calender month or policy month consideration could be handled similarly.

This example utilizes a top-down approach, where we start from the output variable(s) being documented and work progressively down to the underlying components. Alternatively, one could utilize a bottom-up approach starting from the lowest level of detail and working up to the ultimate output variable of interest. In that case, one would start

from the mortality assumption input through persistency assumptions to the components of the core variable. In addition, this approach is particularly useful when documenting multiple outputs resulting from a common set of core calculations. Ultimately, the documentation could utilize a hybrid approach consisting of both the bottom-up and top-down methods.

Further Considerations

A key methodology documentation challenge arising with vendor systems is the potential need to rely significantly on vendor documentation. Therefore, especially for closed or semi-closed systems, the model documentation should cover the specific/custom configuration/implementation by the user. In addition, for these closed systems it is important to:

- Create appropriately detailed description of any calculations performed on the inputs and assumption data used to populate the model.

- Explicitly describe how the assumptions are intended to be populated. For example, a pricing model may contain distinct mortality/morbidity assumptions for projecting benefits and reserving. The documentation should make it clear how each of these assumptions flow into the ultimate calculations to help the user understand how the product is modeled. This can be done by describing the calculations, when known, or by providing links to the vendor's online help.

  - Provide a description of calculations with references to the vendor-supplied documentation; if the latter is not complete, provide formulas compliant with corresponding regulations, Actuarial Standards of Practice and so on.

  - Indicate and describe, without necessarily attempting to reverse-engineer the calculations, which methods are being used (i.e., what is the model's supporting theory) and be very explicit around **core** calculations. Often the business unit that uses a closed system will have spreadsheets or small programs used to test some key model calculations. These can often be referenced to illustrate a calculation and should be included in the documentation appendix.

OUTPUTS

Model output, as defined earlier, is generally a collection of tables populated by the calculation engine. For example, model output can consist of files, external reports or reporting facilities within the software. Usually third-party vendor-supplied systems have more developed reporting facilities, pre-packaged reports and, typically, customized reports, as well. Non-vendor provided (homegrown) applications are usually custom-built for the business. The model's output should be documented, but external reporting facilities that perform further manipulations on the output can be documented elsewhere.

Any output structure generated by the model should be documented:

- Similar to the input, the documenter can use a data dictionary to describe the output structure.

- At a minimum, the data dictionary should contain the following two components:

  - The description of line items and

  - The formulaic expressions that relate them to other variables, if applicable, provided they are not documented in the calculation sections.

Table 1
Example of Output Structure Documentation

Field Name	Field Description	Notes
t	Time step	This is end of month or end of year depending on $\delta t_k := t_k - t_{k-1}$, setting of the projection time steps
. . .	. . .	. . .
InvIncome(t)	Income from investments: coupons, principal etc.	Documented in the calculation section
RCGL(t)	Realized capital gains and loss	Documented in the calculation section
InvExp(t)	Investment expenses	Documented in the calculation section
NetInvIncome(t)	Net investment income	= InvIncome(t) + RCGL(t) InvExp(t)

- Each output report may correspond to an (output) structure, as shown in Table 1.

In some cases, there may be multiple outputs, in one or multiple tables, corresponding to different scenarios/sensitivities. It is helpful to document these as well. In either case, it is necessary to provide a comprehensive description of outputs, including key variables, calculations generating the outputs and reserving. Other purposes of the output such as accounting basis and financial statements, should be specified. The descriptions provide context and make the documentation understandable to someone unfamiliar with the model. These details enhance the modeler's ability to maintain the model.

CONCLUSION

In this article we focused on three items related to technical model documentation. Proper documentation reduces key person risk, decreases a new modeler's learning curve, provides a consistent standardized companywide process, and helps perform corporate audits, deep dive validations and model conversions.

Comprehensive documentation should also include other items, such as model objectives and limitations, control structure, testing, change management and other relevant information. Many of these are not just informative for the modelers but may constitute a requirement from various governing bodies, such as enterprise risk management, internal and external audit and rating agencies. As a result, the final documentation product becomes a powerful and valuable store of institutional model knowledge base. ■

Dodzi Attimu, FSA, CERA, CFA, MAAA, Ph.D., heads the model validation program at MassMutual. He can be reached at *dattimu06@massmutual.com*.

Ryan Bijur, FSA, MAAA, is an actuary at MassMutual. He can be reached at *rbijur28@massmutual.com.*

Jeffrey Jardin, FSA, is a model risk management actuary at MassMutual. He can be reached at *jjardin61@massmutual.com.*

Michael Minnes leads model governance within enterprise risk management at MassMutual. He can be reached at *mminnes@massmutual.com.*

Veltcho Natchev, FSA, MAAA, is a model validation actuary at MassMutual. He can be reached at *vnatchev@massmutual.com.*

ENDNOTES

1 Attimu, Dodzi. 2018. What's a Model? A Framework for Describing and Managing Models. *The Modeling Platform* November. *https://www.soa.org/globalassets/assets/library/newsletters/the-modeling-platform/2018/november/mp-2018-iss8.pdf*.

2 For example, the statement "This cash flow test model projects all relevant cash flows for the book of business" is technically accurate, but does not provide useful information. The model documentation should provide sufficient detail into its calculations as it is important to accurately capture the model calculations without being too high level. However, once we start adding details of the calculation risks of misrepresentation may arise. Sometimes it means adding a caveat to certain aspects to point out the existence of further lower-level complexity.

3 The model produces aggregate results over the policy calculations that follow.