Article from

The Modeling Platform

December 2015
Issue 2

System Access and Change Controls

By Michael Failor

For many reasons, the insurance industry is paying increasing attention to the control of actuarial systems. Although each organization has its own current level of controls (some more developed than others), model-based valuations will be increasing model complexity and will require higher levels of actuarial systems control to reduce model risk. This increased focus has led many actuaries to realize that their company's actuarial systems require upgraded access and change control processes similar to those followed by their IT counterparts.

Many companies in the industry have fallen behind the controls curve, and those who have recognized their predicament are now seeking workable solutions. This article discusses effective control processes that can help get actuarial systems back on the controls track. One thing is certain—times have changed and new habits are needed to keep pace with the new direction that modeling has taken. Actuaries will soon be expected to apply higher levels of due diligence in actuarial systems management. In fact, we are heading into an environment where documentation and process will be receiving as much focus as the modeled results.

SYSTEMS VERSUS MODELS

The first thing to clarify is what I mean by "system" versus "model." This is a gray area; some models may evolve into full-blown systems, and some systems may be reduced to individual models. But the terms need to be put into context in order to capture the essence of a system and how it is differentiated from a model. Some key defining characteristics of each are shown below.

Characteristics of Actuarial Systems:

- Often contain a myriad of options and settings in which to build and execute actuarial models.
- Provide the underlying code (actuarial formulas) needed to execute projection models. "Open systems" allow users to modify system code and formulas, whereas "closed systems" prevent user-level changes to underlying system code.
- Coding errors can affect every model built on the system (similar to DNA coding errors).

Characteristics of Actuarial Models:

- Are often executed on actuarial projection systems.
- May be defined by selecting options, settings, and inputs accommodated by an underlying projection system (similar to genes in DNA).
- May be independent of each other.
- Are dependent on underlying system calculations.

With these characteristics in mind, we can address the increased need for system controls.

REASONS FOR SYSTEM CONTROLS

There has been an increase in systems development efforts to accommodate changing modeling demands. Some major drivers are stochastic modeling, Solvency II, VM-20, systems consolidations, and continued product creativity. Code development to accommodate these drivers is taking place across many applications and programming languages, including the following:

- Vender-provided "open code" projection systems
- Excel (VBA)
- C++
- SAS
- And many others

Modeling is also increasingly reliant upon larger volumes of data, prompting an increase in SQL code development and utilization of database interfaces. This is happening in addition to new applications of big data analytics.

Given the large amount of code development taking place, there is naturally a greater chance of systems-generated modeling error. However, the model flexibility afforded through coding changes cannot be at the expense of model integrity.

It should be clear that actuarial modeling and systems have evolved—but have our system controls actually kept pace with the changes? The answer to this question can be found in the SOA-sponsored survey conducted by Deloitte, "Actuarial Modeling Controls: A Survey of Actuarial Modeling Controls in the Context of a Model-Based Valuation Framework" (December 2012; a second updated survey on actuarial modeling controls is currently under development.) This survey compared the then-current state of controls against those expected to be in

Modeling Governance Theme	Score	Current State Synopsis
Governance Standards	3	While many companies employ a variety of model governance policies, few companies have a holistic, formal and documented model governance structure.
General Modeling Process	3	Many companies have multiple models and modeling platforms and few companies incorporate a model steward role in the modeling processes.
System Access and Change Control	4	Model changes are not generally governed by a formal change process.
Model Assumption Management	3	Assumptions are regularly reviewed and updated, but with few controls in place to ensure assumptions are approved and input appropriately.
Model Input Management	2	Many companies use automated feeds from admin systems for model inputs of liabilities. Other model inputs are often less automated.
Model Output Management	2	Model output used for financial reporting purposes is generally well controlled, while model output for analysis and other purposes is generally less controlled.

Of the six governance themes analyzed by Deloitte, System Access and Change Control was rated the worst in the industry. (Source: "Actuarial Modeling Controls: A Survey of Actuarial Modeling Controls in the Context of a Model-Based Valuation Framework." SOA, December 2012.)

place for model-based valuation approaches, including an analysis of controls surrounding system coding changes. On a scale of 1 to 5 (1 being the best), System Access and Change Control received a 4—the worst score of all the categories. Note that controls for system coding changes fall into this category.

The majority of survey respondents had no formal systems change control process. Because of this finding, the report accompanying the survey results strongly recommends that companies "implement a formal change management process for governing model code changes and model updates." To create such a change control process, it is important to understand some typical components.

SYSTEM ACCESS AND CHANGE CONTROL COMPONENTS

Actuarial system code changes are typically performed by actuarial staff, because they best understand the theory and actuarial mathematics underpinning insurance products. However, good code management practices (a hallmark of IT professionals) are often underappreciated or inconsistently applied in actuarial units. Many actuaries view systems controls as overbearing and unduly burdensome. In some instances this sentiment is justified. Nevertheless, good control processes pay off in the long term. Whether achieved through direct IT departmental oversight or more independently managed within the actuarial units, companies should have customized control processes designed for efficient execution.

Some important components to consider when developing a control process are addressed below.

Code Comparison Tools

When actuaries make code changes, or compare different versions of an open code application, they can use code comparison tools to make the job easier and provide valuable documentation of system changes. Vender-supported actuarial systems may contain integrated code comparison tools. However, when working with Excel VBA or other coding platforms, many third-party code comparison tools are available (e.g., UltraCompare). Most IT departments maintain well-tested code comparison tools available on your network.

Systems Peer Review

Not only should actuarial models be peer reviewed, but changes to actuarial systems should be peer reviewed as well whenever system code changes are made. (See Reviewing and Validating Actuarial Models, "Systems Peer Review" presentation at the SOA 2013 Valuation Actuary Symposium.) Peer reviewing actuarial system modifications can be a significantly different task

than validating an individual actuarial model. In fact, many (if not most) system errors that I have seen could have been discovered in a proper systems peer review.

Model and System Stewards

For model and system stewards to remain effective, top-down management support is crucial. These roles can be staffed by one or many individuals throughout the organization. But, if the authority of any steward is undermined by weak or faltering management support, then the control process may lose its footing as hurried modelers push for procedural shortcuts or the "quick fix." Although flexibility has its place, procedural controls should be explicitly defined to expedite small, isolated system changes without foregoing critical control steps. Not every scenario can be anticipated, but reducing the number of procedural "exceptions" is best achieved through an effective change request classification system. Alternatively, overly broad and unduly cumbersome control processes are more likely to be circumvented, thus reducing the steward's effectiveness.

The following are examples of some of the potential duties of a steward:

- Secure production models and systems
- Archive models and systems when changes are made
- Apply system versioning controls
- Ensure that control processes and procedures have been properly followed
- Assign system access levels
- Manage system and model documentation
- Communicate with modelers and update procedures
- Participate in model governance committee
- Coordinate testing and peer reviews
- Obtain approvals
- Schedule new system releases and software upgrades

System Access Controls

Access to system files, data, and models should be restricted to users, modelers, developers, testers, stewards, and others demonstrating a need. Access should be granted at the appropriate levels and reevaluated on a regular basis to remove or modify access levels as individual roles or processes change. In general, it is best to limit a user's access to the lowest level needed for them to perform their duties (e.g., read, write, copy, or run). Access should be set separately for production grids versus testing servers. Without proper access controls, the other control processes and procedures may be circumvented—either purposefully bypassed or by accident. The time-honored adage is "If something is not locked down, then you do not have control over it."

Test Beds (or Test Packs)

Whether you are modifying existing system code, converting to a completely new system, or upgrading to a more recent version of actuarial modeling software, test beds are a must for your testing arsenal. Test beds help to identify errors and inconsistencies among different software implementations by running identical input (e.g., seriatim policies, interest rates, mortality tables, and product settings) through each system and then comparing the results. Test beds are usually a subset (or complete set) of your organization's business modeled on the systems under comparison. When new products or features are modeled, test beds should be updated accordingly so that future test bed comparisons will be sensitive to errors associated with these new enhancements.

When a test bed is run through a system, the results should be archived along with any supporting files and system settings. This will aid in future analysis when unexpected discoveries need to be traced back to their origins. Although test beds are a vital component of a system change control regime, they cannot be expected to catch all errors.

Management/Departmental Approvals

Whenever proposed system changes are made, those who directly use the system or the system's output may take close interest. Depending on the extent and nature of a systems change, interdepartmental approvals may be required along every step of the process—from the development of the initial business requirements through testing and setting an implementation rollout date. The actual approval process will

need to be fleshed out and will depend heavily on the modeling environment.

Communication and Documentation

The value of system documentation cannot be overstated as more parties become involved in modifying, analyzing, peer reviewing, and testing actuarial systems. Also, as systems become more interconnected, sufficient documentation may be expected for use in downstream systems analysis. Customers (including other systems) that rely on system output will benefit greatly from well-organized and detailed descriptions of their upstream sources.

System documentation should not be relegated to an end-of-project exercise. It should be a continual process starting at the beginning of a systems project and continuing through the final implementation steps. As the systems project morphs in scope or design, the documentation should be updated accordingly. A good systems actuary will not only document high-level descriptions but also provide helpful comments in the actual code. Documentation should also be archived and linked to production models, test results, and corresponding rollout schedules.

Modeling Environment Considerations

The modeling control process goes hand-in-hand with the modeling organizational structure. Gaining in popularity are organizational structures having a single centralized modeling environment where the actuarial models are built and maintained on one actuarial system. Organizations with centralized modeling environments often maintain a single "model of record" for each modeled block of business that serves as a base model for pricing, valuation, capital, and risk management modeling endeavors. In contrast to the fully centralized approach, a decentralized modeling environment disperses model development and maintenance among the respective functional actuarial units. Decentralized environments are often supported by different actuarial software systems that best satisfy the modeling demands of the respective actuarial unit.

The modeling control process goes hand-in-hand with the modeling organizational structure.

Although the centralized modeling environment incorporating models of record may be preferred, it requires a modeling system that can accommodate changing requirements in features and functionality typically demanded by different functional actuarial units. In addition, because the model of record may undergo an increased number of simultaneous changes in response to ongoing change requests, a formal code aggregation process may be required to ensure that simultaneously modified code works as intended when combined. Additionally, whenever simultaneous changes are made to a system, a formal code module check-out/check-in process can help prevent specific code modules from being modified at the same time. Even with a code check-out process, integration testing is still required because simultaneous changes to interdependent modules may produce unintentional effects.

Decentralized modeling environments present their own issues. For example, duplication of effort may occur when maintaining duplicate models and supporting multiple actuarial systems. Modeled results for a given portfolio may also differ between models, often requiring additional cross-validation.

LEADING PRACTICES

The results from the SOA Actuarial Modeling Controls survey included a number of leading practices pertaining to system access and change control. These practices typically contain the following four high-level steps:

1. Establishment of a procedure to identify and prioritize model changes (i.e., a change request process).
2. Evaluation of coding changes in a test environment and analyzing any impacts on financial results.
3. Performing additional testing on the model code changes. Depending on the nature of the changes, this can include regression testing, sensitivity testing, and peer reviews.
4. Producing proper documentation and seeking formal approvals.

If the final tested changes have been approved for use, then a system release should be scheduled for production. Note that implementation of new production code should be coordinated well in advance of any reporting close dates (e.g., quarterly or annual closes).

The creation of modeling teams and IT involvement is also recommended. Modeling teams responsible for managing and prioritizing change requests and determining change request procedures may be new to many organizations. However, these well-chosen teams are crucial, because they will also be expected

to publish and maintain required system documentation and change request logs. IT involvement in the code change process will leverage their expertise and increase control and code consistency—which is of greater concern for the open code actuarial systems and centralized models.

SYSTEM DEVELOPMENT LIFE CYCLE

Although the SOA-sponsored survey covered much ground, this article would be incomplete without mentioning one of the major topics in system controls: the System Development Life Cycle (SDLC).

SDLC really merits its own article, but it is worthwhile to note that SDLC methods provide a clearly defined process for planning, creating, testing, and deploying systems and systems modifications. SDLC benefits include model risk reduction, well-defined roles and responsibilities, improved communication and documentation, and an auditable process. Lest one think that SDLC methods are just for IT professionals, be aware that the Institute and Faculty of Actuaries in the UK has included an SDLC method as part of their best practices since 2009.[1]

STRENGTHENING OUR PROFESSIONAL PRACTICE

Actuarial systems control is a multifaceted endeavor requiring an organizational structure that orchestrates skill sets and processes into a well-controlled yet highly efficient modeling environment. Because of their increasing importance, many systems control components will continue to take hold within the actuarial profession as we more fully recognize and adopt the tenets of the system development life cycle. Although processes and organizational structures do not change overnight, continued progress in actuarial systems controls will reinforce confidence in our modeling and ultimately add value to the profession. ■

Continued progress in actuarial systems controls will reinforce confidence in our modeling and ultimately add value to the profession.

ENDNOTE

[1] Institute and Faculty of Actuaries, "Report from the Actuarial Processes and Controls Best Practice Working Party—Life Insurance," May 31, 2009, *http://www.actuaries.org.uk/research-and-resources/documents/report-actuarial-processes-and-controls-best-practice-working-party.*

Michael Failor, ASA, MAAA, is a modeling actuary involved in research and development at SCOR Global Life. He can be reached at *mfailor@scor.com.*