

Version Control Framework for the SAGE Model

Prepared by National Center for Environmental Economics (NCEE), U.S. EPA
SAB CGE Model Review
August 22, 2019

The purpose of this memo is to describe the proposed version control framework for the computable general equilibrium (CGE) model, SAGE.¹ The SAGE model refers to a collection of inter-related files - containing source code and data - that is designed to construct datasets and simulate the effects of policy changes on the United States economy. Aligning SAGE with the latest economics literature and the best available data is a moving target. Incorporating changes in the model files might encompass relatively small bug fixes to the code, moderate updates to underlying baseline data or parameter estimates, or larger structural changes that better represent the nature of the simulated regulatory changes on the affected universe of regulated sources. Ensuring transparency and consistency across the application of these improvements to SAGE requires a version control system.

Version control is a management tool that tracks changes to a collection of files. In its simplest manifestation, a version control system is a unique identifier, a number or letter code; the higher the number, the more recent the version. Print editions of books are an example of this type of simple version control system. In the digital world, changes to software or models often occur more frequently and the changes may vary in importance and magnitude. In these instances, more nuanced systems of unique identifiers can be used to communicate routine changes to model structure and/or underlying data. When applied in a consistent fashion a version control system enhances long-term model use and maintenance by enforcing a logical structure on the management of changing source code. Documentation of the specific model changes that correspond to a given version is also important for implementing a transparent version control system.

This memo is organized as follows: The first section of this memo describes the proposed version control structure for the SAGE model. The second section discusses the relationship between the version control structure and peer review of model updates. The appendix illustrates the application of this version control framework by describing the historical development process of SAGE.

¹ SAGE is a recursive acronym, as in SAGE is an Apply General Equilibrium model.

1 Versioning Control Framework

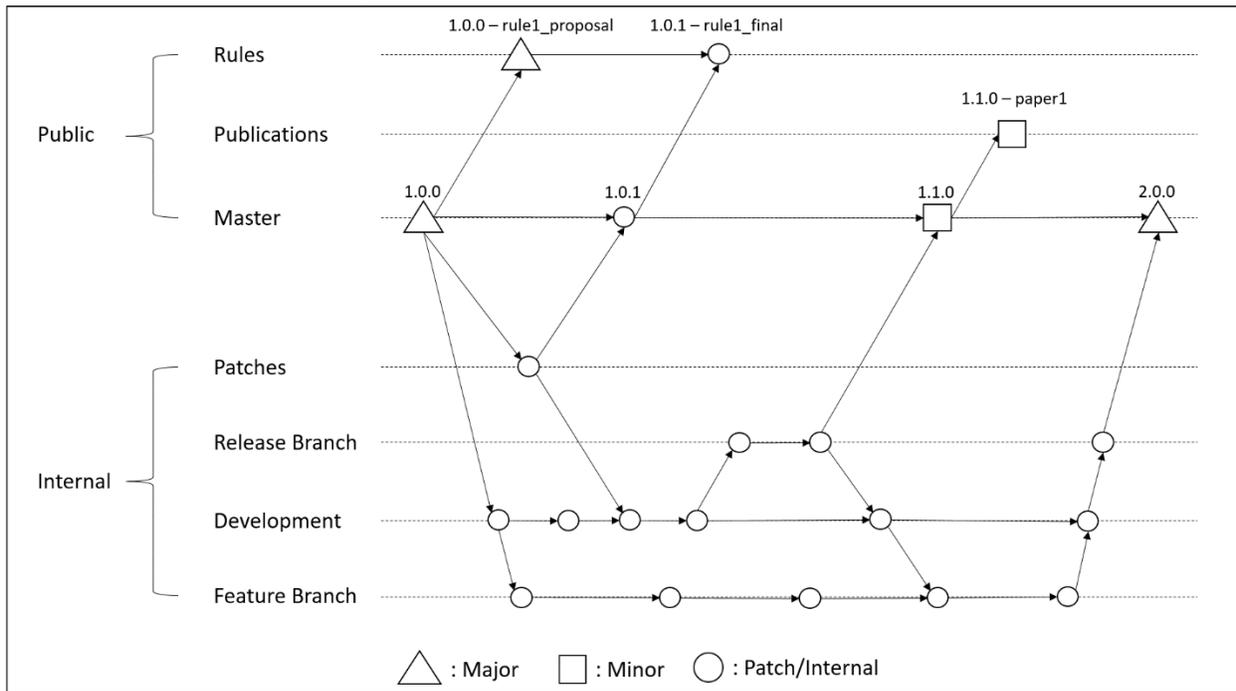
A relatively common approach to digital version control is to use a series of numbers (and/or letters) where each number can change independently, and a number's placement in the sequence communicates the nature of the underlying change relative to the previous version. For instance, a change in a number at the end of a sequence typically indicates a more minor change, while a change in a number at the beginning of a sequence often indicates a larger scale change. We view the collection of the source code, documentation, and data build stream files as key components of the model and opt for a version control system that is applied to the overall collection of files. Thus, a single change in the version control number may indicate changes to multiple files. This makes documentation of changes between versions more essential but ultimately allows for a more tractable system over time.

The SAGE model is regularly revised to update and improve its representation of the U.S. economy and ability to model environmental regulations. To manage these changes, semantic versioning is employed to codify versions of SAGE following updates. Semantic versioning is a sequence-based scheme designed to distinguish between major, minor and patch changes to the modeling framework by assigning a cumulative, three-digit unique identifier. This methodology is often used in software development. A semantic version identifier looks like: Major.Minor.Patch, where each segment of the identifier is a numeric value. When new versions of the code are made available, it is necessary to also change some component of the version identifier. Small edits to address bugs or streamline code are associated with an increase in the patch component (e.g. 1.0.1 to 1.0.2). Increases in the minor component are either characterized as a collection of patch edits or moderate changes (e.g. 1.0.2 to 1.1.0). Increases in the major component may be reflective of a collection of minor version updates that, cumulatively, substantially distinguish the modeling framework from previous major versions, a singular significant change to the underlying model structure, and/or changes that prevent backward compatibility of the model to previous versions (e.g. 1.1.0 to 2.0.0).² Updating the model to a new major version would require setting the minor and patch components of the version identifier to zero (e.g., from 1.7.1 to 2.0.0). Similarly, updating the model to a new minor version would require resetting the patch identifier to zero (e.g., from 1.1.4 to 1.2.0).

² Backward compatibility in reference to models differs somewhat from the case of software development. In this context we use backward compatibility to refer to the extent that code for conducting previous analyses can be easily used to replicate the analyses with newer versions of the model or the extent to which new versions of the model can operate using data from previous versions of the model.

The semantic version identifier helps distinguish between public facing versions of SAGE. The top portion of Figure 1 depicts an illustrative example of this framework (the internal process in the bottom portion will be explained later in the memo). Each level of the diagram can be thought of as a distinct branch of code. Here, branches essentially refer to separate snapshots of model code allowing for storage of multiple versions of the model (for release or subsequent code development).³ The “Master” branch holds all publicly released versions of the SAGE model. The “Rules” and “Publications” branches highlight the anticipated method for storing instances of model code used in rule making or in applications outside of the regulatory context such as publications. Updates to the three components of the identifier are represented as different shapes. Starting from major version, 1.0.0, the diagram illustrates an update in the patch component following a bug fix to 1.0.1. Additional model development results in a minor update, 1.1.0 and the inclusion of a new feature results in a major update (to 2.0.0).

Figure 1: Illustrative Versioning Design⁴



³ Technically speaking, this process is more complex than described here. See: <https://git-scm.com/book/en/v1/Git-Branching-What-a-Branch-Is>.

⁴ Note that this diagram is for illustrative purposes only and represents a hypothetical versioning sequence. It is not reflective of an actual rule making or model development process.

There are two contexts in which version control for the SAGE model is relevant within the EPA: general model updates, and modifications to the model for a specific regulatory application. General model updates are changes that become fully integrated into the core model. Best professional judgement should be used to designate general model updates as either major, minor, or patch changes. Patch level changes are more obvious: they typically are restricted to bug fixes and changes to streamline code or enhance backward compatibility. Examples of minor changes might include updates to the underlying data or assumed elasticity values drawn from peer-reviewed literature or estimated based on peer-reviewed methods using more recent data. Major changes would include significant new features that fundamentally change some aspect of the model. For instance, this might include re-specifying consumer preferences through functional form assumptions.

There is a gray area when determining whether a change is a major or minor update. A variety of factors should be considered in judging the significance of changes in the model and whether a major version update is warranted. For instance, if many minor changes are made simultaneously, this may rise to the level of a major change. In some cases, it may be possible to judge whether updates are minor or rise to the level of a major change based on the output from a series of standardized runs. If the model produces output that is sufficiently different from the previous version following the implementation of new features, then the modifications may be considered a major change. In keeping with EPA's Peer Review Handbook (2015), we would take a case-by-case discretionary approach to minor vs. major designations with care taken to document the justification.⁵

1.1 Single Use Regulatory Applications

General model changes are distinct from modifications to the model for use in a single regulatory application or a potential study outside of the regulatory context. These changes may be rule or problem specific and not reflective of general modeling needs in other contexts. For instance, analysts may decide to further disaggregate one of the SAGE industrial sectors to more closely identify the subset of sources subject to a proposed regulation and/or to better reflect substitution possibilities between these plants

⁵ The EPA's Peer Review Handbook (4th Edition) notes that the EPA has "significant discretion in deciding the timing and frequency of peer review". In a similar context, EPA also has discretion over whether modeling updates are interpreted as "major" as this relates to the peer review process as illustrated in the next section. Further, any use of words like "may" or "should" does not create a mandatory requirement for the agency but indicates a preference toward a specific course of action (U. S. EPA, 2015).

and closely related subsectors. Likewise, the way in which pollution abatement possibilities are captured in the model may differ with the specifics of a given regulation. These changes would not necessarily be incorporated into the core model.

It is important to track one-time modifications to SAGE for specific regulatory applications or other studies to ensure transparency and reproducibility. Figure 1 illustrates this in the top two branches. If the model is modified for use in a regulatory application, for example to analyze the costs of rule “rule1”, the modified version of the model will be denoted as major.minor.patch-rule1, where major.minor.patch refers to the base version of the model prior to the regulation specific modifications. Because regulatory analysis undergoes various stages of development, model use in the early stages (here labeled as “proposal”) versus the later stages (here labeled as “final”) may use a different base model version if updates have been made since the model’s previous use for the rule. If different versions of the model are used in the analysis, the stage of the rule is appended to the version identifier. For instance, in Figure 1, version 1.0.0 is used in the proposal stage of the rule and version 1.0.1 is used in the final stage of the rule. Correspondingly, there are two versions of the model for “rule1”: 1.0.0-rule1_proposal and 1.0.1-rule1_final.

Holding the identifier major.minor.patch constant will allow analysts to rely on documentation for the core model to explain structural aspects and key parameter assumptions. Analysts will then only need to document (and justify) the changes made to SAGE for the regulatory application relative to the core model.⁶ The extent to which these changes may be subject to peer review is discussed in Section 2. To enhance transparency and the ability of stakeholders to replicate results reported in a regulatory analysis, the branch of the model and its data used for the regulatory analysis will not be further updated, except in the context of the rulemaking, and will retain the same version identifier that is referenced in the rule package. If any of the features implemented for the regulatory analysis are deemed useful for other applications, they may be brought back into the core model through a patch or minor update.

1.2 Internal Model Development and Versioning

Underpinning the publicly facing branches is an internal process that establishes the rules for organizing the development of model code between versions.⁷ Not all internal code development will be individually stored in separate versions of the model. The bottom part of Figure 1 depicts this process in an illustrative

⁶ The same is true for studies outside of the regulatory context.

⁷ This process is an adaptation from the one proposed at: <https://nvie.com/posts/a-successful-git-branching-model/>.

setting. There are four main branch types in the internal organizational structure: the patches branch, feature branches, the development branch, and release branches.

The patches branch houses code for quick bug fixes. Once the code has been sufficiently reworked to correct the issue in question, the code is merged into the master branch as a new patch version of the model (e.g. 1.0.0 to 1.0.1). Model updates can either be small patch fixes or longer-term significant feature developments. Feature developments that fundamentally change some aspect of the model are housed in feature branches. In Figure 1, a longer-term feature development is stored in the bottom branch. Version 1.0.0 of the model is merged into this branch to act as a starting place for this feature.

Between the initialization and completion of a feature, smaller changes are stored in the development branch. The development branch represents the ongoing development of the model and may accumulate several updates before the changes are pushed to the master branch as a new version. All completed model developments following the establishment of a new model version are stored in this branch. In Figure 1, each additional circle in the development branch indicates some type of update to the model code. The development branch is held in perpetuity alongside the master branch. Note that when a patch is applied, all changes are pushed both to the master branch and development branch to correct for the issue. When the development of the model reaches a point where the updates are determined to be ready for elevation to the master branch, the code in the development branch is first sent to a release branch. The release branch is used to run quality assurance and quality control (QA/QC) tests to verify the efficacy of the model code before publicly releasing a new version.

The diagram indicates the potential for updates as the result of the QA/QC process after the release branch was created (denoted as additional circles). All changes in the release branch are pushed both to the new version in the master branch and back into the development branch. These changes would also be merged into any active feature branches to sync code with the most recent publicly available version of the model. Following the completion of the feature development, this code is merged back into the development branch, and in this example, eventually constitutes a new major version of the model. Note that feature branches are finite. Following the completion of the feature development, this branch is no longer in use.

This version control framework is implemented using the Git version control system.⁸ Git provides an efficient means of implementing the described versioning strategy, in part, because the branching

⁸ <https://git-scm.com/>

structure was designed in tandem with the capabilities of the software, which tracks incremental changes between versions for a detailed (i.e., line specific) changelog. “Tags” are used within the Git system to designate major, minor, patch, or regulatory specific model versions so they may be preserved.

2 Peer Review and Versioning

While the main purpose of this SAB panel is to review SAGE’s current core modeling framework, future major and minor changes to the model also may warrant additional peer review. This section describes how the version control system relates to and might help inform the determination of whether model changes are substantial enough to require additional peer review.

According to the EPA’s Peer Review Policy, “[p]eer review of all scientific and technical information that is intended to inform or support agency decisions is encouraged and expected,”⁹ though the Agency also recognizes that peer reviews can be time consuming and costly. Scientific and technical work products that are more likely to warrant peer review include those that establish a significant precedent, model, or methodology; address a controversial issue; focus on significant emerging issues; involve a significant investment of agency resources; or consider an innovative approach for a previously defined problem, process, or methodology, among others (U.S. EPA , 2015). In addition, the U.S. EPA recommends peer review of its scientific models “to evaluate whether the assumptions, methods, and conclusions...[are] based on sound scientific principles, and to check the scientific appropriateness of a model for informing a specific regulatory decision” (U.S. EPA, 2009). Criteria that should be used to judge whether peer review of changes to existing models is warranted are the level of scientific/technical complexity and/or the novelty of the change (U.S. EPA, 2009).

While establishing explicit criteria to judge *a priori* what SAGE model changes may warrant additional peer review is difficult, it is expected that major updates will generally meet the threshold for additional peer review. This is particularly true in cases where the update significantly changes the model structure and key assumptions.¹⁰ For example, when an update is associated with implementing concepts that are well established in the economics literature but are significant enough to qualify as a new major model version,

⁹ See: https://www.epa.gov/sites/production/files/2015-01/documents/peer_review_policy_and_memo.pdf.

¹⁰ U.S. EPA (2009) lists aspects of a model that should be judged independently for scientific credibility. These include: the appropriateness of input data, boundary condition specifications, and parameter values; documentation of adjustments to model inputs to improve model performance (calibration) as well as key assumptions; model application with respect to the range of its validity; and supporting empirical data that strengthen or contradict conclusions that are based on model results.

the peer review will likely focus on whether these practices have been properly incorporated into the model. The form this peer review will take, the selection of peer reviewers, and the charge will be consistent with EPA's peer review guidance (U.S. EPA, 2015). Note that the U.S. EPA also has considerable discretion regarding the timing and the frequency of peer review (U.S. EPA, 2015).¹¹

The level of peer review required for minor updates is a gray area that will require the modeling development team to exercise its professional judgement. One possible criterion the team might use is the degree to which minor changes are relatively routine. For example, incorporating the latest data from the Bureau of Economic Analysis or Annual Energy Outlook requires little or no change to underlying model structure. These changes likely will not require additional peer review. Likewise, while disaggregation of sectors for a specific regulatory application that proceeds along dimensions already clearly delineated in the data will need to be clearly documented, these changes may not require additional peer review. On the other hand, a minor update to incorporate new elasticity estimates from the literature may qualify for peer review if these changes have a significant effect on the model output or are utilized in a novel regulatory application.

When model modifications are made for a specific regulatory application, stakeholders may comment on the use, inputs, and interpretation of the output if used to inform a rulemaking. The EPA often engages with stakeholders, or individuals potentially impacted by the outcome of the policy, to ensure that all relevant information is considered. The Agency must respond to comments received during the rulemaking process. While analysis conducted as part of the rulemaking process must be transparently documented so that stakeholders can review it via the notice and comment period for a proposed regulation, updates to the SAGE model for one-time regulatory applications may still require additional peer review. Public comments can help inform the peer review process through suggestions of related experts and isolating technical issues to be considered by independent peer reviewers (U. S. EPA, 2015). If changes made to the model in the context of a one-time application are a general improvement in the way SAGE captures certain aspects of economic behavior, then EPA also may ask peer reviewers whether these modifications should be incorporated into the core model.

¹¹ Scientific or technical work products that are deemed "influential scientific information" (ISI) are also required to undergo peer review per the Office of Management and Budget's Peer Review Bulletin. See: https://obamawhitehouse.archives.gov/omb/memoranda_fy2005_m05-03/.

References

Marten, A., R. Garbaccio, and A. Wolverton. Forthcoming. "Exploring the General Equilibrium Costs of Sector-Specific Environmental Regulations." *Journal of the Association of Environmental and Resource Economists*.

U.S. EPA. 2015. *Peer Review Handbook*. 4th Edition. Science and Technology Policy Council. EPA 100-B-15-001. Available at: https://www.epa.gov/sites/production/files/2016-03/documents/epa_peer_review_handbook_4th_edition.pdf

U.S. EPA. 2009. Guidance on the Development, Evaluation, and Application of Environmental Models. Office of the Science Advisor. Council for Regulatory Environmental Modeling. EPA/100/K-09/003. Available at: <https://nepis.epa.gov/Exe/ZyPDF.cgi?Dockey=P1003E4R.PDF>

Appendix: History of SAGE Versioning

This section describes a partial history of SAGE developments up to version 1.1.0 as a way to further illustrate the proposed versioning strategy. Figure 2 illustrates this in the context of a flow diagram. A description of incremental changes between patch and minor versions is contained in Table 1. Most code developments from versions 1.0.0 and 1.1.0 were patch edits. This is reflective of early stage model development, and the structure of the versioning framework is expected to converge to something like Figure 1 in the long-term. Long-term features such as the reconciliation of marginal income tax rates within the model dataset and updates to the model documentation, depicted at the bottom of Figure 2, are eventually incorporated into the new minor version of the model (1.1.0). Note that transparent gray arrows reflect updates to feature branch code following a patch fix. Version 1.0.7-cost_paper is the version of the model used in Marten et al. (forthcoming).

Figure 2: Flow Chart Depicting a Partial History of SAGE Versioning

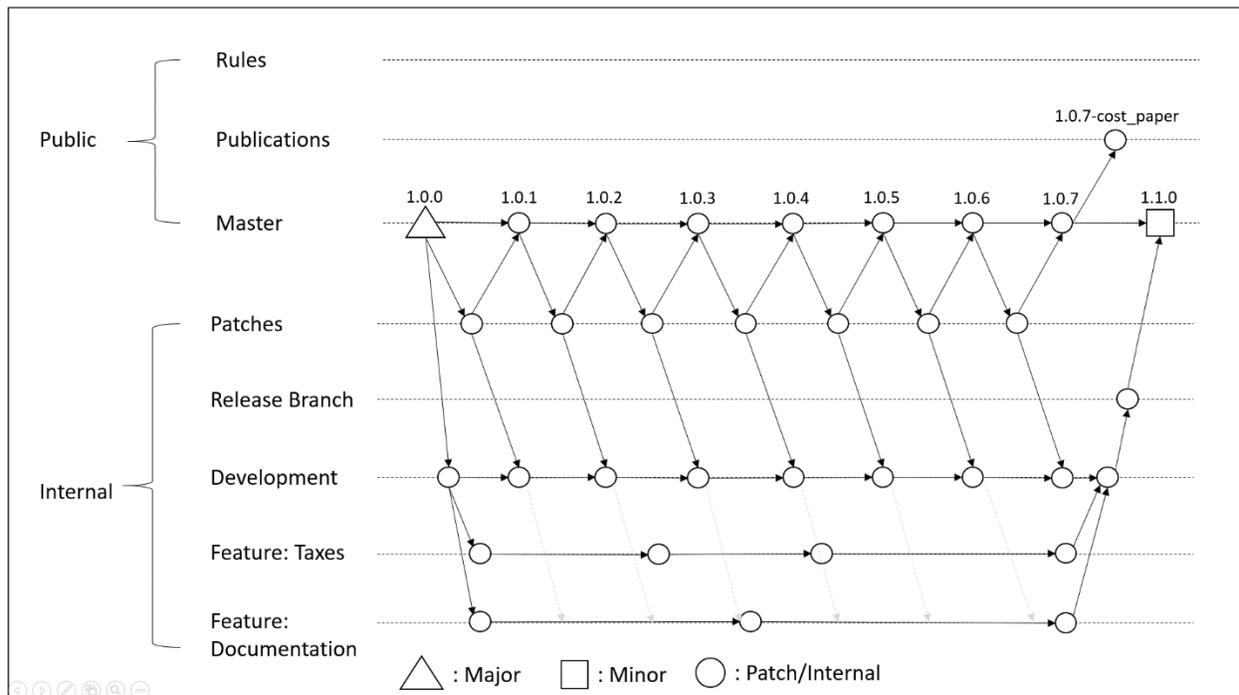


Table 1: A Partial History of SAGE Versions

Type	ID	Description of Change
Minor	1.0.0	* Initial version of the model.
Patch	1.0.1 – 1.0.7	<ul style="list-style-type: none"> ○ Improve calculation of equivalent variation ○ Fix division by zero error in the static model. ○ Allow for user defined time steps between model periods. ○ Fix production calibration in resource sectors in the static model. ○ Update example policy shock representation to account for previous changes. ○ Remove outdated and unused code or files.
Minor	1.1.0	<ul style="list-style-type: none"> ○ Inclusive of all previous patches. ○ Update effective marginal labor income tax rates by region and household. ○ All capital is extant in the initial year by default. ○ Update documentation on running the model and modeling separate abatement activities for each sector. ○ Performance improvements.