Quality Assurance Systems in Transport Modelling

Sinead Giblin[1], Ruimin Li[2],

[1] Sinclair Knight Merz, Sydney, NSW, Australia

[2] Transurban Ltd, Sydney, NSW, Australia

1. Introduction

Wikipedia defines '**Quality Assurance** (**QA**) is the activity of providing evidence needed to establish confidence among all concerned, that quality-related activities are being performed effectively'.This technical paper details the outcome of the development of a QA system for transport modelling tasks.

To avoid errors in traffic modelling projects, rigorous QA processes need to be developed for all procedures. An effective QA system ensures that the appropriate checks and balances are imposed on modellers, so that reliable forecasts are produced and the peer reviews are more straightforward and unproblematic. Documentation of best practice in this area is not found in published material. Therefore, these recommendations are developed from our wider knowledge of QA systems and best practice in SKM, Transurban and in some of our other client organisations. Many of the procedures discussed in this paper are currently implemented in the management of modelling projects.

The most important element of an effective QA system is that persons must conform and implement the QA protocol incorporated. A good QA system is one that is simple and efficient and not too cumbersome or time consuming to implement. A QA system constantly updated and adhered to can prevent modelling errors, be instantly available and easily accessible.

ln this technical paper we primarily focus on general QA procedures which are relevant to all modelling projects. We set out good practice operating procedures for the naming, storage and logging of critical input and output data, covering model directories, interfaces and log files and the use of version control and configuration management. All model updates and scenarios need to be reviewed and approved by a nominated technical reviewer with knowledge of the model and the modelling software.

We also look at the QA systems for setting up the base model and for scenario testing. For the base models there is a focus on calibration and validation acceptance tests while clear file naming and configuration management are needed to reduce the possibility of errors in scenario testing, where large numbers of scenarios and input assumptions are incorporated.

This procedures and protocols detailed in this technical paper can be applied to all modelling packages. Documentation **must** be done on a continuous basis.

2. General Procedures

General QA procedures apply to all data associated with modelling. The model should have a logical directory structure for the naming and storage of the model data and runs and log files of all model runs should be maintained (including a master log file, and input and output log files). Model input and output files should be designed, checked and archived.

Version control and configuration management should be implemented, affecting the management and storage of all script or macros files. Agreed procedures for approving model applications need to be pre-defined and adhered to. Persons responsible for sign-offs must also be identified. Documentation must be done on a continuous basis and kept up to date. If possible, files should be self-documenting.

2.1 Model Directory Structure and Log Files

The model directory structure is important to ensure that each data set is clearly named and located in a logical place. This reduces confusion and loss of time searching for inputs, outputs, scenarios and reduces problems and errors in the synchronisation of files between local drives and network servers. This process should be automated so that both drives are constantly up to date.

Maintaining an up to date centrally located master log file facilitates the management of large quantities of model runs and helps minimize the re-running of models. Each model is given a unique number which is detailed in the master file. The model inputs and outputs are also detailed in the master log file. Model input files such as matrices, planning data, demographics are all periodically updated. Each time an input file is updated it should be allocated a unique version number. This version number will be logged in the automatically generated log file so that the inputs are easily recognized. Automated model outputs should include the unique model run identification number and all input files, with referenced version numbers. Therefore the inputs details and for each model run is readily available and easily accessed.

2.1.1 Directory Organisation

Every folder and sub-folder should be consistent for each model run - an example directory structure is given in **Figure 1**. File naming protocols need to be defined such that they both identify the model/data configuration used and also uniquely identify each application. The structure of the model folder should be designed to create automatically consistent file names according to these protocols.

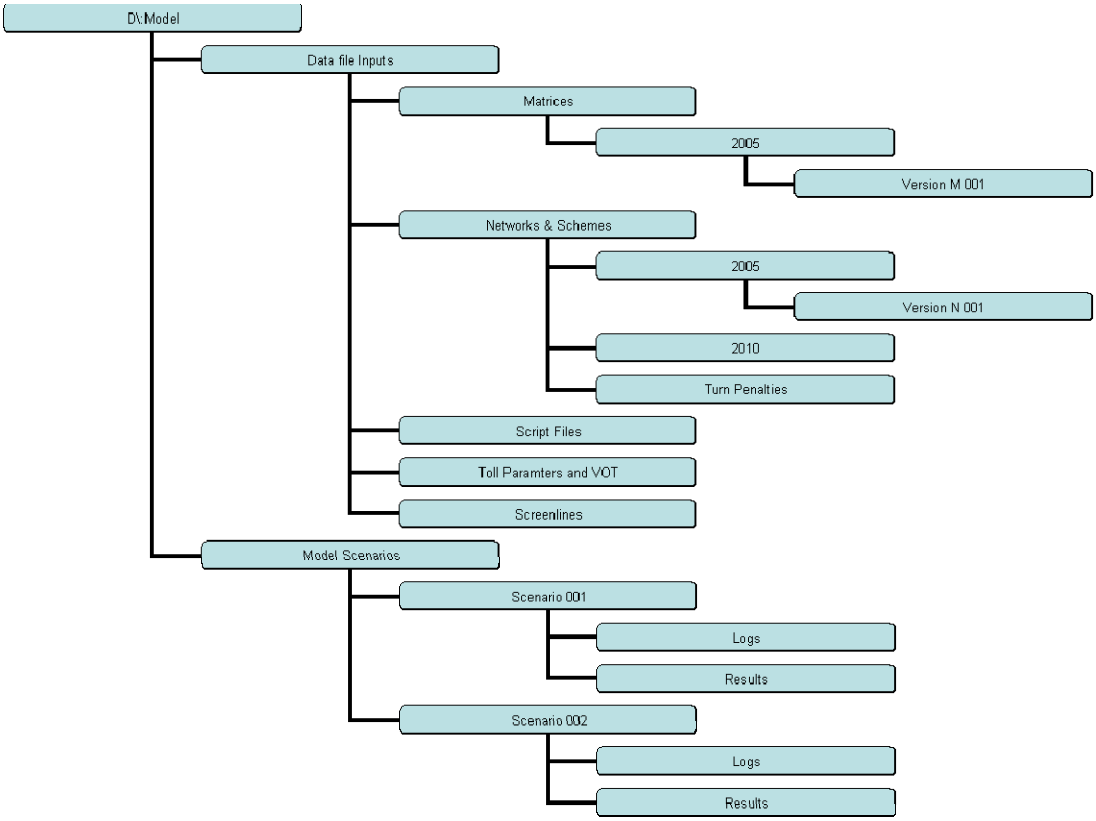


Figure 1 An example of a consistent filing system

The models should be run on a local drive and the directories should then be updated on the network to allow access by others and have a central location for the storage of data, model runs and outputs. A convenient way of doing this is using a file/folder synchronisation utility. A file/folder synchronisation utility is a solution for keeping files up to date between workstations and the server.

By simply specifying source and target folder paths and the file types that are required, the file/folder synchronisation utility will produce a comparison list of the relevant files showing which need to be copied and in which direction. You then have the option to filter the list, examine specific files for differences and change copy directions before going ahead with the synchronisation process.

2.1.2 User-Friendly Operation

The model front-end should facilitate the design of a user-friendly model interface and the minimisation of user errors. The front-end should be designed to enable user access to model features which are permitted to be changed in standard applications, but to restrict access to other 'controlled' features to specialists. Thus some parameters and processes could be adjusted for individual applications while others would be fixed. Additionally it must be ensured that scenario-specific numbering is used for every model run completed. This scenario

specific number will correspond to the number in the model log file (see Section 2.1.3), so that all runs and run descriptions can be reviewed at any time.

2.1.3 User-Maintained Log File

A log file of all model runs should be maintained by model users. An entry in a centrally located model log file (i.e Model_Log file) for each model run, should contain the date the run was completed, the matrices used, the network, specific schemes included etc. It should also contain a quick health check on the outputs in terms of, for example, vehicles hours travelled (VHT) and vehicles kilometres travelled (VKT). The VHT and VKT values can be automatically generated for each model run. If an error has occurred in the input files and/or assumptions it will be frequently highlighted by these values. The directory should be entered when the scenario is created.

Excel will provide the most common and useful format for the log file. This spreadsheet will include the unique scenario number and should also contain a column where the reviewer will enter their initials and the date the model run was reviewed and approved. An example of this spreadsheet can be seen in

Figure 2.

Each run should always have a Run ID. This will include a unique scenario number which will be consistent with that in the Model_Log file. A protocol, for the run ID should be developed in which the run ID includes: the run title, the run date and a model configuration ID (combining the version number/configuration ID of the model implementation with additional codes for specific model application inputs).

Model Log												
	Inputs								Outputs		QA	
Scenario ID	Scenario Location	Scenario Description	Matrices Year	Network Year	Network Description	Time Period/s	Toll Paramters	VOT used	VHT (m)	VKT (m)	Reviewed By	date
SC 001	Model - V01	2005 Base Year	2005 - M004	2001 - N001	2001 Network	24 Hr		Tolls_2005_VOTAppended.csv	12	82	DA	4/05/2007
SC 002												
SC 003												
SC 004												

Figure 2 Example of a Model Log File

2.1.4 Master Log File

An input/output log file should be automatically created, at the beginning and end of each model run. The log file should include the scenario name and unique number, and should be generated in the scenario specific results folder. This unique scenario number will correspond to the unique scenario number in the Model_Log file.

The Run ID/ Scenario ID should be included in each run and should be the first entry on the log file. Other parameters recorded should be the versions of:

- networks;
- matrices;
- the values for the input assumptions;

- script or batch files that control the files called upon in each model run;
- model inputs.

The latter parameters serve to define the model configuration exactly. Thus together this data can be designed to uniquely identify the characteristics of the application and the version of the model which is being used.

Automated filing and archiving set-ups are needed for the log files (see directory structure in **Figure 1**).

2.1.5 Model Input Files

General

In general there are a number of different types of inputs to model runs which are used repeatedly and should be given version numbers and including in a configuration ID. For example networks and planning data files may exist for different years and different scenarios. Documentation of all run inputs, relating to the IDs should be archived.

Network Coding Process

This is the most complex and error-prone input to a model application. Consequently, for each scheme coded in a database, an audit/review should be completed as this will be where the highest frequencies of error will occur in the coding. The coding should then be signed off by the reviewer and this will then be documented.

There are means by which network updates and developments can be systematised. For example: a hierarchical coding process can be used whereby base network *scenarios* are distinguished from sequential partial network upgrades and a central nodes file helps to prevent multiple use of node numbers.

2.1.6 Model Output Files

Automated model outputs should include:

- run/configuration identification on all outputs;
- archived material:
  - records of run instructions;
  - records of run inputs;
  - summary outputs for checking/diagnostics (should be automatically generated by the model run);
  - detailed standard outputs and reports (should be defined);
- temporary files: these may be intermediate run diagnostic outputs which are deleted subsequently and non-standard outputs.

Automated filing and archiving set-ups are needed for the output files (see directory structure).

2.2 Model Version Control and Configuration Management

Where possible, version numbers should be appended to the data file name (eg "_v01") to prevent the use of old data. A process will need to be followed to ensure that any links to other files are updated as well. The version control of scripts/macro files, which effectively specify the model configuration, is an effective way of ensuring that all script files are documented correctly. Script or macro files are coding files that determine what files and processes are called upon during the model runs. The model configuration is the composition of the input files for each model run.

There is a danger in modellers changing a script file "quickly" and not documenting changes that have been made. These changes could then be carried across other runs leading to erroneous results. All script files should be stored in a centrally located folder and should be logged in the Model_Log.xls file, where a unique version number is assigned to each script file. All script files should be read only so they cannot be over-written. They however can be saved as a new script file with its own unique version number.

When a script file is saved as a new version, the purpose of the script file should be documented in the script file, with the date, the modeller's initials and approval confirmation. The script details should then be updated in the scripts log. An example of this can be seen in **Figure 3.**

As a general principle, base model updates to versions/configurations should be subject to acceptance testing, review and sign-off by the nominated modelling reviewer.

```
; Script S021
;  THIS SCRIPT READS A TRIP MATRIX, CREATES A MATRIX OF CONFIDENCE
LEVELS
; AND OUTPUTS THESE TWO MATRICES FOR USE IN CUBE ME
; BY Sinead Giblin, 04 April 2007
```

Figure 3 An example of a documented script file

2.3 Run Verification, Review and Sign Off

As the risk of error are significant, agreed procedures for approving model applications need to be defined and adhered to. Typically this should include ensuring clearly documented run instructions, self-documenting model runs and summary diagnostic outputs. Persons responsible for sign-offs must be identified.

Acceptance tests for each model application are therefore desirable. Quick checks are very efficient and effective for run processing – vehicle kilometres travelled and vehicle hours travelled, should be created automatically in the outputs, for every model run, but other

diagnostics are likely to be relevant depending on the nature of the application. Configuration plots of key outputs are very efficient in completing quality checks, especially where multiple scenarios are being run.

A verification sheet will be checked by the modeller/checker at the end of each model run. The model run should then be reviewed and signed-off by the nominated technical reviewer. Persons responsible for sign-offs must be identified and take responsibility for the reviews completed. Suitable knowledge of the scenario inputs is required by the senior staff member signing off on the model run, as to whether the run results make intuitive sense. The reviewer and the date of the review should also be detailed in the Model_Log.xls, as shown in

Figure 2.

2.4 Documentation

Documentation **must** be done on a continuous basis. Where possible, programs and files should be self-documenting, so that there is no need for separate documentation. In models in use over a long period of time, issues are found, issues resolved and perceptions discovered etc. It is useful to have a system of numbered technical notes enabling this knowledge to be accumulated without the effort of more formal documentation.

3. QA System for Base Models

Acceptance tests assess the validity and accuracy of traffic models. These acceptance tests should be pre-defined and adhered to. An effective QA system should have consistent criteria for model validation and calibration. The calibration and validation should be reviewed and signed-off by a reviewer with appropriate seniority and knowledge of the model. In addition, the base models require version control and configuration management for ensuring that well-defined versions of models and data bases are used as this ensures that the most up to date versions are being used by all modellers.

3.1 Acceptance testing for model calibration and validation

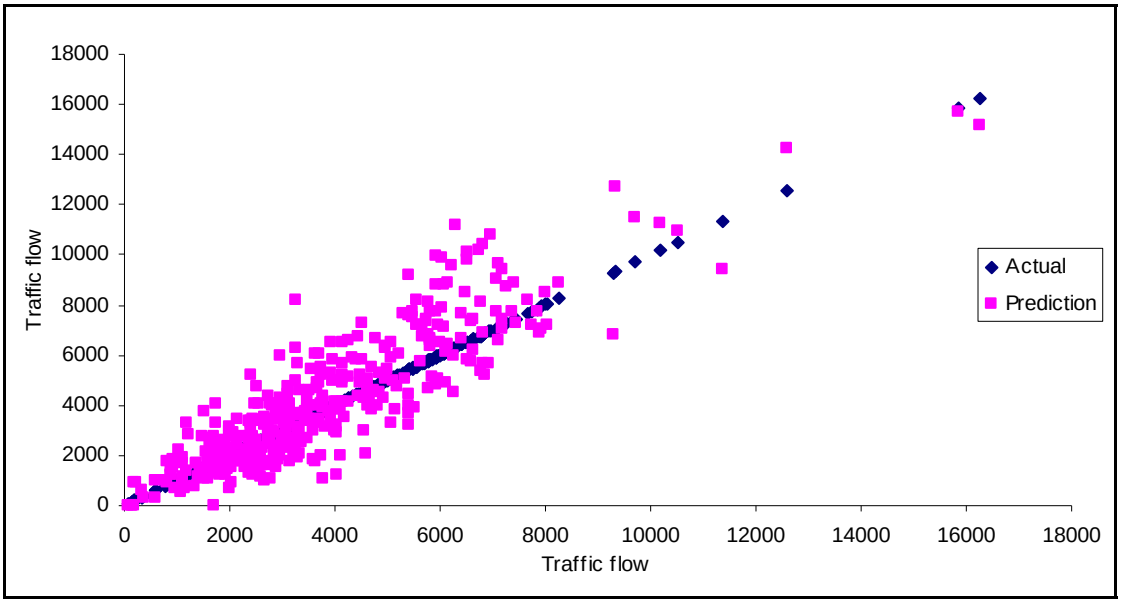
The acceptance tests can be implemented at three levels: network, screenline and link level. The measures of network level include VKT, VHT and the average VKT and VHT per person. These values can be evaluated based on the historical trend and growth rate from other resources. The test results at network level are ideal inputs for the model log file introduced in the preceding section. U.S. Department of Transportation, Federal Highway Administration (FHWA) suggests that the following acceptance criteria based on VKT relative difference should be met in urban area by road type (1):

Road Type	Population		
	Small (50-200 K)	Medium (200 K-1 M)	Large (>1M)
Motorways	18-23%	33-38%	40%
Major Arterials	37-43%	27-33%	27%
Minor Arterials	25-28%	18-22%	18-22%
Collectors	12-15%	8-12%	8-12%

Table 1 Acceptance criteria for VKT by road type (Source: model validation and reasonableness checking manual (FHWA))

At the screenline level, the modelled traffic flow and the traffic counts are compared by screenlines. The screenline check analyses the traffic movements into and out of a region and aggregates the total traffic flow on a screenline. Screenline traffic checks are a measure of traffic prediction accuracy on corridor traffic flow. The major screenlines defined by the local road authority following natural barriers such as watercourses and railway lines should at least be checked. In the large population area, minor screenlines should be included. The accepted forecasts can be decided based on the percent deviation. The maximum desirable deviation should be different for corridors with different volumes. The desirable values in terms of the total screenline volumes can also be found in (1).

The acceptance tests at the link level can be implemented either by plot check or quantitative measurement check. Plot check provides a convenient way to compare the modelled and measured traffic flow on each link. **Figure 4** shows an example of the comparison of assigned traffic with the measured values.



- **Figure 4 Plot check of traffic flow at the link level**

The Root Mean Square Error (RMSE) is one of the most common quantitative measurements of the relative accuracy of model:

$$\text{Root mean square error (RMSE)} = \sqrt{\frac{1}{\text{n}}\sum_{i=1}^{n} e_i^{\,2}}$$

Where e_i is the difference between the modelled and measured traffic numbers for link i. In practice, the RMSE should be calculated based on volume groups. The RMSE error over 100% for a low volume group will have much less impact on the assignment procedure than a high volume group with a smaller RMSE value. Table 2 shows the desirable percent deviation recommended by FHWA.

Average Annual Daily Traffic	Acceptance criteria based on percent deviation
<1,000	60
1,000-2,500	47
2,500-5,000	36
5,000-10,000	29
10,000-25,000	25

25,000-50,000	22
>50,000	21

Table 2 Acceptance criteria for link flows based on percent deviation (Source: model validation and reasonableness checking manual (FHWA))

To avoid the problem of multiple acceptance thresholds, the GEH statistic can be used to measure the prediction accuracy of peak-hour volume. This empirical measurement is introduced by Geoff Harvers of the Greater London Council in 1970s and widely used by British engineers (2):

$$GEH = \sqrt{\frac{2(v_i - V_i)^2}{(v_i + V_i)}}$$

Where v_i and V_i are modelled and measured hourly traffic volume on a link i, respectively. The modelling results are considered as acceptable if GEH values of more than 85% traffic volumes are less than 5. The GEH statistic enables a single acceptance threshold to be used over a wide range of flows with which either the absolute or the relative difference is unable to cope.

The aforementioned acceptance test procedures should be repeated every time the model is updated. The base models once completed, reviewed and signed-off by the reviewer, should be locked by the model developer. Therefore the base models are password protected against any changes/updates unless approved by the model developer.

Once reviewed (or updated) each Base model should be run under the "Model_Runs" directory in a uniquely named sub-directory. The naming convention will be BM_{year}_{desc} (eg. 2020_BM). Each of these model runs will be logged in the Model_log.xls file with details of the critical inputs and parameters (e.g. demographics, model version, application settings, network, toll assumptions, and maybe such details as special generator generation) – anything that defines the Base model as special in comparison with other base model runs.

3.2 Version Control and Configuration Management

Version control and configuration management are important for ensuring that well-defined versions of models and data bases are used. A model configuration is the combination of individual components, such as base year matrices, base matrices for individual future scenarios, base network, assignment methodology, software and parameters. These components can all change and be updated, but all changes need to be under rigorous version control and any model run should done using an identifiable 'configuration' of different versions of each of the base models and data sets.

When a model has reached an appropriate stage – typically a full set-up ready for detailed acceptance testing – then it should be brought under version control. This means:

- it has been subjected to very detailed acceptance testing designed to ensure that there are no errors and for which there is some documentary written record[1];
- it is then allocated a version number (eg 1);
- further changes cannot be made to this version without doing formal acceptance testing of the revisions, for which a record is maintained, and then updating the version number (eg 2).

If a model consists of a number of more-or-less independent modules/databases, it is safest and more convenient to control each separately, as well as the whole system. In that case each would be allocated its own version number and the system would be brought under 'configuration management' in that the version number for a system would describe the configuration (eg a system with 3 modules, at stages v1, v3 and v6 respectively would have the configuration identifier '136'). In this situation each component version would be subject to acceptance testing as would be the combined configuration of the model system.

Evidence of the suitability and performance (validation) of any particular update configuration (the acceptance tests) should always be produced and documented. A standard report contents for providing this is worth considering.

4. QA System for Scenario Testing

Strict QA processes are required to produce consistently accurate modelling results and reduce the chance of errors. These are founded on the general procedures in Section 2 and the processes involve:

- clear file naming conventions for the various networks, land use sets, toll scenarios and future years;
- update of networks through a rigorous and repeatable process (i.e. networks are not updated / changed interactively with no chance for recall of changes made);
- in processing/coding the inputs into the model mistakes are made, so a documented checking process is required, verifying the coding and confirming the correct configuration has been used.
- simple numeric numbering of model runs with transparent copy down of input files so as the model run can easily be identified / described by the input files and their underlying assumptions;
- a standard set of outputs and a rigorous review of differences between model runs to ensure they accord with expectation/previous runs.

Where a new future scenario is to be the basis of many model applications, it is particularly important that it is subjected to comprehensive acceptance tests.

[1] The record may be electronic and it need not necessarily be text – it could be electronic test results etc.

5. Conclusion

The QA procedures detailed in this technical paper are frequently implemented in well managed modelling projects. The real challenge in implementing such a system is that it may be time consuming and therefore costly to initially implement. However, once implemented it can help reduce errors and provide clarity for the modellers and reviewer involved in modelling projects.

There can be serious consequences from the application of models subject to errors – in delays, additional costs or, at the extreme, sub-optimal decisions.

This paper has sought to document the main attributes of QA systems in modelling, both in regard to model set-up and applications. The emphasis is on systematic procedures, filing and checking, all unavoidable if errors are not to proliferate.

But, as with any QA system, the approach has to be convenient and easy for the modeller and not impose large costs on the project.

Reference

1. Travel Model Improvement Program Federal Highway Administration, "Model validation and reasonableness checking manual", USA, 1997

2. UK Highways Agency, "UK design manual for roads and bridges," London, UK, 1996.