Government Commercial Function

SHOULD COST MODELLING

Technical Build Guidance

MAY 2021

Version 1.0

Contents

1. Foreword

1.1 Overview

1.1.1 The requirement to produce a Should Cost Model (SCM) when making sourcing decisions and contracting outside suppliers for the delivery of **public services** is set out within the Sourcing Playbook (see Chapter 3) and for **public works projects or programmes** within the Construction Playbook (see Chapter 5).

1.1.2 The Sourcing and Construction Playbooks set out when contracting authorities should produce an SCM, which functions are responsible for them (see Ownership, Knowledge, Understanding and Awareness framework), and how SCMs fit within the procurement lifecycle. The accompanying SCM Guidance Note provides high-level guidance on SCMs. It is part of a set of Cabinet Office guidance relating to SCMs:

- SCM Guidance Note - outlines what SCMs are, when and why contracting authorities should produce them, and key considerations around developing and/or procuring them;
- **SCM Development Guidance** - provides contracting authorities with guidance on using internal resources to design, develop, test and manage SCMs; and
- **SCM Technical Build Guidance** – the guidance in this publication is based on good practice principles for building SCMs. It is technical in nature and aimed at people who will be building SCMs.

1.1.3 A number of practical Tools and Templates have also been produced by Cabinet Office to support the development of SCMs and to help reinforce good practice approaches. These, together with the guidance set out above, are aligned to different phases/ stages of the model development lifecycle (see Figure 1).

1.1.4 Prior to reading and following the recommendations within this publication it is recommended that both the SCM Guidance Note and the SCM Development Guidance are read.

1.2 Contact

1.2.1 You should consult the Cabinet Office Sourcing Programme for further information or before planning an SCM for complex services, projects or programmes via sourcing.programme@cabinetoffice.gov.uk.

2. Introduction

2.1 Overview

2.1.1 This publication is a set of good practices that provide a structured, principles based, approach to the development of a Should Cost Model (SCM) using Microsoft Excel. Many of the principles are also applicable to other software applications. It explains how to apply good practice and does not focus on the substantive content of the SCMs themselves.

2.1.2 An SCM Build Template, which embodies the relevant good practice guidance set out herein, has been produced by the Sourcing Programme. It can be adapted as required to form the basis for development of an SCM.

2.2 Why have Guidance

2.2.1 This publication provides a mechanism to drive consistency in the approach to developing SCMs across contracting authorities. It provides a vehicle to codify and document good practice principles and techniques that inform the model under construction.

2.2.2 Embedding good practice into the development of a model serves to improve the overall quality of SCMs, reduce the time taken to develop and test them, enhance their usability, improve transparency, and help manage the risk of errors. Adherence to good practice provides benefit to the user community and contracting authorities as a whole.

2.3 Who is this Guidance for

2.3.1 This publication is particularly relevant to model developers. However, it also has utility for quality assurers as it provides a baseline against which to test a model's adherence to good practice when undertaking a Good Practice Critique (see SCM Development Guidance). It can also help to inform the expectations of model customers.

2.3.2 The development of SCMs can demand specialist skills and experience and this publication assumes that the reader has a reasonable degree of proficiency in MS Excel. It is the responsibility of those overseeing the development and use of SCMs to ensure that model developers, quality assurers and other key personnel are suitably qualified and experienced to discharge their responsibilities.

2.4 Principles of Good Practice Model Development

2.4.1 Subsequent sections of this publication present good practice principles for the development of Should Cost Models. In summary, Should Cost Models should be:

- **Planned** – SCM development should follow a structured approach that begins with progressively detailed up-front planning;

- **Logical** – the design of an SCM should be akin to a book, with logic flowing from top to bottom, left to right and front to back;
- **Aligned** – the SCM should be structurally aligned and presented in a consistent manner throughout;
- **Separated** – the SCM should be arranged in modular fashion and there should be clear delineation between inputs, calculations and outputs;
- **Transparent** – the SCM should be intuitive to use and provide transparency over calculations and logic, with no hidden data or calculations;
- **Integrous** – the SCM should be accurate and provide insight in an unbiased, easy to understand and transparent way. There should be no manipulation of the calculation logic to produce a desired result;
- **Checked** – Quality Assurance (QA) should be integral to the SCM and span the entire lifecycle (see Figure 1), from up-front QA planning through in-model checks to formal QA and testing.

Figure 1: Model Development Lifecycle (Including SCM Governance Process Overview and Supporting Cabinet Office Guidance, Tools and Templates)

Project Phase / Stage	Plan	Design	Develop	Test	Use
Activities	Quality Assurance				
	Initial Model Assessment	Scope	Specification & Design and Data	Build & Populate	Review & Formal QA	Implement & Use
Description	• Assessment of inherent model risk performed • Assess criticality of the decision and level of sophistication of the supporting model	• Requirements documented • Selection of appropriate tool / software platform • Plans for delivery, resourcing and quality assurance prepared	• Formalised model deliverables, outputs and inputs in prototype model • Agreed model logic / methodology • Codified data plan	• Model built following good practice guidance • Data prepared and populated • Developer testing performed • Additional documentation produced	• Appropriate QA and testing performed • QA documentation prepared • Release processes and requirements followed	• Decision support sign-off • Model maintenance • File management
Key Documentation to Produce*	• Initial Model Assessment	• Model Scope • Delivery Plan • QA Plan	• Model Specification (inc. Design) • Data Plan	• Draft Model • Book of Assumptions / Data Log • Updated Model Specification (inc. Design) • [User Guide] • [Technical Guide]	• QA Report • Supporting Test Memos	• Updated Model • Updated Book of Assumptions / Data Log • Updated Model Specification (inc. Design) • [Updated User Guide] • [Updated Technical Guide]
Key Sign Offs	IMA Tool	Scope / QA Plan	Specification & Design	Draft Model	QA	Final Model / Documentation
Guidance	SCM Guidance Note Initial Model Assessment Tool	SCM Guidance Note Scoping Template Planning Template QA Plan Template Development Checklist	Development Guidance Technical Build Guidance Specification Template Example SCM Build Template Book of Assumptions / Data Log Template Development Checklist	Development Guidance Technical Build Guidance SCM Build Template Book of Assumptions / Data Log Template Good Practice Build Toolkit Version Control Log User Guide Example Development Checklist	Development Guidance Technical Build Guidance Testing Procedures Book of Assumptions / Data Log Template Good Practice Build Toolkit Development Checklist	Development Guidance Book of Assumptions / Data Log Template Development Checklist
Tools and Templates						

** Key documentation to produce included in square brackets '[ ]' are optional, depending on requirements.*

2.5 Structure of this Document

2.5.1 This publication is focussed on the develop phase of the SCM development lifecycle and specifically SCM build (see Figure 1). It provides an overview of SCM Planning and Design and thereafter focuses on good practice in relation to the following, more structural aspects of SCM Build:

- **Workbook Considerations** – including organisation, features and protocols;
- **Worksheet Structure** – including sheet design, principles and presentation;
- **Input Sheets** – including input data arrangement and source referencing;
- **Calculation Sheets** – including layout, principles and formula construction;
- **Formulas and Functions** – including those which should be avoided;
- **Output Sheets** – including usage restrictions and end user communication;
- **Checking and Control** – including in-model error checks and overall QA; and
- **Other Governance Elements** – including documentation and version control.

3. Workbook Considerations

3.1 Overview

3.1.1 The guidance set out below is generally applicable at the workbook-level but will also impact individual worksheets.

3.2 Separate and Organise Worksheets

3.2.1 A workbook should be designed with clearly structured and separate categories of worksheet. Worksheets within these categories should be kept together in a clearly signposted section, as depicted in the example below:

Figure 2: Workbook Organisation

3.2.2 Separation of worksheet by category can help to reduce the risk of formulas being overwritten during data entry and inputs being overlooked when updating the SCM. Worksheet categories should typically be organised into the following groups:

- **Inputs** – these should contain only inputs to the model and be devoid of any form of calculations that form part of the model's overall logic;
- **Calculations** – these should contain only calculations and no inputs other than formulas that 'call-up' the inputs to perform calculations on them;
- **Outputs** – these sheets should contain links or 'call-ups' to calculations or inputs in order to display outputs in pre-agreed formats. They should not contain inputs or calculations (other than basic summary calculations);
- **Constants** – Sheets that contain elements used across the workbook, such as timelines and constants (e.g. days in a week); and
- **Governance** – performing checks and governance, including summary error checks, Book of Assumptions / Data Log, change and version control.

3.2.3 An SCM should 'read like a book'. Worksheets should be organised logically so that data flows from inputs at the front of the workbook through calculations to outputs at the back of the workbook. This helps usability and aids detection of errors in calculation logic. Acceptable deviations include summary output sheets at the front of the SCM, which can aid understanding and usability.

3.3 Define Formats and Styles

3.3.1 An SCM should include and adhere to a defined format and colour scheme that guides model operators. It should clearly define colours/styles that are to be used for input cells, calculation cells and output cells in order to guide the user on how to use and interpret the model.

3.3.2 These formats and styles should be documented on a discrete sheet in the workbook, so everyone interacting with the model will have a reference point to understand what each colour or style means.

3.4 Adopt a consistent signage convention

3.4.1 A consistent signage convention (the treatment of positive and negative numbers) that that supports model usability should be considered and adopted. There are several different approaches to signing within a model. For example, adopting positives as default (all figures within the model are positive apart from exceptions); the inflow/outflow convention (using inflows/outflows to separate from positive and negative values); and using natural signage in accordance with accounting principles.

3.4.2 What is important is that a consistent approach is adopted and that the treatment applied to numbers is documented within the model. Consider trapping inappropriate data entry through error checks (see Section 9.1), Data Validation (see Section 5.5) and incorporating the desired signage into the labels (see Section 3.10) to instruct the user on how to correctly enter or interpret data (e.g. “+ve Values Expected”).

3.5 Use Intelligent Formatting

3.5.1 Microsoft Excel’s inbuilt Conditional Formatting is impactful and can be used for error checks (see Section 9.1). It can also be used to guide the user to enter specific inputs that, for example, may only be required under certain switch settings or to ‘grey out’ those that are not currently in use. In addition, it can also help to dynamically highlight certain outputs or outliers (e.g. highlight all values over a certain threshold).

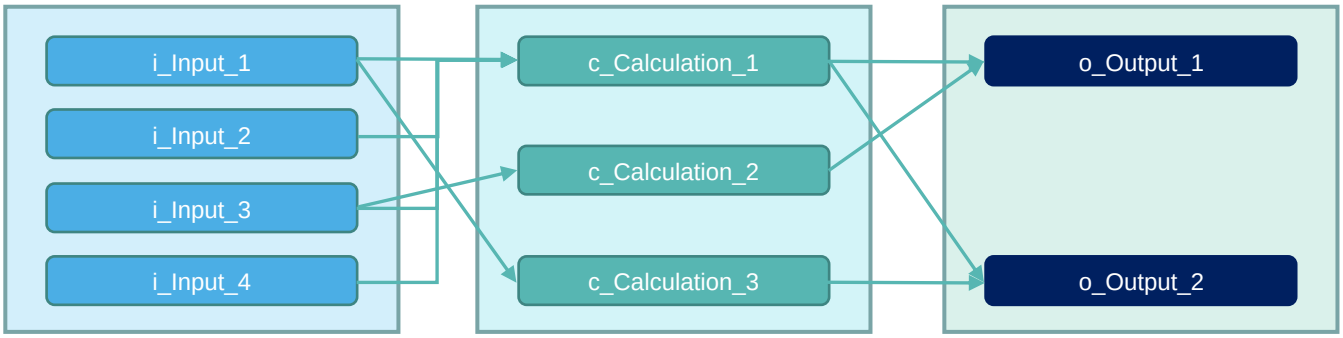
3.5.2 Despite its advantages, if over used Conditional Formatting can start to detract from usability and, moreover, can adversely impact a model’s size and performance. Whilst the use of Conditional Formatting is encouraged, its use should be assessed against whether it will enhance usability of the model or adversely impact its performance.

3.6 Contents Page, Model Description and Disclaimers

3.6.1 All SCMs should contain a contents page with a Model Map (see example in Figure 3 below). This map will give model operators and users of model outputs insight into how the model has been constructed and the interrelationships between inputs calculations and outputs. It will support QA and testing of the model and aid navigation when in use.

3.6.2 Although not essential, hyperlinking the contents page to the individual worksheets that are included can speed navigation and improve overall usability. Including a return hyperlink back to the contents page, in a consistent place within the freeze pane of each worksheet, will further aid usability.

Figure 3: Example Model Map



3.6.3 SCMs should also contain a worksheet that describes the purpose for which the model has been designed, what it can be used for and what its limitations are, including limitations arising from QA and testing. The purpose of this is to educate model operators on what tasks the model is and is not designed to support.

3.6.4 Some SCMs may also require a disclaimer page, particularly if they are being shared with third parties, or an appropriate Important Notice added. For example, this may include information on the model status (e.g. 'draft and not to be relied upon'), the SCM's Protective Marking and any associated handling instructions, and who or which organisations the model is intended to be used by and the basis of distribution and use.

3.7 Include a User Guide within the SCM

3.7.1 A model should contain a User Guide that covers how a user should interact with the model. This should extend beyond how to update and maintain inputs to cover how the outputs from the model should be used and what decisions they are designed to inform.

3.7.2 The User Guide should provide an overview of each worksheet within the SCM. This should describe its purpose and highlight salient points, such as the requirement for user input or key calculations performed, and how the worksheet relates to others.

3.7.3 For more sophisticated models it may be appropriate to produce an additional, more detailed, User Guide in a separate document. Some models may also benefit from a Technical or Developer Guide (e.g. to support changes to the model). The management of User and Technical or Developer Guides over their lifecycle should be addressed appropriately (see SCM Development Guidance for further information on file management).

3.8 Use Succinct and Logical Worksheet Names

3.8.1 Worksheet names should infer meaning to the user and be simple to understand. They should be succinct and prefixed with an indicator to denote the worksheet category they belong to (e.g. Inputs or Calculations).

3.8.2 Inclusion of spaces within worksheet names should be avoided in favour of a hyphen, underscore or similar. For example, use "Input_1" or "Input-1" in favour of

"Input 1". When spaces are included, Microsoft Excel automatically includes quote marks around the name of a referenced worksheet. This increases formula complexity and detracts from formula readability.

3.9 Use Consistent Worksheet Headers

3.9.1 A header to be used consistently on all worksheets should be designed. It should contain key information such as worksheet name and the model's protective marking, consistent column headings and reference to the master and worksheet level error check status (see Section 9.4). It should also contain a hyperlink back to the contents page to facilitate navigation (see Section 3.6).

3.9.2 The header should present the model's timeline (as appropriate). The timeline itself should be constructed once on a dedicated sheet in the constants section and each worksheet should link to it. The timeline should not be repeatedly constructed on different worksheets within the model.

3.9.3 Freeze Panes should be applied at a worksheet level to ensure that the key information in the header, such as the timeline, is visible when scrolling down or across the worksheet.

3.9.4 Column consistency should be maintained throughout the workbook wherever possible and even if it means including blank, unused, columns on some worksheets.

3.10 Signpost Using Labels

3.10.1 Labels help the user community to navigate, interpret and interact with the model. Models should contain sufficient, dedicated labelling columns to guide the user through the model. Label descriptions should be succinct but sufficient to clearly articulate the object.

3.10.2 Labels that refer to the same thing should be consistent throughout the workbook in order to avoid confusion and reduce the risk of misinterpretation. A consistent capitalisation approach should also be adopted.

3.11 Maintain a Unit of Measurement Column

3.11.1 There should be a column, in a consistent place across all worksheets, with the dedicated purpose of informing the user of the units of measurement attached to the modelling element. Each element should have a label to show what it is being measured in, for example, unit, currency value, weight or percentage.

3.11.2 For units that pertain to monetary values the underlying assumptions should be overtly clear, for example, if they include or exclude VAT or inflation. Different terminologies may be in use, for example, constant and outturn or real and nominal and, if used, a definition should also be included within the model. For example, outturn costs are assumed to include VAT and reflect the impact of inflation.

3.11.3 Maintaining a clear unit of measurement in a consistent column throughout the model reduces the risk of errors, aids understanding and facilitates QA and testing.

3.12 Timelines Should be Horizontal

3.12.1 Timelines should be constructed horizontally across columns and not vertically down rows. Potential deviations may include a need to present time-based outputs in a vertical format or to ease the user selection of time-based inputs. In these situations, the addition of vertical timeline, driven off the master horizontal timeline, may be necessary.

3.13 Timelines Should be Consistent

3.13.1 In all worksheets that include a time range the placement of the time line should be consistent. It should start and end in the same column on each worksheet and it should reference back to the master timeline in the constants section. This will reduce the risk of errors and improve SCM usability.

3.13.2 Where there are primary (e.g. months) and secondary or summary time periods in a model (e.g. years) they should cover the same duration. For example, the secondary or summary time period for a model that has 36 months as its primary time period should be 3 years.

3.13.3 Where timelines are based on financial as opposed to calendar years, this should be defined within the model/model documentation and the header should make it overtly clear which years or periods are covered (e.g. 2021/22 c.f., FY22). If different financial years are included within the model then the month should also be included in the header (e.g. Apr-21/Mar-22).

3.14 Timelines Should be Appropriate

3.14.1 Timelines should be at a suitable level of detail for the model, for example, monthly or annual. Where monthly or more granular values are not required, timelines should be annualised to reduce overall complexity of the model. Timelines should generally be constructed on calendar basis (i.e. using date-based formats).

3.15 Avoid Duplication of Elements

3.15.1 Inputs should only be entered once within a model and should never be duplicated. For example, there should only ever be one place to enter the Treasury Discount Rate within an SCM and all calculations that require it should refer to that single entry. This makes the SCM easier to understand, easier to update and reduces the risk of inputs being missed during updates.

3.15.2 The same calculation should not be reproduced multiple times within a model. Rather than performing the same calculation several times in a model, it should be performed once and then ‘called-up’ when used in subsequent calculations within the model. This makes the SCM easier to understand, improves performance and reduces the risk of calculations not being amended consistently. The exception to this is error checks (see Section 9.2), where the duplication of calculations is often necessary.

3.16 Use a Single Workbook

3.16.1 As a general rule, an SCM should not be split over multiple workbooks. Inter-linked workbooks can be difficult to manage and maintain, and pose additional risks. If source data is contained in a different workbook it is preferable to copy and paste it into the model, removing formulas, rather than through the use of links (see Section 5.4 on the use of external links).

3.17 Avoid Multiple Copies

3.17.1 Use of multiple copies of the same SCM to run different scenarios should generally be avoided in favour of designing a model that can accommodate scenarios. The administration of multiple models is more challenging, consumes resources and raises the risk that they are not consistently updated or maintained.

3.18 Avoid Visual Basic for Applications

3.18.1 Visual Basic for Applications (VBA), sometimes referred to as 'macros', is a programming language contained in MS Excel. In comparison to native Excel is typically less transparent, less accessible and more susceptible to errors. The use of VBA to perform calculations that impact SCM outputs should therefore be avoided. If VBA is deemed necessary its development should be restricted to Suitably Qualified and Experienced Personnel (SQEP) and, furthermore, it should only be undertaken if:

- The VBA code is developed in line with good practice principles; and
- Independent SQEP are available to quality assure and test the VBA.

3.19 Avoid use of Custom Functions

3.19.1 Designing a Custom Function through VBA that allows for the performance of a custom calculation (e.g. to apply a mark-up to a cost) should be avoided. Custom Functions lack transparency compared to formulas that are native to Microsoft Excel, present an increased risk of error and are less accessible (see Section 3.18).

"An SCM should 'read like a book'. Worksheets should be organised logically so that data flows from inputs at the front of the workbook through calculations to outputs at the back of the workbook."

4. Worksheet Structure

4.1 Overview

4.1.1 The guidance set out below is generally applicable to all types of worksheets within the SCM.

4.2 Worksheets Need a Discrete Purpose

4.2.1 Worksheets should serve a discrete purpose. In the main they should be used for capturing inputs, performing calculations or presenting outputs. There will also be worksheets that perform governance functions and those which enhance usability (e.g. a contents page). These discrete purposes should not be combined into single worksheets. Separating purposes across worksheets reduces the risk of errors, improves usability and makes a model easier to understand and navigate.

4.3 Worksheets Should Flow Logically

4.3.1 A worksheet should 'read like a book'. The logic and calculations should flow from left to right and top to bottom. References to calculations performed further down a worksheet should be avoided. If required, they are typically best performed on a separate worksheet and 'called up' as required. Any deviations should be clearly marked.

4.3.2 The organisation of calculations across different worksheets should, in part, aim to reduce the number of inputs to and outputs from a worksheet. The requirement for significant numbers of inter-linkages between worksheets may be indicative of a suboptimal model design.

4.4 Columns Should be Used Consistently

4.4.1 Each column on a worksheet should serve a unique purpose and this should be consistent across worksheets. Having columns that consistently display a particular data point (e.g. unit of measure or time independent values) or locating all table headings in a particular column will bring consistency to the user, make a model easier to navigate and understand and reduce the risk of error.

4.5 Include Totals in the Freeze Pane

4.5.1 The inclusion of display totals can aid understanding and help with QA and testing. Where totals are based on summed time-series data they should be:

- **Separate** – considered as time independent and held in a column that sits outside of the time series columns (see Section 4.6); and
- **Frozen** – be located on the left-hand side of the worksheet within the freeze pane so they are visible when scrolling.

4.6 Separate Constant and Time Series Data

4.6.1 When designing the model, it is important to define data that is time dependent and that which is not. Columns should be organised to reflect the time dependant or time independent nature of the data that is being included. Time dependant data should be arranged into consistent columns using the model's timeline sheet as a guiding reference point. Time independent data, such as a discount factor, should be consistently held in a column that does not overlap with any time sensitive data points or calculation. An example layout for Constant and Time-Series data is shown below:

Figure 4: Example Constant & Time-Series Layout

			Ref.	Units	Constant	2020	2021	2022	2023	2024
1	Costs									
	1.1	Insurance Costs								
		Insurance Costs	4.02	£Nominal		5,000	5,250	5,513	5,788	6,078
2	Financial Assumptions									
	2.1	Discount Rate								
		Discount Rate	3.07	%	3.5%					

4.7 Arrange in Labelled Sections

4.7.1 Worksheet content should be arranged in discrete sections and subsections. Indenting subsection labels according to hierarchy aids interpretation. Indenting labels across separate columns aids navigation (e.g. by using Ctrl+↓ and Ctrl+↑).

4.7.2 Unique calculations should be given unique labels in order to delineate them from others. Where calculation blocks are repeated it may be appropriate to preface labels with section headers (e.g. Division A Total Personnel Costs, Division B Total Personnel Costs). This may best be achieved with dynamic labels linked to a master label.

4.8 Keep all Model Elements Visible

4.8.1 There should be nothing hidden within the SCM. All worksheets should be visible and there should be no elements of the model in hidden columns or rows. Having all components visible and transparent aids understanding, increases trust, reduces the risk of elements being unchecked and, significantly, of elements being inadvertently overwritten. Exceptions to this include:

- **Outline** – Use of Microsoft Excel's Group & Outline feature, which can improve usability and aid navigation;
- **Edges** – Hiding columns to the right of or below the entire used range, which can aid model navigation; and
- **Sheets** – by exception, if data has to be hidden it should be organised so that the whole worksheet, rather than parts of a worksheet, can be hidden.

4.9 Display the True Value of Numbers

4.9.1 Numbers should be presented at their true value. They should not be formatted to display at anything other than their true value (e.g. displaying 000s as MM or displaying -ve as +ve values). An exception to this may be the formatting of error checks. Rounding up or down should also be avoided.

4.10 Avoid Cell Merging

4.10.1 Avoid the merging of cells. It hampers cell selection, can be problematic for VBA and the unmerging of cells can compromise the integrity of calculations. Use of Microsoft Excel's Centre Across Selection feature can provide an alternative to cell merging.

4.10.2 Cell merging can also hamper the application of verification or logic testing software and, in some cases, may need to be removed to enable testing to be undertaken.

5. Input Sheets

5.1 Overview

5.1.1 Input sheets are the mechanism for importing data into a model. They should be used solely to capture data and assumptions and no calculation or analysis that feeds the model's calculation logic should be performed on them.

5.2 Arrange Inputs logically

5.2.1 Input sheets should be arranged with a logical structure. There are no prescribed rules although typically they are organised according to:

- **Nature** – organised by their nature such as constant or time series;
- **Form** – organised by form such as historical and actual balances;
- **Function** – organised by function such as costs and revenues; and
- **Provider** – organised to align with individuals who provide them.

5.2.2 In general, the manner in which inputs are organised should mirror the way that the calculations are organised.

5.3 Reference Input Sources

5.3.1 Input sheets should contain a column that has the sole purpose of referencing the data to the Book of Assumptions / Data Log (see Section 10.2). For example, the 'Ref.' column Figure 4, where the reference number corresponds to an item in the Book of Assumptions / Data Log. This enables the provenance of input data to be established and allows for model calculations and outputs to be understood in the context of the data that underpins them.

5.3.2 Include a dedicated comments column to add supplementary information and use this in favour of Microsoft Excel's Cell Comment feature, which is less transparent.

5.4 Avoid External Links

5.4.1 The manual interchange of data between an SCM and other models or 'feeders' is preferable over inter-linked workbooks. If direct links are required, for example to pass live values dynamically in order to perform calculations, the splitting of the model should be bought into question.

5.4.2 If direct links are deemed necessary then they should adopt the following good practices:

- **Called-Up** – formulas linking to external files should be akin to 'call-ups' and simply link to the source data. Calculations should not be performed within call-up formulas;

- **Delineated** – links to an external file should be considered as quasi inputs. They should be clearly delineated and organised on a dedicated data import sheet. Similarly, links out of a source file should be considered as outputs and organised on a dedicated data export sheet;

- **Range Named** – the source data should be included in a Named Range and the destination file should link to the data via the Named Range. This will help to maintain the integrity of the link in the event the structure of the source file is modified; and

- **Sign Posted** – the source file should clearly signpost that specific data is being exported and include details of the onward destination usage.

5.5 Apply Appropriate Validation

5.5.1 Inputs that need to be restricted should be restricted using Microsoft Excel's Data Validation features. For example, they may need to be restricted to particular sets of values, ranges of values or types of values. Providing feedback to the user community on the restrictions, for example via labels or via the labelling feature within Microsoft Excel's Data Validation itself, can aid usability.

5.6 Include Error Checks

5.6.1 Whilst Microsoft Excel's Data Validation can help to ensure that inputs are appropriate it may not always be possible to apply the desired restrictions. For example, it can't prevent the deletion of values and ensure that they are populated. It may be appropriate, therefore, to include checks that all of the required input cells are populated. Furthermore, Microsoft Excel's Data Validation cannot always be relied upon and the inclusion of additional in-built checks that feed into the error check network (see Section 9.4) may be appropriate.

6. Calculation Sheets

6.1 Overview

6.1.1 Calculation sheets apply mathematical functions to inputs to produce the analysis and insight designed into a model. As a rule, calculation sheets should not contain inputs or require model operator interaction.

6.2 Use Consistent Formulas

6.2.1 Calculations performed over a series or block should be consistent both horizontally and vertically and should cover the entire range of the calculations. Formulas should not change part way across a row or down a column without separation by a blank column or row respectively. If exceptional circumstances demand a change it should be very clearly delineated (e.g. through formatting).

6.2.2 Whilst they may be consistent, formulas that include reference to a cell range should refer to the entire range and not sub-elements within that range. Partial range references can be more difficult to test and increase the risk of error in the event of change. Where reference to a sub-element is required, the entire range should be referenced and the specific element extracted using functions such as INDEX or SUMIF.

6.3 Section Calculations into Blocks

6.3.1 Calculations should be organised into distinct, appropriately labelled, blocks. Each block should be separated from other model elements by blank rows and/or columns and clearly signposted through section headers and/or labels. Totals, if included, should be delineated from calculation blocks and clearly identified.

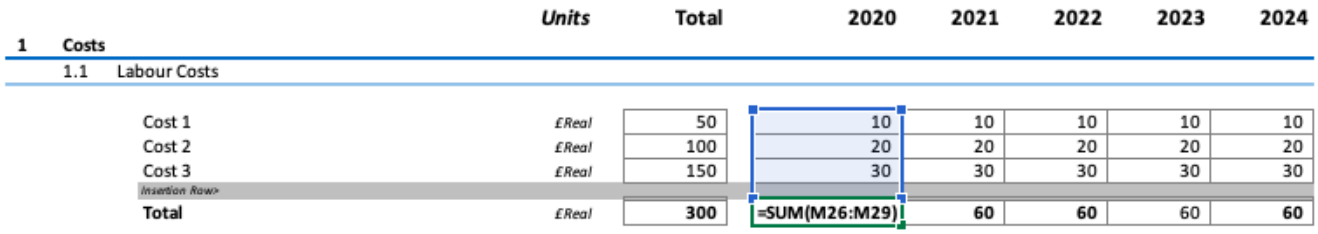
6.3.2 Akin to worksheet logic, calculation blocks should use a consistent formula that works from left to right and top to bottom across the entire block. This reduces the risk of errors, makes them easier to identify if they are present and expedites the testing process as fewer unique formulas require checking. Any deviations should be very clearly delineated.

6.3.3 Blocks should be constructed with the minimum required cell anchoring (‘$’ sign) in order to reduce the visual complexity of formulas and enable them to be copied and re-used as required.

6.4 Use Insertion Rows

6.4.1 If total lines are required to sum a block of calculations then an insertion row should precede and be included in the summation. This reduces the risk of additional items, if added at a later date, not being included in the SUM calculation. This is a frequent cause of errors in models. An example of this technique is as follows:

Figure 5: Example Insertion Rows in Sum Formula



			Units	Total	2020	2021	2022	2023	2024
1	Costs								
	1.1	Labour Costs							
		Cost 1	£Real	50	10	10	10	10	10
		Cost 2	£Real	100	20	20	20	20	20
		Cost 3	£Real	150	30	30	30	30	30
		Insertion Row>							
		Total	£Real	300	=SUM(M26:M29)	60	60	60	60

6.4.2 Similarly, insertion rows should also be included beneath input ranges that have potential to expand in the future and the insertion row included in range references to them.

6.5 Use Consistent Ordering

6.5.1 Where calculation components are repeated the ordering of components relative to others should, wherever possible, be kept consistent. For example, the pattern of 'call-ups' followed by timing flags followed by calculation block should be repeated consistently where applicable.

6.6 Single Purpose Formulas

6.6.1 Formulas should be designed to serve a singular purpose where practicable. Having multiple calculation steps in a single formula should be avoided. It is better to break down calculation steps into a logical and simple sequence than to create a complex formula that performs several calculations at once. Having single purpose formulas makes understanding them and using the model simpler and makes them easier to modify if required. Furthermore, it is easier to identify and remedy errors if calculations are approached sequentially.

6.6.2 A formula that contains more than one function or is longer than the length of the reading pane should likely be broken down into multiple steps. This, however, needs to be balanced against any impact that this may have on the overall complexity of the model and the ability to understand it.

6.7 Use Flags & Factors

6.7.1 SCMs are invariably founded on time-based assumptions and, where applicable, timing flags should be used. These should typically be based on a Boolean logic approach (i.e. 1s and 0s) and display constituent components on separate rows. This provides a visual aid that helps drive understanding and facilitate testing activities.

6.7.2 Depending on where and how they are used it may be appropriate to include flags and partial period factors on a dedicated worksheet in the constants section (see Section 3.2.2).

6.8 Keep Formulas Simple

6.8.1 Formulas should be kept as simple as possible to achieve the desired result. If a formula is difficult to readily understand or explain it may be too complex. Complex Excel formulas should be avoided if a simpler solution can achieve the same result. Use of unfamiliar formulas, such as Array formulas, should be avoided to increase the usability and make the model easier to test, and understand by the general user population.

6.9 Avoid use of Array Formulas

6.9.1 Array Formulas (that are presented visually within '{ }') can perform one or more calculations on an array of data. Their use should be avoided as they can decrease workbook performance, increase model complexity and make the calculation logic more difficult to understand.

6.9.2 If there is a need to perform one or more calculations on an array of data within a model the use of Microsoft Excel's native array-based formulas should be considered (e.g. SUMPRODUCT).

6.9.3 It is important to note for any formula, such as an Array Formula, that relies on a predetermined range, that additions to the model will likely necessitate its reconfiguration to ensure that formulas remain correct and produce appropriate outputs.

6.10 Exclude Current Sheet References

6.10.1 Formulas with references to the current worksheet that include the name of the current worksheet should be edited to exclude the name of the current worksheet. These references can be included automatically by Excel when constructing formulas. Inclusion of current worksheet names can be confusing and adds unnecessary visual complexity to formulas.

6.11 Avoid 3D Formulas

6.11.1 Three dimensional formulas, which perform calculations using data from a different sheet in a workbook, should be avoided where practicable. They can be difficult to understand, are more complicated to test, and have a higher risk of error. The approach of using 'call-ups' to pull data from one worksheet to where it is required on another is preferable.

6.12 Avoid Circular Calculation Logic

6.12.1 Producing models that have a circular calculation logic should be avoided. They can be difficult to understand, test and, moreover, can produce erroneous results. Alternative approaches are almost always available. Where they are not, VBA (see Section 3.18) could be considered to manage circular calculations, with integral checks constructed to help ensure that the SCM has resolved to a stable solution.

6.13 Do Not Include Constants in Formulas

6.13.1 Formulas should not contain constants or 'hardcoded' values. Having a constant embedded into a formula (e.g. to divide by 100) should not be designed into the model. Having a defined input section that contains the constant value referenced by the formula provides the same result whilst maintaining transparency, increasing model auditability, easing the update process and reducing the risk of error. Exceptions to this rule include requirements for predefined formula arguments (e.g. a 1/0 for True/False).

6.14 Ensure Formulas are Readable

6.14.1 Construct formulas in a manner that facilitates readability. Do not over use brackets or use excessive anchoring ('$' sign). Only use brackets and anchoring to achieve the desired mathematical or functional objective. Considered inclusion of spaces and carriage returns can make formulas easier to read. These techniques will keep formulas free from clutter and make them easier to read, understand, and test.

6.15 Signpost Temporary Code

6.15.1 Code should generally be progressed to a completed state before development is paused (e.g. over a weekend). If a model is saved with temporary code, the code should be clearly signposted and visually identifiable as such.

"Having multiple calculation steps in a single formula should be avoided. It is better to break down calculation steps into a logical and simple sequence than to create a complex formula that performs several calculations at once."

7. Formulas and Functions

7.1 Overview

7.1.1 Formulas and functions are the mechanisms that Microsoft Excel uses to perform calculations and organise data. Oftentimes, the same result can be achieved through the use of different formulas and functions and choice can be based on model developer preference or style. However, certain formulas and functions have key advantages over others and should be employed accordingly.

7.2 Avoid the NPV Function

7.2.1 Microsoft Excel's inbuilt NPV function lacks transparency and can produce unintentional results as it applies end of period discounting that may be inappropriate. Its use should be avoided.

7.2.2 Calculating Net Present Value through the creation and application of a time series discount factor is the preferred approach. Furthermore, through explicitly displaying the discount factor, the time value of money is more readily comprehendible. This can support testing and give greater confidence to model operators.

7.3 Avoid the OFFSET Function

7.3.1 The OFFSET function has considerable utility when performing depreciation calculations, and in some situations, it affords the only solution. However, in general, the OFFSET function should be avoided as inter-dependencies are difficult to trace and this hampers testing.

7.4 Avoid the INDIRECT Function

7.4.1 The INDIRECT function can have considerable utility for dynamic selection, and in some situations, it affords the only solution. However, as it relies on defined names and ranges the results obtained through INDIRECT are volatile and have an inherent error risk. Use of the function should be avoided.

7.4.2 The INDIRECT function is also at risk of error if worksheet or cell references are included as constants and the underlying structure of the workbook is changed. Where the use of the INDIRECT function is deemed necessary, efforts should be made to improve its robustness through, for example, use of named ranges or dynamic cell address labels.

7.5 Use INDEX Over CHOOSE

7.5.1 The CHOOSE function has utility for making selections across worksheets. However, in general, the INDEX function should be used in favour of the CHOOSE

function if viable. With the INDEX function it can be easier to expand the range of selectable options and it can be less memory intensive than the CHOOSE function.

7.6 Use INDEX Over IF

7.6.1 Use the INDEX function in favour of the IF function for picking values. It can be more robust and easier to expand the range of selectable items.

7.7 Use INDEX and MATCH Over V/HLOOKUP

7.7.1 Use a combination of INDEX and MATCH over VLOOKUP or HLOOKUP. It can be more robust, more memory efficient, easier to modify and easier to test. V/HLOOKUPS are often cited as causes of model errors.

7.8 Avoid Blanket Wraps

7.8.1 Avoid the use of blanket error wraps such as ISERR and ISERROR as they can mask genuine errors. Instead, use traps that catch specific errors or the underlying driver of errors such as dividing by zero (e.g. trap the ‘0’ not the resultant ‘#DIV/0!’).

7.9 Use Named Ranges Cautiously

7.9.1 Microsoft Excel’s Named Ranges should be used to reference cells from within VBA code (see Section 3.18.1) and for passing data between inter-linked workbooks (see Section 5.4.2). They may also be required for dynamic selection (e.g. dynamic Named Ranges) and can have utility for particular items such as VAT. However, excessive use of named ranges can hamper understanding and flexibility and their over use should be avoided.

7.9.2 Akin to worksheet names, Named Range names should be succinct, meaningful and, where applicable, adhere to a logical naming convention.

7.9.3 It is good practice to include a list of all Named Ranges in model documentation (see Section 3.7), together with details such as their location and purpose. This is particularly important for dynamic Named Ranges owing to their reduced transparency.

8. Output Sheets

8.1 Overview

8.1.1 Output sheets contain the product of the SCM's calculations in the desired output format. They draw from calculations, and potentially inputs and controls, and may also include tables and graphs or charts.

8.2 Include Dedicated Output Sheets

8.2.1 SCM outputs should be shown on dedicated sheets that 'call up' the appropriate section of the calculations or inputs. The use of output sheets minimises the good practice compromises that may otherwise have to be made in order to construct and arrange calculations in a manner that meets the output format requirements.

8.2.2 With the exception of links to other sheets (i.e. 'call-ups') or basic summations, calculations should be avoided on output sheets. If calculations are found to be necessary then they should be performed on the calculation sheets themselves or additional, interim, calculation sheets should be included.

8.2.3 Output sheets should not include any inputs. If inputs need to be presented on output sheets, they should be included in the appropriate input sheet and pulled through to outputs sheets via 'call-ups'. Potential exceptions include inputs to scenario or sensitivity switches. However, a dedicated scenario and/or sensitivity control sheet or the use of switches connected to appropriately located inputs should be considered.

8.3 Outputs Should be Fit for Purpose

8.3.1 Output sheets should articulate the answers to the key questions that the model was designed to support. Where there are prescribed presentational formats, these should be considered in the assessment of the SCM's overall fitness for purpose. Additional considerations relate to the level of detail and/or summary provided and whether outputs, taken holistically, provide the appropriate information needed by model customers.

8.4 Outputs Should be Clearly Labelled

8.4.1 Outputs should be appropriately labelled and in a way that minimises the risk of potential misinterpretation by model operators. Without the benefit of detailed calculation logic, the output labels may need to be more descriptive than those within calculation sheets.

8.5 Errors Should be Clearly Flagged

8.5.1 As a minimum, the status of any in-model or known errors and reference to limitations should be clear on the output sheets. This is necessary to reduce the risk of outputs being used inappropriately. More stringent measures, which may be appropriate for some SCMs, include obfuscating the outputs to prevent access in the event of an in-model error.

8.5.2 The ability to help identify errors and provide confidence in the authenticity of the SCM can also be provided through the inclusion of Key Performance Indicators (KPIs) and graphs or charts as part of the outputs.

8.6 Graphs Should be Appropriately Constructed

8.6.1 Visual outputs from the SCM should not be misleading. For example, the labelling of graphs or charts included in scenario models should make explicit which outputs are being presented (e.g. what scenario they relate to).

8.6.2 The data that is used to power graphs or charts should be clearly delineated within the SCM. This is often best achieved by having a dedicated sheet or section on a calculation sheet for graph or chart data feeds, which may also include graph or chart labels (e.g. chart title including the scenario name).

8.7 Restrict Pivot Table Use

8.7.1 The use of Pivot Tables to perform calculations that feed the overall model logic should be avoided. They can adversely impact performance and their static nature and lack of configuration control presents an increased risk of error. Their utility lies in displaying model outputs in a structured way that can be rapidly reconfigured and investigated by the model operator. However, if outputs are to be presented in a consistent format, a preconfigured table should be used rather than a Pivot Table.

8.7.2 Where Pivot Tables are deemed necessary, the settings for Rows, Columns and Fields should be displayed consistently and not hidden in order to reduce the risk of misinterpretation. Clear instruction on any requirement to refresh or configure Pivot Tables should also be included.

8.8 Restrict Data Table Use

8.8.1 Data Tables can adversely impact the performance of a model and, compared with traditional approaches, they can hamper flexibility. In general, their use should be avoided.

> “Output sheets should articulate the answers to the key questions that the model was designed to support.”

9. Checking and Control

9.1 Overview

9.1.1 All SCMs should include in-built checks and controls. Their inclusion and application form an essential part of helping to ensure that SCMs are fit for purpose.

9.1.2 The application of checks and controls is discussed in HM Treasury's Aqua Book.

9.2 An SCM Needs In-built Checks

9.2.1 Notably, the inclusion of in-built checks and controls within an SCM is complementary to, and is in no way any form of substitute for, developer testing or formal QA and testing. All SCMs need to be subject to appropriate QA and testing (see SCM Development Guidance).

9.2.2 Checks, which include error checks and warning flags, should be designed and built into the SCM from the outset. They should form part of the underlying SCM design and should not be treated as an afterthought.

9.2.3 The inclusion of checks can give confidence to the user community that the SCM is being operated correctly and that the SCM's inputs and outputs are in-line with expectations. The nature of these checks, which need to be designed and built into the SCM, are highly varied but may typically include:

- **Arithmetical** – e.g. do the numbers add up, which might include checks based on calculating them in a different way;
- **Outliers** – e.g. do values fall within expected ranges, which might include checks against upper and lower bound values;
- **Format** – e.g. are inputs in the expected format, which might include checks for numerical or alphanumerical formats; and
- **Completeness** – e.g. have all inputs been populated, which might include checks for blank input cells.

9.3 Checks Should be Robust

9.3.1 A well-established error check approach is the 'sum zero' method. With this, errors are flagged in check cells if the check cells contain any value other than zero. This method supports the summing of individual error checks into a 'network'.

9.3.2 Error checks should be constructed in a consistent manner throughout the model in order to reduce the risk of misinterpretation and/or errors amongst the check-cells themselves (e.g. adopt sum-zero approach throughout).

9.3.3 Error checks themselves should be robust. Notably, values returned by sum-zero based check cells need to be sign restricted to prevent +ve and -ve check-sums

cancelling each other out. Additional measures to prevent sum-zero error checks registering as 'OK' if the formula that underpins them is inadvertently deleted or hardwired to zero should also be considered.

9.3.4 It may be appropriate to include a small error tolerance within sum-zero checks to prevent false triggers. These can, for example, occur as a result of small rounding 'errors' produced by Microsoft Excel. Notably, tolerance values should be included in the constants section and not be included as hardwired values within error checks (see Section 6.13).

9.4 Checks Should be Networked

9.4.1 Error checks performed on individual worksheets should be consolidated into both a worksheet and a workbook level check that displays the error status for the entire workbook. This should be undertaken on a dedicated error check summary worksheet that shows the status of error checks on every worksheet as well as for the workbook as a whole.

9.4.2 The error check summary sheet is useful for reviewing and checking the status of the model when in use and for providing a quick reference to any worksheets that contain errors.

9.5 Checks Should be Visible

9.5.1 Error checks should be overt and the presence of an error should be highly apparent to the model operator. The use of Microsoft Excel's Conditional Formatting (e.g. green for OK and red for ERR) is more impactful than simply returning the value of an error check.

9.5.2 Error checks should be included at appropriate locations in the model, such as where errors can be triggered. This will alert model operators, including those making changes, of any potential data entry, operation, corruption or other errors.

9.5.3 Worksheet level error checks that indicate the presence of an error anywhere on the worksheet should be included in every worksheet. These checks should be located in the worksheet header within the freeze pane at the top of worksheet so that they are always visible, even when scrolling.

9.5.4 Each worksheet should also include a workbook-level error check adjacent to the worksheet-level check. This check should inform the user if an error is present on another worksheet or anywhere else in the workbook.

9.5.5 Worksheet and workbook-level error checks should be in a consistent position across every worksheet within the workbook. This will reduce the risk of errors trapped by the in-built checks not being noticed by the model operator.

9.6 Cell Protection Should be Applied

9.6.1 Cells within a model that are not designed to be input into by the user community should be protected to limit the risk of corruption. Applying cell protection followed by worksheet protection will reduce the risk of errors through unintentional

changes. For example, the overtyping or hardwiring of calculations or outputs. Use of passwords should generally be avoided and, if used, should be appropriately managed (see SCM Development Guidance).

9.7 Undertake Developer Testing

9.7.1 The model developer should undertake their own stress and other testing throughout the SCM's development. These tests might include:

- **Extremes** – examining model behaviour when subject to very large or small inputs (e.g. how does it cope with zero or negative values);
- **Precision** – examine any provided data to see if it appears overly precise;
- **Patterns** – examining how the model behaves under simple inputs (e.g. 1%, 10%' 1, 10, 100), which can better enable visual evaluation of behaviour;
- **Relative** – examine the relative difference between values or metrics to see if they make sense (e.g. price > cost; days worked < working days);
- **Trends** – examine if trends make sense (e.g. cost growing over time). Add graphs or charts and KPIs to support sense checking as required; and
- **Break** – can you 'break' the model (e.g. deleting input values or formulas) and are in-built checks and protection sufficient to help counter this.

9.7.2 In addition to ongoing testing throughout SCM build, the model developer should undertake a comprehensive set of basic tests before the SCM is issued for formal QA and testing. As a minimum this would include the following:

- **Fitness** – confirm the model Specification (inc. Design) is up-to-date and that the SCM meets the requirements, including producing the required outputs;
- **Guidance** – confirm the SCM aligns with this SCM Technical Build Guidance (e.g. Planned, Logical, Aligned, Separated, Transparent, Integrous, Checked). Use verification or logic testing software to produce maps and unique formula lists. Review maps for consistency and confirm that formula do not contain any constants;
- **Documentation** – confirm it is current and sufficient, including flow diagrams;
- **Book of Assumptions / Data Log** – confirm that the data documentation is sufficient and it is up-to-date;
- **Checks** – confirm that in-built checks are sufficient and working as intended;
- **Controls** – confirm controls are included and current (e.g. version, config);
- **Ranges** – Check Named Ranges cover correct areas, do not include #Ref! errors and are included in the model/model documentation (especially dynamic Named Ranges);
- **External Links** – confirm external links are documented and no inadvertent ones introduced;
- **Protection** – confirm application and that the SCM operates with it in place;
- **VBA** – if used confirm operation and that code is in line with good practice;
- **System** – check that the SCM operates in target system and/or Excel version;

- **Run** – run data sets through the SCM, confirm it provides the expected results and switches, sensitivities, scenarios, KPIs and graphs work as expected; and
- **Test** – undertake analytical review on the impact of input changes on outputs, undertake stress testing and review KPI's and graphs to confirm expectations.

9.7.3 Notably, developer testing is not independent. An SCM, or its outputs, should not be released for use until it has been subject to the appropriate level of QA and testing and the QA and testing has been appropriately signed-off (see SCM Development Guidance).

9.7.4 Developer testing can usefully be supported by a checklist that provides both a guide and record of tests undertaken by the model developer. The Sourcing Programme have produced an SCM Development Checklist that covers developer testing and can be adapted as required.

9.7.5 The Sourcing Programme have also produced a Good Practice Build Toolkit that can be used to structure and inform an assessment of a model's adherence to many of the good practice principles set out in this publication.

9.8 An SCM Should Undergo Formal QA and Testing Before Use

9.8.1 Before being used an SCM should be subject to an appropriate and proportionate QA and testing regimen. An SCM or its outputs should not be used or released until quality assurers have tested the SCM, issued their reports and appropriate sign-off that the SCM is fit for purpose has been obtained. Records of the QA and testing undertaken and the accompanying results should be stored appropriately (see SCM Development Guidance) and referenced within the model (see Section 3.6.3).

9.8.2 Notably, QA and testing is time restricted and may cease to be valid in the event of data changes, structural changes, scope changes or after an elapsed time period. As a guide, the maximum validity period should be considered as three years but may be much less for some SCMs. This should be set out in the QA Plan (see SCM Development Guidance).

9.8.3 The Sourcing Programme have produced an SCM QA Plan Template to support QA planning and SCM Testing Procedures (including QA Report and Test Memo Templates) to support the performance and documentation of model testing.

9.9 Documentation Should be Provided for Formal QA and Testing

9.9.1 When submitting an SCM for formal QA and testing it should be accompanied by documentation (e.g. the model Specification and Book of Assumptions / Data Log). All forms of testing test an SCM against something (e.g. against the model Specification or this SCM Technical Build Guidance). In the absence of model documentation that, for example, sets out the SCM's purpose, it may not possible to confirm that the SCM is fit for purpose.

9.9.2 Where documentation is deficient the time taken to undertake QA and testing may be increased substantially.

9.9.3 The Sourcing Programme have produced a number of templates that can be adapted as required to support the provision of model documentation. These include an SCM Scoping Template, an SCM Specification Template Example and a Book of Assumptions / Data Log Template.

9.10 Changes Should be Minimised After Starting Formal QA and Testing

9.10.1 In the event that formal QA and testing identifies issues that require the model developer to make changes to the SCM, the model developer should:

- **Document Changes** – update the SCM's Version Control Log with all of the changes made (see Section 10.3.1);
- **Minimise Changes** – keep structural changes to a minimum. In particular, deleting columns or rows should be avoided as this can hamper the checking process (see below);
- **Submit for Checking** – following any changes the SCM should be passed back to the quality assurers to check that the issues have been appropriately resolved and no new issues have been introduced as a result of changes; and
- **Maintain Independence** – The model developer should not seek or take detailed guidance from quality assurers on how to rectify issues as this could lead to effective self-review.

10. Other Governance Elements

10.1 Overview

10.1.1 In addition to in-built error checks and appropriate testing, an SCM should include a number of other governance elements, as set out below.

10.2 Book of Assumptions / Data Log

10.2.1 An SCM should contain a Book of Assumptions / Data Log (BoA / DL) that provides details of each and every source of input data employed. Data and assumptions within the model should be referenced, where appropriate, back to the BoA / DL using a dedicated column (see Section 5.3). As a minimum, the BoA / DL should list where the data has been obtained from and with sufficient detail to enable its provenance to be confirmed. Whilst some information about the data should be included in the SCM adjacent to the data (e.g. units) it is good practice to include this alongside a more detailed set of information in the Book of Assumptions / Data Log, for example:

- **Reference No.** – to link the entry to model inputs;
- **Area** – area the data relates to (e.g. wage inflation);
- **Input Label** – the data label used in the model;
- **Description** – a description of what the data is;
- **Rationale** – why data/ data source is being used;
- **Assumptions** – what's included (e.g. inc. VAT)
- **Units** – the unit of measure (e.g. £k, $m, #)
- **Base Year** – Economic Conditions (if appropriate);
- **Filename/Path** – details of source path or files;
- **Source/Provider** – source name or data provider;
- **Date Produced** – how current or up-to-date the data is;
- **Date Provided** – date when the data was provided;
- **Owner** – details of who owns the data/assumption;
- **Review Date** – date when data was last reviewed;
- **Refresh** – the date when data should be updated;
- **Validity** – details of any known data validity periods;
- **Issues** – details of any known issues with the data;
- **Manipulation** – details of any pre-processing applied;
- **Maturity Level** – assessment of data readiness level; and
- **Maturity Plan** – how the data will be matured over time.

10.2.2 Depending on, for example, the volume of data sources and the details included within the BoA / DL, it may be appropriate to produce it as a standalone document.

The risks associated with management of a separate BoA / DL (e.g. keeping it current) require careful consideration (see SCM Development Guidance for more information on file management).

10.2.3 The Sourcing Programme have produced a Book of Assumptions / Data Log Template that can be adapted as required.

10.3 Version Control

10.3.1 Every SCM should include a Version Control Log. This log, which is also a recommendation of the Aqua Book, should document changes to the model over its lifecycle, capture the QA status, and help to ensure robust version control. Key fields that should be captured include:

- **File Name** – i.e. the name of the file/model that the change relates to;
- **Version Number** – i.e. the file version that contains the new change;
- **When Change Made** – i.e. the date that the change was made;
- **Who Made Change** – e.g. name of model developer or data provider;
- **What Change Made** – e.g. data, structural, which worksheets impacted;
- **Why Change Made** – e.g. to correct an error or refresh the input data;
- **Impact of Change** – e.g. if outputs changed or new features added;
- **Authorised** – i.e. if changes have been approved by the model owner; and
- **Current QA Status** – i.e. current QA status (e.g. pending or completed).

10.3.2 The SCM Build Template, produced by the Sourcing Programme, includes a Version Control Log that that can be adapted and used within the SCM Build Template or another workbook as required.

10.4 Configuration Control

10.4.1 Configuration Control pertains to the settings used to produce a particular set of outputs and is different to Version Control. It is more relevant to SCMs that are capable of running scenarios or sensitivities and may not be relevant to all SCMs.

10.4.2 If Configuration Control is relevant to an SCM, a Configuration Control Log should be included within the SCM and it should provide sufficient information to enable the results to be reproduced and/or confirmed. Configuration Control may form part of Version Control or may in fact be separate and distinct.

10.5 SCM Background

10.5.1 In addition to setting out the purpose and limitations of the SCM (see Section 3.6), details should be provided of key personnel such as the model developer and Model Senior Responsible Owner (Model SRO).