**Session Two**

SLIDE 2 (Contents)

(Introduce the Session)

The next Slide represents the start of the formal evaluation, and to maintain a uniform approach with each participant, the tutorial which provides guidance on the comparison and compliance assessment steps of the process follows a written script. There will be time for any questions at the end of the tutorial.

As we seek your feedback on the clarity of the written process instructions, no consideration of the written process steps themselves is made in this tutorial.

SLIDE 3 (Tutorial: Comparison)

In the introduction to the framework and process we provided in Session One, we concentrated on the modelling steps of the process (Steps 1 to 4). In this session, we will consider how the models created in Steps 1 to 4 can be assessed.

There are both comparison steps (Steps 7 & 8), and compliance assessment steps (Steps 5 & 6). Whilst Steps 9 and 10 are comparison steps, they are conditional and are not discussed in this tutorial. Whilst they are **not** considered in this tutorial, they **are** detailed in the process instructions you were provided with as part of these two evaluation sessions.

SLIDE 4 (Tutorial: Internal Consistency)

We compare the different elements of software safety practice with the aim of identifying any non-conformances, differences, deviations, and impediments – which will ultimately enable an assessment of whether, and how, software safety guidance and / or software safety practice may need to change.

The first part of any comparison or conformance assessment is to determine whether the specified practice is capable of completion – i.e. whether it is internally consistent. Whether an element of software safety practice is internally consistent is determined by an assessment of whether there are enough activities detailed in the model to allow the element of practice to successfully complete.

Once an analyst is satisfied that enough activities exist, the next step is to consider whether sufficient detail in the description of the aspects of each activity exists – i.e. the inputs to, and outputs from an activity; the time by which an activity should start and / or complete; the techniques or methods used to carry out the activity; any controlling factors; and the resources expended by each activity.

SLIDE 5 (Tutorial: Internal Consistency (Cont’d)):

The assessment of internal consistency also extends to the artefacts required and produced by the activities of software safety practice. Similar to the assessment of activities, the analyst must assess where there are sufficient artefacts to enable the successful completion of all activities. Additionally, the analyst must determine whether each activity produces an artefact – and whether each activity has an adequate set of artefacts describing the aspects of activities (such as inputs and outputs).

An analyst must also assess whether the aspects of each artefact are described in sufficient detail. This includes the consideration of whether the artefacts exist or need to be created (perhaps by stakeholders outside of the software safety team).

It is important to note that any potential deficiency observed by an analyst may not necessarily be an impediment to achieving best practice for software safety practice, as it may be reasonable to leave some considerations of practice to the as-required or as-observed elements of practice. Confirmation or otherwise will only be realised through further investigations – but such investigations are not in the scope of this evaluation.

SLIDE 6 (Colour Coding – Internal Consistency)

When assessing the internal consistency of an activity, a red / amber / green colour coding is appended to the aspects of activities in the FRAM model. The colour coding follows the scheme shown on screen:

GREEN. An aspect is coloured green if it has been defined in a manner sufficient for the activity to successfully conclude. Alternatively, the colour green signifies that no consideration of this aspect is necessary for the activity to successfully conclude.

AMBER. An aspect is coloured amber if it has been partially defined, but not necessarily complete, nor sufficient for the activity to successfully conclude.

RED. An aspect is coloured red if no consideration has been given to it.

SLIDE 7 (Colour Coding – Comparison)

When comparing different models of software safety practice, a similar red / amber / green colour coding is appended to the aspects of activities in the FRAM model. The colour coding follows the colour coding scheme shown on screen:

GREEN. An aspect is coloured green if its definition is in full agreement with that of the model of practice being compared with.

AMBER. An aspect is coloured amber if its definition is in limited agreement with that of the model of practice being compared with.

RED. An aspect is coloured red if its definition has no agreement with that of the model of practice being compared with.

SLIDE 8 (Practical)

The Tutorial is now complete, and you will shortly be asked to undertake a short exercise. Do you have any questions on the use of colour-coding to compare different models of software safety activities?

SLIDE 9 (End of Session)

(Discuss next steps)