Requirements for a Guide on the Development of Virtual Instruments

Luke Emmet and Peter Froome
Adelard

Abstract. Adelard is producing a good-practice guide and training course on the development of virtual instruments as part of the DTI's Software Support for Metrology programme. This paper describes our requirements capture process and presents some of the principal issues that are emerging.

1 Introduction

The need for assistance to UK enterprises on the development of virtual instruments (VIs) was identified during the formulation of the Software Support for Metrology (SSfM) programme, and Adelard has been awarded the contract to produce a best-practice guide and training course. This paper describes the first requirements capture stage in which we identified where help is most needed and what constitutes current good practice. This has mainly been done by means of a series of interviews with personnel representing the important stakeholders and viewpoints on the use and development of VIs.

The second stage of the project will see the development of the guide and course, and the third stage involves dissemination activities. The project is scheduled for completion at the end of 1999.

The term *Virtual Instrument* does not have a standard definition, so we have adopted the following:

> *A VI is a reusable measurement instrument created by adding hardware and/or software to a generic computer, typically a PC, Mac or workstation, and which uses a computer screen to provide the visual interface to the instrument.*

A majority of VI developers use National Instruments' LabVIEW and therefore some of the issues relate specifically to that system. However, other languages such as MS Visual C++, MS Visual Basic, Quick Basic, HT Basic and HP Basic are also used, and a number of more general issues that apply to all these languages have also been identified.

The remainder of the paper is organised as follows. The next section describes the methodology that we used for the interviews, and Section 3 briefly discusses the principal issues that we identified, which need to be addressed in the guide and course. Section 4 summarises our conclusions.

2 Methodology for requirements capture

We have applied a general interview-based requirements elicitation approach, which we have developed and used before in a number of industries. Our basic approach is to identify the various stakeholders in the industry, and also to identify specific viewpoints. In the case of VIs the stakeholders include:

- national standards laboratories
- industrial developers of VIs
- industrial users of VIs

- organisations relying on measurement systems incorporating VIs

Within such stakeholder organisations there can be a number of different roles or viewpoints, who may have differing objectives, e.g.:

- quality controller
- production manager
- company lawyer
- operator
- system developer
- software engineer
- sales manager

We carried out requirements capture by a process of direct structured interviews aided by a checklist. There tends to be a law of diminishing returns in repeating interviews covering the same stakeholder and viewpoint, and we used a coverage matrix to assess the degree of coverage achieved of both stakeholders and viewpoints.

Prior to the interviews we developed a brief for the interviewees that was sent ahead of the interview, and also developed a checklist that was used by us during the interview.

The survey was structured around the key attributes of VIs, which include:

- accuracy
- reliability
- usability
- integrity (i.e. it indicates when it *cannot* deliver a measurement)
- traceability (to have confidence in the measured value)
- throughput and capacity

3 Results of requirements capture

The requirements capture process showed that there is a clear need for the guide and course. VI programming is often carried out by a single instrument engineer rather than a team of professional programmers, and these engineers need guidance on an appropriate lifecycle and techniques to provide assurance for the sometimes critical measurements the VI performs. This need is greatest for the users of commercial off-the-shelf (COTS) packages, since the manufacturers' documentation tends to concentrate on the features of the package rather than on the process, standards and assurance. There is also scope for guidance on corporate support for approved VI packages.

The remainder of this section discusses the most important issues that were identified in the interviews.

3.1 VI development process

Although many of the interviewees used ISO 9001 quality management in other software areas, there is no standard process for VI development in general and for LabVIEW in particular. Specific areas where guidance was requested include:

- *A VI development lifecycle*. This should take account of the rapid application development (RAD) style development process that is often adopted, and should also address maintenance of the VI.

- *Suitable methods and notations for specifying VIs*. Currently requirements documents are usually in English text although sometimes object-oriented methods are used, but there seems to have been little work on the most appropriate means of capturing and recording VI requirements.

- *Design methods and notations*. A separate design step is quite rare in VI developments but there can be problems from poor design of VIs leading to "spaghetti" code, particularly with LabVIEW's dataflow programming language. There are often many sub-VIs leading to a deep hierarchy.

- *Coding standards for LabVIEW*. These should address layout and documentation of the code, and known pitfalls such as the interaction between LabVIEW modules and the execution sequence.

- *V&V*. Methods for verification and validation of VIs, commensurate with the required measurement accuracy.

- *Configuration management*. Formal configuration management is rarely applied. Users need to be made aware of the features in LabVIEW to assist with keeping revision histories and source code control, and also be given guidance on overcoming practical problems, such as backing up very large VIs.

3.2 Reuse

Reuse is a key motivation for adopting VIs, and there are several areas where guidance is appropriate. Reusable software needs to be adequately specified and documented so that users have all the necessary information to use it correctly, including limitations, assumptions and accuracy. VIs have to be specifically designed for reuse, so that for example they are more tolerant to out of range inputs than would be necessary if a constrained environment could be assumed. Also, reusable VIs have to work reliably on differing hardware platforms possibly running different variants of operating systems, etc.

3.3 Design of HCI

Guidance was requested on the design of consistent user interfaces embodying good human factors principles. HCI design features include the need to prompt for a confidence check of the result, and the need to give clear, positive instructions to operators. HCI design should be appropriate for the software implementation and not just copy hardware controls.

3.4 Assurance of VIs

Several interviewees raised the problems of assuring VIs, and particularly those built using COTS products, to an adequate level for their measurement requirements. One solution that should be addressed in the guidance is the use of some level of diversity, ranging from a "sanity check" by the

operator to a complete recalculation using diverse code and libraries. One interviewee identified the need for trusted mathematical modules for VIs, and/or special test data-sets.

3.5 Corporate infrastructure for VIs

The adoption of a particular COTS product for VIs within an organisation as an "approved package" is usually a corporate decision. However, to gain the greatest benefit from this policy, a support infrastructure needs to be set up within the organisation, which should include:

- *Training materials for users*. This is important since a good deal of VI development is undertaken by instrument engineers not professional programmers, who may encounter difficulties due to the complexity of COTS packages.

- *A central support facility*. This is to enable the sharing of users' experiences (perhaps through a user group), to provide a first point of contact for technical assistance, to publicise known software faults, and inform the COTS product developer of faults found by the corporate users. Often pockets of expertise may evolve separately within an organisation; encouragement for sharing experiences is a valuable means to improving the overall knowledge base.

- *Corporate-wide assurance of the COTS package*. Many users assume that because the package is "approved", its results can be relied upon. The infrastructure should provide acceptance testing of new releases of the package and assurance of arithmetic precision etc.

3.6 Selection of components and tools

Different development tools exist for creating VIs, each with advantages and disadvantages (e.g. in the areas of timing, speed of development and determinacy of measurement). The guidance should include criteria for choosing the best tools and components for the type of instrument being developed.

4 Conclusions

We have carried out requirements capture for the development of a best-practice guide and training course on the development of VIs, mainly by means of a series of structured interviews, which has confirmed the clear need for such a guide and course. The interviewees identified a number of issues on which guidance would be of value, including the definition of a standard process for VI development, reuse of VIs, HCI design for VIs, assurance of VIs, the provision of a corporate infrastructure for the use of approved VI packages, and the selection of components and tools.