



# FESHM 5201: DEVELOPMENT, MAINTENANCE, PROCUREMENT, AND USAGE OF SOFTWARE PRODUCTS RELATED TO ENVIRONMENT, SAFETY, AND HEALTH

## Revision History

<b>Author</b>	<b>Description of Change</b>	<b>Revision No. &amp; Date</b>
J. Donald Cossairt	Title changed to present, content changed to better reflect present technology and usage.	October 2011
J. Donald Cossairt	Addressed editorial changes reflecting Laboratory organization and function.	November 2010
J. Donald Cossairt	Initial release FESHM 5201 Original Title: "Usage of Computers in Calculations Affecting Environment, Safety, and Health"	February 1998



## TABLE OF CONTENTS

1.0	INTRODUCTION.....	2
2.0	RESPONSIBILITIES .....	2
2.1	Division/Section/Center Heads .....	2
2.2	Users of software products .....	2
2.3	Personnel responsible for the use of software products that could affect environment, safety and health.....	2
3.0	PROGRAM DESCRIPTION .....	3
4.0	TECHNIQUES FOR PROVIDING ASSURANCE .....	3
4.1	Validation .....	3
4.2	Testing.....	4
4.3	Verification.....	4
4.4	Independent Analysis .....	4
5.0	BENCHMARKING .....	4
6.0	SPECIAL CIRCUMSTANCES .....	5
6.1	Software Products Porting to Different Platforms .....	5
6.2	User Compiled Codes.....	5
6.3	Upgraded and Improved Software Products .....	5
6.4	Standardization.....	5
6.5	Error Reports for Commercial Computer Source Codes.....	6



## 1.0 INTRODUCTION

Software products are widely used at Fermilab to aid engineering design. They are also used as a part of the Laboratory's program for evaluating operational conditions related to environment, safety, and health. These items range from large software products, supported by their vendors, to non-proprietary specialized software maintained solely by users. These software products are nearly always the quickest, and potentially most accurate, means to solve the complex stress, thermal, electrical, magnetic and radiation shielding problems encountered in the design of accelerators and detectors. They are commonly used to evaluate environment, safety, and health conditions to assure successful designs that provide for worker safety and environmental protection. The intent of this chapter is to include both software developed at Fermilab and at similar institutions and commercial products.

It is Fermilab policy to avoid reliance on a computer as an essential element of any system that is necessary to protect people from serious harm, to protect the environment from significant impact, or to protect property the loss of which would have a serious impact on our mission. The use of computers for monitoring, data logging, and reporting is encouraged, however computers used for these purposes must not be essential for protection. Contact the Fermilab Computer Security Executive for any variance.

## 2.0 RESPONSIBILITIES

### 2.1 Division/Section/Center Heads

Division/Section/Center Heads have responsibility for maintaining an awareness of the use of software products that affect environment, safety and health in their organizations. They are responsible to take steps to assure that those who make such use of software products within their organization implement the guidance provided by this chapter by performing such calculations in a manner that provides reliability and verifiability using techniques exemplified by those specified here.

### 2.2 Users of software products

Users of software products have the responsibility to identify those circumstances in which the use of a particular product has the potential to have a significant impact on environment, safety, and health. For those applications where such impacts are identified, a variety of techniques may be used to assure that the objectives of this chapter are met.

### 2.3 Personnel responsible for the use of software products that could affect environment, safety and health

Personnel responsible for the use of software products that could affect environment, safety, and health should have a level of technical competency, based upon education, experience, or special training, commensurate with the calculations which need to be performed. This can be established in the course of other routine assessments of employee performance and development.



### 3.0 PROGRAM DESCRIPTION

With multi-user platforms, networks, and software products, the opportunity for corruption of the source code is always present. Each critical usage of these software products should therefore be scrutinized to assure the viability and reliability of the results.

This chapter does not distinguish between centralized, distributed, or personal computing. All software products used under the supervision of Laboratory employees that could significantly affect environment, safety, and health—fall under the requirements of this chapter, regardless of platform. Where providers of subcontracted services employ software products as a part of their work, the primary responsibility to assure of the reliability and verifiability of the results rests with the subcontractor.

This chapter provides examples of measures that should be taken to insure that calculations resulting from the use of software products that could have an impact on environment, safety, and health are carried out in a manner that assures their accuracy. Examples of methods by which this assurance can be achieved are presented later in the document. Professional knowledge and experience that remains necessary to assure the overall reasonableness of the work.

Certifications of software products by governmental bodies, standards organizations, and/or professional societies may be considered in the initial choice of computer application to be employed. However, the person performing calculations retains the responsibility to establish the validity of the results of usage of the software product using one or more of the other techniques described here.

This chapter is intended to be consistent with the Fermilab Policy on Computing found at: <http://security.fnal.gov/policies/cpolicy.html>.

### 4.0 TECHNIQUES FOR PROVIDING ASSURANCE

Each computational problem presents its own considerations with respect to verification. A variety of techniques exemplified by those described here may be adapted in evaluating the reliability and verifiability of calculations. Professional judgment should be used to select one or more of these techniques in a specific application. The users of software products should seek out other approaches or adaptations to these suggestions, which might be used to provide more effective or efficient assurance of accuracy for their particular application.

Techniques chosen, whether from the following list or created by the analyst consistent with the intent of this chapter for specific analysis, should be included in any documentation of the work performed. Each calculation or set of calculations of the type addressed by this chapter should be evaluated by the user of the software products to verify that corruption has not occurred and that the results can be used with confidence. When practicable, these evaluations should be documented.

#### 4.1 Validation



Validation is the process of evaluating a system or component during or at the end of the development process to determine if it satisfies specified requirements. In some cases, it is possible to compare calculations made using established analytical formulae or textbook examples to specific results from software products. The computational model can be modified to match the analytical model as needed. Then the correspondence of the results of the computational model made using the software product and those obtained using the analytical results can be taken as verification that, for a very similar, though perhaps not identical circumstance, the software product performs accurately. Professional judgment and experience must be brought to bear on the assumptions necessary to modify the computational model to represent the typically simpler analytical problem, such that the relationship between the analytical model and the final computational model is well-understood, and the accuracy of the computer model verified as acceptable. When practicable, a second experienced person should review such results.

## 4.2 Testing

Testing can be used if the person performing the calculations has access to and is able to compile the source code. The components of a model can be isolated and tested as to whether they are performing their correct function. Input from other code users may be invaluable in tracking down errors or deficiencies in the code by the person who has access to the source code. Some software products also employ self-testing features which should be used where available and appropriate.

## 4.3 Verification

Verification is the process of evaluating a system or component to determine if the products of a given development phase satisfy the conditions imposed at the start of that phase. Engineering and scientific literature, including international, national, or trade standards documents, often contain results which can, with careful consideration, be used to verify the particular computational model used in the software product. This work may be in the form of measurement, analytical results, or computational modeling of a design similar to, or having aspects salient to, the engineering design in question. In each case, the person performing the verification must understand exactly how the measurements or other calculations are done, and be convinced of the validity of the data before using it for verification.

## 4.4 Independent Analysis

Comparison with an independent analysis can be considered to be part of the validation process. The results of an independent analysis may be used to verify those of the person performing the calculations. This independent analysis should be performed by a competent individual who is experienced in the type of calculations being reviewed. This verification may be a set of independent analytical calculations, the creation of an entirely new computational model, detailed review of the principle analyst's model, or any other approach agreed to by both principal analyst and the reviewer to provide verification.

## 5.0 BENCHMARKING



For codes used which are under development, benchmarking should be deployed. Benchmarking is a suite of the above four tests to verify the accuracy and speed of the program. Commercial programs maintained by their vendors routinely perform such benchmarks, and make them available to the analyst as part of the documentation package. User-maintained software products are typically benchmarked by the person performing the calculations, who has the responsibility to design this benchmark suite such that those aspects of the calculations most relevant to environmental health and safety are exercised and verified. Benchmarking should always be done using the same version of the software product and the same platform as will be used in the actual analysis. Certain designs will severely test the limits of a particular program, and benchmarking should probe these limits to enhance understanding of program behavior. The aspects of “benchmarking” related to computation speed are not directly related to the objectives of this chapter.

## 6.0 SPECIAL CIRCUMSTANCES

### 6.1 Software Products Porting to Different Platforms

Since different computers can have different architecture and methods of doing internal arithmetic, it is important to compare and understand the differences between the same calculation performed on different platforms. It is suggested that this can be accomplished by *benchmarking* the results using a new platform against those obtained using an established platform in order to understand the significance of any differences which may be encountered and, if possible, their origin. This kind of benchmarking will likely be needed only once, when the software product is translated to a new platform.

### 6.2 User Compiled Codes

Compilers may vary from one computer to another; even between platforms with the same architecture. Object libraries referenced by the code may also vary. Like the case for different platforms, user compiled codes need to be tested when first used on a particular combination of platform, architecture, compilers and object libraries.

### 6.3 Upgraded and Improved Software Products

The user of a code should follow such changes and updates, and if necessary, the codes should be updated to the current version. Care should be taken, especially when “centralized” software products are used, to assure that the input data specifications, which are typically retained for indefinite periods of time remain compatible with the updated code. In some environments, multiple users access computer source codes in association with their own applications. In these circumstances, it is especially important to use one or more of the techniques of this chapter to verify the validity of the results in order to assure compatibility among the various applications used.

### 6.4 Standardization

In situations where the software product is available to many users, it is common that several customized variations are produced by the different users. This can make comparative checks and



reviews exceedingly difficult. Such modifications of the software product should be clearly identified as being such and coordinated with the provider of the software product when possible.

## 6.5 Error Reports for Commercial Computer Source Codes

Quite often, vendor-supported commercial programs maintain a record of the errors known to exist in their code, and release error reports to their customers. The individual performing calculations subject to this chapter should review these error reports to ensure that his/her results is not within the regime known to have produced incorrect results for some users. If critical errors of this type are discovered, measures need to be taken to assure a correct understanding of the results and their possible impact on environment, safety, and health. Lab management shall be notified as appropriate if significant environment, safety, or health consequences are identified.