PROJECT WORK PLAN ADDENDUM

2004

Organization: CERN – LLG Project

SEAL
Common Core Libraries and Services for LHC Applications

Document Revision #: 1.1

Date of Issue: 02.03.2004

Editors: Pere Mato, Lorenzo Moneta

Approval Signatures

Approved by: Project Leader

Approved by: LCG Project Leader

Prepared by: Project Manager

Prepared by: LCG Project Manager

Reviewed by: Quality Assurance Manager

Document Change Control

This section provides control for the development and distribution of revisions to the Project Work Plan up to the point of approval.

Revision Number	Date of Issue	Author(s)	Brief Description of Change
1.0	17.02.2004	P. Mato, L. Moneta	Initial Draft
1.1	2.03.2003	P. Mato	

Table of Contents

1. Introduction

This document is an addendum to the SEAL Project Work Plan document [1] to describe with some detail the proposed work items for the period from January-2004 to December 2004. The main driving force for establishing this plan is the mandate to fulfil the needs of the LHC experiments in the domain of core software foundation and basic services. In addition, an important input has been the report of the LCG-AA Internal Review [2] and the wish from the line management to define a more coherent collaboration model with the ROOT project. Two areas have been identified as initial areas to start an attempt of doing common developments: Dictionary and Mathematical Libraries.

Many small items described in this plan are simply the evolution of the development already started in the past year and to achieve a degree of completion and quality that make it useable to experiments.

The current work packages of the SEAL project are listed in the following table.

Table 1 SEAL Work Packages

Foundation	Foundation and Utility Libraries and Plug-in Manager
MathLibs	Math Libraries Support and Coordination
Dictionary	LCG Object Dictionary
Framework	Component Model and Basic Framework services
Scripting	Scripting Services
Grid	Grid Services (not yet active)
Documentation	Education and Documentation

The document is organized in three sections. The first section we describe the short term work items, which are basically the goals we have in mind for the next major release of SEAL (1.4.0). The second section is the detailed plan for the Math Libraries work package that was not included in the original SEAL work plan [1]. The third section we list the longer term work items for the rest of the work packages that constitute the SEAL project.

2. Goals for SEAL 1.4.0

2.1 Foundation Work Items

- Remove unnecessary dependencies in external packages. Study the feasibility of eliminating dependencies to external packages like pcre, uuid, re, by either using an alternative package or copying the code in SEAL.
- Improvements in the test suit. Merging the small individual tests into bigger executables.
- Storage adaptors. More use cases are needed from CMS.
- Simplification of the PluginManger to reduce the number of classes the user has to provide.

The estimated resources required are 3-4 FTE weeks.

2.2 Framework Work Items

- Help POOL to adopt component model. Incorporate all the changes needed for POOL in the component model. This work is needed to simplify the POOL code related to plugin management by basically halving the number of required classes and facilitating the component configuration.
- .INI style configuration service. Finalizing the development of a new implementation of the configuration service using this syntax.

The estimated resources required are 4 FTE weeks.

2.3 Dictionary Work Items

- Dictionary service. Finish the development of a service to interrogate about any class that is not found already loaded. This work is needed to simplify and automate the loading of LCG dictionaries by dictionary clients (e.g. POOL)

The estimated resources required are 2 FTE weeks.

2.4 Scripting Work Items

- Introduce the Python courses and tutorials as part of the SEAL release.
- New implementation of PyLCGDict. Finish the ongoing development of the new version and replace the old one.
- Make PyROOT not dependent of Boost.Python. Study the possibility to be added as part of the ROOT distribution.

The estimated resources required are 6 FTE weeks.

2.5 MathLibs Work Items

- Minuit major cleanup. Take the opportunity of a major release to perform a major clean-up and performance optimization.
- Development of a test suit for GSL/ROOT/NagC.

The estimated resources required are 4 FTE weeks

2.6 Documentation Work Items

- Workbook. First implementation of the SEAL workbook.
- Development of topical User's Guide (e.g. PyLCGDict, SealBase, …)

The estimated resources required are 4 FTE weeks

3. MathLib Program of Work

This work plan describes the organization of the SEAL MathLib sub-project for the next year in term of work items, expect deliverables and personnel resources. The recommendations of the MathLib [3] and Blueprint RTAG [4] are the starting points for the work plan for the Math activities in the SEAL project. Support and development of mathematical and statistical libraries used in physics analysis, reconstruction and simulation should be coordinated in a common LCG project within SEAL.

The purpose is to provide a coherent system to the end-users. In addition, it is recommended to coordinate the activities with ROOT, bringing the needs of LCG (experiments, Geant4, etc.) and ROOT together, trying to avoid maintenance and support of various mathematical libraries providing similar functionality. The goal is to share a common mathematical library between ROOT and the rest of LCG activities and experiments. ROOT will be able to build layered functionality based on this common library if a number of requirements are fulfilled.

The idea is to re-use as much as possible of the existing open source mathematical libraries, such as GSL. New developments should be made only when the requested functionality is not found in any of those libraries. Collaboration with the LCG community (experiments and projects) and outside (other HEP labs and institutes) is envisaged and anybody who wishes to contribute should feel free to do so.

3.1 Proposed Work Items

The main goal of the project is to provide a coherent set of mathematical libraries, fulfilling the functionality needed by the HEP experiments for analysis, reconstruction and simulation. Two types of mathematical libraries exist:

- Generic libraries covering mathematical functions, linear algebra, complex numbers, etc. GSL is for example one of those libraries.
- HEP libraries covering functionality especially needed by our community, such as Lorentz Vector, fitting and minimization libraries (Minuit), etc.

Mathematical libraries are used directly in C++ program, for example in the experiment reconstruction, or during the analysis phases from an interactive environment, using either Python or ROOT/CINT. A major requirement is to use the same basic mathematical library in all these cases, and not having different ones, which could contain different algorithms.

In the following, we describe the work items that we have identified, with their respective deliverables and estimate resources needed.

3.1.1 MathLib Web Site

We should first provide an inventory of most common Mathematical functions and algorithms used by the HEP community. This list can be deduced from use cases of analysis, reconstruction and simulation in LHC experiments. High quality user documentation should be provided as Web pages, describing the palette of functions and algorithms used by HEP. It should be documented how to use each function (API), what precisely they do, providing in addition a list of FAQ and references. When multiple implementations exist, recommendations should be given on which one to use, taking into account the quality of the algorithm, of the code and license issues. A cross reference table with links to and from CERNLIB list of routines should be provided, to facilitate users, which are used to consult the CERNLIB write up and have used it extensively.

The deliverables of this work item are:

- Produce inventory table with list of commonly used function and algorithms
- Document each item in the table and specify origin (GSL, Root, etc…) with links to detailed documentation (e.g. GSL reference manual)
- Produce cross reference table with CERNLIB routines whenever is possible
- Produce list of FAQ
- Study and document difference between multiple implementations and recommend best one

The estimated resources are 0.5 FTE in the following 12 months.

3.1.2 Evaluation of GSL

It is expected that the majority of needed mathematical functions is provided by GSL. In order to be re-assured of the quality of the library, we need to study its functionality, numerical stability, performance, accuracy, etc , and comparing it with other existing libraries of known quality such as Nag. The result of these studies will be a validation and a test suite, complementing the one supplied with the GSL distribution, for the parts needed by HEP. This new test suite can then be run to check every new GSL release.

From the inventory table produced in the previous work item and from the evaluation studies we should identify the missing functionality in GSL as requested by the experiments. If other existing HEP libraries such as ROOT, CERNLIB or CLHEP can provide this missing functionality, we should propose their usage, either directly or re-writing the algorithm in C++ and adapting them to the MathLib conventions. Otherwise we should develop the missing algorithms, inside the MathLib project, but trying to start always from reviewing what could be present in existing scientific mathematical libraries. It is essential to be in contact with the GSL authors, and collaborate trying to add any new functionality developed by the project into GSL. However, we expect that the number of functions missing in GSL and not found in other HEP libraries is very limited.

Deliverables:

- Produce validation and test suite of GSL for HEP specific use cases

- Produce results on the comparison of these GSL algorithms with Nag, ROOT and CERNLIB.
- Produce list of missing or not adequate functions and algorithms in GSL
- Study a solution to implement the missing functionality

Estimated resources are 0.5 FTE in the next 12 months excluding eventual new developments

3.1.3 Linear Algebra Evaluation

Evaluate and review existing linear algebra packages (from CLHEP, ZOOM, ROOT, GSL, etc...), performing comparison and validation studies in an HEP application environment, for example in a reconstruction program. If we arrive at the conclusion that the existing packages are not satisfying the HEP requirements, we will develop a new C++ package, trying to re-use as much as possible of the existing packages in C (GSL) or Fortran (BLAS, LAPACK). This new developments should proceed in collaboration with existing projects in HEP (ROOT, CLHEP, ZOOM). We will provide, in addition, interfaces to Python/CINT and persistency.

Deliverables:

- Collect results of existing evaluation, verify them and complete if necessary and make them available
- Study solution and establish plan for new developments of a C++ linear algebra library.

Estimated resources are 0.5 FTE in the next 6 months

3.1.4 C++ MathLib

The first goal is to provide a thin wrapper layer in C++ for the set of chosen functions and algorithms. The functions should be those identified in the first work item and the recommended best implementation should be the used. This layer should be the recommended way of using the mathematical functions in C++ applications. We need to collaborate with ROOT, which has already a similar layer, TMath, to provide together a single library with extended functionality. We plan to evaluate also the existing GSL C++ wrappers and in particular to collaborate with ZOOM from Fermilab, which have already developed a C++ wrapper for the GSL special functions.

The second goal is to develop a higher layer of C++ classes describing mathematical properties of functions and related algorithms, such as derivatives, integrals, function operation and compositions, etc… These classes can serve users directly, for example in a C++ or an interactivity environment (Python or CINT), in a fitting framework such as the proposed one (see Fitting and Minimization) or higher level analysis frameworks. We should not here start from scratch but from what already exists, for example in the CLHEP Generic Functions packages or in the TF1 class of ROOT. Furthermore, we have to produce the dictionary libraries for both the basic classes and higher level function classes in order to have them used in the PYTHON and CINT environments and for persistency.

The structure of the proposed MathLib C++ and its relations with existing libraries and tools are shown in Figure 1. Detailed technical plans for this new C++ Mathlib including the Linear Algebra part will be described in a separate document.

Deliverables:

- Produce C++ wrapper layer for mathematical functions and algorithms identified in the MathLib inventory list
- Design and prototype C++ classes describing mathematical functions and their usage
- Produce dictionary library using the LCG dictionary and CINT for both layers

Estimated resources are 0.5 FTE in the next 12 months.

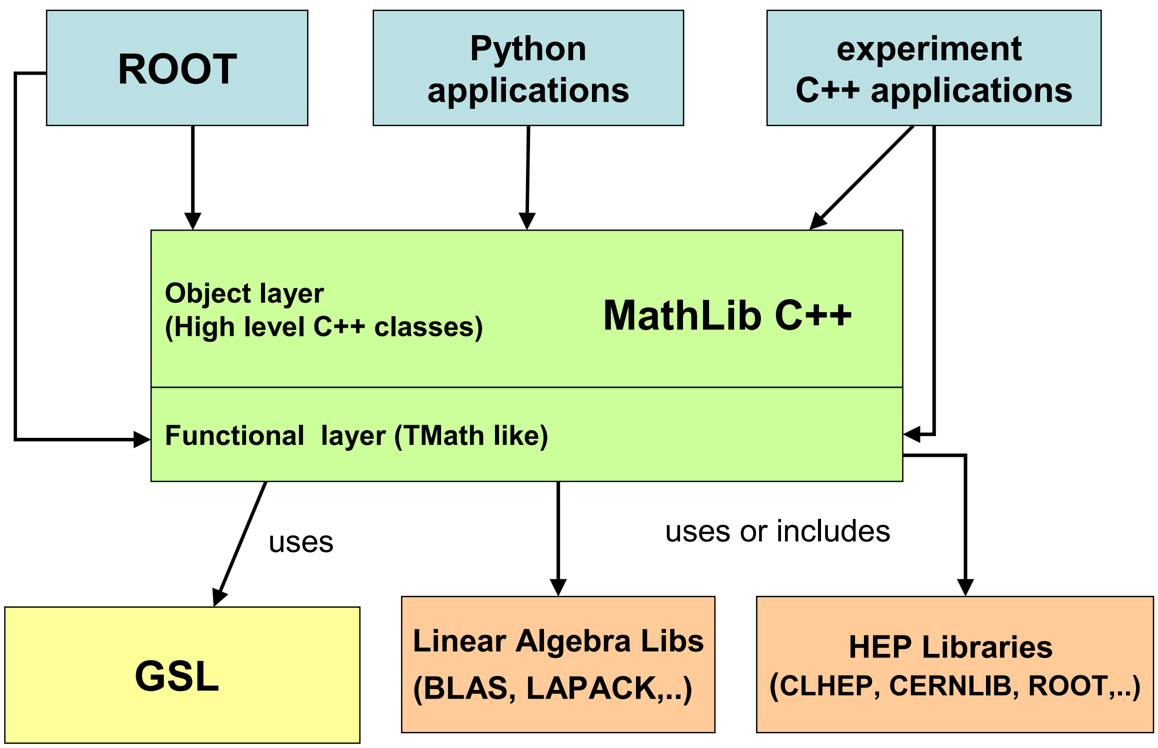

Figure 1 Structure and relations between the proposed two layers MathLib C++ and existing libraries and applications.

3.1.5 CLHEP

We expect to continue to provide local support for CLHEP, in terms of installation, porting to the needed platforms, testing and helping in the maintenance, since it is in use by the experiments and Geant4. We need to increase the collaboration with the CLHEP editors and to participate more actively in the discussions, proposing new developments according to the needs of the LHC experiments. Together with them, we should define a new development plan. If needed and requested, new developments, including changes at the level of the API for CLHEP classes should be carried out, but trying to keep, at least for some time, a backward compatibility layer for not breaking user code.

We should work on the validation and tests of CLHEP, improving the existing ones and combining them in a common test suite to be run automatically for every new release. In addition interfaces to CINT/Python and persistency should be provided.

Deliverables:

- Continue to provide support for CLHEP
- Collect requirements from experiments and discuss them with CLHEP editors. Establish a new developments plan
- Enhance and update existing tests and produce a validation and test suite
- Participate in new CLHEP developments

Estimated resources are 0.3 FTE in the next 12 months for the first three items and 0.3 FTE in the next 12 months for new CLHEP developments, varying according to needs.

3.1.6 Fitting and Minimization

The plan is to complete the current development of the Minuit C++ library. We have already produced a production version which is used by CMS in their reconstruction programs and by analysis tools such as HippoDraw.

To complete the work we need to add the following new functionality, which was present in the Fortran version: minimization with Simplex, possibility to perform contours and scans, possibility to define minimization strategy and to control the verbosity level. Some work will be devoted to improve the testing and to add more examples. Documentation for Minuit needs to be completed producing a user manual, reference guides and updating and improving the tutorials.

In addition to Minuit, we plan to develop a fitting and minimization framework to facilitate the usage of Minuit and to integrate it with additional minimizer engines which can be provided in external libraries (e.g. GSL). We expect to develop this system, following the architectural vision described in the Blueprint RTAG to facilitate the integration with existing analysis frameworks such as ROOT, HippoDraw or JAS. Some tools (e.g. ROOT), already having a fitting framework, should have the possibility to interface at a lower level, for example only at the minimization level, while others can profit from the whole system. We plan to re-use as much as possible of existing software already developed, adapting it to our needs.

Deliverables:

- Complete C++ Minuit functionality (Simplex, Scan, Contours, etc…)
- Produce complete Minuit documentation
- Prototype of fitting and minimization framework using existing code
- Integrate prototype in existing analysis tools and experiment frameworks and give to users. Collect feedback and re-iterate on design/implementation

Estimated resources are 0.7 FTE for Minuit in the next 12 months and 0.3 FTE for higher level fitting and minimization library

4. Program of Work for the other Work Packages

4.1 External software guidelines

We need to produce a document describing the process for evaluating, selection criteria and integrating external software. This was already an established deliverable in the original SEAL work plan but it was always delayed for one reason or another. This time is essential to have written guidelines on which to base other decisions.

- Produce a draft document and circulate it among the projects and experiments to get feedback and eventually an agreement.

The estimated resources needed are 3-4 FTE weeks.

4.2 Foundation Work Items

We do not foreseen major new developments in this work package in the next months. The work that is needed is mainly to educate users by developing and setting up tutorials, user-guides and coaching developers. The work items are:

- Development of Tutorials.
- Development of a set of web pages describing and categorizing all the available classes.

The estimated resources needed are 0.2 FTE for the next 12 months.

4.3 Framework Work Items

As it was stated in the internal review, this is the work package one must engage the experiments if it wants to avoid irrelevance. The objective is then, for the next coming months, to integrate SEAL component model into the existing Gaudi/Athena framework and evaluate its costs and benefits. Before new functionality is being added, we need to get the feedback from the experiments. Therefore, the development of other services (i.e. Whiteboard) is put on hold until there is a firm commitment from at least 2 experiments. The work items are:

- Provide the necessary help from the project to integrate the existing SEAL component model into Gaudi/Athena.
- Provide the necessary help for POOL to adopt the component model.

The estimated resources needed are 0.2 FTE for the next 6 months.

4.4 Dictionary Work Items

The dictionary activities will be done in collaboration with the ROOT team. The final aim is to converge with a single dictionary between LCG and ROOT. This common dictionary will be the base for enabling the interoperability of many other layered software components between ROOT, LCG and non-LCG. A new set of packages (Reflection, Generation) will be developed and the idea is that it will shared between LCG projects and ROOT in a similar way to what is aimed for the MathLib. The deliverables are:

- New Reflection API. Development done in collaboration with ROOT team. Iterate the current proposal until an agreement is reached.
- Provide reference implementation of the agreed API as baseline.
- Adapt code generation tools (gcc_xml based) to produce new dictionary descriptions.
- Develop an implementation using available CINT data structures with the goal to eliminate the need of having the LCG<->CINT gateway (POOL).
- Adapt existing ROOT reflection classes to the new common dictionary.

The estimated resources needed are 1.0 FTE for the next year.

4.5 Scripting Work Items

The main goal is to seek feedback from the experiment physics community in the usability of python in interactive analysis. This work should be coordinated with the other projects (POOL, PI, ARDA,…) The idea would be to identify physicists that would like to act a guinea-pigs.

- Continue the python tutorials and provide help to the user and developer community.
- Provide access to key services in such a way that Physicists can asses Python in a useful context, rather than as an abstract language choice in isolation from their real problems.

The estimated resources are 0.5 FTE for the next year.

4.6 Documentation Work Items

We plan after SEAL 1.4.0 has been released to start a serious campaign in documentation for all aspects of SEAL. This will be by providing user guides to complement the existing reference documentation and by enhancing the existing workbook.

The estimated resources needed are many small fractions of each member of the SEAL development team.

5. Summary

5.1 Resources

The following table shows a summary of the estimated required resources for the described work items in this document. The available manpower associated to the SEAL project is of the order of 6.5 FTE, therefore there is a reasonable match between the required and available manpower.

Table 2 Summary of the required resources for each major work item. The estimated FTE is calculated assuming an effective fulltime year of 40 weeks

Work package	Work Item	FTEweek	FTE
Foundation			**0.4**
	Modifications for 1.4.0	4	0.1
	External Software Guidelines	4	0.1
	Development of Tutorials		0.1
	Development Web Pages		0.1
MathLibs			**3.1**
	Modifications for 1.4.0	4	0.1
	Mathlib Web Site		0.5
	Evaluation of GSL		0.5
	C++ MathLib including Linear Alg.		1.0
	CLHEP support		0.3
	Fitting and Minimization		0.7
Framework			**0.3**
	Modifications/Additions for 1.4.0	4	0.1
	Integration into Gaudi/Athena	4	0.1
	Help POOL adopt component model	4	0.1
Dictionary			**1.0**
	Additions for 1.4.0	2	0.1
	New Reflection API	8	0.2
	Reference implementation	8	0.2
	Adapt generation tools (gccxml)	4	0.1
	Implementation using CINT structure	10	0.3
	Adapt ROOT reflection classes	8	0.2
Scripting			**0.7**
	Modifications/Additions for 1.4.0	6	0.2
	Python tutorials		0.2
	Help to Physicists		0.3
Documentation			**0.3**
	Documentation for 1.4.0	4	0.1
	New documentation		0.2
Total			5.8

5.2 Milestones

The following is the list of proposed Level-2 milestones for the SEAL project. It includes at least one milestone for each major work package.

- External software guideline document (31st May 2004)
- Workbook for SEAL including complete HowTos and examples for each SEAL component (15th June 2004)
- New Dictionary API and reference implementation (30th June 2004)
- MathLib project Web (15th July 2004)
- First version of the C++ MathLib package (30th September 2004)

5.3 Feedback and Priorities from Experiments

5.3.1 CMS

For CMS the priorities are:

- Definition of the "external software guidelines" and their implementation
- Development, integration and support of a Mathlib for HEP
- Software components required by POOL

At the moment, CMS does not have any specific urgent requirements in the other areas but expect that the SEAL team will be able respond promptly if and when they will emerge.

5.3.2 LHCb

Integration of the available SEAL component model (including the plugin manager) is one of the priorities for LHCb. This integration should take place in the next coming weeks/months. In addition, LHCb would like to get support from SEAL for testing the ideas of using Python as a scripting language for physics analysis.

5.3.3 ALICE

Alice is not interested in SEAL at all.

5.3.4 ATLAS

6. References

[1] *SEAL project plan*, P. Mato et al., CERN-LCGAPP-2003-01
http://seal.web.cern.ch/seal/documents/SEAL_Project_Plan_1_0.pdf

[2] Applications area internal review report 2003, CERN-LCGAPP-2003-16
http://lcgapp.cern.ch/project/mgmt/rev200310/aa_review_report_2003.doc

[3] Final Report of the RTAG on Mathematical Libraries, F. James et al.
http://lcg.web.cern.ch/LCG/sc2/RTAG3/finalreport.pdf

[4] Architecture Blueprint RTAG report, T. Wenaus et al., CERN-LCGAPP-2002-09
http://lcgapp.cern.ch/project/blueprint/BlueprintReport-final.doc