DRAFT Persistency (POOL) Project Workplan for 2002

This workplan describes the initial organization of the POOL project in terms of work packages, their responsibilities, and personnel. It further describes a release schedule for the project that is tailored to delivering quickly the highest priority functionality required by the experiments, within the limits of the available resources. This plan was developed largely at the first LCG Persistency Workshop June 5-June 6 and a subsequent Architect's Forum meeting at CERN. This plan is based on the assumption that the first external release of the project is required to deliver all of the following key features:

- Transparent navigation to single objects (O(100)-O(1000) per event) integrated with a grid aware file catalog (EDG/Globus RLS)
- Persistency for arbitrary transient C++ objects (no inheritance from TObject or user code instrumentation required)
- Object collections and iterators including associated meta data (eg event level, file level and collection level metadata)
- Dictionary read access (introspection) and write access (runtime class creation) via a new interface which is persistency technology and language neutral.

We would like to point out that this is a list of requested features - not necessarily the set of features which can realistically be implemented by November with the resources available to the project. Since the feature set has only been agreed on recently, neither the estimation of the required effort (based on working proof of concept prototypes) nor the estimation of the available effort (based real people contributing to the project) is fully completed yet. We therefore assume that all resources assigned to the project will be available immediately and that any major new functionality in Root I/O (eg foreign class support) and EDG components (eg replica location service) are available to the project in time. The release plan proposed here is our best current estimate of what is attainable if these assumptions hold true.

A more comprehensive workplan encompassing the expected three year duration of the project will be submitted later this year.

Proposed Work Package Split

We suggest creating the following work packages, which are closely aligned to the software components, identifiedby the RTAG. The assumption is that each work package will have significant effort (>0.25FTE) of at least 3 people with experience in the particular problem domain assigned to insure that a balanced discussion can be achieved on the work package level. If the resources available to the project should not suffice, some work packages may need to be combined.

In the following we list the main task of each work package and the active participants at this point.

Refs and Object Lookup

- Definition of IStorage Manager and ISmToken interfaces

- Implementation of these interfaces based on RootI/O
- Definition of smart pointer (Ref) types and cache manager interfaces
- Implementation of a prototype cache manager (to be used as reference for experiment cache manager integration)
- Implementation of templated Ref classes
- Evaluation of the Root I/O checkpointing mechanism

Participants:
Markus Frank, 50%
Giacomo Govi, 50%
Fons Rademakers, 25%

File Catalog and Grid Integration

- Definition of the IFileCatalog interface
- Implementations of this interface based on native MySQL and EDG/Globus RLS (replica location service)
- FileUID generation
- Provision of end user catalog management tools
- Evaluation of RDBMS checkpointing/transaction mechanism

Participants:
Zhen Xie, 50%
Maria Girone 50%
Mathias Steinecke (to be confirmed)

Collections and Metadata

- Explicit and implicit collection implementations for RDBMS and RootI/O backends
- IAttributeList implementation for RootI/O and RDBMS backends
- Interface for query definition and iteration over query results

Participants:
David Malon 30%
Chris Lain 50%
Sasha Vaniachine, 20%
Julius Hrivnac, 20%

Reflection and Conversion

- Definition of a Reflection interface (transient dictionary) and a interface to the a description of data la yout (persistent dictionary)
- Implementation of a gateway between the Root I/O dictionary and the Pool transient and persistent dictionaries
- Implementation of a conversion service (to be used from the storage manager) which converts arbitrary objects between their transient and persistent representations

Participants:

Craig Tull, (to be defined)
Stefan Roiser, 50%
Victor Perevoztchikov, 30%

Common services and Integration

- Project specific infrastructure (project web, cvs repository maintenance, etc.)
- Common RDBMS C++ interface
- Definition and implementation of the PersistencyManager interface (façade to user visible services provided by internal components like storage manager, reflection and conversion)

Participants
Giacomo Govi, 50%
Zhen Xie, 50%
Maria Girone 50%

Proposed Release Schedule up to end of 2002

We propose to organise the software development for the persistency project as a sequence of developer releases, which starting from a few core components integrate additional components (and provide additional interfaces) with each release. This approach should not delay the development of more external components (such as collections and meta data handling) but rather keep their development decoupled from the central core until sufficient agreement about the required functionality and their user interface has been achieved.
Since the RTAG component model has a well-defined structure, without cyclic dependencies, starting from the core navigation components should minimize the need for code changes.

In the following we summarize the projected release content and release schedule for the POOL project this year.

September '02 - Release 0.0.1 – Transparent Navigation

Release Focus

As initial set of components we suggest to focus only on the three components that form the basis of the navigation system:

- StorageManager
- FileCatalog
- Refs and CacheManager

The main goal of this release is to validate the RTAG proposal on how to extend the existing Root I/O streaming layer to provide transparent object navigation.

Other WP milestones
Still outside of the release, but milestone to the respective WP:

- Proof of concept reflection interface for reading

October '02 - Release 0.0.2 – Collections

Release Focus
Provide in addition support for object collections. In particular:

- Explicit collections - eg all objects denoted by a list of references
- Implicit collections - eg all objects in a given (set of) file(s)

Both should available as RDBMS implementation and Root I/O implementation. All implementations share the same basic interface for iteration and population.

Other WP milestones

- Expect that the EDG/Globus file catalog has become stable enough to be integrated as an additional IFileCatalog implementation.
- Proof of concept for Root I/O checkpointing
- Ref extension to allow navigation into embedded collections (eg vector)
- Proof of concept dictionary interface for writing

November '02 - Release 0.0.3 – Metadata & Query

Release Focus
This release should add metadata support to file catalog, collection catalog and events in an event collection based on extensible property lists. All implementations are expected to provide the same basic population and query interface.

Other WP milestones
The complete dictionary interfaces, Root checkpointing and Ref extensions should now be integrated.

This release will allow to make arbitrary C++ objects persistent, navigate transparently between them (implicitly consulting one of the file catalog implementations) and keep queryable, annotated collections of object references to those objects in either the RDBMS or Root I/O layer.