Project Status (Aug 2002)

LCG Software Process & Infrastructure

A.Aimar IT/API
CERN

Infrastructure for Software Dev.

Software development phases and activities

User Requirements

Analysis and Design

Development

Planning

Release

Testing

Distribution

Deployment and Installation

Documentation

Validation

- Provide services specific to the phases
  - Tools, Templates, Training, Examples, etc.
- Provide general services
  - CVS repository, Web Site, Software Library
  - Mailing Lists, Bug Reports, Collaborative Facilities

Break-down the infrastructure

User Requirements

Analysis and Design

Development

Release

Deployment and Installation

Testing

Distribution

Validation

Documentation

Planning

- Break the project in "components"
  - Each is a sub-project
  - A responsible person in the LCG SPI
  - Understand and learn the subject
  - Know/find who knows about the subject
  - Provide practical solutions, usable independently

SPI development infrastructure

- Web address
  ("development web" for now)
  http://lcgapp.cern.ch/project/spi
- Mailing lists
  project-lcg-peb-spi
- Support email address
  project-lcg-peb-spi-support
- SPI infrastructure:
  - CVS server
  - Development web pages
  - Templates
  - Standard way of working on each SPI service/component

Component development

- Each component of the infrastructure has:
  - A responsible person in the project
  - Similar SIMPLE approach
    - Define the goal of the component
    - Standard procedures and documentation
- Standard procedure
  - Survey of possible/existing solutions in HEP and free software
  - Meet the people expert and responsible of the component in real projects (LCG or experiments or big projects)
  - Discuss and agree/decide/verify a solution ← **WE ARE HERE**
  - Present the solution
  - Implement the solutions
  - Use in the LCG SPI project itself
  - Use it in a real project (LCG or experiment or big project)

Services and components

- Services (A.Pfeiffer)
  - AFS delivery area
  - CVS server
  - Build platforms
- Components
  - Code documentation (L.Mancera)
  - CVS organization (I.Papadopoulos)
  - Testing (M.Gallas)
  - Software documentation (A.Aimar)
  - Coding and design guidelines (M.Sang)
- Other services and components started

Service: AFS delivery area

- The AFS delivery area is below **“/afs/cern.ch/sw/lcg”** and shall be structured such that there is :
  - an area to install software created by projects in the LCG application area comprising libraries, include files as well as sources and (internal, reference) documentation; in the following called **”app”**,
  - an area where external and third party software -- which is needed by one of the projects in the LCG application area -- is installed, in the following called **external** , and
  - an area where software is installed which is provided by people for evaluation within a project; in the following called **contrib**.
  - In each of these areas, the installed software shall be in directories specifying the package name, the package version and the “**OS_Compiler”** for which the software is installed. Links may be set up for convenience for another "ordering" of these.

AFS delivery directories

```
/afs/cern.ch/sw/lcg/app/pool/<version>/<OS_comp>/
                       /spi/<version>/<OS_comp>/
                       /... # other LCG App Area projects

                   /external # installed Software library

                  /contrib # open to all contributors
```

- "app" one directory per LCG project
- "external" we have a list of software
  installing what is needed by LCG projects
- "contrib" we create a home directory with permissions

- Example: mySQL++ in
  ```
  .../lcg/external/mysql++/1.7.6/rh61_gcc2952/
  ```

Service: CVS server

- Installed on lcgapp.cern.ch
- Hosting LCG projects
  - POOL, persistency
  - SPI, Software Process & Infrastructure
- Browsable via the web (cvsweb) http://lcgapp.cern.ch/cgi-bin/cvsweb/cvsweb.cgi
- These are “project repositories” and output will be in the AFS delivery areas described before
- For SPI information/artifacts email me or lcg-peb-spi-support

Service: Build platform

- Presently the following build servers are publically available, so everybody can login using their AFS account
- Operating system / Machine name

RedHat 6.1	lxbuild001	with oprofile
RedHat 6.1	lxbuild002	
RedHat 7.2	lxbuild003	
RedHat 7.2	lxbuild004	
RedHat 7.2	lxbuild005	with oprofile
Solaris-7	sundev	

- Window 2000 and Windows XP are needed too, and be remotely accessible (help needed)

Other Services

- Software Library
  - We have a list of packages
    http://lcgapp.cern.ch/project/library
  - Installing with the help of POOL what they currently need
- Project web
  - Have a portal for the project
  - Investigating “savannah”
- Development tools
  - For tools needed
- Our rule: Use/configure/pick existing IT services not redo them (SDT, etc.)

Component: Code documentation

- Purpose
  - Provide LCG projects with system(s) to browse, search and document the code via a standard Web browser
  - Provide a uniform mean to learn about the development of a project
  - Use what is already available and used in CERN and HEP
- Deliverables
  - Install and maintain systems for code documentation
  - Provide documentation and tutorials about the used systems (Doxygen, LXR, cvsweb,...) for new developers.
  - Propose guidelines for comments in the code for new developers or to people from other projects to understand the code and contribute to the projects.

Component: CVS Organization

- Purpose
  - CVS in every software project in HEP. For the LCG software projects it is therefore the obvious choice.
  - Emphasis will be given to the organization of configuration units involving items with source code.
  - No assumptions will be made on the top-level configuration management tool.
  - This component is expected to have loose dependencies to the components addressing release management, testing and deployment procedures.
- Deliverables
  - Description of the directory structure and the contents of a CVS repository for an LCG software project.
  - Customizable scripts to prepare the CVS structure for a new project, filled with templates for basic units.

CVS structure

```
lcg/app/ProjectName/
                   Packages/
                            interfaces/
                            src/
                            tests/
                                  test1/
                                        input/
                                        src/
                                        output/
                                  test2/ (………)
                            userDocumentation/        # userDoc
                            developerDocumentation/   # devDoc
                            Component1/
                                       interfaces/
                                       src/
                                       tests/
                                       (………)
                                       SubComponent1/
                                                     interfaces/
```

CVS structure (cont'd)

```
lcg/app/ProjectName/
                   integrationTests/
                                           test1/
                                                 input/
                                                 src/
                                                 output/
                                           test2/
                                             .
                                             .
                                             .
                         systemTests/
                         acceptanceTests/
                   userDocumentation/
                   developerDocumentation/
```

- Testing and documentation directories are defined in more detail in other components

Component: (Unit) Testing

- Purpose
  - Select simple techniques, tools and methodologies that will be used with software developed within the LCG project.
  - All level of testing should be run as part of automatic nightly and pre-release builds.
  - We focus on unit (work package) testing for now in which each unit of software is tested to verify that the main functions has been correctly implemented.
- Deliverables
  - Simple tools and templates to describe and implement unit testing

Component: Software Documentation

- Purpose
  - Provide the minimal amount of documentation to describe a project and its work packages
    - Purpose of a work package
    - Work done on the work package
    - Status of the work package
    - How to use the work package
- Deliverables
  - Template to describe the status of a project
  - Template for describing the purpose and status of a work package
    - Being defined and used by Pool

Component: Coding Guidelines

- Purpose
  - To provide a concise and readable summary of generally accepted good practice in C++ programming and OO design.
  - The aim was to keep them short and readable, so that they may be bookmarked and checked often.
  - When guidelines are too long: conscientious developers will read them once but not consult them regularly; less conscientious ones will not even read them.
- Deliverables
  - One or two documents describing the rules and guidelines for programming LCG applications in C++.
  - Definitions of and templates for code reviews, automatic rule checkers and other quality assurance issues.
  - A system to automatically check some rules and help with code reviews.

Coding Guidelines (cont'd)

- Checkability of rules is inversely proportional to their importance except in a few cases (e.g. no public data members).
  - It is very easy to write dreadful, unmaintainable, unreadable spaghetti code which scores 100% on all automated tests.
  - If we want to enforce rules, the only way to do it is with code reviews conducted by experienced developers.
  - We should discuss it with Torre and the project leaders (Dirk, etc).
- We will use automatic check but also help/organize reviews if the project leaders will want that approach
- Purely automated checking will not help people to learn and improved from experienced colleagues

Other Components

- Nightly Builds (L.Moneta)
  - Provide a standard way to describe releases and builds and the standard infrastructure to perform them
- Project Web
  - Have a default web template when a new project starts, with all features for participants and users
  - Will have a **project workbook** to follow for participants and users
- Bug and Feedback Reporting
- Configuration management tool (release mgmt)
  - The discussion on the way

Component development

- Each component of the infrastructure has:
  - A responsible person in the project
  - Similar SIMPLE approach
    - Define the goal of the component
    - Standard procedures and documentation
- Standard procedure
  - Survey of possible/existing solutions in HEP and free software
  - Meet the people expert and responsible of the component in real projects (LCG or experiments or big projects)
  - Discuss and agree/decide/verify a solution
  - Present the solution
  - Implement the solutions **← WANT TO MOVE HERE !!!**
  - Use in the LCG SPI project itself
  - Use it in a real project (LCG or experiment or big project)

Next steps

- Publish the SPI web site where all material is available
- Receive feedback on the different services and components
- Support LCG project(s) (I.e. Pool)
- Finalize and implement the proposed solutions

- Please mail me or phone me (16 3158) your comments and disagreement
- We are trying to do something useable and provide all possible help in getting it used