GAIA

Quality Assurance Activity
Quality Assurance WP

- GWP-T-x03-00000 Quality assurance and conf. management for CUx
- GWP-T-x03-10100 Quality assurance
- GWP-T-x03-10100 Write Soft. Quality Assurance Plan
- GWP-T-x03-10200 Document reviewing
- GWP-T-x03-10300 Software reviewing
- GWP-T-x03-10400 Quality Assurance Control

■ To obtain a satisfactory software quality level for application software
  - Reliable
  - Easy to maintain
  - With Performance and load capacity

■ By giving common rules and recommendations

■ Essential for a project
  ✷ Where the final system will integrate components from
    - Hundreds of developers from a large number of institutes
  ✷ Where corrective and enhancements maintenance will cover the period 2012-19
QA CNES activities

■ Leading CU1 QA WP
  ♦ Provide a generic Product Assurance Plan (PAP) for CUx
    • To be validated by CU1
    • To be agreed by DACE
  ♦ Tools for Quality Assurance (GWP-T-103-11000)
    • Control, validation and testing tools

■ Leading QA activities for CU4, CU6, CU8
  ♦ QA Support to DU
    • To specialize PAP for CUx purpose
    • For Document reviewing
    • For Software reviewing
    • For Quality Assurance Control
  ♦ Engineering Support to DU
    • QA Tools
    • Other Tools (from GWP-T-110-00000 Coordination common software resources)
      – Environment, development, ….
Through Next figures:

- **function:**
  - part of a logical model used to decompose the functionalities expected to be performed by a system. This logical model is an implementation-independent model of software items used to analyze and document software requirements.

- **Scientific algorithm:**
  - mathematical, formal or graphical description of a scientific processing. This description will be an input for the software implementation.

- **interface:**
  - depending the need, the following interface terminology must be used:
    - Data interface: data exchange between Software products
    - Service interface: description of means that are set between systems to provide exchange of data.
    - Dependency: API call or link between different software products or units.

- **system/subsystem:**
  - set of interdependent hardware and software elements constituted to achieve a given objective by performing a specified function. System is composed of subsystems.
    - For GAIA, the term System is implicitly related to the overall GAIA data processing system (overall system)
    - The subsets of the overall system that are installed and operated in the various Data Processing Center (IDT, First look, AGIS, Photometry, Object processing...) are subsystems: a GAIA subsystem will be the result of the integration of Software products and/or Software Units within an host infrastructure in the Data Processing Center
    - The term system may be used more widely with the condition to be precede by its name: i.e. the GIBIS System

- **software component:** part of a software system
  - NOTE 1 Software component is used as a general term
  - NOTE 2 Components can be assembled and decomposed to form new components. In the production activities, components are implemented as modules, tasks or programs, any of which can be configuration items.

- **software product:**
  - set of computer programs, procedures, documentation and their associated data. A software product is the result of a unit of work (DU, WP, ...) for which an organization is set (responsible, developers, ...). The software product life cycle and management is defined through a common set of rules.

- **software unit/module:**
  - program or library that is part of a software product and which could not be divided. In particular, software interfaces shall be identified and described at this level.
ideal case

« Functional » view

« Project organisation » view

« System CUx » view

(done during design phase in architecture WP)

Legend

data flow
data interface
organisation unit
service interface
software component

Host Software Framework
« Functional » view

Software Product A
- Software unit A1
- Software unit A2

Software Product B
- Software unit B1
- Software unit B2
- Software unit B3

Software Product C
- Software unit C1

« Project organisation » view

« System CUX » view
(done during design phase in architecture WP)

Host Software Framework

Legend
- data flow
- data interface
- organisation unit
- software component
- service interface
- dependency
PAP & Engineering Dispositions document

■ Comes from
  • “Simplified Quality Assurance provisions for software development”
    • Part of ECSS-E-80

■ Addresses CUx for Scientific development (DU/WP)

■ Contents
  • QA provisions
    • For instance about the development cycle
      – What has to be provided during specification, design, …
  • Engineering dispositions
    • For instance:
      – Tests shall be automated with JUNIT.
      – Developers shall have separated environment for development and for execution
        When needed, those dispositions are given with example to be used directly
  • Coding standards are in separated documents
    • Java coding standard and Guideline document
    • …?
Important point

The development cycle is iterative

- Each iteration provides more functionalities to the CUx System
  - May reveal issues quicker
- Each CU, through the development plan:
  - Will have to determine the number of iteration and their
    Objectives,
    Activities planned,
    Completion criteria.
  - In concordance with the overall GAIA system planning
Programming Language

- **questionnaire**
  - **main objective**
    - Define and dimension the JAVA support actions
    - Determine if an additional language is needed
  - **41 replies**
  - **Main observations:**
    - Good availability of developers until the end of mission
    - Strong determination to reuse existing libraries or SW already developed in laboratories
    - Most developers have no experience in cooperative development
The language skills are extremely varied:

- Fortran (77, 90, 95), C, C++, JAVA, R, MatCad, MIDAS, Basic, Visual Basic, Pascal, Labview, IDL, Perl, TCL, Yorick, Mathlab, AWK, Python, Bash, …)

Opinion on JAVA:
- Of those who do not know Java (41 – 6 = 35 people), 25 have a positive or very positive opinion on the transition to JAVA and are prepared to learn this language.

Main objections on Java choice
- Need to translate codes previously developed and validated through years of use.
  - We cannot expect to obtain the same level of validation for codes transcribed into Java.
- Java computation performance is insufficient
  - (it appears, on the basis of the ESAC evaluation, that this objection is no longer relevant)
- We under-estimate the training time required
- Greater development time will be required due to lack of experience
- Transcription into JAVA should be handled at CU or DPC level.

Note: those familiar with JAVA and other languages see no special difficulties in implementing in Java for GAIA.
Language

• Java is the programming language recommended by the consortium and recommended by CNES for CUs 4, 6 and 8.
  - for both developed reused laboratory:
    • reused codes must be transcribed into JAVA, otherwise an existing Java library offering equivalent features will be used.
  - Actions required (CU1 and DPC from 2006):
    • JAVA training,
    • support for the translation of existing codes into JAVA,
    • support for setting up development environments,

• authorization to use Fortran on a case by case basis:
  - With the following conditions:
    • need to operate in a standalone environment
      - no interaction between different languages in the same executable
        (no calls to Fortran functions from Java or vice versa)
    • comply with the Fortran ISO standard (no use of features specific to a particular compiler)
    • Compliance with the coding standards,
  - With the following disadvantages:
    • Less support and tools
    • Data access may turn out to be much slower since less flexible than with JAVA
Environment & Workshop

- Wiki version of PAP document will be available
  - Ensure to use an up to date Version
  - Allow an easy “copy and paste” use

- Developer Toolkit
  - A developer Toolkit will be provide
    - This Toolkit has to be defined in CU1 framework
    - Include
      - Environment tools (language, library, ..)
      - Development tools (IDE, …)

- Quality, Java, Tools Workshop
  - Exact contents will be defined in march
  - The idea is
    - To give “à la carte” course
    - To get to a common way to work
Planning

- **Mid 2006**
  - PAP validated

- **October 2006**
  - Specialized PAP for CUx ready

- **Workshop QA, Tools, Java**
  - March 2006: schedule defined
  - June 2006: Number 1
  - October/November 2006: Number 2

- **Developer Toolkit**
  - Ready: mi 2006

- **Support Web Site:**
  - October 2006