European Committee for Standardization
Comité Européen de Normalisation
Europäisches Komitee für Normung

CEN ISSS WS/BII

Project Report Template – Report Annex to WG4 CWA

(Pilot Report Template)

Version: 0.1

Released: 2009-06-26

Date of approval:

ABOUT THIS DOCUMENT

This document presents requirements for a Project template and report template. The input for the content of this document is outlined in the “2008_03WG4_Report” document. The purpose of the report is to present requirements for the Project Template and the Project Report. To illustrate possible ways to manifest a project plan, an example is provided at the end of this document.

Note: It is important that this document be reviewed continuously to keep the information relevant and up-to-date. This document is subject to further elaboration.

PILOT (PROJECT) REPORT TEMPLATE

The pilot report template is be used to provide a summary of the current status of any pilot running. The pilot report is in essence a **progress report** that summarize how the pilot progresses. The main focus is to highlight any area issue area so that the pilot management might act on the issues.

Content:

PILOT ID

<Provide the specifics for the project, such as

- Date
- Project manager
- Version #>

GENERAL COMMENTS (FREE FORM)

<Provide a textual overview level description of the situation considering 1 – progress made since the last report, 2 – new challenges and 3 – any especially noticeable events>

MANAGEMENT VIEW (EXECUTIVE SUMMARY)

<Summarize, using a table, # of issues in the areas described below>

CHANGE REQUESTS STATUS

<List any change requests proposed during the pilot (so far) and the corresponding activities that any change requests has initiated>

QUALITY LEVEL SUMMARY

# OF DEFECTS

<Enter the # of defects known as a result of the test efforts conducted>

MITIGATED DEFECTS

<Enter the # of defects mitigated (handled) as a result of the test efforts conducted>

RISK SUMMARY

<Enter all risks, separating those entered before the pilot started and those identified during the execution of the pilot>.

MITIGATION SUMMARY

<For each listed risk, describe how the identified risk has been mitigated. To provide an easy to follow overview, consider using a table where the risk and corresponding activities initiated to mitigate the risk are listed>

TIMETABLE STATUS

COVERED AND OUTSTANDING MAIN TASKS

<Present the tasks that have been covered and the outstanding tasks, separated and in relation to the time table>

PROJECT PLAN TEMPLATE – ALTERNATE EXAMPLE

Below is an example of a project plan template used for traditional software projects. The intention is NOT to propose that pilot plan template present above should be replaced, but to illustrate how a project plan template alternatively could be manifested. Look as this alternative example as "inspirational".

Comments have been inserted to provide further clarification where the heading / content is not self explicatory. The project plan example is not to be viewed as a proposal, but rather as an example. Considering the PEPPOL pilot scope and content there are sections omitted in the below example, as well as sections that either are included in other documents or all together unnecessary / out of scope. Hopefully, though, the example will aid in finding out the any missing elements when composing project template and a project report.

Project Plan Example

Version 2.2

Edition 1

Document ID:	<document name goes here>
Author(s):	<author>
Reviewed by:	<revised by>
Date:	<date goes here>

Comment: To have the project plan revised is recommended (if not required).

Revision History

Date	Modifications	Reason	Version
1999.04.07	Various sections combined (AK)	Initial document collation	0.1d
1999.04.08	Document contents formatted (AK)	Conformance to Style Specification	0.3d

Table Of Content

1. System Development Plan

1.1 Major Milestones

The milestones have previously been defined in the <milestone definition> document. The following table lists the **milestones** in relative chronological order along with their dependent **tasks**, the number of people working on them, and the date they are due.

Comment: The actual Milestones as well as number of resources and due dates are only provided as examples.

Milestone	Milestone description	Dependency	Resources	Due Date
1	Requirements Analysis	0	7	30 October 2008
2	Verification	1	4	ongoing
3	Validation	1	4	ongoing
4	Requirements Solution Statement	1	7	30 November 2008
5	Requirements Refinement	2	5	ongoing
6	Solution	3	7	TBD

	Analysis			
7	Pilot Specification & Design	4	7	30 January 2009
8	Pilot Implementation	5	7	TBD
9	Pilot Testing	8	4 - 6	30 April 2009

Table 1.1 - 1: Major Milestones

1.2 Gantt Chart

The schedule as outlined below only defines the first five tasks be completed. For each task, there is one or typically more work items defined.

Note that the verification and validation tasks here only indicate the processes in regard to the functional prototype, not the final pilot. This is why in the above milestones chart they are listed as ongoing (open ended) and here we state their start and end dates.

<Gantt chart goes here>

Figure 1.2 - 1: Gantt Chart

In the above chart, color coded bars might indicate the planned start, duration and end dates while another color represented bar might represent the "slack" time in which we can finish the task without adversely affecting the time specified for later tasks. This might provide better exposure to task that have large adverse impact on the project.

1.3 Procedures

1.3.1 Requirements Analysis

In order to build the *correct* pilot the given requirements need to be analyzed in detail. The business requirements statement typically only specifies the broad idea regarding what functionality the system needs to provide. From this we need to flush out what is needed and what is not on a more specific level.

In order to do this effectively a **functional prototype** will be implemented.

The prototype will assist in verifying that the thus far defined requirements are correct, comprehensive and specified at the necessary level of detail. The prototype will also provide insight into the qualities of the proposed infrastructure and the technologies required to implement the requirements. The prototype will also provide an opportunity to establish development and quality assurance processes.

1.3.2 Verification

Verification is the process of determining whether the right solution is being developed. Once a prototype is complete the result of the prototype will be analyzed with the main objective to verify that the requirements have been satisfied in the prototype. This will involve an iterative two step process:

1 Determine what's missing or not required

2 If updates still required

2.1 Update the prototype

2.1.1 Return to step 1

2.2 Otherwise stop verification

It should be noted that verification is an ongoing process. At the each completion of each milestone, we will need to perform a verification analysis to make sure we are still on track. Similar to the above process, this will be in conjunction with the reference group.

Following the completion of this verification phase, it should be possible to begin defining the pilot we intend to build more precisely. This will be in the form of the **Requirements Solution Statement**.

1.3.3 Requirements Document Development

The requirements solution statement is an abstract definition of the **functional** and **non-functional** requirements of the system as interpreted. It will serve as the contract between the pilot <supplier> and the <commissioning body> by defining exactly what to develop and on what basis (standards, profiles, functional and non-functional requirements). The requirements document also serves as a basis for QA to develop testing procedures for the development/testing phase of the project.

Comment: the “Requirements document” is the collective name for what in effect very well could be an aggregate document, referring to several artifacts / documents.

Note: each major task / phase of the project will require a verification process, similar to that of the prototype verification. The purpose, to secure the result and level of quality, is basically the same, independent on type of task / activity.

1.3.4 Validation

Validation is the process of making sure that the solution is constructed correctly and according to the requirements specification (including legal and non-functional specifications). Throughout development of the pilot, weekly builds of the solution will be declared and handed to the **Quality Assurance** (QA) team.

QA will then perform various testing procedures on the product and present the defects to the development supplier(s) to mend. Once all the defects found by QA have been fixed, another build is declared and the process repeats itself.

The actual testing process is defined more precisely in the <**Implemented Test Guidelines document**>.

1.3.5 Solution Analysis

Once we have our final solution statement, we need to design the final solution. Given that we have already built the functional prototype by this stage, we should have a good idea as to how to go about building the system.

As a result of building and analyzing the prototype it should be possible to determine:

- what worked well
- what needs to be redesigned
- what needs to be thrown away altogether
- what needs to be added.

This will form the basis of the Solution specification and design.

Note: In order to avoid confusion, it should be pointed out that the term "**pilot**", above, and "**solution**" from this section and beyond might very well be interchanged, as the "pilot" (in the case of PEPPOL) is in fact the "solution", that is the final result – the solution/product.

1.3.6 Solution Specification and Design

The Solution Specification and Design will be based on the solution statement and solution analysis. It will include implementation specific information such as class diagrams, object models and so forth. At this stage the specification and design process is relatively undefined. As the project develops more information will become available.

1.3.7 Solution Implementation

As above, the process of implementation will become clearer as it comes closer and will be defined at a more appropriate time.

Note: The following section assumes that the PEPPOL pilot will be build in a "traditional" and solution oriented fashion. This might very well NOT be the case. The PEPPOL pilot might in fact be based on and assembled by various entities and building blocks, both traditional products (software) and services, available in the cloud and/or through on-site secure channels accessible at the the appointed suppliers' and/or clients' respective environment. The need, however, for CM and defined procedures is assumed to be unaltered.

2. Configuration Management Plan & Procedures

From the point where there is a functional user interface skeleton, builds will be made at intervals set by the **Application Engineer** (or similar role).

Each build will be a tagged version, named sequentially build-1, build-2, etc. These builds are sent to the Quality Assurance team (QA) and if they are deemed suitable to be released as versions, they are released as internal versions. The criteria used by QA to determine what qualifies as a version is specified in this document under the QA and Test Plan sections (alternatively referenced – with reference to the external Test plan).

When /new/ versions are to be constructed, they will be named sequentially version-1, version-2. These builds and versions will be performed by the **Configuration Manager**. Any problems should be reported to the Application Engineer (author) and the Configuration Manager.

Instructions on checking out builds and versions will be published when the first build is created. The build and version numbers are for internal use only and do not relate to the public releases. These will be made at the discretion of the Application Engineer and subject to quality assurance approving a build. These will be given names such as 'Cohesion' and 'InSync'

Projected release dates will be published by the Application Engineer when they become available.

3. Development Support Environment

The software development environment is a suit of applications which allow quality software to be written in a flexible way. All of the development software is available at the <development site/sites>. The following software is supported:

- <List of software used for development>
- Example (**note: partially fictional**):
    - Windows Vista
    - SUSE Linux version 3.0 or higher
    - Java 6 JDK from Sun Microsystems, Inc.
    - JRE 3.0from Sun Microsystems, Inc.
    - Notepad
    - Symantec Visual Cafe 6.0
    - JBuilder 5.0
    - Visual Studio Team Suite 2.0

Developers are free to use other development tools. Before any code is checked in to the repository, it must compile using the supported tools. Developers may not receive support for problems caused by using unsupported tools.

As new tools become available and are tested, they may be added to the list of supported software.

The software listed above requires for following *minimum* system hardware specification to run:

<The minimum Hardware specification goes here>

Any system not meeting this requirement will not be able to install and/or run the required software.

Development under other operating environments and or operating systems is not supported. It may be possible to develop and build under the Solaris OS, or on private Linux installations, but this is not supported and any problems must be solved by the individual developers. All code developed under these environments must be tested under the supported environment before being checked in to the repository.

4. Verification & Validation

4.1 Testing Plan Introduction

4.1.1 Testing Plan Purpose

The purpose of this *Verification and Validation Plan* for the *pilot*, is to document the planned steps taken by <the project group> to ensure that the solution developed is of high quality in relation to what the <commissioner> has requested. In other words, it assures that we take steps so that:

- We build the right solution.
- We build the solution right.

4.2 Testing Plan Scope

Please see **Report 1, requirements for Test Guidelines** for further information.

4.3 Lifecycle Phase Independent Activities

Lifecycle Phase Independent Activities might be outside of the scope for the PEPPOL pilot. It is, however, recommended that the solution is built in such a way that lifecycle specific considerations are at least considered and if possible built into the solution. This might include (but not be limited to):

- Built-in status check reporting
- Built-in monitoring and alert handling

- Built in near-real-time analysis and reporting

- Built in change management

4.3.1 Critical Analysis and Risk Assessment

Analysis and mitigation of risk is an integral and critical part of the project reporting. The project manager (or, if the necessary, the risk manager role) is responsible for maintaining the risk list as well as securing that all critical risks are highlighted and that activities to mitigate risk are planned and executed.

4.3.2 Document Review

Documentation reviews are conducted to observe measurable progress in the software completion process by reviewing and analyzing delivered solution design and development documentation. There are three kinds of document reviews conducted by the project team, namely:

- Document revisions received via the Customer as a Configuration Change Request (CCR). The project team's responses might be submitted in the form of **Impact Analysis Reports**. An example may be a CCR proposing changes to Level 3 requirements that could present a potential impact to the detailed design.
- Checking for missing documents. This occurs when extra requirements or extra wanted functionality is received from the Customer. The goal is to secure that all Change Requests are fully analyzed and that no activities are initiated without a clear definition of what is to be done (I e a functional and/or non-functional requirement specification).

- Checking for redundant documents or document sections. This occurs when requirements or functionality wanted is removed by the Customer.

Note: "Customer" in this context could be replaced for "steering committee" or "commissioning party" (or equivalent),

4.4 Lifecycle Phase Dependent Activities

If applicable (see the comment above), further Lifecycle specific activities might need to be planned and executed, such as:

- Analysis Evaluation (Prototyping)
- Design Evaluation

- Software Development Evaluation

- Test Evaluation

4.4.1 Analysis Evaluation

Analysis evaluation consists of examining both the process used when the project team performed the analysis for the solution and the actual requirements generated by the effort. Then, these requirements are compared to the known requirements received from the Customer, and the functionality required by the Customer.

4.4.2 Design Evaluation

Design Evaluation consists of examining both the process in which <the development team> produced the design for solution and the actual deliverables generated by the effort. Follow-on design analysis will focus on reviewing the progress of software development processes and changes, and enhancements to design the solution (and whatever entities that together form the solution).

4.4.3 Software Development Evaluation

Software Development Evaluation consists of analyzing software code and related documents to assess whether the implementation is traceable to the design and of high quality. The solution will also be checked for **standards compliance**, **internal code consistency**, **appropriate functionality**, and support of desired **user interaction**, as appropriate.

The Solution Development Evaluation is primarily solution oriented (I e focus is examination of software code); however, the implementation of software development processes (e g. adherence to any development standards used) will be examined also.

4.4.4 Test Evaluation

Test Evaluation consists of the witnessing and independently analyzing results of system tests performed by the QA Team. The following process in performing test evaluation could be used:

- Obtain system test and verification plans, and test schedules.
- Review plans for sufficiency and completeness of test coverage and requirements traceability.
- Identify tests to witness, critical requirements.
- Witness tests.
- Analyze test results provided by the QA Team to assess test conduct compliance to plans and procedures, and to verify that the results accurately and completely reflect the outcome of the tests.

The Test Process Evaluation focuses on how the QA Team test process is implemented. The evaluation examines the test plans for the system as well as the verification plan, and the system integration and test plans to assess the likelihood that the process will (continue to) yield the required implementation end solution.

5. Quality Assurance Process Plan

This document describes the following:

- The structure the <team goes here> will work in.
- The procedures and protocol the <team goes here> will follow.
- Assumptions and expected standards from other <team goes here>.
- Actions taken for possible issues or scenarios that may arise.

5.1 QA Team

<Description of one team – could be applied on any team part of the Pilot project>

5.1.1 Team Structure

Name	Skills	Type of work assigned
<Name goes here>	**<Skill set>**	**<List of responsibility and work items>** **Note**: ultimately generated from the project report tool and/or synchronized with the development and CM tool used.

Table 6.1.1 - 1: QA Team and Responsibilities

5.1.2 Team Meetings

The <team name> will be meeting weekly at 16.30 on Monday to discuss ongoing work and problems that may arise. These meetings are compulsory for all QA team members.

Conference calls are scheduled for each Friday at10.00.

5.2 Documentation

Formal deliverable documents written by the project team will be reviewed by a QA team member for content correctness and completeness according to:

- The user requirements and defined goals.
- The analysis and design outcome.
- An expert peer reviewer’s opinion (may be a member outside the QA team, if the expert knowledge is lacking within the QA team regarding the document subject).

Non-content related issues, such as the spelling, grammar and style of document, are to be checked by the ..<if applicable>.

5.3 Internet Site

The project Site set up for the PEPPOL pilot will be reviewed by one or more QA team members for content correctness and completeness according to:

- The user requirements.
- HTML Style Guide.

- An expert peer reviewer’s opinion (may be a member outside the QA team, if the expert knowledge is lacking within the QA team regarding the document subject).

Non-content related issues, such as the spelling, grammar and style of website, are to be checked by the <if applicable>.

For the functional aspects of the project Internet Site, software testing procedures will be followed according to the procedures and issues detailed in <list of procedures goes here / is referenced here>.

5.4 QA Release Software

The following section outlines the procedure for and the standards expected for the release of software to be tested.

5.4.1 Procedure for Releasing Software to QA

- <List of procedures goes here>

6.4.2 Standard of QA Release Code

Any code that is released to QA must adhere to the following standards:

- <List of release code standards goes here, if applicable>

5.5 Testing Process

This section describes the procedures QA will follow in order to assure a complete and unbiased testing of the solution. **Please reference the Report 1 – requirements for Test Guidelines for further information**.

5.5.1 Procedure for Testing

For each section of the solution to be tested, the following steps will occur:

- <Procedure list goes here>

5.5.2 Issues Relating to the Testing Procedure

The following are issues relating to the process of the QA Team:

- <Procedures for the QA team>

5.6 Bug Notification

The <bug report system> tracking system will be used for logging, viewing and editing bugs. The information on each bug logged will be as follows:

- Example:
- Date and time found.
- The component name (I e the Client Class).
- The software version.
- Concise description of the problem.
- Type of bug (I e Interface).
- Severity of bug using scale of 1 - 5 [1: Fatal error; 3: Non-fatal functional error; 5: Cosmetic errors (ie. No on-line help content)].
- Long description of test step, behavior and result.
- Current state: Not fixed, Open (I e being fixed), Fixed.

5.7 Test Cycle Completion and Iteration

Each test cycle will be iterative. To move on to the next test cycle, either no bugs or only bugs of severity 5 must be present.

There must be no bugs logged in the Internet Site bug tracking system before any stress or performance testing can be commenced.

5.8 Code Inspections

All code inspections will be performed according to the Software Inspection Process document obtainable from the QA Team Leader.

6. Documentation Plan

6.1 Document Completion Procedure

The various steps to be followed for a given document to be completed are as follows:

- The content of a document should be supplied to the project manager <or appointed responsible manager> by the group that is responsible for the document.
- A draft is written of the document and is checked and verified for compliance with the *Document Style Standards* defined in the project documentation plan. At this stage the document should also be given

a Version number and a document ID.
Note: The draft should use the generic document template. This template contains the proper formatting and style according to the Document Style Standards.

- The document is then passed from the <responsible role> to the Quality Assurance Team.
- The Quality Assurance Team checks and verifies the documents for compliance with their standards.
- The document is returned to either the <responsible role> or the group that initiated the document (for content related issues). Feedback on corrections or improvements to be made before the final draft should be supplied.
- Repeat steps 3 to 6 until no further corrections/improvements are required for the deliverable.

6.2 Transfer Procedure

To ensure that a document has been transferred from one sub-group within the project to another and that each group is aware of the transfer the following steps must be adhered to:

- A copy of the document, in HTML format, is placed in the 'docs' module of the project internet site, documents folder and sub folder, according to topic / activity.
- An email is generated to the entire project group (to inform them of the completion and handover of the document), the entire Quality Assurance Team (whose members will be delegated the task of checking the document by the Quality Assurance Head). The email will contain the following information:
    - Document ID
    - Author
    - Filename
    - Version number
    - Corrections/improvements made (only relevant if Quality Assurance has already seen the document and had sent it back for corrections)
    - Any necessary comments.
- Finally the milestone, deliverable, task or work item associated with the document should be transferred between the appropriate groups in the <project report system> (if available).

6.3 Feedback

Once the Quality Assurance Team has checked and verified the document for compliance with their standards, the changes are registered in the project log book.

7. Delivery Plan

7.1 Project Deliverables

For a deliverable to be ready for submission, it must have first followed the procedures outlined in the Documentation Plan. This process verifies that the deliverable has met the Quality Assurance plan (which included being subjected to the necessary test steps). Finally, the deliverable is published on the project web site and an email is sent to the assessor.

Deliverable	Due Date	Criticality
<Deliverable goes here>	<Date goes here>	<Impact value>

Table 7.1 - 1: Major Deliverables

8. Distribution Procedure

8.1 Deliverable Distribution Procedure

When a given deliverable is ready to be released, the following steps are to be taken:

- <Description of the release plan>

9. Marketing Plan

9.1 Solution Advertisement

<If applicable, describe how the pilot is to be advertised, including any marketing efforts>.