Guideline

Project Acronym:	**PEPPOL**
Grant Agreement number:	**224974**
Project Title:	**Pan-European Public Procurement Online**

Post Award/ Transport Infrastructure Life Cycle Management Model

Release Management Processes

Version: 1.00
Status: In Use

Editors:
Giacomo Franco, IBM
Raffaella Migliorini, Consip
Emanuele Colombo, IBM

Project co-funded by the European Commission within the ICT Policy Support Programme		
Dissemination Level		
P	**Public**	**X**
C	**Confidential, only for members of the consortium and the Commission Services**	

Revision History

Version	Date	Editor	Org	Description
1.0	27.02.2012	Giacomo Franco Emanuele Colombo	IBM	Released final document version

Contributors

Organisations

IBM, www.ibm.com
Consip (Italy)
Bremen Online Service (BOS)

Persons

Birgitte Lund, IBM
André Højgaard, IBM
Raffaella Migliorini, Consip
Lars Thølken, Bremen Online Service
Susanne Elken Thomsen, IBM
Nonny Nyberg, IBM

Table of Contents

1 Introduction

PEPPOL Release Management is used to collect Requests for Change (RFC's) into a release of software and documentation. When solutions to a collection of RFC's have been developed, they are gathered into a Release Package for joint test and deployment.

As indicated above, there is a close relation between PEPPOL Change Management (referencing the PEPPOL Pilot Lifecycle Methodology - PPLM) and PEPPOL Release Management. Change Management is related to the collection, assessment and management of RFC's received from various sources, while Release Management is the technical development, test and deployment of software and documentation to satisfy the RFC's in question.

In an environment like the PEPPOL Enterprise Interoperability Architecture (EIA), where many stakeholders and organizations are involved, coordination and communication of new or updated software and documentation is pivotal. To secure the coordination and the communication especially 3 things must be in place:

- Change Management Board(s)
  The Change Management Board decides which RFC's is part of a release. The Change Management Board also manages the release from planning to deployment. The Change Management Board can group different boards according to OpenPEPPOL organization, e.g. three different boards:
  - OpenPEPPOL post-award Coordinating Community
  - OpenPEPPOL eSignature Coordinating Community
  - OpenPEPPOL pre-award Coordinating Community.

- Tools for Change Management and Configuration Management
  JIRA is a support tool where RFC's are recorded. JoinUp is used as a Configuration Management tool, i.e. to store information about the current baseline of software and documentation.

- Communication Channels
  Various PEPPOL internet locations are used to publish information about a release, being software, technical documentation or end-user documentation.

1.1 Scope

Release management includes:
- Planning the rollout of software and other non-software Artefacts (e.g. specifications)
- Designing and developing the release components/specifications, and correctly documenting them
- Effectively communicating and managing expectations during the planning and rollout of new releases
- Managing the release validation and acceptance processes
- Recommending which actions to take when a new release is made
- Controlling the distribution of changes.

Release Management does not include support requests, incidents or problems to be solved by 1st or 2nd Level Support. These are managed by the PEPPOL Incident Management and Problem Management processes defined in the PPLM.

1.2 Long Term Sustainability considerations

The release management process is designed to handle PEPPOL software components and specifications. As for PEPPOL hosted services at long term sustainability strategy and governance structure is expected to be established within the framework of OpenPEPPOL.

In that context it would also be useful to have a plan for how to handle open source contributions from development teams which are not or only partially under the control of PEPPOL.

In general, the release management process should be revised after the Long Term Sustainability (LTS) organisation and strategy is defined. In the current version the process is based on the existing PEPPOL structures, e.g. Work Packages, and will therefore need to be aligned with the LTS organisation.

2 Process description

The overall responsibility for PEPPOL Release Management lies upon the Change Management Board (coordination and management) and 3rd Level Support (technical development). Activities are executed by 3rd Level Support, 2nd Level Support and 1st Level Support. Details are shown in the process flow in the following section.

All releases of Artefacts in the PEPPOL EIA will be performed using PEPPOL Release Management. Though there is a difference in the way they are handled in the different activities e.g. release of a new version of a service specification is more complicated and requires more in the activities than releasing a new version of a guideline to understand some sample implementation.

2.1 Process Flow

In the diagram the high level process flow for PEPPOL Release Management is shown.
The scenario described in the picture summarizes a general process that when it is implemented for a specific artefact could have few differences (see detailed flows in the following chapters of this document) in terms of activity complexity/aggregation and/or performers.

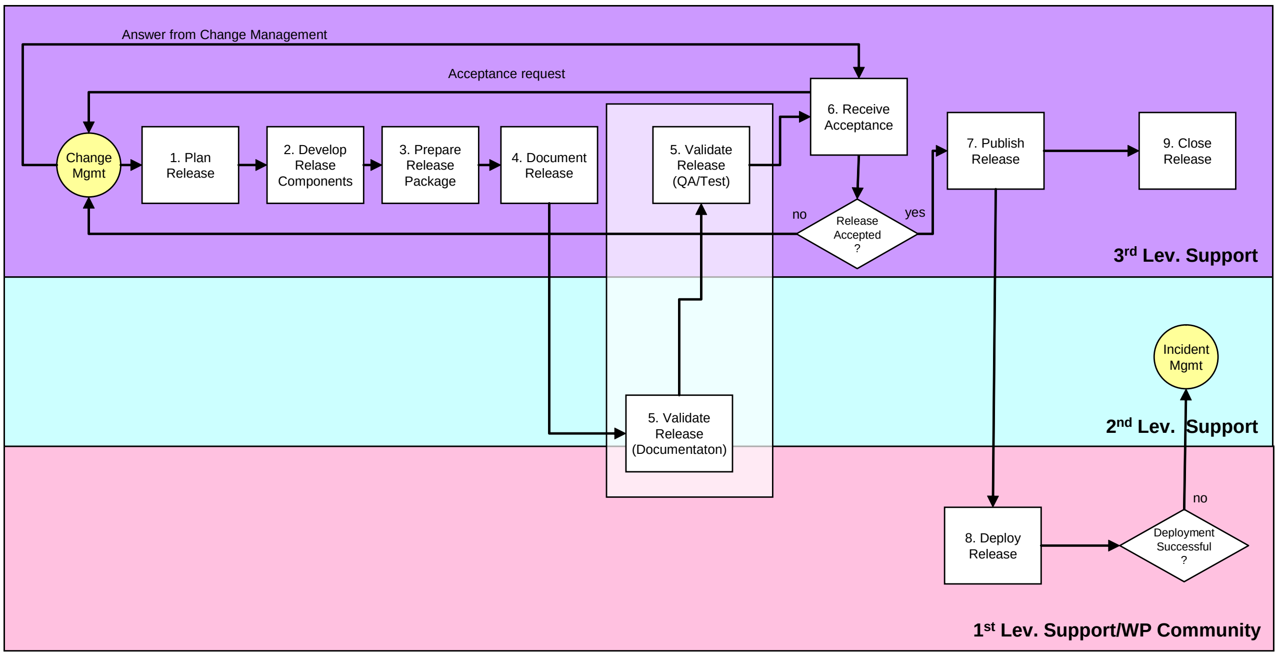


Figure 1: PEPPOL High Level Release Management Process Flow

The release validation action is depicted in the picture as split in two distinct items (Documentation and QA/Test). In fact, the validation can involve software or/and documentation, depending on the Artefact type. In case of non-software Artefact the validation will be applied only to documents (e.g. specifications), performed by 2nd level support and/or the involved WP community. For the software Artefacts the documentation will be reviewed by 2nd and 3rd level supports, while the QA/Test on the code by 3rd level support.

2.2 PEPPOL Artefacts

The PEPPOL Artefacts (software and documentation) managed through the Release Management process can differ by nature (documentation/software), complexity level and event/process driving the change.
The Release process will be described in this document by:

- **Software Artefacts**

  - <u>Major software releases</u>
    Major software releases normally containing large amounts of new functionality, some of which may make intervening fixes to problems redundant. A major upgrade or release usually supersedes all preceding minor upgrades, releases and emergency fixes.

  - <u>Minor software releases</u>
    Minor software releases, normally containing small enhancements and fixes, some of which may have already been issued as emergency fixes. A minor upgrade or release usually supersedes all preceding emergency fixes.

  - <u>Emergency software fixes</u>
    Emergency software fixes, normally containing the corrections to a small number of known problems.

- **Non-Software Artefacts**
  Non-software Artefacts of the PEPPOL Artefacts are specifications, guidelines, business rules, etc.

2.3 Release Policies

As mentioned in the previous paragraph, the involved Artefacts can be documentation and/or software. In order to version their release, the following general system and rules have to be applied:

[Version].[Upgrade].[Update]

- **Version**
  Major modifications (e.g. 4.5.7 –> 5.0.0), including major software releases or similar documentation changes or extensions.

- **Upgrade**
  Bundle of several major corrections (e.g. 4.1.3 –> 4.2.0), including backward compatibility, or minor documentation changes/corrections.

- **Update**
  Bundle of several minor corrections (e.g. 4.1.3 –> 4.1.4), including actions on code having backward and forward compatibility or focused corrections to documentation without impact on the overall content of the documents themselves:
  - *<u>Workaround</u>*
    Temporarily bypass of a bug without a change to specification and/or SW.
  - *<u>Patch</u>*
    Temporary elimination of a specification and/or software bug.

2.3.1 Release Policies for Software Artefacts

2.3.1.1 Software/Packages

For the release of software, the following guidelines will apply:

1. The objective will be to have maximally two releases operational in the field.
2. New major release will be delivered every 12 months.
3. A new major release contains all fixes and Service Packs that have been implemented in the meantime.
4. Only strict emergency fixes will be issued in between formally planned releases of enhancements and non-urgent corrections.
5. Releases are divided into:
    a. Major software releases (Versions): containing large areas of new functionality, some of which may make intervening fixes to redundant problems. A major upgrade or release will supersede all preceding Service Packs and emergency fixes.
    b. Minor software releases (Upgrades): normally containing small enhancements and fixes, some of which may have already been issued as emergency fixes. E.g. a minor release can be issued when API changes are incompatible.
    c. Emergency software fixes (Updates): normally containing the corrections to a small number of known problems.
6. Releases will be uniquely identified according to the following schema:

    <name>_<x>.<y>.<z>

    Where:
    - *<name>* is the name of the package/component
    - *<x>* is a number for the major release
    - *<y>* is a number for the minor release
    - *<z>* is a number for Service Pack or emergency fix.

7. RFC's will follow normal CR process and need prior approval of the Change Management Board.
8. RFC's will be rolled up in Service Pack's and, finally, in the first upcoming major/minor release.
9. Support will be provided for RFC's until the moment the first upcoming release becomes available that contains the rolled up RFC's.

2.3.1.2 Software bug fixes

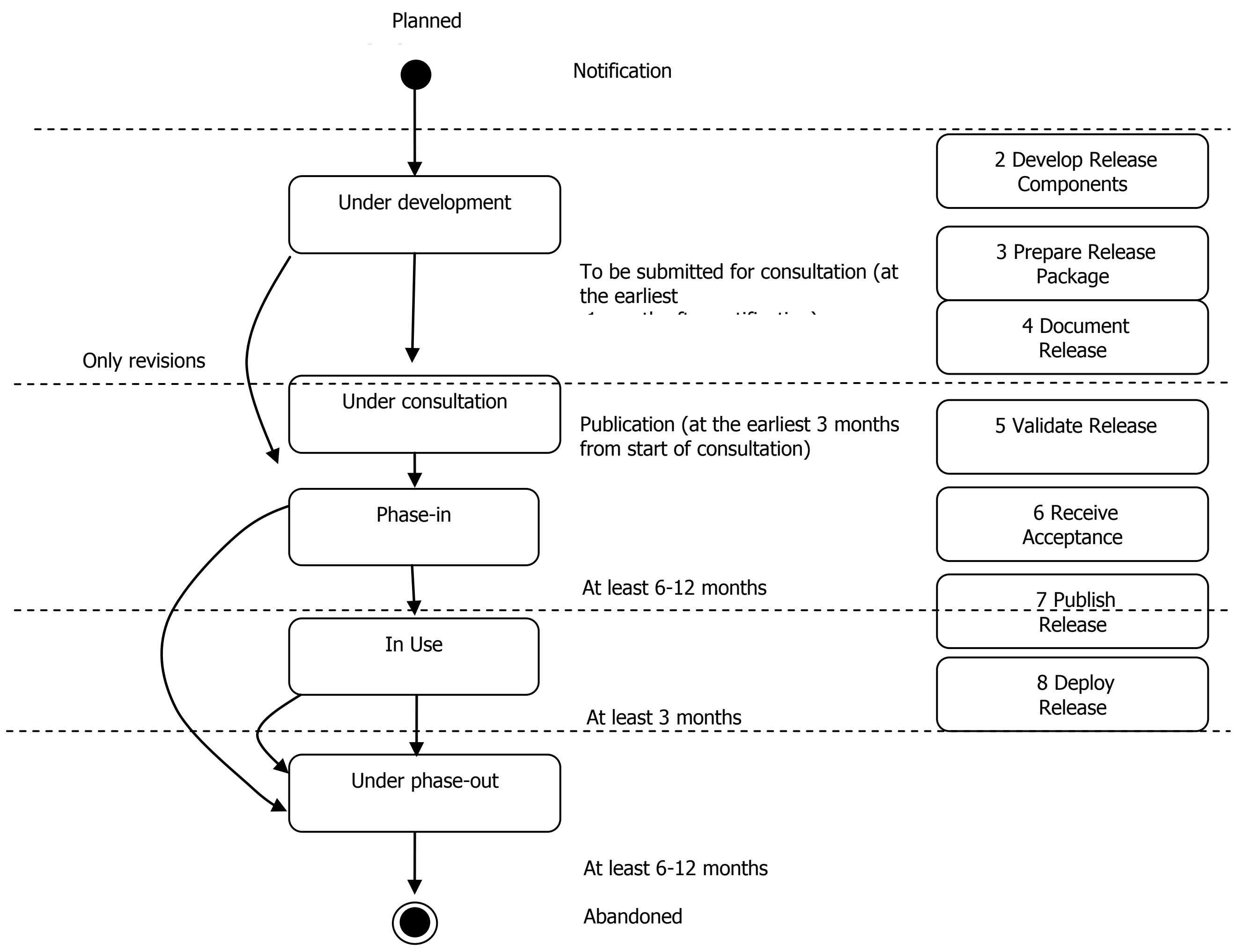
For the releases concerning bug fixes:
1. Version number will not change. Each fix will be numbered sequentially.
2. Change log will be published describing each bug fix stating explicitly the impact of the bug fix.
3. Bug fixes will be implemented on latest available General Availability release only.
4. Each source code fix made to a released branch will go through a code review process by the QA team. The team will look for validation of the fix, but more importantly look for regressions.

2.3.2 Release Policies for Non-Software Artefacts

For this kind of Artefacts it is necessary to introduce the concept of phases, describing the different status of specification during its lifetime:

- **Under Development** - It has been notified that work on a new version has started
- **Under Consultation** – The Artefact is not yet ready to be used, but is publicized because it gives some information on the direction for the community
- **Phase in** – The Artefact is replacing another (phase out) Artefact
- **In use** – the Artefact is the official PEPPOL Artefact to be used
- **Phase out** – The Artefact is in use, but it is being replaced by a new (phase in) Artefact
- **Abandoned** – The Artefact is not used anymore

The old version is “in use” until the new version changes from “phase-in” to “in use” – at which point the old version changes to “phase-out”. When the old version is not used anymore it passes to the state “abandoned”.
The diagram below shows the relationship between the various phases and the mapping with the Release Process.



2.4 Process Activities

The process activities are described in detail in this section by Software Artefacts and Non-Software Artefacts.

When information is sent between different support levels, the support tool JIRA will be used, i.e. comments are added to the requests in JIRA and automatically the next level is notified by email. (Refer to ISU JIRA Support tool, User's Guide).

2.4.1 Process Activities for Software Artefacts

Name	Description	Performed by
1. Plan Release	One or more approved change requests are considered for a release of PEPPOL EIA. The RFC's are analysed in order to decide what may be included in a release considering time frame, available resources, budget, and the priority of the requests. In the release to be planned, open contributions can be included if the WP Release Manager has evidence that they can be useful to be added, they are stable and they are going to be largely used by WP community. A time schedule based on the analysis is prepared by the WP Release Manager(s). The changes including technical and other requirements and the time schedule are composed into a Release and Deployment plan by the Change Management Board. 2nd Level Support is informed by mail about the release and deployment plan. 1st Level Support is informed when relevant.	3rd Level Support (WP Release Manager) and the Change Management Board
2. Develop Release Components	The release components are designed, developed and tested including integration test.	3rd Level Support (WP Development Team)
3. Prepare Release Package	The release components are gathered into a Release Package for transfer to the EIA, and it is delivered to 2nd Level Support. Compile a complete set of instructions for build of the release components. When it is possible, roll-back activities must be included. When a release can rollout in many different environments, it can be prepared as Sample Implementations.	3rd Level Support (WP Release Manager)
4. Document Release	Perform necessary end-user documentation, software compatibility requirements for other artefacts and technical documentation of the software explaining: - Release notes (what has changed) - What are the changes since last release - What are the recommendations (required or not, they optional or not, etc.) - Developer guides for service providers - Deployment guidelines If required, education material is performed.	3rd Level Support (WP Development Team)

Name	Description	Performed by
5. Validate Release	The Release Package is reviewed and evaluated to decide if is the release is supportable; e.g. the documentation is reviewed for completeness.	2nd Level Support
6. Achieve Acceptance	Ask for acceptance of release readiness from the Change Management Board(s).	Change Management Board(s)
7. Publish Release	When the release is approved for deployment, the Release Package is published on the Internet including all software and documentation artefacts (as described in section 3.3). 1st and 2nd Level Support is notified about the new release.	3rd Level Support (WP Release Manager)
8. Deploy Release	The new release is deployed and verified by PEPPOL Adopters. In case of an error or fault in the release, a request must be created to be handled by PEPPOL Incident Management.	1st Level Support
9. Close Release	The release is locked in the Configuration Management System (JoinUp) when it has been deployed. This version of the released components is now the current version (baseline).	3rd Level Support (WP Release Manager)

2.4.2 Process Activities for Non Software Artefacts

Due to the different nature of the Non-Software Artefacts, there is an impact on activities of general process because some of them need to be grouped to be aligned with previous described status specification phases (see paragraph 2.3.2).
In the following table, the process activities are described associating them to the corresponding phases.

Name	Phase	Description	Performed by
1. Plan Release	Plan Specification Release	While PEPPOL Specifications are grounded on existing standards major changes from Standard Bodies are the primary cause of changes. The changes are analysed in order to define the impacted artefacts. One or more RFC can be derived from a change in underlined standard. The involved WP communities are notified that work on a new version will start.	Change Management Board and 3rd Level Support (WP Release Manager and TC)
2. Develop Release Components	Under Development	The new Artefact release is designed and built.	3rd Level Support (WP Development Team) or OpenPEPPOL community related to WP
3. Prepare Release Package		The released documentation is gathered into a Release Package to be published on the suitable repository (see paragraph 3.3). Due to the nature of the Artefact, the Document Release activity is covered by the two previous activities (2 and 3).	3rd Level Support (WP Release Manager)
4. Document Release			

Name	Phase	Description	Performed by
5. Validate Release	Under Consultation	The version developed has been submitted for consultation and is awaiting comments. Depending on the team who produced the new release (WP Development Team or Community) the other team (Community or WP Development Team) will validate the release content.	OpenPEPPOL community related to WP or 3rd Level Support (WP Development Team).
6. Receive Acceptance	Phase In	The consultation phase has ended (without exceptions by people involved in the release validation and requests of changes), and related acceptance by the involved WP Community and WP Development Team. The WP Release Manager notifies the Change Management Board.	OpenPEPPOL community related to WP and 3rd Level Support (WP Release Manager)
7. Publish Release		The WP Release Manager made available the new version of the Artefact specification via the PEPPOL Internet Location. From this moment, any related Software components can be released and verified. If applicable, it will be announced that the current version will be withdrawn.	Community 3rd Level Support (WP Release Manager)
8. Deploy Release	In Use	The specification has become mandatory.	
9. Close Release			
Additional step	*Under phase-out*	*It has been announced that the version will be withdrawn.* *All the Artefacts related to this version will be withdrawn.*	*3rd Level Support (WP Release Manager)*

3 Tools

In this section the tools and documents used for PEPPOL Release Management is shortly described. They are listed in alphabetic order. For references to tools, documents, guidelines or similar not mentioned in this section, please refer to section 6 under.

3.1 JIRA

JIRA is the tool used for support management, issue tracking in the overall PEPPOL Project. Please refer to ISU JIRA Support Tool User's Guide for details (link is provided in section 7 under).

3.2 JOIN-UP

JOIN-UP is the community to share and reuse open source software, semantic assets and other interoperability solutions for public administrations.

3.3 Publishing releases

Non SW Artefacts

- All documents must use the format found in http://bscw.uni-koblenz.de/bscw/bscw.cgi/1936450
- All artefact must be entered into PEPPOL EIA - BSCW http://bscw.uni-koblenz.de/bscw/bscw.cgi/1824913
- The release log in the PEPPOL EIA - BSCW cell is updated http://bscw.uni-koblenz.de/bscw/bscw.cgi/1824913
- All artefacts must be entered into PEPPOL EIA - JOINUP: https://joinup.ec.europa.eu/svn/peppol/PEPPOL_EIA/
- All artefacts must be described (meta data) in the PEPPOL EIA – www.peppol.eu overview: http://www.peppol.eu/peppol_components/peppol-eia/eia
- The new release log in the PEPPOL EIA - JOINUP: https://joinup.ec.europa.eu/svn/peppol/PEPPOL_EIA/

SW Artefacts

- All SW artefact must be entered into a SW Folder in PEPPOL EIA - JOINUP: https://joinup.ec.europa.eu/svn/peppol/PEPPOL_EIA/
- All artefacts must be described (meta data) in the PEPPOL EIA – www.peppol.eu overview: http://www.peppol.eu/peppol_components/peppol-eia/eia and linked to the JoinUp Library

PEPPOL EIA Metadata

All artefacts must be described (meta data) in the PEPPOL EIA – www.peppol.eu overview: http://www.peppol.eu/peppol_components/peppol-eia/eia and linked to the JoinUp Library

PEPPOL EIA Metadata:

[Name]
[Filename]
Type: [Type]
[Short Description]
Version: [Version]
Status: [Work In Progress/Phase In/In Use/Phase Out/Abandoned]

Example

PEPPOL BIS 3a Invoice Only Specification
ICT-PostAward-PEPPOL_BIS_4a-300.pdf
Type: Specification
Specification on Basic eInvoice
Version: 3.00
Status: In use

It is the responsibility of the WP Release Manager(s) that all the Release Components provided by his or her WP is distributed as mentioned above.

3.4 Release and Deployment Plan

The Release and Deployment Plan is used by the Change Management Board to manage a release. Input to the plan comes from the 3rd Level Support Release Managers in the form of a Time Schedule, refer below. At the moment, there is no template for the Release and Deployment Plan but to be able to manage releases, the plan should at least include the information mentioned below. A spreadsheet is recommended because it is possible to filter on each column. Each RFC – preferably each Release Component - should have a row in the spreadsheet with information as listed below because one RFC can result in several Release components (e.g. software and documentation) while one Release Component can satisfy more than one RFC.
All information in the Release and Deployment Plan should be connected to an RFC.

- Release ID or name. (Used to uniquely identify a release and its related components. Should be left blank for RFC’s not yet planned for release)
- Release description. (E.g. the release category mentioned in section 1 over or main subject for the release. Should be left blank for RFC’s not yet planned for release)
- Planned Release date. (Expected date for the release. Should be left blank for RFC’s not yet planned for release)
- Related RFC. (Reference to RFC in JIRA. Input from Time Schedule mentioned below)
- Priority. (High (H), Medium (M) or Low (L). Used to prioritize RFC’s into releases. Input from Time Schedule).
- Status. (Accepted, Rejected, Planned, Tested, Release approved, Released, Closed. Status “Accepted” and “Rejected” are related to decisions made by the Change Management Board on the list of RFC’s received as input from the WP Release Manager(s). When status is changed to “Planned” – and onwards – “Release ID or name”, “Release description”, “Planned Release date” and “Release category” has to be filled in for the RFC/Release Component).
- Release Component. Reference to components to be implemented (sample implementations should point to the specifications they implement)
- Release category. (Refer to section 1 over for details. Should be left blank for RFC’s not yet planned for release).
- Work Package. (Reference to WP responsible for the development. Input from Time Schedule)
- WP Release Manager (Reference to WP Release Manager who gave this RFC as input. Input from Time Schedule)
- Release test results. (Status (pass / fail) for this particular RFC / Release Component when ready. Used to decide whether to deploy or defer).
- Actual release date. (Date for the actual release of PEPPOL Artefacts and communication of the release).

3.5 Time Schedule

The Time Schedule is the individual WP Release Manager’s input to the Change Management Board of RFC’s ready to be developed and collected into a release. The Time Schedule contains many of the same information mentioned in the section above about the Release and Deployment Plan. Information needed in the Time Schedule is repeated below.

A spreadsheet is recommended because it is possible to filter on each column. It will also make the consolidation of Time Schedules from several WP Release Managers easier for the Change Management Board. Each RFC – preferably each Release Component - should have a row in the spreadsheet with information as listed below because one RFC can result in several Release Components (e.g. software and documentation) while one Release Component can satisfy more than one RFC. All information in the Time Schedule should be connected to an RFC.

- Related RFC. (Reference to RFC in JIRA)
- Priority. (High (H), Medium (M) or Low (L). Used to prioritize RFC's into releases
- Release Component. (Reference to components to be implemented, e.g. unique references to software and/or documentation)
- Work Package. (Reference to WP responsible for the development)
- WP Release Manager (Reference to WP Release Manager who gave this RFC as input)
- Roll-back plan. (Yes (Y) or No (N). Used to assess risk of deployment of a solution. In general, a roll-back plan is considered mandatory).

4 Roles and Organizational Units

This section describes the teams and roles involved in PEPPOL Release Management and their responsibilities. They are listed in alphabetic order.

4.1 Change Management Board

The Change Management Board consists of representatives from various stakeholders and organizations involved in the overall PEPPOL Project.

The main responsibilities of the Change Management Board are to:

- Accept / Reject RFC's into releases
- Plan and Coordinate accepted RFC's into releases
- Manage releases by means of the Release and Deployment Plan mentioned above
- Coordinate Communication about new releases.

4.2 WP Release Manager and WP Release Team

The WP Release Manager is responsible for the collection of information about Problems and RFC's related to his or her Work Package. This information is initially stored in JIRA and transferred by the WP Release Manager into the Time Schedule mentioned above. The Time Schedule is input for the Release and Deployment Plan managed by the Change Management Board both of which are mentioned above.
It is an open point who will substitute the WP Release Manager after the piloting is ended. The new OpenPEPPOL organization should take care of to implement a new role equivalent to WP Release Manager.

The WP Release Manager is also the focal point for communication between the Change Management Board and the Work Package he or she represents, especially with regard to implementation of emergency fixes in or between releases.

During the development phase of the release, it is the responsibility of the WP Release Manager to report test results and other relevant information to the Change Management Board which could or should affect the decision of the board whether to deploy Release Components from the Work Package or not.

Finally, it is the responsibility of the WP Release Manager to update the baseline of his or her deliverable in JIRA and on the internet after deployment.

The WP Release Team collaborates with WP Release Manager to perform activities previously described.

4.3 Technical Work Package (WP)

The Technical Work Packages (WP's) are responsible for managing and improving the PEPPOL Artefacts and infrastructure. WP Release Team collaborates with WP Release Manager to perform activities previously described.
The WP's are also responsible for providing 3rd Level Support.

4.3.1 WP Development Team

The subteam of a WP in charge of developing the PEPPOL Artefacts implementing an RFC.

4.3.2 WP QA Team

The subteam of a WP in charge of performing the Quality Assurance process for PEPPOL Artefacts.

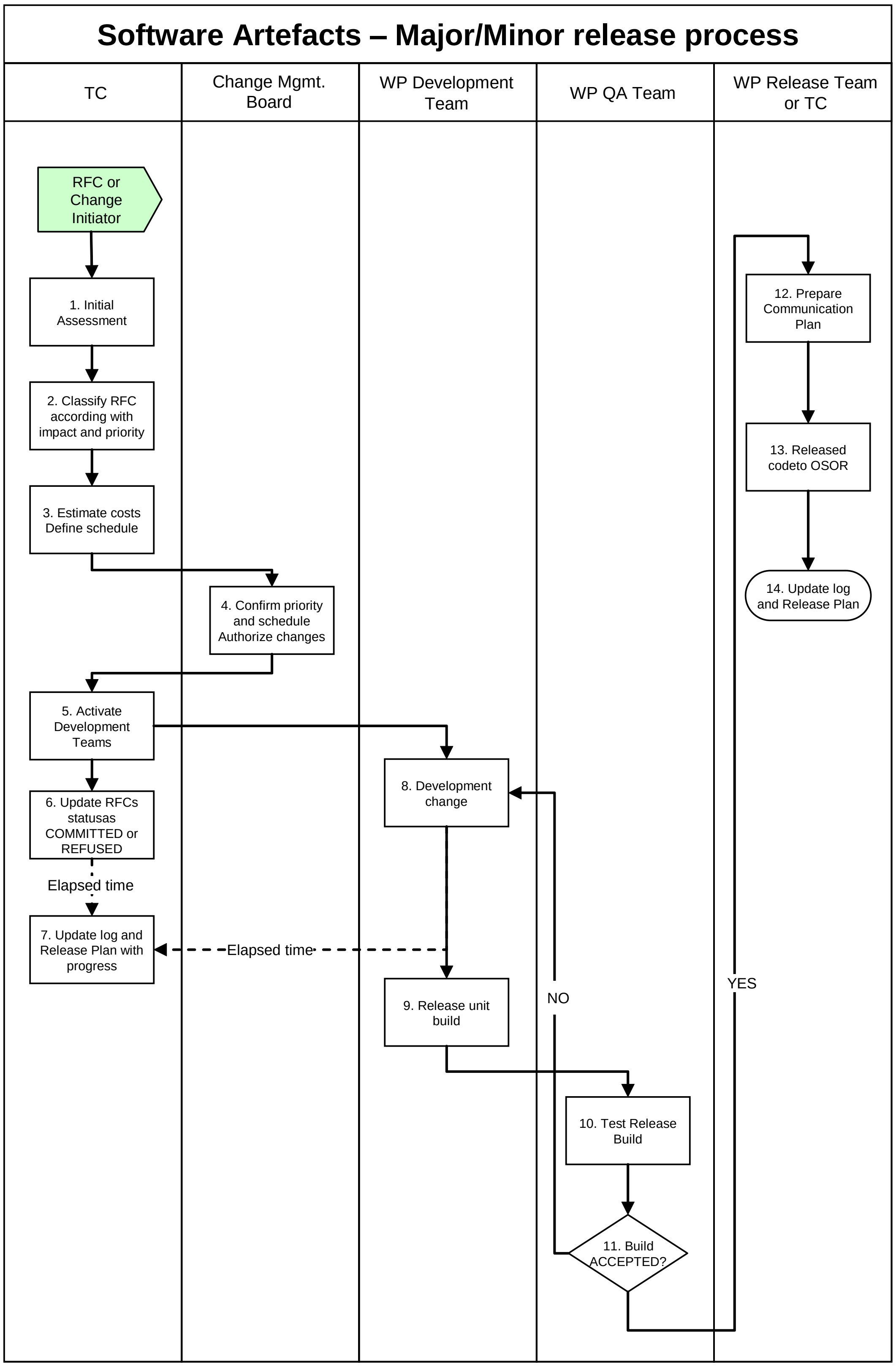
Quality assurance (QA) refers to the planned and systematic activities implemented in WPs, so that quality requirements for the software and not software artefacts will be fulfilled. QA implements a systematic measurement, the comparison with a standard, and monitoring of processes (like Systems Management, Software Quality Checks or Integrated Testing) with an associated feedback loop that helps error prevention.

4.3.3 WP Release Team/Technical Committee (TC)

The WP Release Team collaborate with WP Release Manager to perform activities previously described.

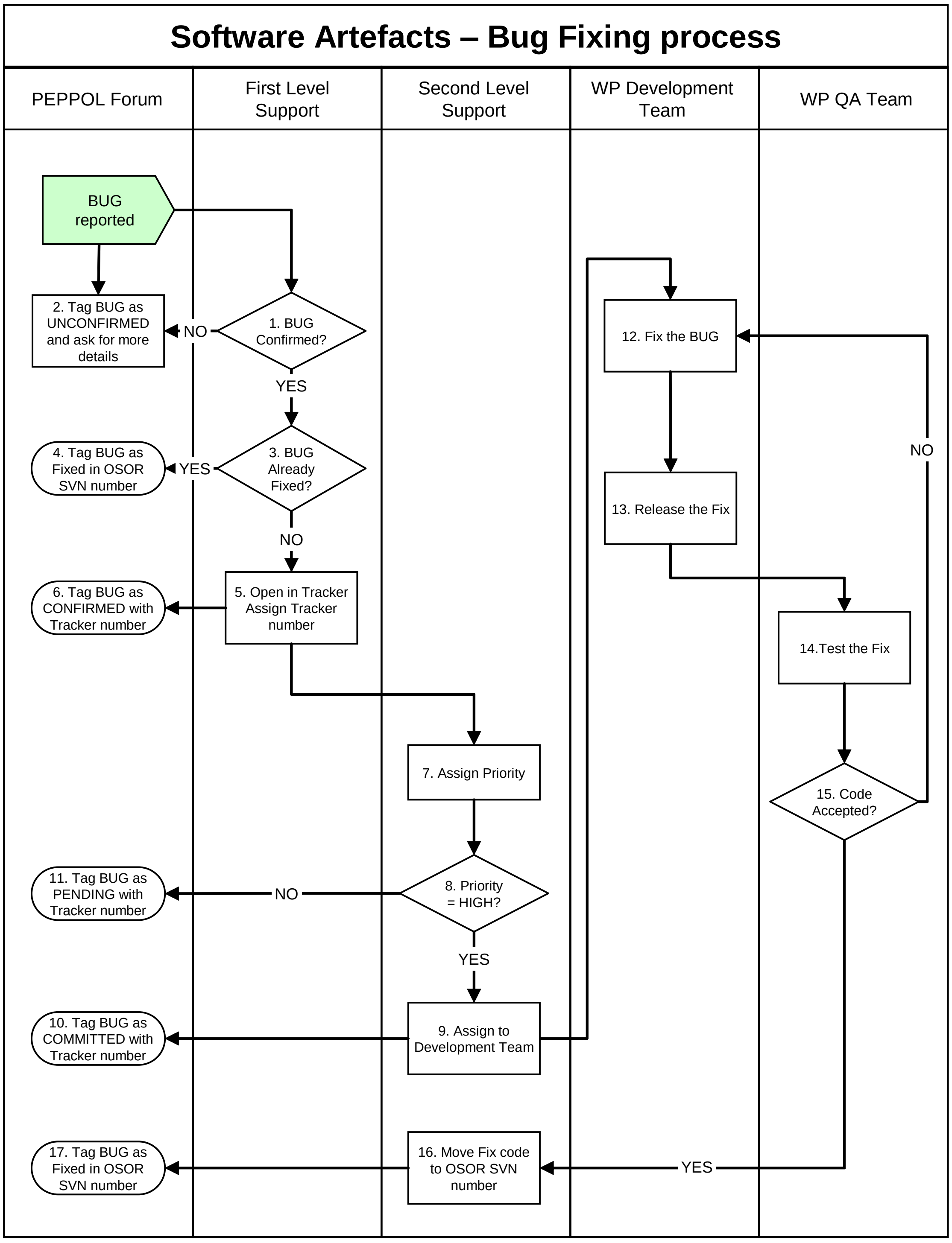
5 Operational processes

5.1 Software Artefacts – Major/Minor Release Process


Software Artefacts – Major/Minor release process
TC
Change Mgmt. Board
WP Development Team
WP QA Team
WP Release Team or TC
RFC or Change Initiator
1. Initial Assessment
2. Classify RFC according with impact and priority
3. Estimate costs Define schedule
4. Confirm priority and schedule Authorize changes
5. Activate Development Teams
6. Update RFCs statusas COMMITTED or REFUSED
Elapsed time
7. Update log and Release Plan with progress
Elapsed time
8. Development change
9. Release unit build
10. Test Release Build
11. Build ACCEPTED?
NO
YES
12. Prepare Communication Plan
13. Released codeto OSOR
14. Update log and Release Plan

1	The TC members will review all the pending RFCs (coming from pending bugs, function required, planned updates etc.). RFCs are tracked (a preformatted form should be made available) numbered sequentially by the system.
2	The RFCs are classified according to impact and priority.
3	Costs are estimated and a tentative scheduling will be prepared.
4	The CMB confirm the priority and authorize changes.
5	The TC activate the development teams.
6	The TC update the status of RFCs to COMMITTED or REFUSED.
7	The TC update log with progress (iterative until the end of the development and test phases).
8	The Development Teams work on the change.
9	The new Build (code and documentation) is released for test on JoinUp as "RELEASE CANDIDATE".
10	The QA Team test the code and control the documentation.
11	If the code is accepted the QA change the tag in ACCEPTED and notify the event to the Release Team. If the code is rejected repeat from step 8.
12	The TC prepare and publish the Communication and support plan.
13	The new code is tagged as General Availability.
14	The log and the Release Plan are updated.

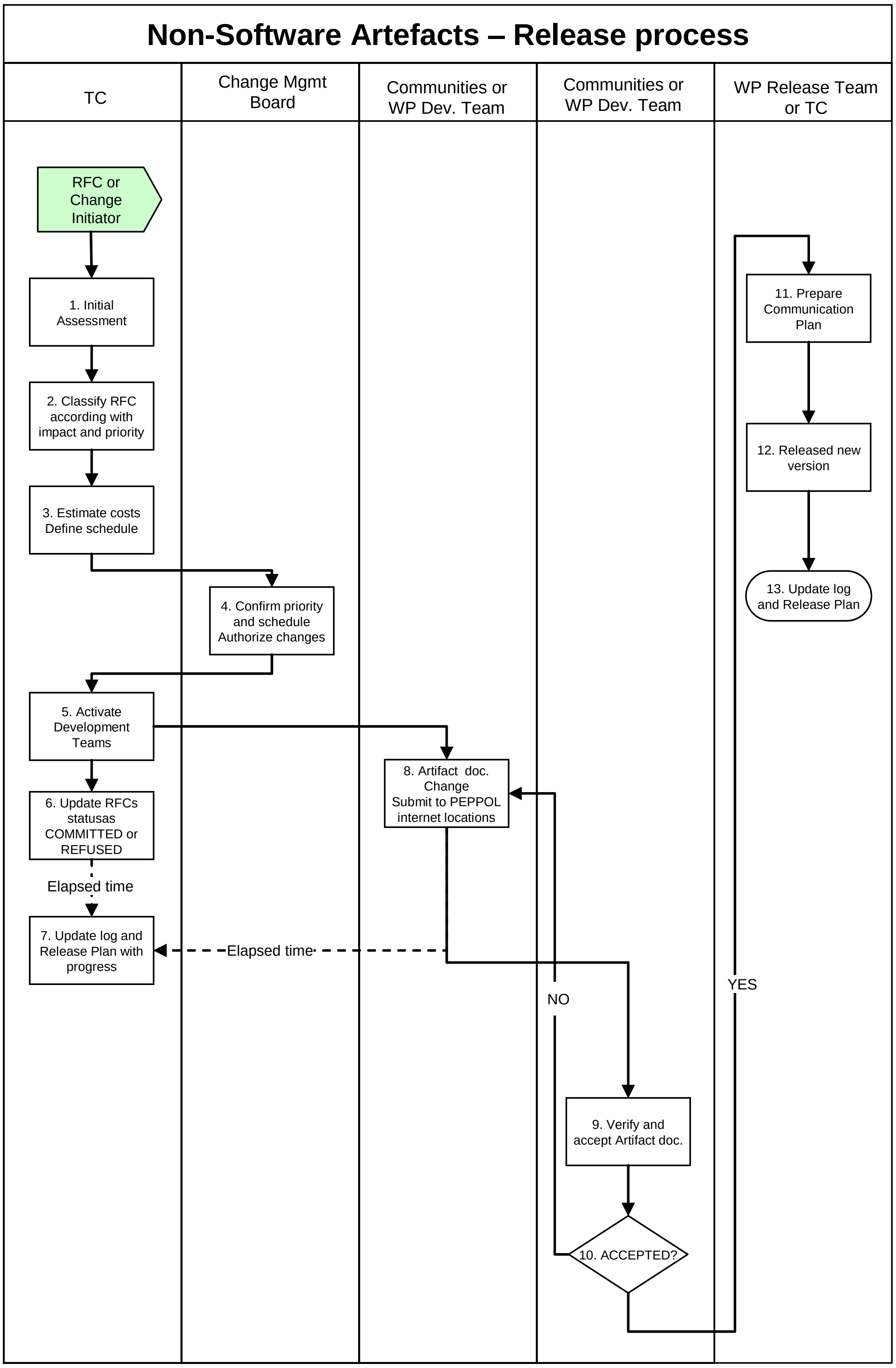
5.2 Software Artefacts - Bug Fixing Process


Software Artefacts – Bug Fixing process
PEPPOL Forum
First Level Support
Second Level Support
WP Development Team
WP QA Team
BUG reported
2. Tag BUG as UNCONFIRMED and ask for more details
NO
1. BUG Confirmed?
YES
4. Tag BUG as Fixed in OSOR SVN number
YES
3. BUG Already Fixed?
NO
6. Tag BUG as CONFIRMED with Tracker number
5. Open in Tracker Assign Tracker number
7. Assign Priority
11. Tag BUG as PENDING with Tracker number
NO
8. Priority = HIGH?
YES
10. Tag BUG as COMMITTED with Tracker number
9. Assign to Development Team
17. Tag BUG as Fixed in OSOR SVN number
16. Move Fix code to OSOR SVN number
YES
12. Fix the BUG
13. Release the Fix
14.Test the Fix
15. Code Accepted?
NO

A bug will be reported on the Developer/User Forum section of the PEPPOL Web site.

1.	A First Level Support member will review the bug and attempt to confirm the report.
2.	If the bug could NOT be confirmed, reply to the thread and request more information from the bug reporter, mark the thread as [UNCONFIRMED] and wait for feedback from the bug reporter and then start over at step 1.
3.	If the bug can be confirmed, test the report on the latest release checked-out on JoinUp to make sure it hasn't been fixed already.
4.	If the bug has been fixed, it is mentioned in JoinUp as FIXED.
5.	If the bug can be confirmed and is not fixed, reply to the report that it was possible to confirm the report and create an entry for the bug with all the information from the bug report.
6.	Set the bug status to CONFIRMED.
7.	A second level support member will assign to the entry a priority based on the following priority guidelines: • High - Something is really broken • Medium - This is not a full breakage but it doesn't work like it is supposed to • Low – Something is not quite right, but basic functionality is still present
8.	Test Priority.
9.	If priority is set to High, assign it to a Developer Team.
10.	Tag the thread as COMMITED with the entry reference.
11.	If priority is not High, tag the thread as PENDING with the entry reference.
12.	Now it is up to a developer to fix the bug branching from the latest General Availability Release.
13.	Once a fix has been prepared, the developer releases the fixed code for test.
14.	The QA Team test the fix to verify that the bug is resolved and the code is stable, the fixed code is marked as ACCEPTED.
15.	If ACCEPTED, the code is moved to JoinUp although repeat from step 12.
16.	The bug is mentioned as "FIXED".

5.3 Non-software Artefacts – Release process


Non-Software Artefacts – Release process
TC
Change Mgmt Board
Communities or WP Dev. Team
Communities or WP Dev. Team
WP Release Team or TC
RFC or Change Initiator
1. Initial Assessment
2. Classify RFC according with impact and priority
3. Estimate costs Define schedule
4. Confirm priority and schedule Authorize changes
5. Activate Development Teams
6. Update RFCs statusas COMMITTED or REFUSED
Elapsed time
7. Update log and Release Plan with progress
Elapsed time
8. Artifact doc. Change Submit to PEPPOL internet locations
NO
9. Verify and accept Artifact doc.
10. ACCEPTED?
YES
11. Prepare Communication Plan
12. Released new version
13. Update log and Release Plan

1	The TC members will review all the pending RFCs. RFCs are tracked numbered sequentially.
2	The RFCs are classified according to impact and priority.
3	Potential costs are estimated and a tentative scheduling will be prepared.
4	The CMB confirm the priority and authorize changes.
5	The TC activate the development teams.
6	The TC update the status of RFCs to COMMITTED or REFUSED.
7	The TC update log with progress (iterative until the end of process).
8	The Development teams work on the change. When their work is finished, the developed version is submitted for consultation to PEPPOL internet locations, awaiting for comments.
9	The communities/private suppliers and/or WP Development Team verify and approve the version.
10	If the version is accepted the WP Release Manager changes the tag in ACCEPTED and notify the event to the WP Release Team and to Change Management Board. If the version is rejected repeat from step 8.
11	The TC prepare and publish the Communication and support plan.
12	The new version is published.
13	The log and the Release Plan are updated.

6 Terms, Definitions and Abbreviations

This chapter includes only terms and abbreviations used in PEPPOL Release Management that could be unknown to the target group.

Terms are listed in alphabetic order.

Term	Description
Artefact	The published results of the PEPPOL project. For example, open source software components, specifications and design documents to describe the function, architecture and design of software.
Blocker	Refer to Severity below.
Change	Refer to Incident and Request for Change below.
Change Management Board	Refer to description in section 4.1.
Component	Used to describe one or more Artefacts in the PEPPOL EIA.
Configuration Management	In some areas, also known as "Version Control". Is used to manage versions (configurations) of software and documentation to the extent where it is known which version(s) are interim, draft or test versions and which are approved and published versions.
Critical	Refer to Severity below.
Documentation	Overall term referring to – but not limited to: ➢ End-user documentation – used by the end-users in the PEPPOL Pilot Projects to use the PEPPOL EIA. ➢ Education material – used to present the PEPPOL EIA for new users. ➢ Technical documentation – used by technicians when performing support, emergency fixes, development etcetera for the PEPPOL EIA.
Emergency Fixes	A solution or work-around for a problem which is considered to be so severe that it has to be solved as soon as possible and before a planned release is scheduled.
Incident	Any issue discovered by a PEPPOL Pilot Project or other stakeholder with the PEPPOL Implementation. The Incident is reported in JIRA and is either solved immediately, e.g. misunderstandings about the use of the PEPPOL Implementation which does not require any corrections, or deferred to PEPPOL Problem Management or PEPPOL Change Management for solution. (Refer to PEPPOL Pilot Lifecycle Methodology (PPLM) for details. Link is provided in section 7 under).
Internet Locations	Please refer to section 3 for details.
JIRA	JIRA is the tool used for support management, issue tracking and Configuration Management in the PEPPOL Pilot Project. (Refer to ISU JIRA Support Tool User's Guide for details. Link is provided in section 7 under).
Overall PEPPOL Project	Term used to embrace all PEPPOL Pilot Projects, Support Levels, PEPPOL Management Levels and other people in any way connected to PEPPOL.
PEPPOL Change Management	Process used to manage Requests for Change in the PEPPOL Implementation. (Refer to PEPPOL Pilot Lifecycle Methodology (PPLM) for details. Link is provided in section 7 under).
PEPPOL Enterprise Interoperability Architecture (EIA)	The PEPPOL EIA is a structured approach to present the PEPPOL Artefacts (project documents, specifications, user guides, software tools, etc.) in a repository so that different stakeholders can access information relative to their specific needs, in a consistent and flexible way.
PEPPOL Implementation	Current version or configuration of the PEPPOL EIA.
PEPPOL Pilot Lifecycle Methodology (PPLM)	A document describing the enablement, implementation, support, monitoring, assessment and evaluation of PEPPOL Pilot Projects. (Refer to PEPPOL Pilot Lifecycle Methodology (PPLM) for details. Link is provided in section 7 under.

Term	Description
PEPPOL Pilot Project	• Involving one or more Pilot participants • Covers all of the pilot phases as indicated in the PPLM (from recruitment to final evaluation. Link is provided in section 7 under) • Covering one or more Pilot engagement processes • Covering several Pilot Sub Projects • Potential complex project structure
PEPPOL Problem Management	Process used to manage problems in the PEPPOL Implementation. (Refer to PEPPOL Pilot Lifecycle Methodology (PPLM) for details. Link is provided in section 7 under.
Problem	Refer to Incident and Emergency fixes above.
Release and Deployment Plan	Plan that gives an overview of planned releases, time schedule and status. Used by the Change Management Board to coordinate software and documentation provided by different stakeholders and organizations in the overall PEPPOL Project. Refer to section 3.4 over for details.
Release Package	A collection of new or changed PEPPOL Artefacts which is delivered as a release of PEPPOL EIA. The Release Package includes PEPPOL Artefacts - developed or changed - by one or more of the WP's.
Request for Change (RFC)	The format used in PEPPOL Change Management to register changes required (problems), proposed and/or requested by the various stakeholders and organizations participating in the overall PEPPOL Project. RFC's are registered in JIRA and managed by the Change Management Board to closure.
Roll-back (activities / plan)	A description of how to roll-back a release – or part of it – if deployment fails. Roll-back is important because the PEPPOL EIA could come to a halt if an error prone component is deployed and it is not possible to perform a roll-back. The entire PEPPOL EIA would have to wait for an emergency fix before normal use could be retained.
Severity	Term used to assess the urgency of a problem. The following table is used in PPLM (Refer to PEPPOL Pilot Lifecycle Methodology (PPLM). Link is provided in section 7 under):

Severity	Characteristics
Blocker	Blocks development and/or testing work, production could not run.
Critical	Crashes, loss of data, severe memory leak
Major	Major loss of function
Minor	Minor loss of function, or other problem where easy workaround is present
Trivial	Cosmetic problem like misspelled words or misaligned text.

Term	Description
Small Enhancement	An RFC that requires little effort to solve AND which does not raise risks, issues or elaborate changes to the PEPPOL EIA, e.g. to change the colour of a button or to correct spelling errors on a screen.

Term	Description
Support Level	The PEPPOL Pilot Lifecycle Methodology (PPLM) operates with 3 Support Levels: ➢ 1st Level Support – in general considered being the level closest to the end user. ➢ 2nd Level Support – in general considered being a middle layer distributing communication about Incidents, Problems and RFC's between the 1st Level Support and 3rd Level Support. In some cases, 2nd Level Support will be able to resolve Incidents or Problems. ➢ 3rd Level Support – in general considered being the technical layer handling Problems and RFC's which require technical knowledge and/or modifications to software / documentation. Refer to PEPPOL Pilot Lifecycle Methodology (PPLM) for details. Link is provided in section 7 under.
Time Schedule	Input developed by the WP Release Manager to the Release and Deployment Plan (refer above) managed by the Change Management Board.
Work Package (WP)	A project team within the overall PEPPOL project. Contributes with software, documents and/or expertise in a particular area of the PEPPOL EIA.
WP Release Manager (3rd Level Support)	Refer to section 4.2 over.
WP Technical Expert (3rd Level Support)	Typically a developer or similar chosen by the Work Package to solve a Problem or develop a solution to satisfy an RFC.

7 References

Reference	Document	Link
JIRA User's Guide	ISU JIRA Support Tool User's Guide	http://bscw.uni-koblenz.de/bscw/bscw.cgi/1970939
PEPPOL Pilot Lifecycle Methodology (PPLM)	In progress	http://bscw.uni-koblenz.de/bscw/bscw.cgi/d2050994/PEPPOL%20Pilot%20Lifecycle%20Mehodology%20v.0.7.doc
PEPPOL Enterprise Interoperability Architecture (EIA)	PEPPOL artifacts (project documents, specifications, user guides, software tools, etc.)	http://www.peppol.eu/peppol_components/peppol-eia