NENA Rules for Use of Version Control Repositories

NENA Rules for Use of Version Control Repositories

NENA-ADM-012.1-2022
DSC Approval: November 08, 2022
Next Scheduled Review Date: November 01, 2027

Prepared by:
National Emergency Number Association (NENA) Data Structures Committee, Interface Specification Development Working Group

1 Executive Overview

The standards development process produces Artifacts, which are concrete items such as documents (especially published specifications), interface descriptions, code, etc. Documents often specify interfaces, protocols, values, and other formal aspects of a standard. Documents are written in English and managed using NENA-ADM-001 [3] procedures. Other Artifacts are derived from a specification and written in a formal language (e.g., XML, JSON, or YAML). These derivative or formal Artifacts are managed using a code repository under the rules of this document. These Artifacts are used by software developers to implement a standard; they are tools for implementors. A common formal Artifact is a schema or an interface description (e.g., documented in XSD or OpenAPI). The development process for formal Artifacts is different than the editing process of a standard, because the Artifact is created to formalize part of a specification. The tools used to create and maintain these Artifacts is different. This document describes how NENA uses the GitHub tool [4] to maintain these Artifacts.

NENA's repositories SHALL be held permanently at https://github.com/NENA911.

Table of Contents

NENA
ADMINISTRATIVE DOCUMENT
NOTICE

This Administrative Document (ADM) is published by the National Emergency Number Association (NENA) to describe the document development and review process for the benefit of NENA members, partners, and affiliated entities. This process has been developed to clearly define the operational procedures used to develop, review, and approve NENA documents for publication that address the complex operations and technology issues related to the provisioning and management of emergency communications services.

This document has been prepared solely for the use of those individuals and agencies involved in the development and approval of NENA documents. NENA reserves the right to revise this document for any reason.

By using this document, the user agrees that NENA will have no liability for any consequential, incidental, special, or punitive damages arising from the use of the document.

Recommendations for change to this document may be submitted to:

National Emergency Number Association
1700 Diagonal Road, Suite 500
Alexandria, VA 22314
202.466.4911
or: commleadership@nena.org

2 Document Conventions

NENA: The 9-1-1 Association improves 9-1-1 through research, standards development, training, education, outreach, and advocacy. Our vision is a public made safer and more secure through universally available state-of-the-art 9-1-1 systems and better-trained 9-1-1 professionals. Learn more at https://www.nena.org.

2.1 Document Terminology

This section defines keywords, as they should be interpreted in NENA documents. The form of emphasis (UPPER CASE) shall be consistent and exclusive throughout the document. Any of these words used in lower case and not emphasized do not have special significance beyond normal usage.

1. MUST, SHALL, REQUIRED: These terms mean that the definition is a normative (absolute) requirement of the specification.
2. MUST NOT: This phrase, or the phrase "SHALL NOT," means that the definition is an absolute prohibition of the specification.
3. SHOULD: This word, or the adjective "RECOMMENDED," means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
4. SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED" means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood, and the case carefully weighed before implementing any behavior described with this label.
5. MAY: This word, or the adjective "OPTIONAL," means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option "must" be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option "must" be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

These definitions are based on IETF RFC 2119 [2].

2.2 NENA Intellectual Property Rights (IPR) and Antitrust Policy

NOTE – The user's attention is called to the possibility that compliance with this standard may require the use of an invention covered by patent rights. By publication of this standard, NENA takes no position with respect to the validity of any such claim(s) or of any patent rights in connection therewith. If a patent holder has filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and

conditions to applicants desiring to obtain such a license, then details may be obtained from NENA by contacting the Committee Resource Manager (CRM) identified on NENA's website at https://www.nena.org/ipr.

Consistent with the NENA IPR and Antitrust Policy, available at https://www.nena.org/ipr, NENA invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard.

Please address the information to:

National Emergency Number Association
1700 Diagonal Rd, Suite 500
Alexandria, VA 22314
202.466.4911

or commleadership@nena.org

2.3 Reason for Issue/Reissue

NENA reserves the right to modify this document. Upon revision, the reason(s) will be provided in the table below.

Document Number	Approval Date	Reason for Issue/Reissue
NENA-ADM-012.1-2022	November 08, 2022	Initial Document

3 Version Control Systems and Repositories

A Version Control System (VCS) is used to track the history of changes as multiple people or teams work on the same project together. VCSes are ubiquitous in the software development industry so multiple programmers can work simultaneously on the same codebase, while tracking and managing multiple versions and providing a way to establish consensus and resolve conflicts. Git [5] is the most popular distributed VCS used today. Git is a free and open source VCS. It allows for developers to see a full timeline of changes, decisions, and progression in one place. Any change can be viewed and, more importantly, easily rolled back. NENA's Development Group (NDG) uses the Git tool GitHub to manage version control for its schemas and interface definitions. This is because a VCS designed for software development is a much more appropriate tool for managing changes to schemas and interface definitions than NENA's traditional means of maintaining a single standing version of a consensus document that includes the normative definition of a schema or interface definition in a static document.

A repository (or "repo") in GitHub is a folder (directory) containing one or more files. The Artifact itself is a file, but there may be other files such as shell scripts, documents, test cases, other folders, etc. Each file has a revision history which is tracked by GitHub. GitHub

commonly has a repository per "project." In NENA, an API is a project, as opposed to, say, a document. This is because documents may modify any number of APIs, some of which may have been defined in different documents. NENA repositories are not limited to APIs; repositories may be created for other purposes.

A working group requiring a new repo SHALL request that the NENA GitHub Administrator create the repo, including minimum content requirements for the initial readme.md (see Section "Readme").

3.1 Git Administrator

The Git Administrator is at least two (2) individuals designated by the Development Steering Council (DSC) to manage NENA repositories. When there is a vacancy, DSC SHALL appoint a new Administrator within a reasonable amount of time.

3.2 About Versions

Artifacts have versions (see Section "Use of Versions"), which follow the pattern of NENA document versions: <major>.<minor><fix>, where minor versions of a major version are backwards compatible with each other. The Artifact is reviewed as part of a document review process and the merge to "main" only occurs on completion of that process. The <fix> designation is for updates to a version that used the "errata" process defined in NENA-ADM-001 [3].

The version number of an Artifact is NOT tied to the same version number of the document that creates or modifies it. The first version of an Artifact is always 1.0, regardless of the version of the document that creates it, and the Artifact's version is incremented as that document, or other documents, make changes to the Artifact. If any document, regardless of whether it is a new document or a major or minor version of an existing document, updates an Artifact in a way that is backwards compatible with all minor versions of the current major version of the Artifact, then the Artifact increments the minor version and retains the existing major version. As a change to an Artifact that is not backwards compatible with other minor versions, by definition, makes the document a major version update, major version updates to Artifacts are always major version updates to the document that makes the change. A minor version update to a document could create a new Artifact (1.0 version) as long as the Artifact is optional to use.

3.3 About Branches

Repositories foster development of Artifacts by allowing creation of a "branch." A branch is effectively a copy of the repository to which changes may be made to any of the files, and specifically the Artifacts, without affecting the (default) "main" branch that always represents the latest released version of the repository. Branches may be merged into other branches, including "main." When a branch is merged to "main," that is the formal

release of the version, and becomes the new release. Older branches are maintained, so that users may download any branch.

The name of a branch is the version the Artifact will become as created or modified by a corresponding document. For example, version 3 of a document creates a Major Version of an Artifact. Later, version 3.2 of the document updates the Artifact in a backwards compatible way. A Minor Version will be created; updates will be made to that branch as development of the 3.2 version of the document proceeds. When the document is approved as version 3.2, the Minor Version branch will be merged into the "main" branch of the Artifact. If another document further modifies the Artifact, a new Minor Version branch will be created in the repository and changes in that branch will be considered as the development of the other document proceeds.

3.4 Commits and Pushes

Users typically "clone" a repo, select or create a branch, and edit files in that branch. The clone is maintained by git as a local copy of the entire repo. When changes to files are made, git tracks the changes. An operation called a "Commit" memorializes what was changed, who changed it, and when it was changed. Conceptually, it creates a snapshot of the repo with the changes applied. A comment may be added to the commit. For NENA documents, comments SHALL be included that summarize the changes. It is possible in some systems to "push" the commit directly to the main repo, but this SHALL NOT be done to a NENA repository. See Section "Pull Requests" below.

Commits on a branch SHALL indicate in their description (commit message) the entity that is requesting the change (usually a working group).

All commits to a working branch SHALL include the name of the corresponding document in the description for the that commit.

3.5 Pull Requests

Pull Requests are requests from an author to apply a set of changes to a branch. GitHub manages this process with lots of automation. The Owner (see Section "Ownership") approves pull requests as changes are reviewed, similar to how documents are updated. This is contrasted with a Push that directly changes the repo branch following a local commit. NENA repos exclusively use Pull Requests and SHALL NOT permit Push operations. Pull Requests are reviewed and then merged to the branch.

Pull Requests on a branch SHALL indicate the entity that is requesting the change (usually a working group) in their description.

3.6 Readme

Every repo and every directory SHALL have a readme.md file. GitHub automatically parses this into a human-readable landing page. The readme file MUST contain:

- A description of the Artifact
- The Owner of the Artifact
- Rules for contributing changes. If not otherwise specified, the default rule for a NENA repository SHALL be "Specification Required." The following rules for changes are adopted and adapted from IETF RFC 8126 [6], and have substantially similar meaning:
    - Private Use
    - Experimental Use
    - Hierarchical Allocation
    - First Come First Served
    - Expert Review
    - Specification Required
    - Standards Action
- Administrator contact information (with a stable URI). (The Owner of the repo or the Administrator for a development branch)
- Version history

4 Ownership

A NENA repository has an Owner. The Owner is normally the working group that created the repo, or its successor. Only the DSC can change the Owner.

The Owner of the main branch has final say on the main branch. This means that another working group making changes on a development branch must have assurance that the Owner intends to approve changes in the development branch. When in doubt, the working group maintaining a development branch may request the Owner conducts an Approval Ballot [3]. Approval of the ballot indicates consensus of the Owner to release the version approved by the development working group.

Development branches are created by the working group that is creating or changing the Artifact.

If required, the DSC can name any entity as an Owner per its discretion.

5 Tags

GitHub has the ability to tag specific points in a repository's history as being important. A tag is a short string attached to a specific commit.

NENA Artifacts and working branches use tags to indicate released versions and review cycles. Other uses of tags are up to the development or Owner working group. When a document undergoes a review, the version of the Artifact that is being reviewed gets a tag that matches the review. Tags used for review cycles of an Artifact SHALL be:

- working-group-review

- all-committee-review" (or "all-committee-review-2" or -3, etc.)
- public-review" (or "public-review-2" or -3, etc.)

When a document is approved for publication and the branch is merged to "main," the merged version SHALL be tagged with the document name that created or modified the Artifact. This allows an implementer to know what the most current released version is: the tagged version on "main."

Tags are maintained so that users may download any prior release. This is the way versions are maintained. Any non-historic NENA document that defines an Artifact SHALL have a stable link maintained to the tag for that Artifact.

6 Issues

NENA should make good use of the issues feature of GitHub. An issue may be a bug report, a feature request, a memory jogger, or a strategic issue being worked by multiple contributors. Any working group member may create an issue. Issues are managed by working group co-chairs or co-chairs may delegate issue management to a skilled member or group of members of the working group. Working group members are encouraged to use issue templates, and working groups are encouraged to create templates for frequently occurring issue sources beyond the default bug report and feature request templates. When creating or working issues, contributors are encouraged to use permalinks[1] to link to Artifacts or other files in a repo. Crosslinking[2] of issues, when they are related, is also encouraged. Larger issues should include a checklist to manage multiple aspects of one issue.

Some working groups use GitHub issues to manage issues that occur within the document development process, beyond the Artifacts that are maintained in GitHub. Similar to a "parking lot," issues provide a mechanism for remembering, processing, and documenting items that the working group has agreed to do. This requires non-developer working group members to learn how to use at least the issue management facilities of GitHub. Use of issues in GitHub does not supersede use of the "Future Work" section of a NENA standards document.

7 Use of Versions

Each version of the Artifact SHALL be tagged with the appropriate version number. This SHALL be recorded in the Version History field in the readme.md file. Each version SHALL update the previous Artifact. Prior version history is available by browsing release tags; for example, if implementers require backwards compatibility or need to implement a legacy

[1] This document does not explore the concept of permalinks in GitHub. For more information see GitHub documentation at [4].
[2] This document does not explore crosslinking in GitHub. For more information see GitHub documentation at [4].

version of an Artifact for some purpose. Implementers SHOULD always implement the current version of any Artifact and are strongly discouraged from knowingly implementing legacy versions.

7.1 Release Tags

The following tags SHALL be used for releases (examples in parentheses):

- **Major** Version (1.0, 2.0)
- **Minor** Version (1.1, 1.2)
- **Fix** (1.2a, 1.2b)

The syntax for Release Tags is:

[Major Version] "." [Minor Version] [Fix Version] " " [Document Version]

7.2 Additional Rules about Versions

Updates to versions other than the current version SHALL NOT be allowed unless it is a Fix. In some cases, two documents may be developing Minor versions simultaneously; in these cases, it is the responsibility of the Owner to referee conflicts and assign Versions.

Major Versions MAY be non-backwards-compatible with previous Versions.

Minor Versions SHALL be interoperable with all Minor Versions within the same Major Version.

Fixes are intended to address obvious mistakes in an Artifact. A Fix may or may not be compatible with previous versions. Fixes are intended only for obvious mistakes in an Artifact. For example, when there is a typo that prevents an interface from working, or when an Artifact is clearly incorrect and does not fulfill the requirements of a published standard that created the Artifact. This mechanism should be used very sparingly. It is up to the discretion of the Owner to determine whether it is reasonable to issue a Fix. For example, if a Fix is introduced within a short time of a standard being published that produced the Artifact, it is reasonable to issue a non-backwards compatible Fix. If a Fix is required after an Artifact is published and there is wide availability of services that match the published version of the Artifact, even if that Artifact has obvious mistakes, it is probably not reasonable to issue the change as a Fix, and the change should probably be made in a Major Version or Minor Version. It is the responsibility of the Owner to determine what is "reasonable" under the guidance in this section.

Typically the Owner will be the workgroup that created the repository. Fixes SHALL be handled through the NENA errata process (see NENA-ADM-001) [3].

8 Abbreviations, Terms, and Definitions

See the NENA Knowledge Base [1] for a Glossary of terms and abbreviations used in NENA documents. Abbreviations and terms used in this document are listed below with their definitions.

Term or Abbreviation (Expansion)	Definition / Description
Artifact	A file containing formal interface definitions, schemas, documents, examples, scripts, or other content that is related to a standard and used by developers to implement the standard. Some Artifacts may be included by reference in a standard. Within NENA, Artifacts are stored in a Git repository. Groups of Artifacts often describe an Application Programming Interface (API).

9 References

[1] National Emergency Number Association. "NENA Knowledge Base Glossary." Updated June 16, 2022. https://nenawiki.org/wiki/Category:Glossary.

[2] Internet Engineering Task Force. *Key words for use in RFCs to Indicate Requirement Levels*. S. Bradner. RFC 2119, March 1997.

[3] National Emergency Number Association. *NENA Development Group Organizational Structure and Operational Procedures*. NENA-ADM-001.5b-2021. Arlington, VA: NENA, approved July 19, 2022.

[4] Github.com. "About Github". Accessed March 17, 2022, https://docs.github.com/en/get-started/using-git/about-git.

[5] Git Project. "About Git". Accessed March 17, 2022, https://git-scm.com/.

[6] Internet Engineering Task Force. *Guidelines for Writing an IANA Considerations Section in RFCs*. M. Cotton, B. Leiba, and T. Narten. RFC 8126, June 2017.

ACKNOWLEDGEMENTS

The National Emergency Number Association (NENA) Working Group, Data Structures Committee, Interface Specification Development Working Group developed this document.

NENA recognizes the following industry experts and their employers for their contributions in the development of this document.

Members	Employer
Brooks Shannon, Data Structures Committee Chair	NENA

Members	Employer
Brandon Abley, ENP, Interface Specification Development WG Co-Chair	NENA
Dan Mongrain, Interface Specification Development WG Co-Chair	Motorola
Roy Artin	E-Comm 9-1-1 Vancouver
Pat Blair	GeoComm, Inc.
Victor Burton	Comtech
Jeremy Crabtree	North Central Texas Emergency Communications District (NCT9-1-1), TX
Lihong Deng	Zetron Inc
Tony Dunsworth	City of Alexandria, VA
Simon Farrow	Stancil Corporation
Randall Gellens	Core Technology Consulting
Mike Hooker	T-Mobile US, Inc.
Tom Hsu	Nokia
Rafal Kaminski	Motorola Solutions, Inc.
Ryan Keeling	Bell Canada
James Kinney	INdigital
Andreas Liebig	GridGears
Patrick Malone, ENP	Comtech
Juan Pedro Martinez	Versaterm, Inc.
Richard Maw	RAM Consulting
George Michallas	MicroAutomation, Inc.
Darrin Morkunas	Intrado - Life & Safety
Stephen O'Conor, ENP	Next Generation 9-1-1 Consulting Services, LLC
Michael Proestler	GridGears
Larry Reeder	Bandwidth
Theresa Reese	Ericsson, Inc.
Wendi Rooney, ENP	Spartanburg County, SC
Brian Rosen	Brian Rosen Technologies, LLC
Jim Shepard, ENP	Motorola Solutions, Inc.
Joe Ben Slivka	Motorola Solutions, Inc.
Michael Smith	Equature/DSS Corp.
Henry Unger	Pulsiam
Michael Vislocky	Network Orange, Inc.
Jeffrey Wheeler	Data Technical Services

Special Acknowledgements:

Delaine Arnold, ENP, Committee Resource Manager, has facilitated the production of this document through the prescribed approval process.

The Document Management Working Group is part of the NENA Development Group that is led by:

- Wendi Rooney, ENP, and Jim Shepard, ENP, Development Steering Council Co-Chairs
- Brandon Abley, ENP, Director of Technology
- April Heinze, ENP, 9-1-1 and PSAP Operations Director