



OSEHRA Forum Configuration Nomenclature

**Version 1.0
March 17, 2014**

©2014 Open Source Electronic Health Record Alliance, Inc.

OSEHRA Forum Configuration Nomenclature by

Frederick D. S. Marshall

is licensed under a

Creative Commons Attribution-ShareAlike 4.0 International License.

Revision History

Date	Version	Description	Author
3/27/14	1.0	Accepted document	Marshall

Table of Contents

1	Purpose	4
2	Executive Summary	4
3	VistA Dialect · Software Life Cycle · Upgrade Stream	4
3.1	VistA Dialect	4
3.2	What Your VistA Dialect Tells Us.....	6
3.3	Software Life Cycle	7
3.4	Upgrade Stream	7
3.5	New Dialects	7
4	VistA Version · Snapshot or Service Pack	8
4.1	VistA Snapshot	8
4.2	VistA Service Pack	8
4.3	Naming Snapshots and Service Packs.....	9
5	VistA Applications · Packages, Versions, & Patches	9
5.1	Applications	10
5.2	Versions & Packages.....	10
5.3	Patches.....	10
5.4	Patch Names	10
5.5	Distribution Names	11
6	Mappings · Streams, Secondary Development, Manifests	11
6.1	Mapping Patches to Software Streams	11
6.2	Mapping Original & Derived Software Streams.....	11
6.3	Mapping Original & Derived Patches	12
6.4	Manifest Section 3 · Application Versions	12
6.5	Manifest Section 2 · Deltas	12
6.6	Manifest Section 1 · Highlights	13
7	Conclusion	13
8	Basic Patch Nomenclature · Names, Subjects, Files	14
9	The Variety of Patch E-mail Subject Formats	15
10	Patch Listings	15

1 Purpose

This document surveys the scheme and nomenclature to be used in assigning version, patch, and sequence numbers to OSEHRA VistA.

2 Executive Summary

How do you answer the question “What version of VistA are you running?” In three ways:

- 1) You name your VistA dialect, which identifies the gold codebase your version of VistA is based on, and which also identifies the stream of upgrades you follow in evolving your VistA system.
- 2) If your VistA system is brand new, you name the VistA snapshot it was created from; otherwise, you name the most recent service pack you applied.
- 3) Optionally, for more detail you can describe the version and patch level of one or more of the VistA applications you are running; the easiest way to do this is to consult your snapshot’s or most recent service pack’s manifest.

3 VistA Dialect · Software Life Cycle · Upgrade Stream

VistA has a complex lineage going back thirty-seven years across many organizations (see The Lineage of VistA chart on the next page).

3.1 VistA Dialect

When a VistA adopter produces a unique, stable version of VistA, upgrades it consistently over time, and continually releases those upgrades to the rest of the VistA community in a complete stream of updates, we call what they have created a VistA dialect. When someone asks “What version of VistA are you running,” often all they want to know is which dialect you are using.

THE LINEAGE OF VISTA

FRIDAY, 24 JANUARY 2014

Defunct dialects
Troubled dialects
Proprietary dialects
Incomplete dialects
Open & Healthy Dialects



© 2014, Carol Monahan & Frederick D. S. Marshall

VISTA Expertise Network's *The Lineage of VISTA* is licensed under the *Creative Commons Attribution-NonCommercial-Share Alike 3.0 United States License* (<http://creativecommons.org/licenses/by-nc-sa/3.0/us/>).

Some VistA dialects have been replaced, some are defunct, some are so unsupported as to be problematic to use, and still others are too incomplete to use, but six or so are production ready.

OSEHRA is creating a seventh—OSEHRA VistA. Unlike the other dialects, OSEHRA VistA is not intended to replace the other dialects but to be a converged common core code base from which the other dialects can be derived, to improve the VistA community's productivity by increasing software sharing and decreasing reinvention of the wheel.

3.2 What Your VistA Dialect Tells Us

We care about the dialect as an answer to the question “What version of VistA are you running?” because your dialect tells us four crucial things.

First, it tells us which constellation of VistA applications you are running. Although there is a large overlap among dialects, the differences in functionality are easiest to summarize in terms of which applications you have or do not have, and this is determined mostly by your choice of dialect—and of course the actual services your hospital provides and needs automation for. If we know your VistA dialect, we know the basic pool of applications you are drawing from.

Second, it tells us about your primary source of innovation, because different dialects are primarily developed by different organizations. When you adopt a VistA dialect, you become partly dependent on its development organization, at worst wholly dependent, at best interdependent. Much like life itself, VistA is far, far too complex to go it alone; dependency and interdependency are necessary parts of adopting and developing an EHR. Your fate is therefore determined at least in part by your relationships, especially by who you choose to be your primary development organization, that is, by who primarily develops your chosen VistA dialect.

Third, it tells us about your rate of innovation, the pace at which your applications improve and new applications are added. Different primary development organizations follow different software life cycles, which results in different rates of productivity. More specifically, it also tells us about the rates of innovation for different applications, since different organizations have different priorities; an organization with an otherwise lethargic development pace might be unusually good with one or more applications you care about.

Fourth, it tells us about your ability to exchange data and software with other VistA adopters, a crucial consideration.

In terms of data, the world is leaving behind the era of medical feudalism, in which hospitals owned patient and treatment data, and moving into a new era of medical-data interchange, in which patients own their medical and treatment information. Despite all the past, present, and future standards for medical-data exchange, it remains true that systems with more compatible data structures have an easier time representing each other's data. If we know your VistA dialect, we know its pedigree, so we know how compatible its data structures are with our own, and thus how effectively we can exchange data.

In terms of software, since we cannot go it alone with VistA, we have to divide up the development work. Different organizations focus on different VistA applications. The more easily we can exchange software, the more easily we can benefit from each other's innovations and accelerate our software's evolution. If we know your dialect, then we know your software architecture and its compatibility with our own, so we know how easily we can exchange software, and thus how much we can help each other out.

3.3 Software Life Cycle

These four things your VistA dialect tells us are important, but they become more important when viewed through the lenses that VistA experts use. The pace of change in medical science and practice is staggering, and it destroys the validity of the ways most of us go about shopping for EHRs. In such a complex field, the hardest principle for non-experts to grasp is the extent to which the software's current capabilities do not matter as much as its future capabilities. If it is wrong today, it can become right tomorrow; if it is right today, it can become wrong tomorrow. Whether it is right or wrong tomorrow is determined not by the software itself but by the pace, direction, and quality of innovation; these are the pre-eminent concerns a VistA expert has when evaluating VistA dialects.

Fortunately, this other system of evaluation is also captured by knowing which VistA dialect you have. The primary developers of the VistA dialects are different organizations, each of which follows its own software-life cycle practices, which in turn results in higher or lower productivity, more or less collaborative relationships with their adopters, and above all a greater or lesser degree of responsiveness to the shifting understanding of their medical experts. That latter quality determines how evidence-based a development organization's software life cycle is, and thus how reality-based its resulting software will be. We do not need to know in advance where specifically IHS, for example, will innovate in their VistA dialect, so long as we know that innovation will be driven above all by their medical users.

3.4 Upgrade Stream

VistA innovation comes to us in the form of a stream of upgrades, as will be described in the rest of this paper. What matters foremost about your upgrade stream is that (a) most of your future innovations will come from it, (b) your upgrade stream is specific to your VistA dialect, (c) your upgrade stream is created by your primary development organization, and (d) your upgrade stream is creating using the software life cycle selected by your primary development organization. Thus, knowing your VistA dialect tells us a great deal about your future.

3.5 New Dialects

When a new VistA dialect is created (rightly, a rare event), its adopters cannot describe their version of VistA just by naming it, because no one will have heard of it or know what it means. It must be explained in terms of existing dialects. The dialect used as the original code base should be named, its main differences from that code base should be described, any borrowings from other dialects should be mentioned to help establish its relationship to them, the primary development organization should be named, that organization's software life cycle should be described, the source of the upgrade stream should be named so we know where to go for updates, and any coherent plans for the future of this dialect should be described.

For example, OSEHRA VistA was created to solve a problem that no existing VistA dialect can solve but that they all need solved—reunification of the VistA dialects, to be a repository for the community's code-convergence efforts. It is derived from FOIA VistA, mixed with contributions from outside VA aimed at improving the core codebase and converging our dialects as closely as practically feasible. It borrows from every healthy dialect, the primary development organization is the OSEHRA community collaborating within OSEHRA, and the sources of the upgrade stream are the OSEHRA VistA Github repository for snapshots and OSEHRA Forum for patches (see next section). Plans for the future are driven by the OSEHRA community within its work groups, which meet and discuss plans openly; everyone is invited to participate.

Now that OSEHRA VistA exists, its adopters can use it to help answer the question “What version of VistA are you running?” We can say “OSEHRA VistA” or more typically in days to come we can say we are running XXX VistA, “derived from OSEHRA VistA.” Likewise, an adopter might say they are running “IHS RPMS” or “VA VistA,” and these are equally powerful summations of a great deal of information about the version of VistA they are running.

4 VistA Version • Snapshot or Service Pack

Knowing the dialect tells us a lot, and it is often all the questioner wants to know, but sometimes it is not enough. Dialects change over time. Unless we know where in time you are within your dialect, we still do not know which version of VistA you are running. When we need to know more than the dialect, usually we want to know the snapshot or service pack. Here’s why.

VistA is among the most complex works of software ever created; it has to be, because medicine is among the most complex of human endeavors. It is too complex to install the way most software is installed, and it is too complex to upgrade the way most software is upgraded.

4.1 VistA Snapshot

VistA is installed by cloning a living VistA system. The same methods we use to keep production VistA systems up to date, we also use to keep each VistA dialect’s reference system up to date. Periodically, we make a copy of that system, what we call a snapshot of it, and that copy is published. In the case of the U.S. Department of Veterans Affairs (VA) and Indian Health Service (IHS), which are federal agencies, the creation and publication of these VistA snapshots is a function of their duties under the Freedom of Information Act (FOIA). Other organizations produce snapshots to support their commercial services or charitable missions.

Each organization produces snapshots on its own rhythm, some monthly, some quarterly, some inconsistently. Time is the right way to identify these snapshots, because VistA is upgraded over time; knowing when a snapshot was taken gives us a pretty good idea which upgrades it includes and which it does not. After knowing the dialect, this is the closest thing there is to knowing what version of VistA you plan to install to set up a new VistA system.

4.2 VistA Service Pack

Unfortunately, VistA cannot be upgraded the same way it is installed. One of the big differences between VistA and most consumer software is that the software and data are not separate in VistA; they are and must be mingled very closely together, intertwined, to allow the complex medical data to drive the execution of the software. Most consumer software is upgraded mainly by replacing the software files with new and improved software files, but to do that to VistA would not only replace the software but also all of the data, erasing all of the patient and other medical information you have so carefully recorded. So once a VistA system has been created, new VistA snapshots are close to useless to that system thereafter; we never use new snapshots to upgrade an existing VistA system.

VistA instead uses a more biological system of upgrades, almost viral, in which small packages of new software and data are delivered to your VistA system using special VistA software, which then installs the update in a way that allows it to properly weave itself into your system. For historical reasons, we call these installation packages “patches,” but they are used as much for delivering enhancements as they are for delivering repairs. VistA’s system of patching is

intricate, precise, and effective, but it is too complex to let us give a short answer to the question “What version of VistA are you running?” For this reason, outside VA most primary development organizations periodically bundle up all their recent patches into service packs, to simplify their release and installation.

Typically, a service pack will include all the patches since the previous service pack that have passed the organization’s certification processes. Whenever the service pack covers a period during which the patches and/or versions of a major multi-application development initiative were released, it is important that the service pack include all of the patches and releases that together deliver that upgrade. We never want to split a major release of interdependent patches across two service packs, because we must ensure that a VistA system is architecturally coherent after installing each service pack.

Unlike snapshots, these service packs can be used to upgrade an existing VistA system, and they are, so identifying the most recent service pack you have installed is another very good way to explain which version of VistA you are running.

4.3 Naming Snapshots and Service Packs

Snapshots and service packs are almost always identified by the time they were taken, such as the FOIA VistA September 2009 snapshot. OSEHRA will create snapshots and service packs four times a year, once per quarter, and they will be named accordingly: OSEHRA VistA 2014Q1, OSEHRA VistA 2014Q2, OSEHRA VistA Service Pack 2014Q1, OSEHRA VistA Service Pack 2014Q2, and so on.

Every so often, a primary developer releases emergency patches that should not wait until the next quarterly service pack to be distributed. In such cases, OSEHRA will release emergency service packs whose names are derived from the most recently released quarterly service pack, such as OSEHRA VistA Service Pack 2014Q1A.

So for new or proposed VistA systems, the best answer to “What version of VistA are you running?” is to name the VistA dialect and snapshot; for established VistA systems, it is to name the VistA dialect and most recent service pack.

5 VistA Applications • Packages, Versions, & Patches

These ways of describing your VistA system’s version are clear and concise, and they will do for most casual situations—certainly they are the best for casual conversation or executive summaries—but they are not precise, and they were not always available. Until recent years, we used a system that is far more complex—too complex to be a polite or casual answer to the question “What version of VistA are you running?”—but far more precise. Even today, VistA software engineers and architects can and should begin with the answers described above, but they soon need to drill down to the more detailed answers we have used for thirty-seven years.

Those answers involve describing the version not of your VistA system as a whole but instead of each of the VistA applications you are running on your system. We describe those versions in three ways: (a) by identifying the applications, (b) by listing the current version of each application, and (c) by naming the most recently installed patch to each application.

5.1 Applications

Each VistA application automates part or all of one hospital or clinic service. Just as different clinics and hospitals offer different suites of services, so their VistA systems will run different applications. Listing the applications running within a VistA system tells us a lot about it, and about the adopter it belongs to.

5.2 Versions & Packages

Major upgrades of applications are called versions, and the distributions of application versions are called packages. When an application is new, that first package is called version 1. Thereafter, new versions may be numbered by integers or decimals. A detailed accounting of the version of VistA you are running therefore requires identifying not just the list of applications but the current version of each one running on your system.

With applications like Laboratory that have not had a new version in nineteen years, knowing the version may not tell us much, because all modern VistA systems are running the same version. But with applications like File Manager, which just saw the release of new version 22.2 last year, knowing the version tells us a great deal about the differences between VistA systems.

Some applications have client software that is versioned independently of the main application's version. For example, VistA's flagship application, the Clinical Patient Record System (CPRS), designed for doctors and other healthcare professionals, has not had a new version in over a decade; it remains stuck at version 3, while its client is approaching version 30. Therefore, we also need to know the version of the client, if there is one, not just the main application.

5.3 Patches

As described above, most upgrades to VistA applications are smaller than complete new versions; most take the form of "patches," small, frequent, incremental upgrades to the application. Patches are often developed in parallel, but they are released sequentially, resulting in a stream of upgrades, the patch stream.

Since patches are almost always installed in order, knowing the most recently installed patch for each application gives us a very precise understanding of the version of VistA running on your system. Therefore, the most precise way to answer the question "What version of VistA are you running?" is to list all the VistA applications you are running, the version number of each, and the latest installed patch for each application.

5.4 Patch Names

Each version of each application has its own patch stream. That is why each patch is identified by (1) the application, (2) the version, and (3) a numeric ID. To this we add a fourth piece of information, (4) the sequence number, which tells us in which order to install the patches within its patch stream. Patches from VA VistA and FOIA VistA follow this pattern, with names like OR*3.0*283 SEQ #272 or DI*22.0*169 SEQ #149.

OSEHRA VistA patches are different, as are patches from any other VistA dialect if they are released from OSEHRA's new Forum system. These patches add a fifth piece of information to patch names: (5) the name of the VistA dialect. To help avoid confusion with VA patches, we also numberspace the patch's numeric ID. Each VistA dialect has its own range of numbers used to assign this value; OSEHRA VistA numeric IDs start with 10,000. So, for example, the first

patch to be released for OSEHRA VistA's File Manager version 22.2 will be called "OSEHRA VistA DI*22.2*10001 SEQ #1."

5.5 Distribution Names

Patches are distributed by e-mail message or by a pair of host files. Message subjects and file names are each derived from the name of the patch they distribute.

Until now, a patch like LR*5.2*413 SEQ #326, if distributed as an e-mail message would have a subject like "Released LR*5.2*413 SEQ #326," if distributed as host files would have names like "LR-5P2_SEQ-326_PAT-413.TXT" and "LR-5P2_SEQ-326_PAT-413.KID." The occasional emergency patch is different only in the e-mail subject line, which would read, for example, "EMERGENCY Released LR*5.2*403 SEQ #319." Likewise test patches, such as "TEST LR*5.2*438 TEST v1."

Distributions from OSEHRA's new Forum system will be just a little different. For example, patch "OSEHRA VistA DI*22.2*10004 SEQ #3," if distributed as an e-mail message would have a subject of "[OSEHRA VistA] Released DI*22.2*10004 SEQ #3," if distributed as host files would have names of "DI-22P2_OSEHRA-VistA_SEQ-3_PAT-10004.TXT" and "DI-22P2_OSEHRA-VistA_SEQ-3_PAT-10004.KID."

6 Mappings · Streams, Secondary Development, Manifests

When we move beyond dialect, snapshot, and service pack, when we need more information, part of what we need is a mapping between these things and patches, and among the patches themselves. It can be a complicated dance, so a little mapping can help simplify things for us by connecting the dots.

6.1 Mapping Patches to Software Streams

Every patch needs to be mapped to its software stream, to avoid any possible confusion about which patches a VistA adopter should or should not install on their system. This is done (a) in the patch name and distribution subject or file name, as described above, (b) in each patch's descriptive text, where the VistA dialect it applies to is named, and (c) in the database itself, to support the distribution and installation process.

6.2 Mapping Original & Derived Software Streams

Some software streams, like VA VistA, contain exclusively primary development, meaning the source of every patch in the stream is the primary developers of each application. This is called an original software stream.

Others, like IHS RPMS, contain a mix of primary development and secondary development, in which the source of many of the patches is some original software stream, but the source of others consists of additional development. This is called a derived software stream. In the case of IHS RPMS, the original stream on which it is based is VA VistA. When the additional development consists of new applications, it is called primary development; when it consists of modifications of or extensions to the patches in the original software stream, it is called secondary development.

The resulting IHS RPMS software stream is more complex than the VA VistA stream upon which it is based, because it contains primary development from VA developers, new primary

development from IHS developers, and secondary development from IHS developers, all carefully interleaved to create a new stream of upgrades. This interleaving usually requires renaming and resequencing the patches—even the unchanged ones—to make room for the new patches inserted between them.

For this to work, the relationships between original and derived software streams must be mapped out. Mainly this is done in the database itself, to support distribution and installation.

6.3 Mapping Original & Derived Patches

Just as the streams need to be mapped, so do the individual patches themselves, so we are clear on the relationship between them. Patches derived from an original stream by renaming and resequencing but otherwise not editing them need to be mapped back to their original patches. Patches derived from an original stream by doing secondary development upon them likewise need to be mapped back, though their more distant relationship needs to be distinguished from mere renaming and resequencing. New primary-development patches that need to be inserted between patches that were adjacent in the original stream need to identify their dependencies upon the original patches, to support proper sequencing.

All three forms of mapping between patches in the original software stream and patches in the derived stream is done (a) in the patch description's header text and (b) in the database itself. This ensures that both human beings and the software itself can trace the provenance of all patches in the derived software stream.

6.4 Manifest Section 3 • Application Versions

To map the high-level description of a VistA version in terms of dialects, snapshots, and service packs to the detailed description in terms of applications, versions, and patches, we need to ensure that each snapshot and service pack includes a version manifest. Each manifest will include three sections, which we will describe in reverse order.

The third section describes the state of the applications. It is a list of all of the VistA applications included in the snapshot or service pack, the version of each application (and of its client, if any and if different), and the latest patch installed for each application. It gives the detailed answer to the question “What version of VistA are you running?” as described above. This list can be automatically generated by the Forum system when a snapshot or service pack is created.

6.5 Manifest Section 2 • Deltas

The second section describes the changes (deltas) in this snapshot or service pack compared to the previous one. It is a list of the new applications, new versions, and new patches that together created this version of VistA from the previous version. For all patches in a derived software stream, it also lists the corresponding original patch, to assist technical experts in mapping this snapshot or service pack back to the version of the original software stream upon which it is based. For a derived software stream, it also lists any original patches excluded from release in this software stream, as sometimes happens. It is indispensable to VistA adopters preparing to apply a service pack or comparing two adjacent snapshots, because it details the differences between them. This list too can be automatically generated by the Forum system when a snapshot or service pack is created.

6.6 Manifest Section 1 • Highlights

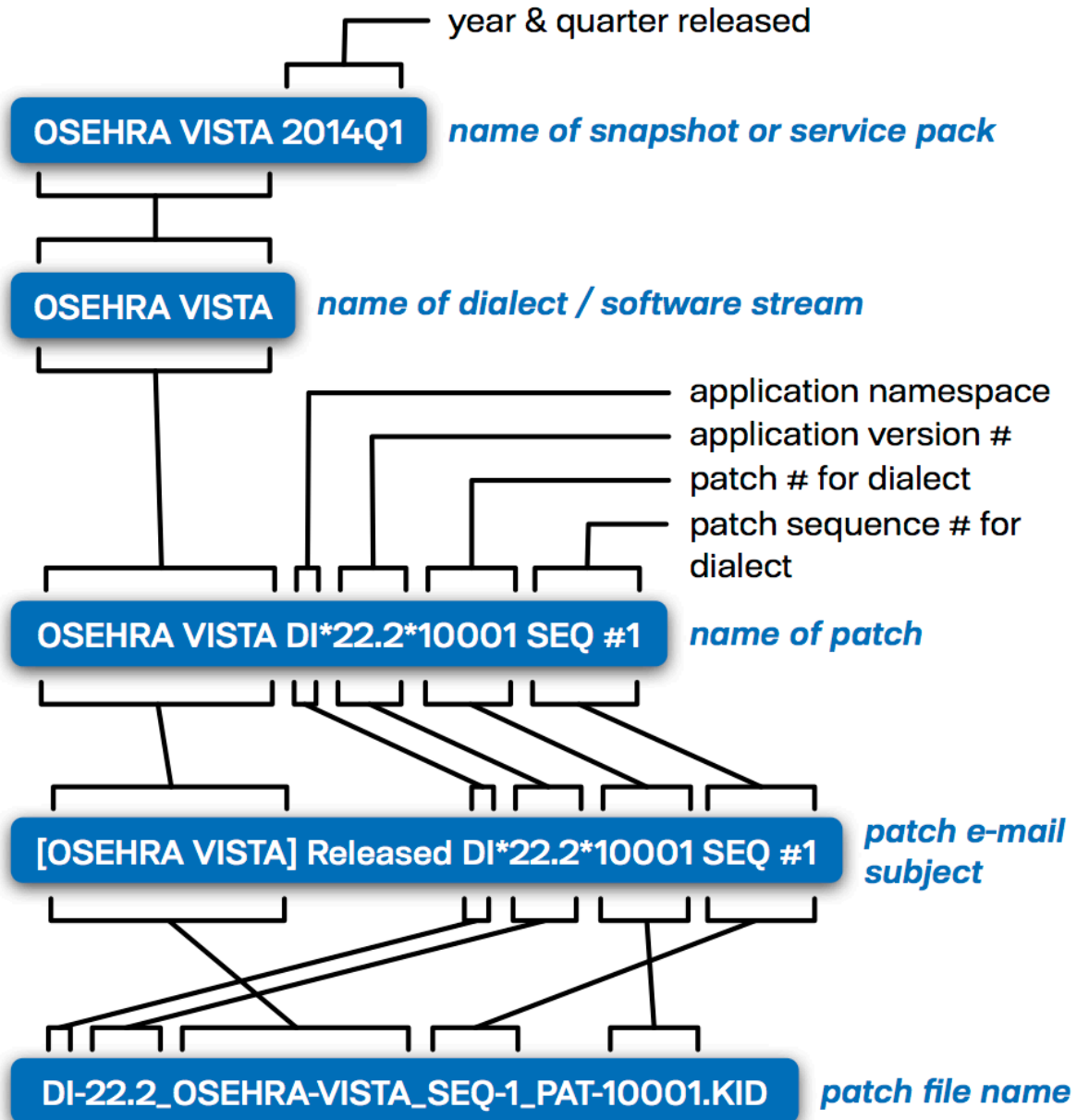
The first section is an executive summary that picks out just the most essential or interesting versions and deltas included in this snapshot or service pack. It should always include the version numbers of indicator packages, such as the CPRS client, which drives a lot of other VistA development, or File Manager, which enables it. It answers the related question “Why am I most likely to care about this version of VistA?” Most people who read a snapshot or service pack’s manifest will only read this first section, but VistA software engineers and superusers will need to study all three sections in the performance of their jobs. This short list must be generated manually by experts who understand the significance of the other two sections.

7 Conclusion

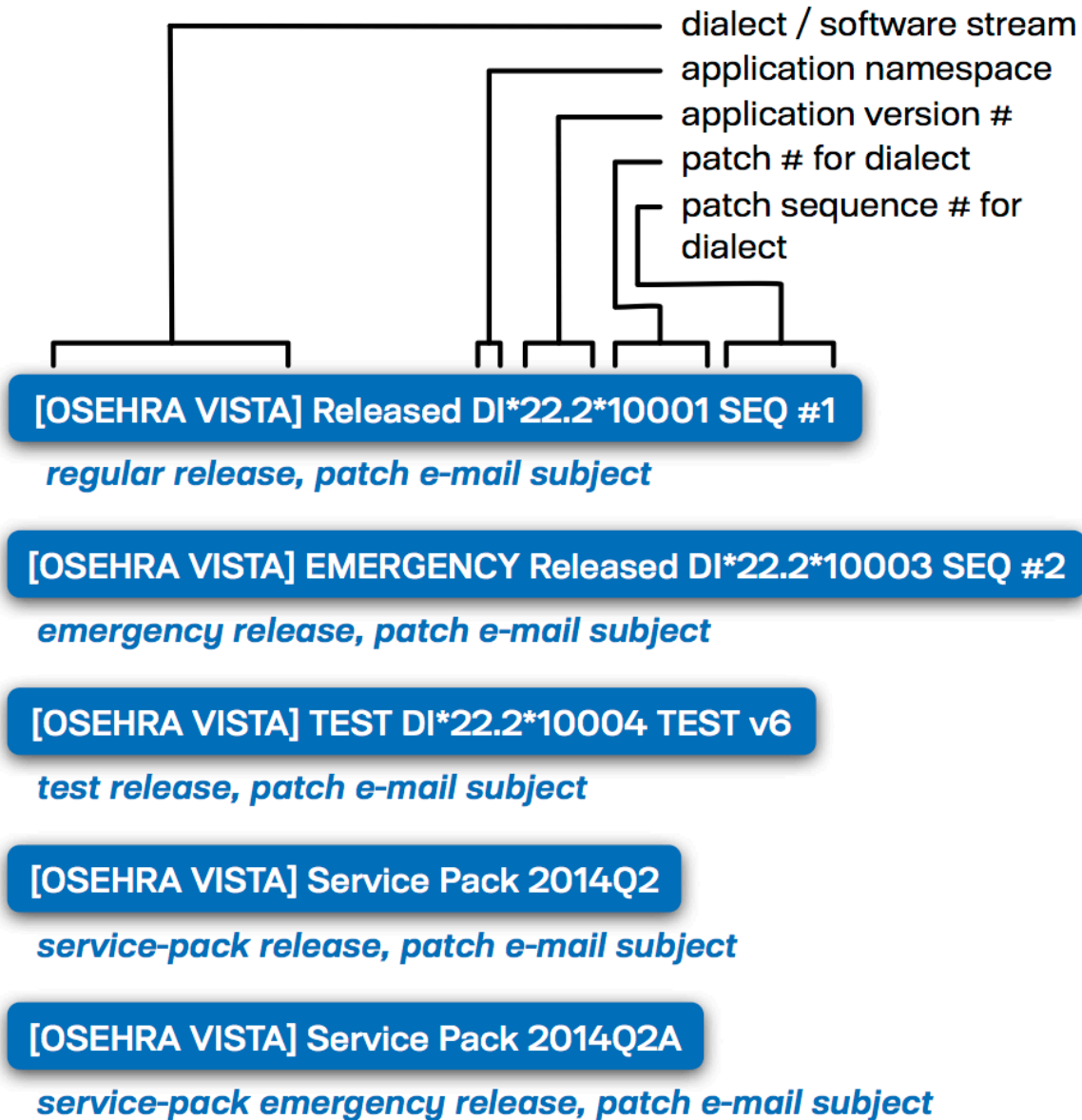
The scheme and nomenclature to be used in assigning version, patch, and sequence numbers to OSEHRA VistA, described in this paper, is an extension of the systems currently in use in VA, in IHS, and outside the U.S. federal government. The main differences are that this system (a) makes it easier to summarize your VistA system’s version, (b) allows you to map that summary to a precise reckoning of the details of your VistA version, and (c) involves new automated support being created as part of OSEHRA’s Forum open-source working group.

We encourage the OSEHRA community to master and adopt this new, extended nomenclature to make it easier for us all to talk about our VistA systems with each other.

8 Basic Patch Nomenclature · Names, Subjects, Files



9 The Variety of Patch E-mail Subject Formats



10 Patch Listings

record #	name	stream	subject	status	developer
80290	PRCA*4.5*10261	[OV]	FIX <NOLINE> ERRORS	IN TH	
	derived from [FV]PRCA*4.5*261				

Every patch stream has an abbreviation, such as OV for OSEHRA VistA, or FV for FOIA VistA, used in condensed displays like patch listings.