VISTA Community Meeting
Day Three: January 16, 2011
Afternoon Session

Topic: FORTHCOMING CHANGES TO THE LABORATORY PACKAGE

Presenter: Rick Marshall, VISTA Expertise Network

Note: Throughout the weekend, a group of VISTA hardhats with expertise in the Laboratory package had been meeting to discuss future versions and enhancements. Mr. Marshall's presentation outlines the results of those meetings.

VISTA Lab hasn't had a new release in 15 years. If VA were following a traditional VISTA model, there would have been at least 7 new releases during that time. Development on Lab has been stagnant, due in large part to policies at VA.

Mr. Marshall believes that there needs to be one VISTA package developed to a new version and released, so that VA is actually behind and there is some pressure for them to catch up. Even if they never adopt it, it will improve the dialog, and it gives developers outside VA a chance to try new things. Lab was an obvious choice to be the first package developed in this way.

Moving VISTA outside the federal government—into hospitals and clinics and so on—involves a radical change of perspective about what the Lab data is doing in the system. It's the distinction between a Health Information System (HIS) and an Electronic Medical Record (EMR). A hospital has to be cautious about chain of custody, about where exactly data came from. If, for example, a patient walks in with a hard copy of lab results, an HIS and an EMR will respond differently. An HIS will be paranoid, and treat these results as something that may or may not be information; they will not be integrated with the patient's record. Doctors need the opposite. They're trying to compile a record; they can't afford to leave anything out.

A lot of things that have to change in moving Lab out of the VA pertain to this difference.

The first step is to identify settings. VISTA identifies some settings, but they are not necessarily the ones that are common outside VA. Specifically, most doctor's offices and small clinics use an external lab, and need an external lab interface. They also could use an improved point-of-care interface, but the team decided that was not as great a priority.

The Lab package covers two aspects of laboratory operations. The first is the "manufacturing" aspect of tracking the steps of the business logic from station to station; the second is recording the information. A doctor's office is concerned mainly with the recording aspect, but some of Lab's business rules and processes are concerned solely with the manufacturing aspect.

Q: Could you elaborate a little on your Inbound Interfacing (on the Power Point slide)? What do you mean by "one-way?"
A: Inbound interfacing can work one of three ways. The first is the "reliable" method, although anybody who's done HL7 interfacing knows that the fields can be interpreted differently. The most reliable scenario is that the outbound interface was automated, and the results came back with the identifiers intact so they could be filed correctly. That's round-trip. It's less reliable if it comes back without the same identifiers and the identification has to be done manually. Worse (and this is

really common) is the scenario where the outbound is a piece of paper. Even if you get a result electronically, the odds are that none of the identifiers are going to make it from the piece of paper into the results. There's a lot of interpretation that will need to get done to safely link the results back to the order for CPRS's purposes, which is what we mean by "one-way." Iffiest of all is when the results come back on a piece of paper, and are transcribed by hand.

In outlining these rules and processes, the group was trying to identify where they should be focusing their efforts. Predictably, the focus points are the problematic ones: cases that are uncommon in VA but very common outside it.

With this as a framework, the focus for the new version of Lab is going to be the Lab data file. The current file is an old structure that has been patched all to heck; it works, but it's kludgy. There will be some work around it, but file 63 is the heart and soul of the new version. It's the driving force.

However, they don't just want to swap out file 63 with a new one and hope for the best. They plan to introduce a new lab data file in which the data elements are being populated, by cross-references and other logic, off of the original file 63. As file 63 is updated, it will keep the new one up-to-date. That buys time to look at the files and ensure that they have the same information.

Then, they will introduce Lab version 6.1, in which the code is converted to point to the new file. Once that is done, version 6.2 will phase out the old file.

Q: So you have data that's going into the old form, and data that's going into the new form. If there's data going into the new form, you have to have some code to put it in there.
A: No, the data always goes into the old file. And then the old file updates the new file. The new file will mirror the lab data file until we're satisfied that we've got it.

Q: There will be cases where the new file knows things that the old file doesn't.
A: That's exactly correct. This is misleading, isn't it? Even 6.0 does more than add the new file and the cross-references; it also involves code changes so that the new information that the new file can know gets put into it. But it's the conversion of the existing logic that's the change in 6.1.

Q: Have we decided where the new data file is going to be?
A: No; right now it doesn't matter. We're saying that any decisions that can be made later, we do. That way, we can focus on what needs to get done now. Exactly what the number will be? Who cares?

The next thing is to apply real rigor to the sorting process of what features to include. There is now a dividing line: after version 6.2 or before. Everything up to 6.2 is what's required to fully bring the new Lab file online. Anything needed for that, and anything that is essential to be done right now, is put on the critical list. Anything that's an improvement not needed for the new file or for Meaningful Use is going to have to come after version 6.2. This includes, for example, the new point-of-care interface and transaction processing.

Q: Quick comment: to the extent that you are able to handle variations between MUMPS systems, one of the important things you can do on GT.M is to use triggers. It will simplify the application logic, and keep the cross-references and the application logic separate.
A: As you'll see on one of the later slides, we're going to do the whole project out in the open. All the team discussions and everything will be where everyone can read them. We're going to start a discussion forum for Laboratory version 6 on Mumpster. Once we do that, I'd love it if you

opened a thread about Lab transaction processing where you talk about triggers.

People complain about the weirdness of the files of the data dictionaries in Lab, but they shouldn't. It's not Lab's fault. Lab needed to do things that File Manager did not support. So, using the other magic of Fileman, they brought MUMPS into play, and they coded around the limitations of File Manager. Thus you end up with weird things like defining every lab test as a field in the a dictionary, and defining a field in another file as a pointer in a data dictionary, even though Fileman can't point to a data dictionary. Or rather, Fileman doesn't know he can point to a data dictionary. They actually lied to Fileman and told him he was pointing to another file, but then they went in and hard-coded the global, so he's not pointing where he thinks he is. And this is not the weirdest thing in the data dictionaries; there's lots of stuff like that.

To undo the weirdness in Lab, the new design for this file needs several new features. The first is crypto-pointers, like that data dictionary example, or something that says it's a pointer, but stores free text instead of a pointer value. There's also a free-text pointer (live or archiving) and super variable pointers. Optional keys are needed because entries with internal data can use it as a key, but not all entries will have that data. New data types are also needed: extensible data types, variable types, and compound types. Triggers need to be re-done as new-style cross-references. Finally, the new design needs URI pointers and subfile pointers. It's a daunting list, but about half of these features were already prototyped for Fileman 22, and the Fileman folks have a pretty good idea what to do for the rest.

This brings up the next issue, which is that Lab 6 can't be done in one step. If Lab 6 is released as described, it will be paralyzed until the corresponding changes are made to Fileman. Instead of trying to tackle this as one humungous project, the team decided to do a series of micro-releases to bring Fileman and Lab along together. Each micro-release will introduce one new Fileman feature (or two if they're small), and a revised Lab structure that takes advantage of the new feature. Very little of this involves code, so it should not be difficult.

So, the team did not define one new version of Lab, or two, or three; they defined 10 new versions of Lab, but they're all micro-versions. They can therefore be put out rapidly. There is an ancient tradition of this in VISTA.

Q: Does anybody inside VA working on Fileman know about this?
A: No. More specifically, yes, but they're working at a slow pace. George Timson develops most of the actual changes, so he and I and some others are going to keep working on it, and hand it to VA, and they can ignore it or not. And I have to say, as bad as VA may feel about being one version behind on Fileman, they're going to get increasingly comfortable as they get two versions, three versions, eight versions behind the rest of the world. The sites will apply some pressure, although it may not do anything.

The group is putting off decisions until they have to be made, but they have decided on the first two steps. The first step is to clean up, document, and release George Timson's MSC Fileman as is, as version 22.3. That will set the baseline for the rest of the project. Then, they will document and re-integrate all patches to the current Lab, and release that as Lab 5.3. The reason to do that is not just to set the numbers. The reason is that you can't have a true software lifecycle unless you are fixing actual users' actual problems. Users have to be taught to believe that they have the right to request changes. With the first release, the team plans to "plant" people to report specific issues, which will be patched and released in the next version. In that way, a release cycle can be started, and users encouraged to report on what they need.

These releases will also need infrastructure to support the software lifecycle. Infrastructure items to be added include the National Online Information Sharing (NOIS) problem reporting process, a Special Interest User Group (SIUG), a primary development team, and various forums.

The new version of file 63 will be introduced in Lab 5.4, supported by new features in Fileman 22.4.

Mr. Marshall moved on to the team's to-do list for what they want to accomplish between now and Lab 6.2. In addition to the redesign of file 63, the team plans to fully integrate the patches that have come out since the old release, clean up the code, refactor it, and create new documentation. There are also plans to update the interfacing software with Laboratory Electronic Data Interchange (LEDI) 4, a rich Application Programming Interface (API) library, and a capacity for inter-facility orders. This last item is a potential source of funding for the project. Meaningful Use also needs to be addressed, as does the VISTA-office EHR (VOE) project. Finally, the Lab packages in VA VISTA, RPMS, and CHCS need to be re-unified, because some of these problems may already have been addressed.

Further requirements include a process for documenting business rules and processes, and a troubleshooting manual or wiki.