How to Cut the Time of Processing of Ryan White Services (RSR) Data by 99% and More.

David Izrael, Michael Costa, Abt Associates
Fizza S. Gillani, Brown University and Lifespan/Tufts/Brown Center for AIDS Research

ABSTRACT

The widely used method to convert RSR XML data to some standard ready-to-process data base uses a Visual Basic mapper as a buffer tool when reading XML data, for example, into MS Access data base. This paper describes the shortcomings of this method with respect to the different schemas of RSR data and offers a SAS macro that enables users to read any schema of RSR data directly into a SAS relational data base, thus avoiding the step of creation of MS Access data base entirely. Applying our macro user can cut the time of processing of Ryan White data by 99% and more depending on the number of files that need to be processed in one run**.**

INTRODUCTION

The three largest public funders of HIV care are the federally-funded Medicare and federal- and state-funded Medicaid entitlement programs, and the Health Resources and Services Administration, HIV/AIDS Bureau's (HRSA/HAB) Ryan White HIV/AIDS Program. At 2.1 billion dollars, the Ryan White program has the smallest budget of these three programs but provides HIV/AIDS services to approximately 500,000 of the most vulnerable individuals each year (1-3). Ryan White is designed to be a "payer of last resort" and specifically targets uninsured and underinsured HIV-positive individuals. Although not the largest public program for HIV care, Ryan White is the major source of care for recently released inmates because this population often lacks health insurance, including Medicaid (4). While this may change somewhat, under the Patient Protection and Affordable Care Act (ACA), such changes have not yet been realized. Starting in 2009, Ryan White funded agencies and their care providers began electronically submitting Ryan White Services Report (RSR) containing client-level data to HRSA/HAB on an annual basis as a condition of their grant awards (5). Data for each individual patient include demographics, major HIV lab test results, clinical and support service dates and referrals to other health and social services. Clients are identified by an encrypted unique identifier assembled from a patient's name, gender, and date of birth.

The RSR data are stored by organizations in XML format. There are a couple of schemas used to create RSR data. The general format of the data, which is an ASCII file, is presented in the Attachment. In general, there are the following sections in any schema of RSR data (highlighted in the Attachment 1 with different colors): Client Report (demographics data), Ambulatory Service, CD4 Test, HIV Risk Factor, Medical Insurance, Race, Service Delivered, Service Visits, and Viral Load Test. Some clients, however, can miss one or more of these sections. The format of presentation of the data, while is always an ASCII file, can differ in the format of some sections. For example, T-REX schema showed in the attachment has ambulatory visits in the following format

```
<ClientReportAmbulatoryService CLD_ID="1586">
    <ServiceDate>07,12,2011</ServiceDate>
      </ClientReportAmbulatoryService>
<ClientReportAmbulatoryService CLD_ID="1586">
    <ServiceDate>08,26,2011</ServiceDate>
      </ClientReportAmbulatoryService>
```

while Casewatch Millenium schema has the data lines combined between the section identifiers:

```
ClientReportHivRiskFactor CLD_ID="48937">
  <HivRiskFactorID>2</HivRiskFactorID>
  <HivRiskFactorID>3</HivRiskFactorID>
      </ClientReportHivRiskFactor>
```

The viral load section in T-Rex presentation looks like this:

```
<ClientReportViralLoadTest CLD_ID="586">
  <Count IsDetectable="false">0</Count>
  <ServiceDate>01,15,2013</ServiceDate>
     </ClientReportViralLoadTest>
```

while Casewatch Millenium schema does not have IsDetectable item at all. The Instruction for transferring the T-REX XML data to MS Access using the VB map script RsrClientXmlToMicrosoftAccessMap.xslt can be found at this link https://careacttarget.org/sites/default/files/file-upload/resources/T-REX_With_X-ERT_User_Manual_4.2.2.pdf. This is a manual process – one RSR XML data set by another (unless you are a VB expert and can automate the whole process). If you receive from the provider a couple of hundred of data sets you have to do the tedious work 200 times. This is a main challenge.

OTHER CHALLENGES

To adjust for different schemas to convert the xml format of the client level data into MS Access tables (which are to be converted to and processed by SAS), some changes to the Visual Basic script have to be introduced. What if you are an expert in the statistical programming but an amateur Visual Basic user (or not a VB user at all) and not able to find a quick adjusting solution to the VB code? But intricacies of VB code is not the last problem you may run into trying to read XML data into Access tables. Sometimes, we have to deal with very large XML files. In this situation, following the instruction at the above link, your transfer might just freeze and you may click several times to get things going and get the Access tables and, even if you managed to write Access tables, you should validate your process to make sure that the correct data was written.

The knowledgeable reader could, of course, say “Why not use the SAS XML mapper?” and would be right if we dealt with a simpler XML schema. Whatever RSR data we tried to map with SAS XMP mapper we got the error message and the log showed us the following message:

```
XMLMap parser encountered XML issue Tue May 3 13:15:53 EDT 2014
```

Clicking on this message the window comes up with the following information:

Exception class	org.xml.sax.SAXParseException
ID	
Message	schema_reference.4: Failed to read schema document 'RsrClientSchema.xsd', because 1) could not find the document; 2) the document could not be read; 3) the root element of the document is not found.
Line	2
Column	152

Apparently, for parsing the XML file (in CaseWatch Millenium schema, for example, shown here) SAS XML mapper expects the schema document 'RsrClientSchema.xsd' for which the header of XML data (the Attachment - pink highlight in the header) refers to a Web location. Obviously, this document cannot be found at this location and our RSR XML file fails to parse in XML mapper.

SOLUTION

We offer a SAS macro that is able to parse multiple RSR XML files using search and match SAS functions and distribute the appropriate sections of those files into the relational SAS data base, thus eliminating the necessity to use the intermediate data base like MS Access along with the pitfalls described above. It is worth mentioning that in addition to the difference in the schema of RSR data, the physical format of ASCII XML files can vary. For instance, in your editor (whether it is VEdit or Notepad++) you can see your data as a "classic" xml file (as in the Attachment) or as one long line of several millions characters. Our SAS macro is able to accommodate both of those formats.

The SAS macro searches for the sections of the RSR data highlighted in the Attachment. Then it converts the information located between the section boundaries into the one of the nine SAS data sets: Client_Report, Ambulatory Service, CD4 Test, HIV Risk Factor, Medical Insurance, Race, Service Delivered, Service Visits, and Viral Load Test. These nine SAS data sets jointly represent the relational database template and can be joined using the Client ID. If there are multiple XML files the data from every file is added to the previously read. Hence, the proposed SAS macro creates a longitudinal RSR data system using the yearly RSR. This will be very useful feature for the RSR data system in the long term.

DESCRIPTION OF THE MACRO

We assume that the RSR data sets to be processed into relational SAS data sets are located in a single folder. The RSR data comes, as a rule, by year and by site of creation. The researches, working with the data want, as a rule, use these two parameters as covariates, so the macro requires that the name of the XML files contain the name of the site and the year of creation separated by underscore; for example PHHS_2013.XML. The created by the macro sas data sets will have two additional variables: site and year of creation.

Different types of schemas are allowed to coexist in the XML-data folder - the macro is omnivorous. The following macro – %searchXML -reads all XML data sets from the specified folder and passes each XML file name to the parsing macro %Parse_RSR:

```
%macro searchXML(path);

filename filelist pipe "dir ""&path\*.xml"" /b";

data _null_;
infile filelist truncover;
input fileline $100.;
call execute (cats('%Parse_RSR(', fileline, ',',  "&path",    ')' ));
run;

%mend searchXML;
```

And here is an example of %searchXML macro call:

```
%searchXML(S:\Projects\LINCS\LA\RSR\TEST1 )
```

All horse work on parsing and distributing the data over SAS relational data sets are done by the macro %Parse_RSR. The macro requires two input parameters: a) full path of an RSR XML file; b) libname of the folder where SAS relational data sets are to be written. As we already mentioned, aside from the issue of various schemas, physical formats of XML ASCII files can be different – it can be, for instance, a standard format presented in the Attachment. Alternatively, the file can present just one long string.

In its initial step, the %Parse_RSR macro converts whatever format the RSR data is into the standard format (Attachment). Then, using the standard format, it parses line by line, identifying to what data base the scanned line should belong, retrieving the identifying information, or identifying the variable and retrieving its value.

For example, if the macro identified the following term -

ClientReportAmbulatoryService CLD_ID=

- it retrieves the client ID from the parsed line and writes it into the data set AmbulatoryService. When parsing the next line the macro will identify the term ServiceDate and, after retrieving the date, writes the variable ServiceDate_Amb with the retrieved date into the data set AmbulatoryService.

Parsing the RSR data, the macro searches for all possible nine data sets listed above but, in practice, some data may not exist. The provider frequently just does not have or does not send CD4 component or Viral Load component. If some components do not exist in RSR data the respective SAS data sets will have 0 observations. User may want to set up the checking of “emptiness” of the certain data base.

Similarly, the macro covers all the possible variables envisaged by the RSR standard. In practice, however, a provider may not provide some variables in the RSR. For example, the Client Report section may lack up to the half of the variables; the Viral Load section may not have IsDetectable component. In such cases the respective variables in the respective data sets will be assigned ‘Missing.’

TESTING THE MACRO AND CONCLUSION

We have tested the macro on two sets of RSR data. The first set had 122 RSR xml files and the second one had 114 *very large* RSR xml files. Both tests ran smoothly and took reasonable time (several minutes). The output data was compared selectively with the original XML information and no discrepancies were found.

Processing the same sets by conventional method using MS Access took a couple of hours for the first set. As for the second set we just were not able to convert some very large XML data into MS Access table. In general, the larger the number of XML files is, the bigger gain in time of processing you would have.

Based on those tests we have made a conclusion that our macro is a universal and very efficient tool to convert any schema of RSR data into the relational SAS data base. Starting from 2014, there have been many changes in the data (e.g. more tables have been included). Version 2.0 of the Macro will be released soon.

REFERENCES

1. IOM. Public Financing and Delivery of HIV/AIDS Care: Securing the Legacy of Ryan White. In: Medicine Io, editor. Washington, DC; 2004.
2. KFF. Fact Sheet: The Ryan White Program. In. Washington, DC: The Henry J. Kaiser Family Foundation; 2009.
3. HRSA USD. About the Ryan White HIV/AIDS Program. In; 2012.
4. Visher CA, Mallik-Kane K. Health and Prisoner Reentry: How Physical, Mental, and Substance Abuse Conditions Shape the Process of Reintegration: Urban Institute; 2008.
5. HRSA/HAB. 2010 Data Report (RDR) & Services Report (RSR) Resources. In: HRSA/HAB; 2010.

CONTACT

Please address your questions or request for the macro to:

David Izrael,
617.349.2434

david_izrael@abtassoc.com

Attachment. T-REX schema of RSR data

```
<?xml version="1.0"?>
<CLD:ROOT xsi:schemaLocation="urn:rsrNamespace RsrClientSchema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:CLD="urn:rsrNamespace">
 <XmlVersion>
  <schemaVersion>3.0.0</schemaVersion>
  <originator>T-REX</originator>
  <versionNumber>4.2.2</versionNumber>
  <technicalContactName>DART Team</technicalContactName>
  <technicalContactEmail>Data.TA@caiglobal.org</technicalContactEmail>
  <technicalContactPhone>678,798,4657</technicalContactPhone>
 </XmlVersion>
 <ClientReport CLD_ID="10039">
  <ClientUci>490389271281674FE5ED8AE1B3A44EEE434201918</ClientUci>
  <FirstServiceDate>10,16,1996</FirstServiceDate>
  <EnrollmentStatusID>2</EnrollmentStatusID>
  <GenderID>3</GenderID>
  <PovertyLevelID>2</PovertyLevelID>
  <BirthYear>1959</BirthYear>
  <EthnicityID>1</EthnicityID>
  <GeographicUnitCode>022</GeographicUnitCode>
  <HousingStatusID>1</HousingStatusID>
  <RiskScreeningProvidedID>2</RiskScreeningProvidedID>
  <FirstAmbulatoryCareDate>11,20,2003</FirstAmbulatoryCareDate>
  <HivAidsStatusID>2</HivAidsStatusID>
  <AidsDiagnosisYear>2005</AidsDiagnosisYear>
  <DeathDate></DeathDate>
  <PrescribedPcpProphylaxisID>1</PrescribedPcpProphylaxisID>
  <PrescribedHaartID>2</PrescribedHaartID>
  <ScreenedTBID>1</ScreenedTBID>
  <ScreenedTBSinceHivDiagnosisID>1</ScreenedTBSinceHivDiagnosisID>
  <ScreenedSyphilisID>1</ScreenedSyphilisID>
  <ScreenedHepatitisBID>1</ScreenedHepatitisBID>
  <ScreenedHepatitisBSinceHivDiagnosisID>1</ScreenedHepatitisBSinceHivDiagnosisID>
  <VaccinatedHepatitisBID>1</VaccinatedHepatitisBID>
  <ScreenedHepatitisCID>1</ScreenedHepatitisCID>
  <ScreenedHepatitisCSinceHivDiagnosisID>1</ScreenedHepatitisCSinceHivDiagnosisID>
  <ScreenedSubstanceAbuseID>1</ScreenedSubstanceAbuseID>
  <ScreenedMentalHealthID>1</ScreenedMentalHealthID>
  <ReceivedCervicalPapSmearID>2</ReceivedCervicalPapSmearID>
  <PregnantID>2</PregnantID>
  <PrenatalCareID>6</PrenatalCareID>
  <PrescribedArvMedicationID>7</PrescribedArvMedicationID>
 </ClientReport>
 <ClientReportMedicalInsurance CLD_ID="10039">
  <MedicalInsuranceID>3</MedicalInsuranceID>
```

```xml
 </ClientReportMedicalInsurance>
 <ClientReportRace CLD_ID="10039">
  <RaceID>1</RaceID>
 </ClientReportRace>
 <ClientReportHivRiskFactor CLD_ID="10039">
  <HivRiskFactorID>2</HivRiskFactorID>
 </ClientReportHivRiskFactor>
 <ClientReportServiceVisits CLD_ID="10039">
  <QuarterID>1</QuarterID>
  <ServiceID>18</ServiceID>
  <Visits>2</Visits>
 </ClientReportServiceVisits>
 <ClientReportAmbulatoryService CLD_ID="10039">
  <ServiceDate>01,12,2001</ServiceDate>
 </ClientReportAmbulatoryService>
<ClientReportCd4Test CLD_ID="10039">
  <Count>8967</Count>
  <ServiceDate>06,05,2002</ServiceDate>
 </ClientReportCd4Test>
 <ClientReportViralLoadTest CLD_ID="10039">
  <Count IsDetectable="false">19</Count>
  <ServiceDate>11,14,2010</ServiceDate>
 </ClientReportViralLoadTest>
<ClientReportServiceDelivered CLD_ID="10039">
  <QuarterID>2</QuarterID>
  <ServiceID>18</ServiceID>
  <DeliveredID>2</DeliveredID>
 </ClientReportServiceDelivered>
 <ClientReportServiceDelivered CLD_ID="10039">
  <QuarterID>3</QuarterID>
  <ServiceID>38</ServiceID>
  <DeliveredID>2</DeliveredID>
 </ClientReportServiceDelivered>
```

Note: all the numbers in the above XML excerpt are fictitious.