



Creating a Usage Metadata Repository

How to show them what can be!

Bob Schork

Sr. Technical Specialist, AVP

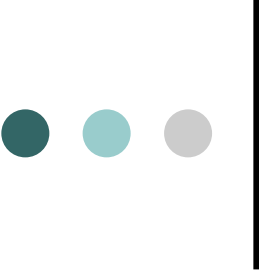
Citigroup Architecture and Technology Engineering

Warren, NJ



Who is this Schork guy?

- Born and raised in New Jersey
- Over 25 yrs of IT Experience (including consulting)
- Over 12 years of Metadata Experience including Metadata Analysis
- Implemented Rochade and Platinum Repositories, including Maintenance and Reporting functions
- Was a Developer, DBA, Data Admin, Data Modeler, Data Architect, and Meta Modeler



The problem with Metadata Repository initiatives

- Too long to implement
- Too costly
- API loads tend to be too manual
- Can't be sure of metadata you are loading
- Usually the Lone Wolf is the only worker
- Lack of true Upper Management support
- Bad job of promoting successes
- Failures usually include “Data” people.



Build verses Buy

BUY

- Costly (incl Maint.)
- Metadata Analysis
- Scanner inconsistencies
- Manual API processes
- More Robust
- Long Implementation time

BUILD

- Low Cost but no Tool
- Metadata Analysis
- Build own Scanners
- Manual processes
- No bells or whistles
- Usually Less Implementation time

Problems encountered when buying

- Vendors have the best product on the face of the Earth or it is in the process of being developed within 6 months.
- Contracts must be negotiated and approved
- Their product is compatible with every other product or it is in the process of being released soon (usually 6 months).



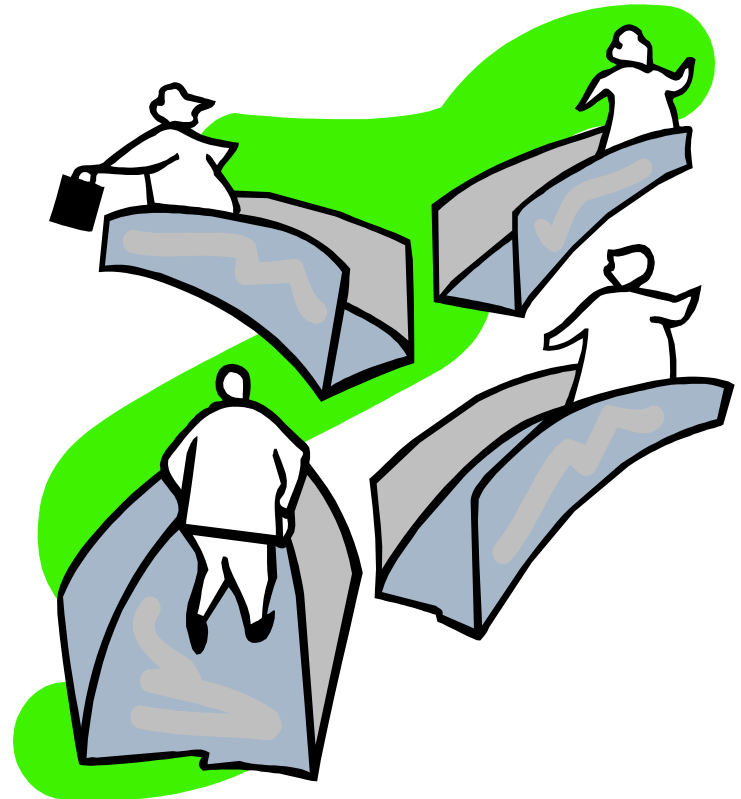
Problems encountered when buying

- Extra cost to hire consultants with Repository Expertise
- Various Program syntax must be accounted for with the Scanners.
- Manpower shortage
- Learning curve



Problems encountered when building

- You have to create your own Meta model. Not as easy as it sounds
- You have to create your own Scanner(s)
- Dev resistance to a non-tool implementation
- Not as sophisticated as a OTS Package





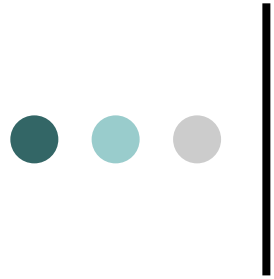
Common problems to both approaches

- Scope Analysis must still be done
- You must have a set of Company Standards and Procedures. This is where a lot of projects fail
- You still have to develop a reporting mechanism. (Intranet, Adhoc, canned)
- Still have to produce something soon.
- “A Repository product has been tried already and FAILED!”



Why Do Repository Implementations Fail?

- Lack of Standards for metadata stores
- “Data” people are in charge
- Too little resources for the initiative
- Lip service by upper management
- The **BIG** roll-out syndrome
- Lack of Maintenance procedures
- Not putting out the right metadata



One Build Approach

Build a Usage
Metadata Repository



Usage Repository

- Definition - A grouping of data that only holds the location of a metadata element within each source code program that it is used.
- The Usage Repository only holds where a metadata element is used (program and line number) and some associated metadata elements.



Drawbacks

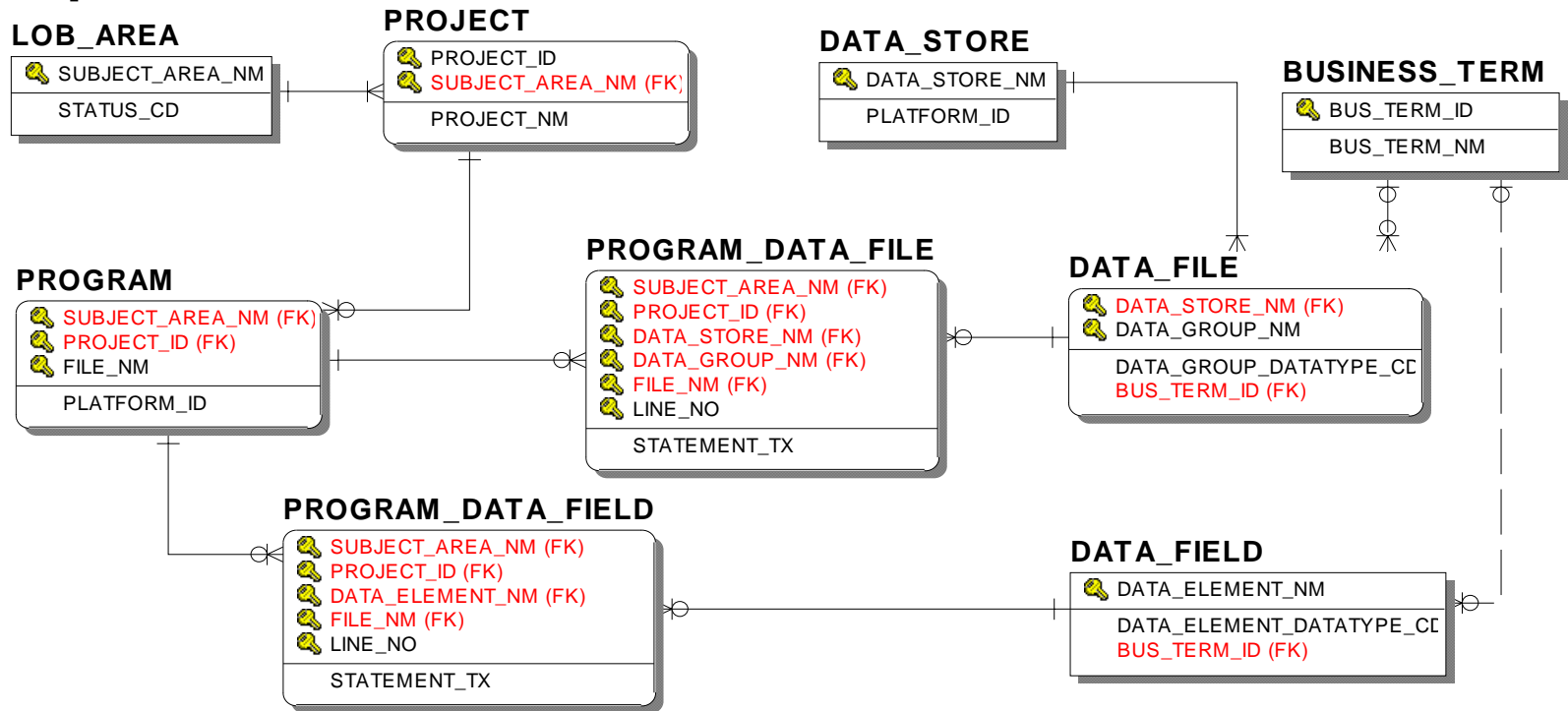
- Does not track the source to target mapping or the origin of the metadata element
- No aliases or synonyms are captured
- No Vendor Scanners exist to extract information - must build your own
- No “What if” scenarios (although it does help)
- Business users may have higher expectations than what you can deliver
- Does not track additional Metadata (ETL processes, Technical Metadata, Queries)



Building a Usage Repository

- Start with analysis
- Next is the metamodel
- Gather all reference metadata (org, projects, etc)
- Focus on what source code uses what data elements (tables, columns, databases, etc)
- Used solely for Lite Impact Analysis

Simple Metamodel





Analyze the Standards

YOUR #1 CAUSE OF PAIN

- Know your Scope before doing your Analysis
- Do you have a set of Standards for all source code libraries or are they in Development libraries. (If not, try implementing the former)
- Are there naming standards, abbreviations, etc.
- Is the Data Pgm Source Code easy to get to?
- Most Client Server Source code libraries are kept in the Developer libraries.



Solidify the Scope

- Don't bite off more than you can DO.
- Start small and accurate. This beats big and incomplete every time.
- Get your contact list (SMEs) at this time.
 - This will become or lead to Data Ownership and Stewardship
- Avoid them leading you toward Business processes. A big time waster.



Develop a Maintenance Plan

- Do it **BEFORE** you start Construction.
- How often are scanners run, from where, etc.
- Allow for changes (source locations, new projects, etc)
- Don't forget easy reference data
- The Plan should be readable to a newbie
- Call it a Repository (positive connotation)
- Modify the plan as “minor” requirements change.

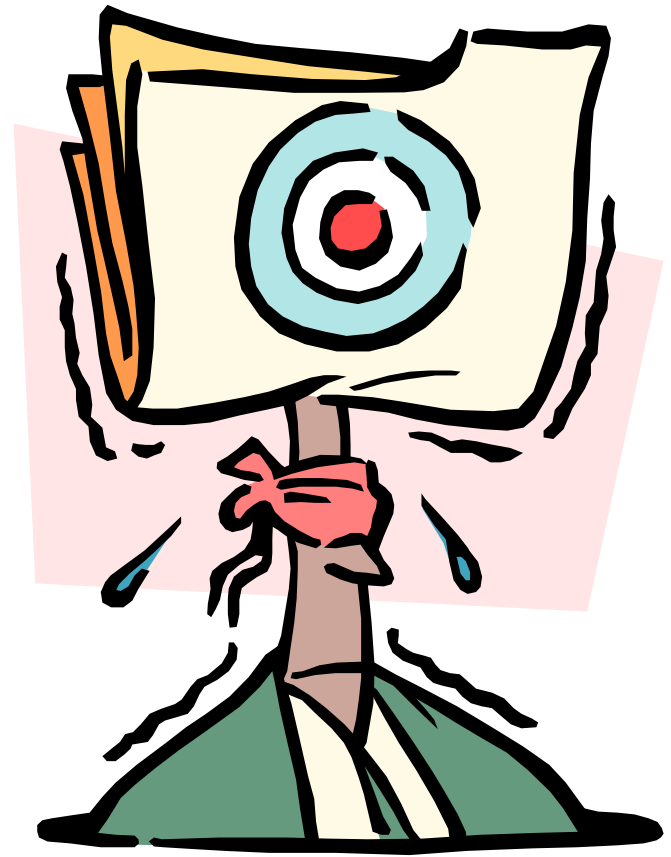


Develop a Maintenance Plan

- Should contain the following:
 - Objectives
 - Major Components and their locations
 - Maintenance Schedule (daily, weekly, monthly)
 - Loading processes and step-by-step procedures
 - Source code access methods (logins, approvals)
 - The Metamodel and Issues encountered
 - Publish a maintenance schedule

Repository Architecture Pitfalls

- Resist using Access to store Metadata
- Set up your libraries so that you know where your scanner source code resides
- Work toward having an integrated system





Develop a reporting mechanism

- Define the terms used from day one.
- Let the users decide how they want to access the metadata in your Repository
 - Intranet links to your Database
 - Adhoc using a search engine to scan DB
 - Links to commonly used canned reports
 - **ASK THEM TO CONTRIBUTE (access their skills)**
- Reports (Adhoc and Canned)



The Politics of Reporting

- Don't develop a Reporting mechanism in a bubble.
- Don't presume what the business wants
- For non-developers, ask what kind of information that they want to see
- Make friends by giving the Business VP's what they need (accurate data) and what they want (bells and whistles).

SCANNERS

- Definition: A Scanner is an executable that will read source code (Java, C++, etc), parse and assemble the occurrences of a metadata element and write that occurrence to a file or directly into the repository (the real goal).





Pre-steps to a Scanner

- Make sure that you are able to extract all information from the existing program source code. (Accesses, centrally located)
- Must have gathered your Table and Column information before running the Scanner. (These will be loaded into the scanner or referenced by the scanner for comparison)
- Read the expanded Source code



SCANNERS

- You have to know the Syntax of the programming language that you are scanning.
- You have to account for your Input/Output
- Read in your Program Tables or Data Groupings into a holding area.
- Read, Parse, Assemble and report.



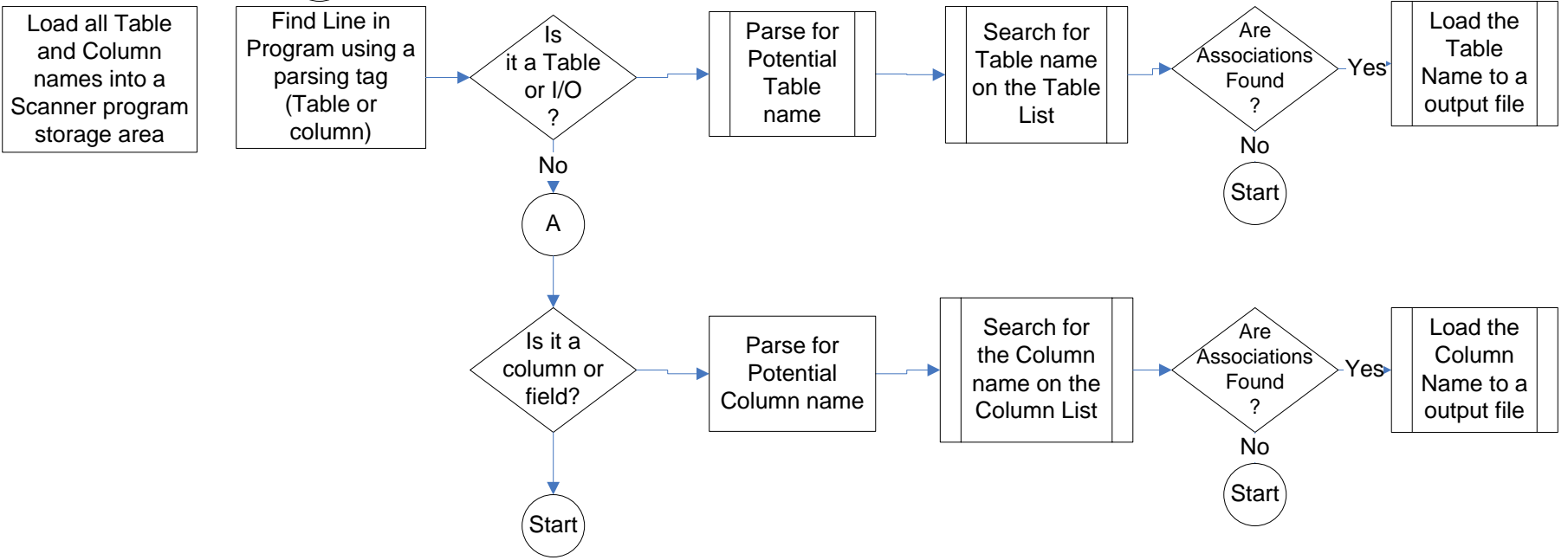
SCANNERS

- Scanners should eventually load directly into the Metadata Repository into the correct data element.
- Account for any errors in an error report.
- Use this error report to modify and enhance the loading of metadata.
- Can create a scanner for each different medium (XML, Java, C++, etc) or have one scanner for all.
- Remember the Reserve words for each language

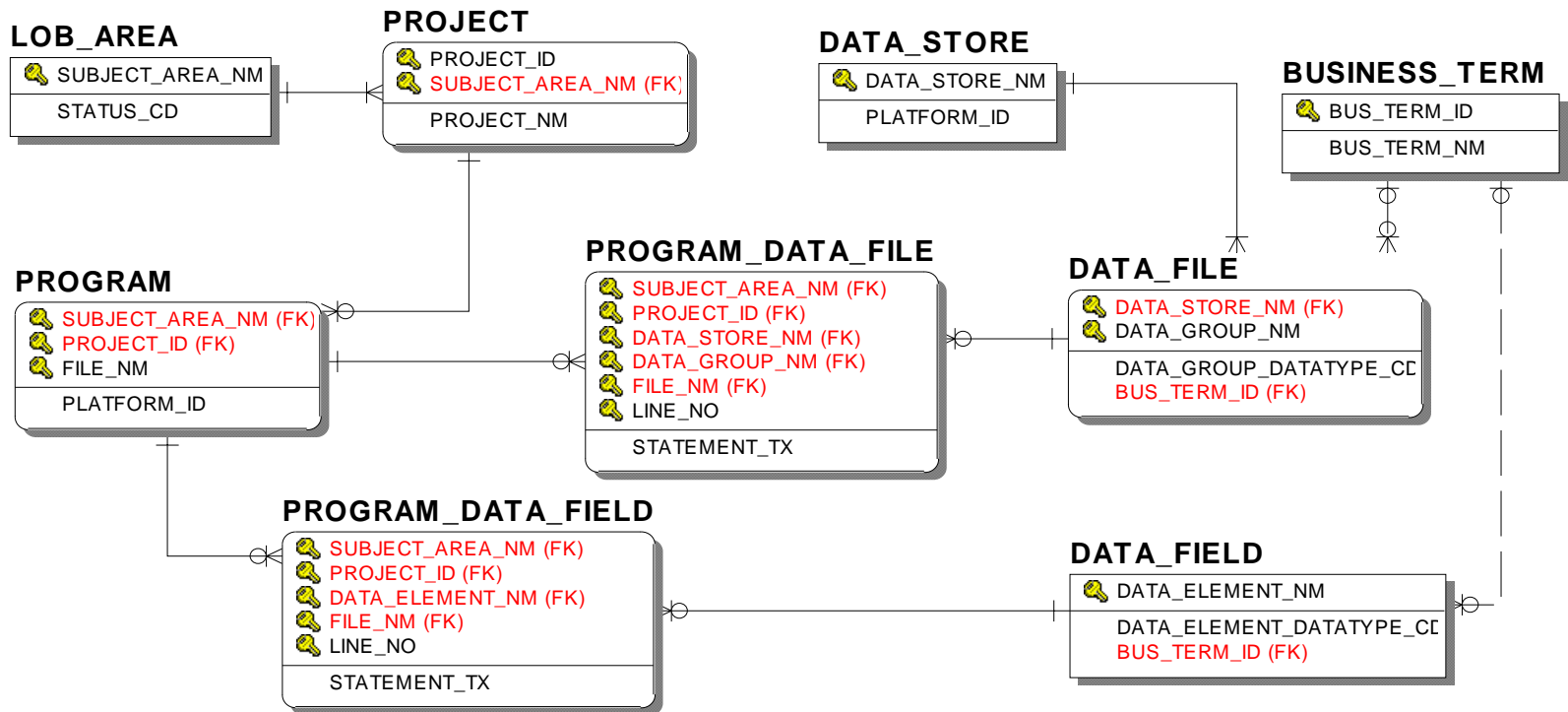
PROCESSING

Start

SCANNING PROCESS FLOW



Simple Metamodel Revisited



Scanner Pitfalls

- Obtaining Project and Owner information takes time.
- If scanning by name, what if field has an attached ID?
- Some obscure syntax may not be captured.
- Does the Language have extra Database syntax attached to the field?





Scanner Pitfalls (cont'd)

- Must run scanner for each different extension and syntax.
- Can set up Line of Business Area and Project keys as metadata elements within the Repository. A real plus.
- Sometimes it will be impossible to scan the source code libraries where they are at. In that case you may have to move them, then scan those libraries.

Politics

Never try to teach a pig to talk.
It annoys the pig and frustrates you.





Final Thoughts

- Avoid Warewolfing (no silver bullets)
- Must have the development skills handy.
- Must have at least 2 dedicated people on the project. Maybe more depending on source code portfolio.
- Don't get hung up on obscure languages.
- Have an XML Parser



Questions

- Robert.Schork@citigroup.com
- Sr. Technical Specialist, AVP
Citigroup Architecture and
Technology Engineering
- Board member of DAMA NJ