

How to Evaluate a Rochade Repository

1. Prologue: Purpose of This Document

The document contains suggestions I have found to be essential for a company that is considering evaluating Rochade or do a Proof of Concept. It might surprise you that I have found that many companies after 2 years or less simply gave up on the implementation of a Rochade metadata repository because they found it was not what they were led to believe, their expectations were way too high, or some other group in their company found a better way to do it.

This document was produced by Dennis W. Simpson of August Computing, Inc., and contains his experiences and opinions. Anything stated here is my opinion, but based on my experiences of 13 years of continuous use of Rochade with many companies world-wide.

You should read this carefully before proceeding to investigate creating a metadata repository, and specifically Rochade. Rochade is not for the faint of heart. You will see positives from the glossies and presentations. You also need to hear other considerations for your analysis to be balanced. This report focuses on some of the negative aspects that may not be obvious. If you can minimize the negatives you can maximize the effectiveness of your repository.

I feel very strongly in the desirability of a Metadata Repository. I also very strongly feel that Rochade, as it exists today, might not be the answer for the majority of companies. So exercise due diligence before you commit.

This document contains hyperlinks to a Yahoo! Group [open rochade](#). This group has a lot of good information about Rochade. Its membership is open to all with an interest in Rochade, so feel free to join. You don't already have to own Rochade to join. If you are thinking about looking into Rochade, this group is a must. (These hyperlinks open in separate windows.)

Table of Contents

| | |
|--|----------|
| <i>How to Evaluate a Rochade Repository</i> _____ | <i>1</i> |
| <i>1. Prologue: Purpose of This Document</i> _____ | <i>1</i> |
| <i>2. Definition of a Repository</i> _____ | <i>4</i> |
| <i>3. Approaching a Rochade Repository</i> _____ | <i>6</i> |
| <i>4. Should I rely on Gartner's Opinions?</i> _____ | <i>6</i> |

| | | |
|-------|--|----|
| 5. | <i>Should I rely on ASG's Evaluation of our needs?</i> | 6 |
| 6. | <i>How Much Time Will It Take?</i> | 6 |
| 7. | <i>Why will it take this long?</i> | 7 |
| 8. | <i>Is Rochade Complete Out-Of-The-Box?</i> | 8 |
| 9. | <i>Assess Your Needs</i> | 8 |
| 10. | <i>Determine your ROI requirements</i> | 8 |
| 11. | <i>Issue a Request for Proposal RFP</i> | 9 |
| 12. | <i>Evaluate the Response to RFP</i> | 9 |
| 13. | <i>Gap Analysis</i> | 10 |
| 14. | <i>Determine Suitability of Vendor</i> | 10 |
| 15. | <i>Security Concerns</i> | 10 |
| 16. | <i>Talk with Rochade Customers</i> | 10 |
| 17. | <i>Do A Proof of Concept (POC)</i> | 12 |
| 18. | <i>Scan Databases</i> | 13 |
| 19. | <i>Scan ERwin Models</i> | 13 |
| 20. | <i>Load Custom Data</i> | 13 |
| 21. | <i>Load Programs/Applications</i> | 14 |
| 22. | <i>Loading the Data</i> | 14 |
| 23. | <i>Validate the Data in Development</i> | 15 |
| 24. | <i>Promote to Test</i> | 15 |
| 25. | <i>Promote to Production</i> | 15 |
| 26. | <i>Do it Again</i> | 15 |
| 27. | <i>User Interface</i> | 15 |
| 28. | <i>Determine Software and Resource Requirements</i> | 16 |
| 29. | <i>What ASG Software is Desired</i> | 16 |
| 30. | <i>Available Software</i> | 16 |
| 31. | <i>Not-Available Software</i> | 16 |
| 32. | <i>Database/Server</i> | 17 |
| 33. | <i>End User Access</i> | 17 |
| 34. | <i>Web Browser</i> | 17 |
| 34.1. | <i>RoAccess</i> | 17 |
| 34.2. | <i>WebAccess</i> | 17 |
| 34.3. | <i>Autopilot</i> | 17 |

| | | |
|-------|---|----|
| 34.4. | <i>Metability</i> | 18 |
| 34.5. | <i>In General</i> | 18 |
| 35. | <i>Versioning</i> | 18 |
| 36. | <i>Configuration Management</i> | 18 |
| 37. | <i>What is the Difference between Field Developed Interfaces (FDI) and GA</i> | 18 |
| 38. | <i>Searching - Large Amounts of Data</i> | 19 |
| 39. | <i>Purchase Software and ASG Technical Services Hours</i> | 20 |
| 40. | <i>Set up Dev, Test and Prod Systems</i> | 23 |
| 41. | <i>Customizing a RIM</i> | 23 |
| 42. | <i>Running Scanners</i> | 24 |
| 43. | <i>Loading Custom Metadata</i> | 24 |
| 44. | <i>Data Life Cycle Management</i> | 25 |
| 45. | <i>Automating Scanners</i> | 26 |
| 46. | <i>Using ASG Technical Services</i> | 27 |
| 47. | <i>Personnel Concerns</i> | 27 |
| 48. | <i>Outsourcing Your Rochade Repository</i> | 28 |
| 49. | <i>IBM Example: Partnering and Cost Containment</i> | 29 |
| 50. | <i>Concluding</i> | 29 |

2. Definition of a Repository

I think of a Repository in a very comprehensive way.

Let's start by listing what a real, modern metadata repository is today:

- **Data Sources:** keeps track of databases, tables, indices, columns, triggers, stored procedures, referential integrity constraints, validation constraints, the definition of all columns (like CUSN would be Customer Number). It should handle all SQL/relational databases, such as Oracle, MySQL, Postgres, Sybase, all the open source databases, plus others. Likewise for files, fields on multiple platforms and different types.
- **Business Data:** processes, sub-processes, best practices, terms, definitions, business rules, data validation rules (column and inter-column). It should contain people, projects, project phases, project life cycle, internal and government regulations, privacy, applications implementing processes. Service Level Agreements for applications and usage contracts. Data stewards and administrators must be clearly defined and relationships established. There are probably about 100 different types of business data types that should be captured and cross linked with all interdependencies.
- There must be easy ways to integrate the metadata the scanners bring in with all the business and other metadata.
- **Applications:** should be able to scan the source code to create breakdowns of the application to sub-applications, variables and object usages. They should show classes, inheritances, interfaces. Must also show what data is input to the applications, what processing is done (including validation rules which reference the corresponding business rules), and what is output.
- Must be able to model a complete Systems Oriented Architecture out-of-the-box. This includes message buses, message formats, WSDL files with breakdowns, UDDI repository extracts, handles BPEL (business process modeling), policy (run-time and register time). Must list interfaces and connectors to other systems. Must handle both Web and non-Web services. Policy, contract and SLA management must be incorporated.
- Must have a complete life cycle. [Click here](#).
- Must be completely automated for running the data loading elements. For a discussion, [click here](#).
- Must have a large number of scanners for business intelligence tools. You should not have to pay to develop a new interface. This is to scan reports to determine what data is used, what calculations are performed, and where the output fields are on the paper or screen.
- Must show end-to-end data lineage paths. That is, you see data introduced into the system, it goes to many programs, ETL and other processing, combined and used with other data, stored in several places

and finally displays on reports and screens. You should be able to see the metadata for every step along the way, and easily produce a report. If an auditor says a report field is suspect, you must be able to show every piece of data associated with it, and every step or processing that was done.

- It must come with complete impact analysis software and reporting. It should answer a question like: I'm looking at a report field which is 10 characters wide, but it now needs to be 12 characters wide. I need to know every data source, data base/table/column or file/field/, every application (Java, dotNet, C#, COBOL, etc.) that reads and modifies it. I need to see all validations, table lookups, translation tables and calculations involved. You should not have to write any code. It should be a standard part of the repository, because this is one of the primary reasons to have a repository in the first place.
- Must have screens that are attractive to the end user, have multiple Web based front ends: one for administration and one for end user use. New applications should be easy to create using tools like FrontPage. Programming should use industry standard accessing techniques using standard APIs (ODBC, JDBC, etc.) and data manipulation languages like SQL. The more proprietary, the less desirable it is.
- All loaders must have full validation runs to verify data is perfect before loading. The loaders must have a consistent front end, parameter formats, output reports usage logging.
- Any competent SQL programmer should be able to easily make new programs.
- The internals of the repository should be easily understood and should be intuitive.
- Applications should be easy to install and maintain.
- Clients should be Web based, not requiring any kind of installation on the end user PC such as Java runtimes, dotNet framework, etc. Thin Web clients are best. No client-server applications.
- End user licensing must be consistent and uniform. A user on a PC should be allocated no more than one license, regarding the number of Rochade screens or applications running on his PC.
- There must be a roadmap document that show how to integrate all aspects of the repository into a fully operational, fully integrated, full life cycle managed system. You should not have to write software to implement any of these basic functions.
- It is almost mandatory that the repository and its Life Cycle integrate with at least one of the industry standard workflow packages?

In summary, if it relates to your data in any way, the metadata should also be in the repository.

I believe this is a reasonable checklist. Each vendor will have its strengths and weaknesses, and the above will help determine how quickly you can get your repository operational.

3.Approaching a Rochade Repository

Creating a Rochade Repository is not as easy as it would seem. Most companies find that it takes a great deal more time, money and resources than they had expected or been led to believe.

4.Should I rely on Gartner's Opinions?

No. You should not let anyone else do your thinking for you. Some companies have found that Gartner's opinion and their own experience did not always match well, especially in the Rochade area, which is my experience. I believe that companies that rely too strongly on Gartner open themselves to entirely too much risk. Read the ASG glossies, read the Gartner report and then take both with a grain of salt. Begin your own investigation.

5.Should I rely on ASG's Evaluation of our needs?

No. You should have independent agents who are not affiliated with ASG evaluate your needs.

There is an old saying, "don't ask a surgeon if you need surgery". Surgery is a surgeon's business. He likes it and he makes money at it. You want a separate specialist to determine if you NEED surgery.

It's the same with a Repository, a SOA or any other large technology area.

6.How Much Time Will It Take?

A great deal more that you might expect.

- The RFP preparation: **3 full time people**
- The evaluation of the RFP: **3 full time people**
- The Proof of Concept: **4 full time people**
- The Implementation: **8 full time people a full year plus 4 full time equivalents from other parts of the company**
- Outside Rochade contractors: 4 for 6 months.

How long will it take?:

- Create RFP: 2 months
- Evaluate RFP: 2
- Proof of Concept: 6 months
- Implementation: 12 months

If you wish to spend less time than this, you Rochade is probably not for you.

After the initial year, your repository will be "production ready". But it will not have much of your enterprise metadata yet. It will take another year to get where your repository is highly populated and is actually in constant use by your end users and stake holders.

So you should expect two years after you decide to implement a repository to have it fully populated.

But you will constantly be re-loading data because of Life Cycle.

This means after the first two years, you can start determining when/if you will get a positive return on your investment. Remember, you will still need probably need between 8 and 10 people constantly on the payroll, because hopefully, you will be getting more and more parts of your company wanting to get their data into your repository. Also, many of the data loaded will only be partially found, and you will constantly trying to find the remaining metadata.

Existing programs and data stores are the most time consuming, because much of the information about them may no longer be available. People quit, die, get transferred to other departments, etc. So a lot of work by hand is needed. The good news is once you do as much as you decide you want to do for your existing applications, getting the metadata in for new development can be much faster, because you will probably have already placed metadata documentation requirements in place that the developers must fill out as part of their Life Cycle.

If you don't have this amount time, talent and resources, Rochade is probably not right for you.

You should supply at least 4 full time people for the Proof of Concept. Do not let ASG do the proof of concept. Otherwise you might later find Rochade is much harder than it looked. Remember, Rochade is unbelievably complicated. ASG's people will make it look easy. And you probably won't know exactly what they are doing. You may give the repository more credit than it deserves and think it does more than it really can.

The actual implementation: 8 people for a year.

7. Why will it take this long?

Rochade is incredibly complex. Also, it is not complete out-of-the-box. You will find it does much, much less than you expected, and you will have to make up the difference. No one in your organization will have the skills to do these things, so you will have to train and the learning curve is extremely steep.

You will be tempted to think merely using ASG Technical people will be the answer to shortening the time. This can be a bad assumption.

Also, getting business information, terms, definitions, projects, requirements, business rules, validations, lookup tables, plus a whole host of other things will require a tremendous amount of time and people working with members of the various departments in your company, first searching for the information, and then trying to determine how it relates to all the other information in the repository. You will need experts in a number of technical and non-technical areas.

8. Is Rochade Complete Out-Of-The-Box?

Not even close. It has collections of parts that don't really fit well together. Also it may be difficult to determine which parts to use, and how to integrate them. Also you will probably find a number missing major components. You will be really shocked to find this. In essence, you get parts, and you have to figure out how to build a repository out of the parts. I think it's getting better, but very slowly.

9. Assess Your Needs

If you are fortunate, you will have metadata repository experts who can codify your requirements.

Many companies hire large consulting firms to do this. If this seems good to you, make sure that company has done at least 4 full scale Rochade implementations. Creating a Rochade repository is probably nothing like what they have done before, and they will make assumptions based upon their past experiences which will not translate to the Rochade world.

10. Determine your ROI requirements

In today's volatile environments, a new technology should break even in 6 months. Then for the next 6 months, it pays for itself by creating subsequent savings.

Some big ticket vendors say you will recoup your investment in 5 years. This is way too long. If you can't get a 100% return on investment in 6 months today, you need to look at another solution.

It will take you at least a year 12 full time people just to get the Repository operational. This means it will probably take another year get all the metadata in

the company into the repository. It will be at that point you will start receiving a benefit from the Repository. So you will have spent 2 years and 144 man months of people to get there. This does not take into account the large expense for the software licenses.

Very carefully determine what cost benefits you will get after this two year period, and determine when the repository will pay for itself.

In most cases, it never pays for itself, and the repository is cancelled.

Knowing this, you need to take a hard look at the repository and how you will implement it.

For an example of Rochade and ROI, [click here](#).

11. Issue a Request for Proposal RFP

Your people alone or your people working with a large consulting organization will do this. Expect 2 months minimum. Do not outsource this. Your people need to be 100% involved.

If you don't have time or people for this, then either completely outsource the repository or don't create one. You Rochade repository will take a lot of time, and if you don't have time, now, you won't have time later.

12. Evaluate the Response to RFP

Have independent Rochade consultants work with your people to determine how well Rochade will meet your requirements. They should have at least 5 years and 3 full implementations in their work history. Experience with other types of repositories will be good, but Rochade is very different, and little technical knowledge will transfer.

You should require that there are at least 20 companies, that are not controlled by the repository vendor, that have good products and solutions that are based on the repository tool. If these third party companies do not exist, you are stuck at the mercies of a sole source vendor, with all the attendant liabilities and shortcomings. Unfortunately, Rochade fails this rule.

Are there a large number of experts available for contract or permanent positions? If there are many, then the risks are less and your costs will also probably be less. If there are not many, why is that?

13. Gap Analysis

Every vendor's reply to a RFP will typically have these categories of information:

- What is true
- What is sort of true
- What mostly not true
- What is a complete lie

The Gap analysis is to determine the category for every line item in the vendor's response. The Gap means the gap between what is stated and reality.

14. Determine Suitability of Vendor

You will probably be looking at other vendors as well. Do not use Gartner as a guide. Use your own expertise. Other vendors may seem smaller, or not have as nice and flashy a presentation, but they really do have good products. You might have to integrate several of them to get a real repository. But that is what you are going to have to do with Rochade anyway.

15. Security Concerns

Rochade security is different. Normally one expects the database engine to enforce security. Rochade's engine only does limited security. Rochade enforces security mostly at the API or application level for programming languages and hardly any security enforcement at all for the RPL/database level. For example, for a relational database, no matter how you access the database, if you violate a security rule, the database engine itself will catch it. Not so with Rochade, because the Rochade engine itself is mostly wide open and you have to enforce restrictions at other layers. An experienced Rochade programmer can bypass a great deal of the security, which is not desirable.

This is not necessarily a high negative, but you need to have your security people review it.

16. Talk with Rochade Customers

The idea is to see what other customers for the product think about it. And if it is predominantly positive, and they had similar needs and expectations from their repository, then your experience might be good if you choose to buy Rochade.

As you can guess, there is an inherent problem. Companies may not talk negatively for fear of legal action. You are aware that if you are considering hiring a person, and you call his former employer, if they say anything negative,

they might be sued. So most companies have a policy of just validating the employment and nothing more.

That is the same case here. If a company spent millions and they had a disaster, employees quit, contractors refused to renew their contracts, and employees were reassigned or let go, and the repository terminated, they probably won't tell you. Here again, it is wise to employ consultants who have a lot of Rochade experience and are not in ASG's payroll or receive a benefit, such as receiving a finder's fee if you purchase Rochade or will be placed at your business by ASG.

What types of companies to ask?

- 3 companies who purchase Rochade last year.
- 3 companies who purchase Rochade the year before
- 3 companies who purchase Rochade two years before

You want a mix of positive and negative responses.

- You must determine if the company has as expansive idea of a repository as you do.
- Are they in full automated production?
- Is the company requiring more and more departments to use the repository, or are the repository folks still trying to convince the company to use the repository?
- Did they have a 100% Return on Investment in 6 months after purchasing Rochade? How about 1 year. How about 2 years. Do they think they will ever get Rochade to pay for itself, or is it just a continuing cost. Review this [Rochade ROI document](#). Many times a repository department will not track a lot of the true costs. If people are loaned to it, they must be accounted for. If Technical services and full time equivalents in other departments were needed, add the costs.
- How many people did they employ in going into production? 8? 10? 12? Remember, the majority of the metadata (like definitions of the columns) has to be ferreted out and takes your and their time. This takes lots of people.
- Do they have both Hardware and a Data Life Cycles with full dev, test and prod as a minimum?
- Ask for how many divisions, departments, applications, regulations, definitions, terms, business processes, etc., they have. If they have only a few, that is not good. If they have thousands, that is good.
- Have they written a large number of custom Rochade applications and screens? Did they do it themselves? Did they hire Tech Services?
- Did all the FDI custom scanners they paid for go into GA status?
- Was the repository a complex one, or was it a less useful, easier to implement subset? Your idea of a repository may be different from theirs.

You especially want to talk to the companies who have abandoned Rochade. You should talk to at least 4 companies.

Here is what you should ask:

- Why is it not active/disbanded/terminated or mothballed? Do not accept, "It was defunded." That is a result. You want the cause. Of course any project that is not working out will be defunded.
- Why was its funding cancelled?
- Had the repository met its targets (budget, time, amount of data, user acceptance)?
- Did the repository have a few thousand applications loaded plus all related business information all loaded, entered and cross linked? Were the fields and columns all linked to validations, applications, quality metrics, etc.?
- Did they have a 100% Return on Investment for the repository at any time?
- Was the repository growing in size with a constant new stream of departments wanting their data to be mined and stored in Rochade when it was terminated?
- How many people were on the project at the end?
- Was the company completely happy with the repository?

The easiest way to get this information is via a phone call, with only 1 other person on the line. The more people, the more formal it becomes, and the less freely information may flow. Emails are "in writing", and company lawyers might not approve this. But informal conversations are usually OK. The best way I've found is to actually visit the person who was in charge of the repository, and talk to them off-site, like during lunch. That way you can develop a sense of trust and comradeship between you. After all, you both are/were in the same boat.

17. Do A Proof of Concept (POC)

Here you will try to model what your concept of a metadata repository is, and determine if Rochade will do all you need.

Most companies find that much of the Rochade software is not only hard to understand but also just as hard to install.

So they decide to let ASG install all the components. This is a good idea. But remember, you should have your people working with them to completely understand how to install the components so if you do purchase Rochade, you will be able to rely upon yourselves.

You must have Life Cycle as an integral part of your POC. If you don't, you will be sorry later. There is a section later that is more detailed about Life Cycle. But for here, you need to simulate your company through its life cycle. Merely

loading data is not enough. You need to simulate development, test and production for typical data. Load data, promote to production. Then make changes, load into test, verify things that have been added, deleted and changed and then see them promoted into "test". Then verify the test is correct. Then promote the test data into production. This includes ASG scanned data, plus you custom data that is loaded into your custom RIMs. You must be able to view the data in test and be able to view the repository as to what it would be like if all of test were promoted to production.

For every data source run the scanners yourself. There some interesting types of configuration files, and you need to become familiar with how they work.

Rochade uses the term "scanner" to refer to a program or a set of programs that get metadata from data source. You might define scanner as a metadata extraction tool.

18. Scan Databases

Scan at least three different ones. Note: you will see many key columns that are the same thing (like CUST_ID). Rochade does not know if they are the same conceptual column, so it will make them separate columns, and you will have to spend an incredible amount of time looking at every column in each database, which can be tens of thousands, to determine if CUST_ID in on table is the same as CUST_ID in another, and if CUST_NO is the same or different. You then need to like the terms/definitions or generic definitions to the columns you just loaded.

19. Scan ERwin Models

Scan at least 3 different models (both logical and physical). If your people do not integrate logical and physical models themselves in ERwin, Rochade will not do it for you, and you will have islands of unconnected physical and logical data.

Do the physical data elements link to your physical data stores? If not, you are in trouble. For instance, suppose the physical model has a table ABC which has columns X and Y. Do you have such a database scanned in? Are there links from the ERwin ABC, X and Y to the scanned Items of the database? No? You will have to write a program to match identical names between ERwin physical columns and those scanned in.

If the physical and logical models are not done together in ERwin, you will have to find some way, probably a program, to try to do the linking between them.

20. Load Custom Data

You will have data like: companies, divisions, groups, projects, programs, terms, definitions, regulatory requirements, processes, sub processes, people, documents, best practices, glossaries, business processes, business rules and such. When you start getting them, remember how difficult and time consuming it is just for a small sampling for the POC. Later you will extrapolate for thousands and thousands of these to understand why it takes years to populate your repository.

You will also have mappings from generic names to actual physical columns. Some of the mappings will say CUST_ID, CUSTNO, CUSTMR in some tables are the "same". But the same names in other tables, programs, reports are NOT the same, even though they are spelled the same. Someone has to know the data well enough to know what each column means.

21. Load Programs/Applications

If you scan a Java program, there is no way to link to business rules or even validation rules. It does not link to data sources, like database/columns or files and fields. So Java programs once scanned in are islands of data. You will have to think how to link them to other parts of your system. For instance, a Java program may be used to pre-calculate data for reports. You will have to find ways to show what data is used by the Java programs/classes/methods, what is done to the data and where it finally written, maybe a database column. Then a report must show these data as inputs. All these are links YOU must define and load. Then extrapolate if you are working with 3000 programs.

This is a case where you might consider having a custom scanner written say for a report writer to some of these tasks for you.

When you are told there are other scanners to load programs, reports, ETL programs, copy books, etc., ask how it maps input data fields, calculations, business/validation rules to other parts in the repository. Have them actually run the scanners and verify the links were indeed automatically created. You may be surprised that sometime the scanners do less that you might want or expect.

22. Loading the Data

Make notes on how the data is captured, scanned and loaded. It is a dizzying array of different things being done differently with seemingly little commonality. Think of how you will document the processes and procedures for your final organization.

Also, ask about how the loading was validated. Is it possible to load bad data?

Can the loaders/scanners be truly completely automated so you can scan hundreds of separate data sources each day with only taking a few minutes to schedule everything?

23. Validate the Data in Development

How do you make sure the data is complete and without errors? Rochade does not have the concept of referential integrity, run time rules and validations. It does not enforce mandatory and optional fields.

Did the various data databases that were scanned get automatically linked to your higher "DATA_STORE" ItemTypes. That is DATA_STORES have databases and files. If you scan a database, it will probably NOT be automatically linked to a data store because the scanner does not know about anything beyond the database, such as projects using the database, etc.

How do you find what is missing? Broken links. Missing Items. Missing data elements.

You must be able to view "dev" in such a way as to show what the repository would look like if all the dev data were to be promoted to production.

24. Promote to Test

Once you validate the data, it can be promoted to test. Test should be on another computer.

25. Promote to Production

After test is validated, promote the data to the production repository.

26. Do it Again

Change a table by deleting a column, re-scan it into development. Verify the column is not in the metadata (because it was deleted in the table). Promote it to test. Make sure the column is not in test. Promote to production. Make sure this column, that used to be there, is no longer there.

27. User Interface

Make custom Web screens, applications (both Web and stand alone) and reports to show users what they can get. Have your Java people work with the ASG tech folks. It is important to understand what can and cannot be easily done with the ASG user interfaces.

28. Determine Software and Resource Requirements

The POC is over. Now you decide what components you need. Some will come from ASG's GA (professional quality) product line. Some will come from ASG's FDI line, which are not as good as GA products.

Some needs can't be met, and you will have to fund their development. Do not underestimate the complexity of doing these. Give yourself time to train your people. Otherwise, you will have to pay others. And paying ASG to develop something is not a guarantee it will work as desired, or be done within time and budget.

29. What ASG Software is Desired

This is usually difficult, because if you are like most companies, ASG may have only one half of what you need. Any you might not completely understand just what each software component does and what benefit it gives you.

30. Available Software

GA (generally available) the higher quality software. It consists of scanners, busses, and such.

FDI (field developed interfaces) were developed for clients and sort of supported by the technical services people. They are usually not full implementations, not as well written and not as well documented. If they were, they would be converted to GA, and supported by the ASG product development staff. See the difference?

31. Not-Available Software

If you are a large company, you will probably see a very large number of missing components you need. You may have report writers, programs written in multiple languages that each need a scanner (remember you need links to data the program read/write, business rules/validations, etc.), scanners for report writers, ETL tools, Business Intelligence tools, and such.

You may also require user interfaces not available in Rochade. Their WebAccess is pretty limited as to what it can be taught to do. So if you want fancy and flexible, you need to look elsewhere.

Everything you need to have custom created will take time. There is then documentation, training, and definitions of best practices and such that need to be created. Don't underestimate the negative impact of all these tasks on your

repository milestones. The longer you delay your repository from going into production mode, the less likely it will be a success. Your users have needs now, not a year from now.

32. Database/Server

You will need a license for each Rochade server you implement. If you have dev, test and prod in different locations or machines, you will need additional server licenses. If you also want additional licenses for development or redundancy, you need to price them out.

33. End User Access

What programs will you have to access Rochade?

It is a collection of client-server, Java, Web and mixes. It is not a clean, uniform approach.

34. Web Browser

For access via the Web, you have two options. One is more for end user needs with attractive screens. The other is more for DBA and data admins.

34.1. RoAccess

This is a third party developed suite of access, entry, forms, reports and a validating loader all integrated. These tools predated ASG's Web tools by 4 years. It is available from [August Computing](#). The RoAccess product link is [click here](#).

34.2. WebAccess

This is ASG's Web system. It's confusing, not user friendly, not very attractive, and extremely hard to extend. The current wisdom is to never try to extend it, but just leave it as it is. You will hear it is "infinitely extendable", but that is not what is usually done in practice.

34.3. Autopilot

This client-server program used to be really useful. Today it is in a very sorry state.

It is the primary administration tool. It was based on 3270 type screens, which today are really difficult to use. It has not been kept up with the newer features of Rochade, like extended names and NameSpaces. Parts of it no longer work properly. So basically you use Autopilot for some administration, and use other programs for other administration or data manipulation.

34.4. Metability

This is a Java program that runs on your PC. It is not Web based. It duplicates some of the functionality of a Web browser and Autopilot, and is a data analyzer and admin tool.

34.5. In General

Many find it is hard to know what data access tool to use. No one tool covers all it needs to cover, so you need to learn all three. Different approaches, different user interfaces, etc. ASG once said that its stated direction was Web based, so this may change someday.

35. Versioning

Your RFP may ask about versioning. Rochade has something they call versioning, but it probably does not match your definition.

You may want to be able to see the metadata by rolling back or being able to see prior changes that are periodically made. For instance, you might see the current width of a table column is 5 characters, but you would like to see what it was before the latest change. You can't do that.

Versioning means more like taking a snapshot at a place in time or a "stage" like dev, test or prod.

You need to ask ASG what versioning means and see if it meets your needs.

36. Configuration Management

Because Rochade does not support normal versioning, it does not support configuration management.

37. What is the Difference between Field Developed Interfaces (FDI) and GA

ASG sells two different qualities of software.

GA, Generally Available, is better, well tested and very well documented. It handles all the conceivable options the tool it "scans" might employ. GA is as good as ASG's software gets.

FDI, on the other hand, is not. If a customer needs a scanner for a particular database, ETL tool, report writer, Business Intelligence tool, and ASG does not already have it, you will have to pay to have it developed. Let's presume for argument sake, it is an industry standard ETL tool. You ask ASG to make a scanner for it. But ASG is not an expert in that ETL tool, so they get representative extracts from the ETL tool and then writes the scanner based on what they see. In the end, the scanner will work as long as the way the ETL programmers don't change how they are doing things or start using new features. You will have to come up with a best practices policy and contract with your ETL staff so they are instructed and they agree not to use any "new features", and you document what features are currently being used.

To be fair to the Technical Services folks, they have a hard job. They are a profit center. If they don't have sufficient revenues, they look bad.

Rochade is lacking in many places, so customers have to pay them to come up with solutions or work arounds. Also, Rochade probably only has maybe 25% of all the scanners they really need. So who is going to develop them? Technical services.

But if Rochade was really complete out of the box with new scanners before they are needed, then what would Technical Services do? Everything would be GA. Also if WebAccess were really easy to extend, end users would do it themselves.

So Tech Services, I feel, has a difficult time defining itself. Also, there are times when they are overwhelmed because they do pre-sales, post-sales and other support as well. So these folks are really overcommitted much of the time.

If every Rochade becomes the complete packages it is presented as, then the Technical Services folks could concentrate more on helping customers develop value-added services.

38. Searching - Large Amounts of Data

Searching related information: If you want to issue a search which would use the equivalent of SQL joins where Items are related to other via multilevel links, and you want to qualify on attributes/columns of each ItemType to get back only the ones you want, you will have to write programs. Rochade does not provide a data language like SQL. Rochade provides limited search capability. There is no concept of sum, min, max, having or group by. There is a \$SDCAC query, but it does not handles multi-level/joins/link relationships. This would be easy in SQL, but not in Rochade. There is also no concept of inner or outer joins.

Searching is also many times extremely slow. This is because Rochade has no concept of indices. It only has sort of a primary index consisting of the Item/Row's name. You can't have a multi-key index. In a SQL repository, you can create primary and secondary indices to speed up searches. This becomes extremely important when you have a large repository. Small repositories will search quickly, but they who has a small repository? If you want to do a single ItemType search (not joined) and you want to qualify on an attribute/column, Rochade has to read every Item/Row to get the contents of those search columns. SQL databases can have secondary indices where the data is contained in the index, so the search is extremely fast. But Rochade must always do in essence a table scan. If you can qualify on the Item name, it merely reduces the number of rows it had to do the full data access scan on. Because this is so extremely slow on large Subject Areas, Rochade provides an XML search server to dump the repository into, and you search via the XML search server instead. Yes this can be faster, but it has the downside of the data is not real-time and must always be considered "old".

For example, sometimes you might like to search the entire Enterprise Repository for the occurrence of a word but you are only searching key columns/attributes. Here you have 3000 full projects plus all the related business metadata. Suppose you have only 10,000,000 Items in your repository. Doing table scans to search all Items for a word would take a long time. Here you would consider the XML search engine. But in a life cycle world, Items are always being changed and going through dev to test to production. If you have 3000 projects, you could expect to be promoting various projects to production every day! That means the XML extracts would always be out of data. As soon as you promote some life cycle Items to production, you XML search cache is out of date. So consider what you can do to speed things up for searching and reports.

39. Purchase Software and ASG Technical Services Hours

You must then determine what all parts of the Rochade product line you need to buy. You will probably find it extremely confusing, and not know what you need. That is normal. You might feel the necessity to buy more components than you need "just to be safe". Try to resist this. Make ASG show you why each is absolutely essential to your current needs. Current needs mean NOW and up to 6 months. Most over spend and never use some of those expensive components.

You will find there are a lot of things that are missing that you have determined to be essential to the success of your repository. ASG's policy is to make you pay for them, even if they would have general usability for other customers. The best thing is to create this "scanner" yourself, because it builds your in-house

expertise. But if you feel your only choice is to hire ASG Technical Services, be really careful.

If you are told that ASG is "working" on something you need, what this means is that maybe sometime in the future something will be delivered, but you don't know exactly what. Then it will have to undergo debugging and there will be bug fixes. You probably would not like to implement anything until it is proven, so this means you might not want to use it until the second set of patches/fixes come out. This will all take time. If your needs are now, not sometime in the future, you might consider the impact on your repository if it doesn't do what you need or is not in the timeframe. Or you can have ASG sign a SLA defining its capabilities and when it is guaranteed to be solid.

Tech Services usually charges by the hour, which is not good. You should always require them to provide a fixed price or a not to exceed. You should also fix the due date, and have penalties if they products/services are late. If you only buy "hours", and they are not through at a deadline, what do you do?

Also specify what is to be done in some detail. Do not just give ASG a blank check asking them to do "what is necessary". Your and their definition of what is necessary may differ considerably. Also specify full testing, and even fuller documentation, with complete run-time diagnostics to help problem solving when it occurs.

There should be a Service Level Agreement. It should say something like for a ETL scanner you need, "This scanner is supposed to handle all the features of this ETL tool. If the usage of a not currently used feature causes the scanner to fail, it will be corrected within so many days, including full documentation, and any suggestions to the ETL policy and contract documents." If you don't create such a SLA, you may have to pay every time the ETL folks take advantage of the ETL tool's vast feature set and the scanner breaks, not to mention the downtime inbetween.

Here is a case study to show what can happen:

Let's create a hypothetical situation:

You desperately need two scanners. Even though the products (say an ETL tool and a Business Intelligence tool) to be scanned are industry standard and in wide use, let's presume ASG does not already have the scanners available. Tech Services hours are purchased to create two FDI scanners. Or maybe a fixed price is quoted. No time limit or penalty clause is defined. ASG is just asked to create the scanners.

ASG Tech people are on-site. After many months of constant work, but the scanners are not finished, the "hours" run out, and/or the ASG people leave for

their next assignment, leaving you wondering how to load the data with the unfinished scanners.

Later, one scanner when fixed, runs for a short time, but then breaks. It is determined it is because the ETL tool is being used in a slightly different way than when the scanner was initially developed. The company may be told they had not provided sufficient direction and definition of what the entire scanner was to handle. But ASG had not required any specifications in advance, and it was presumed ASG would do what was necessary. (This is the normal scenario when a company needs a non-existing scanner.)

The Field Developed Interfaces (FDI) are problematic from the start. They are not GA quality, and they are not full functioned. They are what I will call "quick and dirty". ASG likes to call it, "Accelerated Delivery" ; -) Let's take this ETL tool scanner. ASG looks at how it was being used, and implemented primarily what they see. They took what they felt to be an appropriate amount of time to understand what was needed. Later your ETL programmers began to use more of the ETL tools built-in power, and the ETL scanner broke. Again, there are all kinds of finger pointing. You don't want this. If a scanner is for an ETL (or any other) tool, it should always be guaranteed to work, regardless what tool features are used.

If the scanner had been GA developed in advance, that would have been great. But ASG's model has not been to develop in advance, but be paid to develop. The problem with this is that it takes time to develop and time to debug, and your Repository schedule might keep slipping all the time you have FDI work being done.

Second Example:

A existing scanner handles version X of the tool it scans. Version X+1 of the tool is released, and the ASG scanner is marketed as handling this new version. You then tell your people to use version X+1 of the tool. Then you find that the new scanner does not support all the features of X+1, but actually only the features that were in version X. So it really didn't support version X+1 completely. You then tell the tool workers to go back and only use version X capabilities, so there was no reason to upgrade to version X+1 of the tool. This actually happened with the ERwin scanner. So when planning your schedules and budgets, do not make too many assumptions. Test everything in advance. That is the reason for the Proof of Concept phase. Test everything.

Who supports a FDI product? Good question. It is supported only by the Tech Services people, not the GA support people. The FDI folks say they have two people fully supporting each FDI "product". I found this not to be always true. We had a case where the original developer of a complex FDI scanner was

unavailable for several weeks, and no one took over in his place. They waited for original developer to return, even though a crisis was in place for us.

A goal of a FDI product is to have it go "GA". That means it undergoes much more rigorous development, testing and documentation. It then is supported by the ASG standard support staff. But all too often, a FDI product does not make it to GA status, or it takes way too long to go GA. From personal experience, it is very disappointing to have a FDI product stay FDI when you expected it to go GA.

40. Set up Dev, Test and Prod Systems

There are two types of "Dev, Test and Prod".

- **The hardware platforms.** These have no relationship to the actual metadata. They are for staging new versions of the software, drivers, operations systems, program patches, configuration changes, and such.
 - The Dev platform is for software developers. They make a lot of changes, and usually work with test/staged data.
 - The Test platform is for integration with other applications and changes, and usually uses a copy of production data
 - The Development platform has the actual data plus all the applications and configuration changes implemented
- **Data Life Cycle on the Production Platform.** The production platform is the only platform that is used. It contains three conceptual areas for metadata to match the Life Cycle of the software and data sources in your organization. There is a following section on data Life Cycle.

41. Customizing a RIM

Each scanner comes with a RIM, which is sort of like a set of relational tables, which will hold the "scanned" metadata.

But if you want to start adding business metadata, and in general things a scanner cannot do, and to handle spreadsheet loading, you have to create your own schemas. You should be able to follow a roadmap showing what needs to be done to make impact analysis handle them. Ask for that roadmap.

The direction of links is important. You need to decide if you want to use NameSpaces or for what portions of your data. You need to determine how you will load this data, what organizations have it, who will specify what is needed and who in those organizations get the data for you? Data Stewards come into play. These all need to be a part of the full Data Life Cycle so data can be added, modified or subsequently deleted while following policies and procedures.

42. Running Scanners

You must read the entire documentation on each scanner you think you might want. Documentation quality, details and clarity varies. There are usually options which must be set. Many people just take the defaults because the choices seem complicated, and never change the settings, some not even knowing what options are available. If you want to get the most out of your scanner, you must decide what you want it to do for you. If it doesn't do what you need, write or have someone else write it for you.

Then install the scanners with ASG. Do NOT let them install them on their own. Your staff needs the experience. ASG may make it look simple, but in reality it may not be simple.

Then the scanners must be run. Again, you must run the scanners, not ASG. You need to see all the steps and things you will be required to do for each scanner run. This will allow you to better understand the processes and procedures you will need to implement, plus the time it takes end-to-end.

If you just sit back, and tell ASG to do it, you may never be able to manage your repository yourself. Being self-sufficient is very important when it comes to your repository.

How does each scanner fit into full loader automation? Is there a full automation framework that comes with the repository? There is more on this in *Automating Scanners* below.

43. Loading Custom Metadata

ASG's scanners load in the "easy stuff". It brings in columns, tables, databases, and portions of extracts from various modeling, ETL, programming and business intelligence tools. What they don't do is connect them somehow all together. That is the hard part. It does not also go and get and organize all the business metadata. You have to do that.

Suppose you have a file field that is input to a Java method. How will the two metadata items reference each other? Good question. The Java scanner won't do it. You must do it by hand or by creating a spreadsheet by hand and loading it.

Suppose you have a table column that has a validation. How will you know it? Suppose it has a list of allowable values? Where will they be stored in the repository, and how will they be linked to the column. What business rule controls that list of allowable values, and what processes use that business rule?

These processes, business rules, validation rules, processes, on and on, must all be defined in your RIM and loaded with complete Data Life Cycle.

There must be full validation. If you are loading a government regulation, and it refers to 10 business processes, first who determines this? When you load the regulations, can you be sure the 10 business processes, presumably previously loaded, have indeed been loaded? How can you tell? Your loader must tell you of anything that is wrong. Rochade can load relationships to data that does not exist, and it won't flag an error. But if the linked to data is supposed to exist, and it does not, you need to know that in advance. As soon as your users start seeing bad data with links/relationships to missing data, they will increasingly distrust your repository, and use is less.

You will then need to add all this custom data loading into your fully automated loading system, too.

ASG has a spreadsheet loader, but it does not do full validations.

August Computing has [RoLoader](#), which does validations and is a loading system, not just a loader.

44. Data Life Cycle Management

Everything changes. Nothing stays the same. Even if you are successful to where you can load data (columns, applications, the 100 types of business related data, etc.), things change. The applications that implement business processes all have a Life Cycle. Some are in production, some are in development and some are in test. They have configuration and change management processes that are enforced. There are source vaults (Clearcase, SourceSafe, CVS, etc.). There are policies and procedures for how to check out software, how to develop, how to check in changes, and how to submit to testing. Then there are procedures on how to test, what to do if parts fail, etc. This is complicated, but necessary. All this is necessary because you don't want bad software prematurely put into production.

The same is with your Metadata Repository.

Applications use databases, files, message queues, etc. Tables get added, columns get deleted, columns have their types changed. Applications can implement new validations based on new business rules. New applications are created. Some applications/databases/files/processes/procedures, etc. are deleted or changed.

If you're normal you will probably have 300-400 ItemTypes (different types of data) in your Repository. All the data in them must be first added, then

periodically changed, and finally deleted, all part of the life cycle of the systems to which they belong.

How will you put your Metadata Life Cycle in parallel with your application and systems life cycle? This is unbelievably important.

If a set of database/column/business data changes are proposed and loaded, is there a way to view the repository such that you can see what the repository would look like if the proposed changes were to be made production? There must be.

Typically only the changes are loaded into the "development" part of the repository. They are analyzed and eventually accepted. Each project has these proposed changes. If you have 10 projects, then there were be 10 different development clusters of changes. After each project's proposed changes are accepted, they needed to be put into an integration development stage. Here you should be able to answer the questions of ALL changes being worked on right now made it into productions, what will the repository look like? Then they must be promoted into test, when the software and actual databases (along with this SQL create scripts and data loading). That is, the application and the metadata life cycle need to work in sync. Finally when the applications in test are approved, they and their metadata need to be promoted to production.

Life cycle should also come with a large number of reports and utilities to handle special cases.

Rochade does not currently come standard with Life Cycle. They recognize this shortcoming and have some ideas of what to do. Traditionally you had to pay ASG Technical Services to make you a custom Life Cycle for you, which is expensive and delays your repository going into production. You might find a Rochade product that has as part of it an embedded life cycle, but be sure to determine it is robust, and completely general enough to meet all your requirements. But this application will probably have an associated cost. Rochade comes in a lot of bits and pieces, and they each cost. You want to maximize your capabilities while minimizing your costs. Remember that it is already incredibly difficult to get a positive ROI even if Life Cycle came out-of-the-box. Adding more costs will make it all the more difficult.

45. Automating Scanners

You may believe that since a scanner can be put in a batch file, that automating the scanners is easy and won't take much time at all. You need to read this to understand what a real loading automation system needs to do. [Click here.](#)

There is a difference between putting a loader program call into a batch file, and having a complete automation system. ASG does not have an automation system out of the box. So you will have to do this yourself. This will take probably 2 people 3 months for real, complete automation. Remember, some scanners run on a mainframe, others on Unix and others on Windows. Your environment will probably have all three. Loading typically requires certain sequencing, so getting them in the right order is a challenge. Then there is error reporting and tracking.

Why does it need to be this complicated? Simple. You have way more things than you can handle by hand. Suppose someone says you have to load 3000 applications and all their associated business metadata, and you have to do it in a short time. If each application requires 10 different scanners to run, you have 30,000 scan/loading operations to perform! Without complete automation, you are in trouble. Remember, before you finish loading, the first things you loaded will be changing because of their Life Cycle. So you will have to re-run scanners on the changed data. Remember, you are trying to reduce your head count, not add to it.

46. Using ASG Technical Services

You will probably find Rochade so complicated and so unlike any other technologies you have ever used, that you may think you may save time by having ASG Technical Services do lots of your work for you.

The mark of a good system is that it is easily learned by you own staff.

This should raise a red flag.

But for whatever you do with Tech Services, be sure to write a very definitive contract being very specific about what you want, the SLA and penalty clauses.

This may sound harsh, but it's not. See the IBM example below.

47. Personnel Concerns

Initially Rochade looks very appealing and new. That may attract some of the brighter talent you have to join your project.

A problem you may find is that many talented people want to learn skills that are in great demand, for job security. Rochade is really different. It's nothing like relational database and its concepts are unique and proprietary. That means your staff may find Rochade a niche product, and then decide to leave to work on more mainstream products.

Also, if you tend to outsource development to ASG or foreign companies, then your staff may feel the challenging and fun jobs are outsourced and the un-fun jobs are what are left. You may need a strategy for dealing this.

48. Outsourcing Your Rochade Repository

You may start seeing more costs, time, people and other resources that are required to support a Rochade repository than you had not imagined.

You may consider outsourcing the entire repository. That is not a bad idea. There are companies that are outsourcing their entire Rochade repository today.

Even though the outsourcing company already has Rochade, you will probably still have to buy all the software anyway. So you won't save any money there.

Also all the custom RIMS, and getting the data to load (the hard part) will still be your problem. For example, you scan a database and it has 10,000 columns. They have names like NL32PTN. Where will you get the actual definitions of what that means? Obviously, you will need a small army of your own constantly meeting with all the other parts of the company. Many of the people who were part of the original development are long gone, and you may never find out. You may have to sift through program code and guess what the columns are. We're talking tens of thousands of columns per database. For your enterprise, you're talking probably about 100,000 columns! This will probably take you a minimum of a year just to find the data, document the programs using the data, linking them to validations and business rules, processes, procedures, etc. Actually, it will probably take much longer, depending how thorough you want be.

Likewise when you start collecting all the various business metadata (maybe 100 Item Types), who will get the data, and who will determine how the separate data are supposed to relate to other loaded data? Who will actually know they are related? Again, it's the various departments in your company.

So you will need to do all the development, Life Cycle implementation, scanner automation, etc., anyway. This may take 6-12 months. Also, you will have to define the processes and procedures to the outsourced to company on how to load your data.

So how much will you be saving? Not a lot. You do all the hard part and the company you outsourced to merely follows your instructions loading the data. If anything goes wrongs, you may look bad, they'll look good because their job is nicely defined. So how much money will you save? Good question.

49. IBM Example: Partnering and Cost Containment

A large company wanted to do an SOA implementation. So they partnered with IBM. This is not the "partnered" some people refer to with ASG. This is a "real" partnership. IBM and the company entered into this jointly. What is best is that there is a bonus/penalty arrangement. Certain representations were made by IBM about how good the system was, the number of people required and the time it would take. SLAs were created. Here is the best part: if the partnership beats its deadlines (there are cost savings), IBM gets a bonus. That is their incentive and profit motivation. But if the agreed upon targets are missed or more people are needed, then there is a penalty for IBM.

You might consider getting ASG to do the same. ASG is always saying how easy and fast it is to get a full function repository up, running and into production. They shouldn't have any problem agreeing to such a partnership contract.

50. Concluding

By this time, you probably have concerns, as well you should. That is why these topics have been given to you to think about as you. Repositories are not easy to build. [Click here](#) and [here](#) for examples. The more you know, the higher the chances of your repository being successful will be.

A lot will depend upon how you manage your company's expectations for the repository. Promise a little, give yourself a lot of time (at least a year or two).

Of course, you might determine that a Rochade Metadata Repository is not for you right now. It is evolving, and you might want to look at it again in a couple of years.