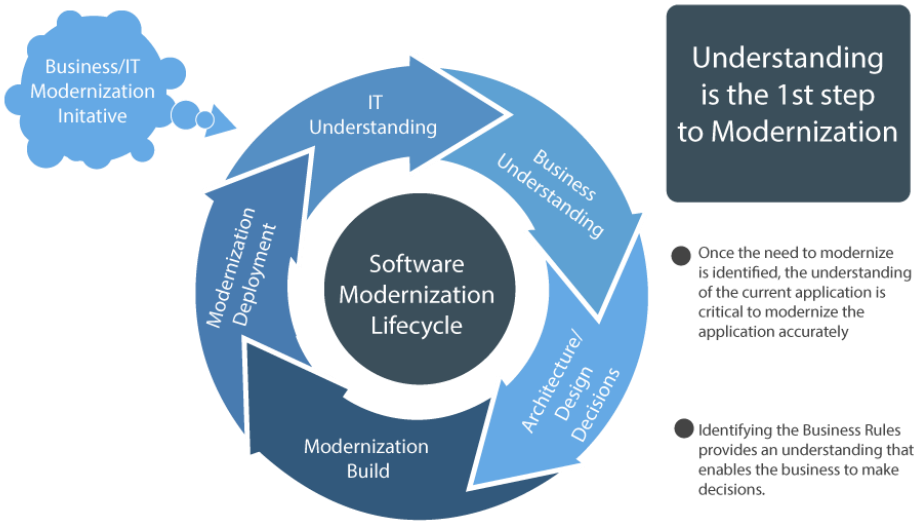
Finding Business Rules in COBOL Systems

Use Case for CM evolveIT

Understanding the first step to Modernization



Large commercial organizations have been developing software to automate business processes specific to their needs since mainframes were introduced in the 1960s. Indeed, many legacy systems have been successfully running for more than 30 years in original software technologies offered with the mainframes in the 1960s and 1970s. Arguably, the most successful legacy mainframe technology is the COBOL programming language. For many organizations, COBOL systems remain the lifeblood of day-to-day operations; however, increasing support costs, lack of application flexibility, and the shrinking pool of available COBOL resources has led most organizations to seek more modern technology alternatives to replace their COBOL systems. For IT, that means choosing the right legacy modernization strategy.

When it comes to Legacy Systems modernization, one size doesn't fit all

Strategy	PRO	CON
Extend existing COBOL application	It's how the business runs today and requires less organizational change	Agility, Costs, Retiring Resources
Migrate COBOL to run on lower cost platform	Significant savings on hardware and costs. No significant change to current processes	Still COBOL: Agility, Retiring resources
Third Party Package	No longer in the "line of code business"	Does the package functionality meet the needs of the business?
Code Transformation	Language migration relatively straight forward	Architecture migration to object oriented paradigm quite difficult
ReWrite	Opportunity to improve process and system functionality	Cost and Risk

System modernization decisions frequently *cannot* be made in isolation. In many cases, particularly when the modernization decisions imply a continuation of key system process essential to the business a shift in

architectural style can have significant impacts on the entire infrastructure not just the migrated systems. Steps must be taken to ensure continued process viability during the entire systems life cycle, not just the initial modernization phase. Understanding the existing legacy environment is a critical first step. It provides an objective framework for making modernization decisions, and can also provide sufficient information to help plan the effort. A traditional top-down approach helps capture at the high-level definition of business process. Collecting more-detailed information about applications, their components and their interrelationships and the business rules they enforce is critical to success.

What are Business Rules?

Rules and Logic− are often confused. Business Rules− *affects a business outcome*. Business Logic− *enforces a business rule*

- **The Rule** *Tells you what the business must do*
- **The Logic** *Tells you how the rule is enforced*

Why is the distinction necessary? Rules are free of process and architecture. They are implicit in code. Logic is tied to process and architecture. On-line systems will typically apply all the rules to each customer in turn. Batch systems will typically apply part of a rule to all customers, and then apply another part of a rule at a later time before an outcome can be determined. Mixed systems often have duplicate logic, share logic and even have logic that adjusts the results of badly shared logic! The actual logic enforcing a business rule can thus be overly complex, distributed amongst many paragraphs, sections, programs and time. The rule/logic relationship is rarely documented, so changed over time adds to complexity and confusion.

---

1. **Only IT systems know what actually happens**
2. **Other sources provide an easy 'heads up' on what to look for**
3. **Ultimately you need to mine code to find or verify rules, look for exceptions & check that systems do what the business expects.**

Why is it hard to find Business Rules in traditional Mainframe analysis approaches?

Mainframe legacy applications are generally older systems developed with an architecture focused on the processing considerations and with little focus on organization of the system from business process perspective. As a result, logic that supports a particular business process or business rule is spread out across the entire system. This makes it difficult to find all of the logic that supports business rules using typical mainframe scan utilities through the manual process of reviewing search results and reading through individual programs. Further complicating the problem, is that input and output data that is relevant to the business is, renamed, reformatted and reused as it gets processed within the mainframe application.

Complexity makes this difficult

Most legacy systems may have hundreds or thousands of changes over 20-30 years that make this even more complicated. Finding business rules in code is a bit like looking for a single needle in a field of hay stacks. Individual applications have grown in size to millions of lines of code. Documentation has become outdated and subject matter experts have moved on to other roles. As a result, the use of existing mainframe tools, reliance on SME knowledge and the manual process of scanning for answers can make mainframe modernization initiatives too high risk.

Technical complexity is compounded by a:

- Lack of modern analytical tooling to enhance understanding of applications
- Lack of tooling that embraces diverse application portfolios
- Inability to transition knowledge across teams or to new team member.

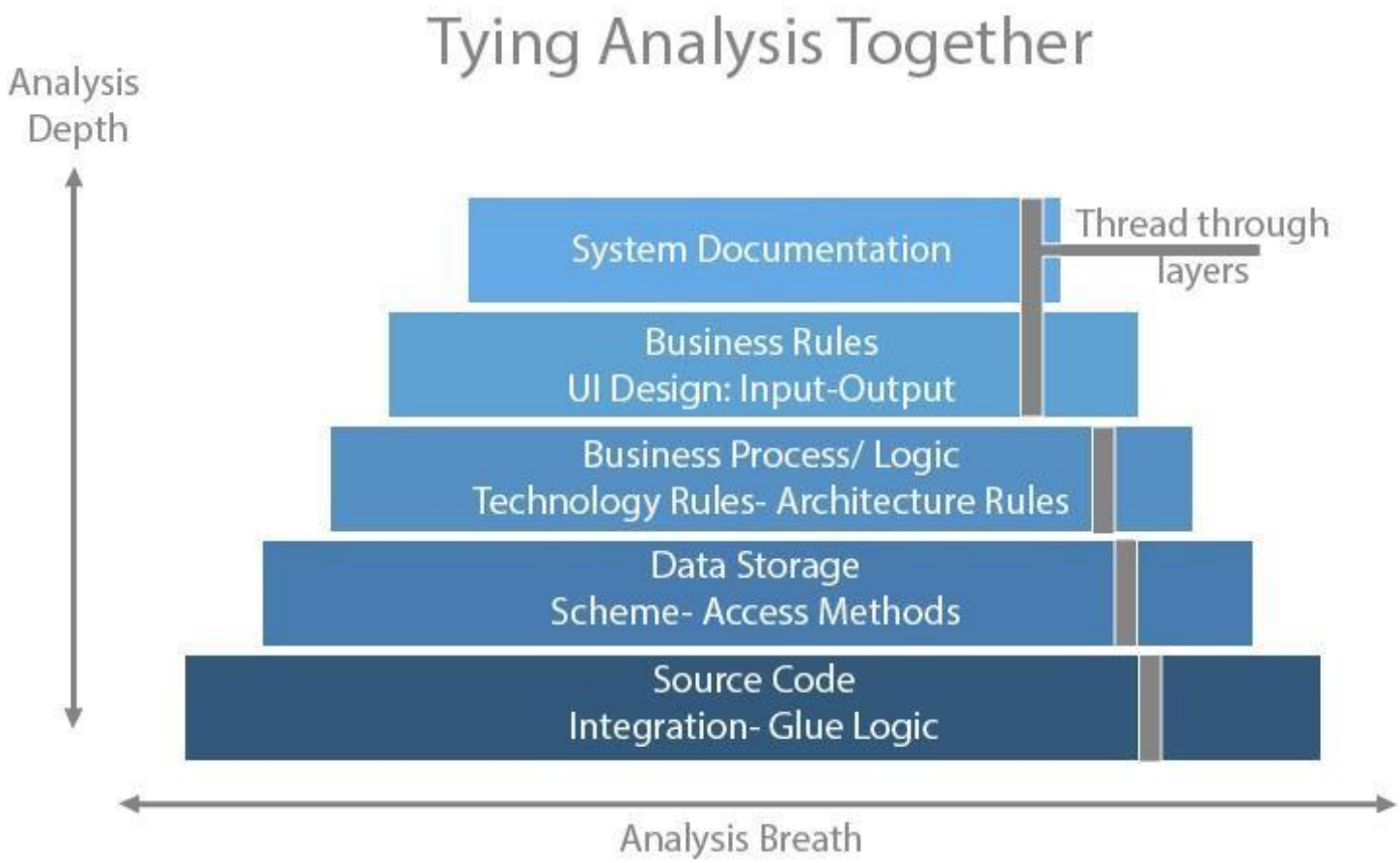
Business complexity

This decline in technical understanding is multiplied by a loss of business understanding. Business users simply cannot keep track of nor communicate all of the varied processes which have been added over the years. System changes to support special customer requirements, new business models regulation changes, etc. can lead to hundreds of variations in the business rules. This becomes especially problematic in companies with highly dynamic business models. Business complexity is compounded by a:

- Lack of common vocabulary between business and IT
- Limited understanding of business functions within applications
- Inability to abstract business understanding out of technical documents

Disbursed Intelligence

The information regarding key business rules is rarely centralized or even linked to applications. External system process may be captured in spreadsheets or other mediums but often are undocumented and are only known by the key users of the system. Compounding the challenge is that there isn't a mechanism to connect various information sources for more complete view.



Solution with CM evolveIT versus mainframe tools

Mainframe Approach

So what is a business rule and how does one find them in a legacy application? A business rule is not typically a small set of code that resides in a single program. A business rule is most often a sequence of code snippets that are spread out across the multiple components and millions of lines of code that make up your mainframe application. With this understanding, it is easy to imagine the challenge of finding them using current mainframe scanning tools. Typically, all that's known about a business rule is that it results in an output of a field on a screen or report. With existing mainframe tools, an analyst must scan to find the output in the system and conduct repeated scans through millions of lines of code. The analyst must gather bits and pieces of information that must be manually organized to "connect the dots" through data name changes, decision logic branches, file and database input and output and connections between different mainframe components. This iterative process is inefficient and prone to inaccuracy because it depends upon the analysts system knowledge and the capability to organize these bits into something meaningful. Some code elements may be missed or overlooked. Others may not be recognized as relevant because of data name changes, links through files, databases and so forth. Once this scanning and manual effort is complete, the analyst must then spend additional time putting their analysis results into some form that is useful for communication to the project team or management.

CM evolveIT Approach

In contrast, CM evolveIT provides the connections between each Business Rule "Code snippet" in the CM evolveIT repository through automated analysis of your source code. Each snippet has already been located and the connections necessary to follow the path between them in your code have already been made through CM evolveIT's compiler-like parsing technology. The challenge of manually identifying data name changes, logic branches, data store reads and writes from the analysis process has already been eliminated before your analyst begins the analysis process using CM evolveIT. Using a screen or report field from the system, CM evolveIT provides a point- to−point analysis capability that enables the analyst to accurately trace and document business logic paths that automatically eliminates the "noise" of unrelated code and leads the analyst to only the relevant code snippets and business logic that implements the report field. Additionally, CM evolveIT provides the ability to substitute business relevant names for data fields, making cryptic data names in code
understandable to a business person. Better still, your analyst need not spend additional time documenting their analysis in flow charting tools or organizing textual reports in word processing applications. CM evolveIT documents analysis results in diagrams, reports and exported documents as the analysis is conducted. All analysis done in CM evolveIT is "selfdocumenting" due to the interactive diagrams, reports and source code views. Documentation can be either stored within CM evolveIT or exported to share it with peers and manageme

CM evolveIT case study

Results with CM evolveIT VS mainframe results

A large insurance company allows their annuity policy holders to take out loans against the value of their annuity policies. The company issues annuity policy letters to policy holders on a quarterly basis, whenever a change to the annuity policy occurs or when a new loan is taken out against the value of the policy. While there is a general understanding of the business rules that govern when an annuity letter is scheduled or requested, there is little understanding of the rules that
govern the population of the loan amount. It's known that several conditions exist that determine how the field is populated, but neither the specific logic decisions made for each of those conditions nor the origin of system data used to populate the loan amount under those conditions is known. In order to identify the complete set of annuity letter business rules, the answers must come from the source code.

In our example we will be documenting the business rule related to the calculation of a LOAN AMOUNT field on an Annuity Letter. We will compare the time required using mainframe tools versus CM evolveIT to find all population paths for the annuity letter loan amount using the output file containing annuity letters PLF.LA.DATA.DAILY.ANNLET:

Traditional mainframe analysis approach

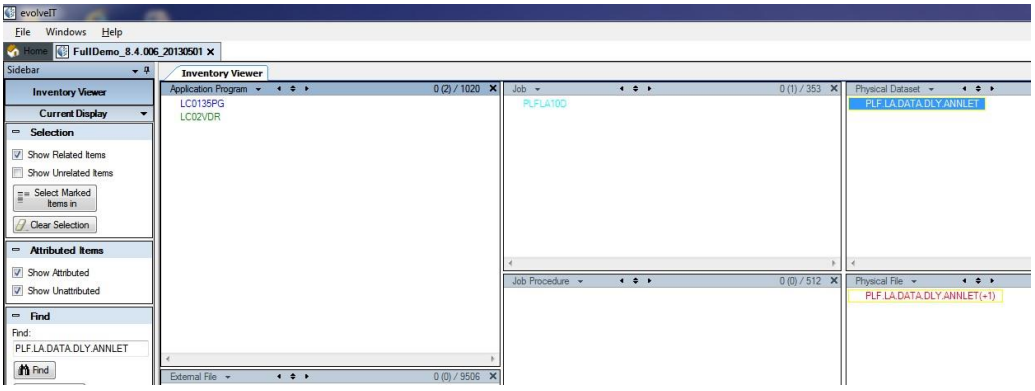
(Elapsed time: 3-5 days depending on Analyst experience)

1. 7-10 Component library scans to identify jobs, job procedures and programs	2. Manual review of dozens of programs with data field references to determine which scan hits are pertinent to the analysis.
3. Dozens of manual source searches in code editor to find the specific statements associated with data population paths.	4. Dozens of manual searches in code editor to find and connect the business logic conditionals that govern the data population.
5. Manual capture, correlation and sequencing of all analysis results.	6. The analyst must then create documentation using diagraming tools and word processors to effectively communicate the results.
7. Manually scan through the statements affecting LC2DX-LOAN-AMOUNT to determine which statements populate the field. Screen capture or print these statements for reference.	8. Execute FINDs in source editor for each of the data items being moved to LC2DX-LOAN-AMOUNT in the statements identified as populating statements.
9. Screen capture or print these statements for reference.	10. For every statement found for WS-LOAN, execute finds on all fields moved to or used in calculations of the value of WS-LOAN (3 fields APL-AMOUNT,
	LOAN-AMOUNT, CF-IMPAIRED-AMOUNT).

The above steps are required to find only the statements that populate the loan amount field. No useful documentation has been generated. Additional time is required to document the job, job procedure, program and files associated with the annuity letter process. Flow charting of the component relationships also requires additional effort.

Steps to find the same information in CM evolveIT:

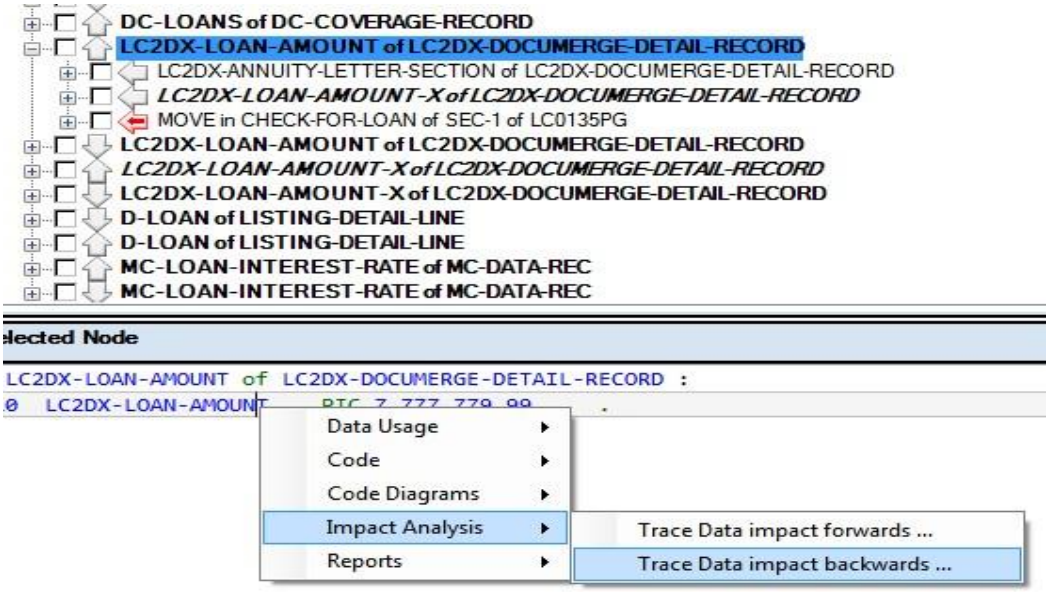
Elapsed time: Less than 15 minutes



1. **Execute a find on the filename PLF.LA.DATA.DAILY.ANNLET in Inventory viewer and use select items in to select the file found.**

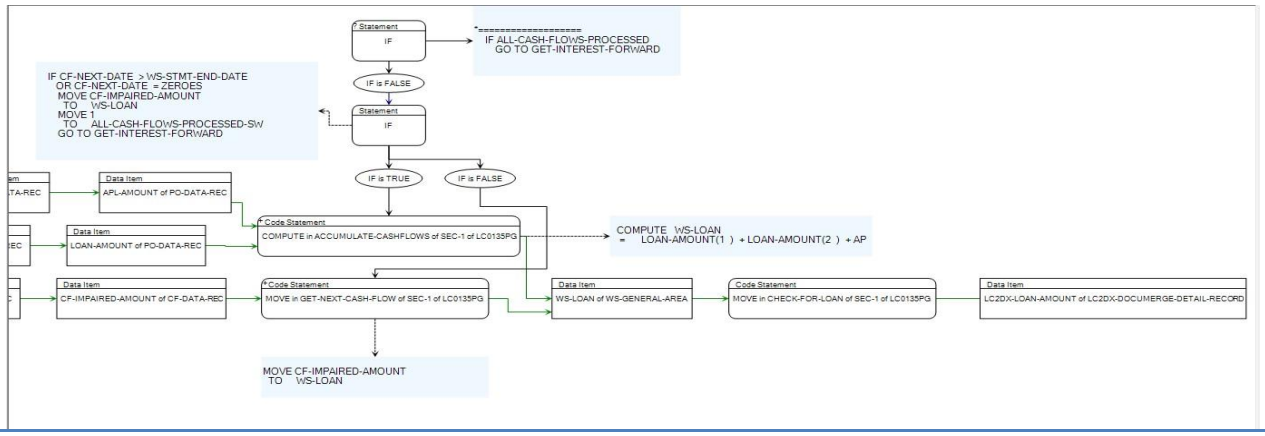
2. **Select program LC0135PG that has been identified as the program writing to the business relevant output file (BLUE colorization).**

3. **Select program LC0135PG, right-click and select DATA SEARCH from the sub-menu and execute a search on a partial name "LOAN". With a short investigation of the LOAN related fields in the Impact Analysis we find that the correct field to work with is LC2DX-LOAN-AMOUNT since it is being calculated as the output data name.**



4. **Right-click on the interactive field name "LC2DX-LOAN-AMOUNT" in the Code View window and execute a backward trace.**

5. **Now expand the diagram backwards twice to find all statements populating WS-LOAN and LC2DX-LOANAMOUNT. With this expansion we find the COMPUTE of WS-LOAN which is moved to LC23DX-LOANAMOUNT as the output field. We also expand the decision logic that determines when the COMPUTE is executed and capture this additional business rule elements.**



The Resulting Business Rule:

- If it's not the end of Quarter
  - ✓ Update the Loan Amount with additional Impaired Amount
- If it is Quarter End
  - ✓ New Loan Total = LOAN AMOUNT 1 + LOAN AMOUNT 2 + Original Loan Amount
  - ✓ Format New Loan Amount for Output to the Annuity Letter

With CM evolveIT interactivity and point- to−point analysis capability, all population paths is now an fields associated with the population of LC2DX-LOAN-AMOUNT can be found, traced and documented in less than 15 minutes. The trace diagram can be captured for external use or saved with analyst notes within CM evolveIT.

CM evolveIT VS mainframe methods

To compare the effort required and the results from a traditional mainframe analysis approach, consider the following:

Time

Efficiency and speed of analysis is a key factor in large mainframe initiatives. CM evolveIT clearly provides an advantage to the analyst because it provides a point- to−point analysis capability not available using typical mainframe scan and source view tools. Starting with a relevant business output file, CM evolveIT then leads the analyst along the path of analysis. The savings in time spent scanning, reviewing and correlating source code analysis results in a significant time savings.

Accuracy

CM evolveIT builds a repository of all relationships in your code down to data relationships. While an analyst using traditional mainframe methods has to make decisions based upon what is connected in the code manually as well as what is relevant, CM evolveIT already has that information built into the repository. The analyst does not have to make a decision but merely follow the path that exists in the source code. There is no fear of missing downstream results with CM evolveIT. With the traditional mainframe analysis method, the analyst is responsible for identifying all pertinent impacts, which opens any mainframe initiative to risk. CM evolveIT is clearly a more accurate option.

Documentation

CM evolveIT's point- to−point analysis capability is self-documenting. The interactive diagrams and reports are themselves the method by which an analyst conducts research in your application. Analysis done with traditional mainframe methods and tools requires additional effort to generate documentation. CM evolveIT documents the results as the analyst conducts the analysis. There is no additional effort required. The resulting documentation from CM evolveIT is better than traditional mainframe documentation. Additionally, CM evolveIT provides the ability to store the analysis results and documentation in the tool and ties all of the documentation to the analysis results using analyst notes and business logic terminology functionality.

CM First products

For over 30 years, CM First and the technology underlying CM First's products have been at the cutting edge of providing IT modernization solutions.

In conjunction with our CM evolveIT product we have a unique methodology and services, that enables you to identify where and how business rules are enforced within your operational code and build a fully cross-referenced inventory of business rules and vocabulary alongside the application code that supports them.