CM FIRST
Rethink Modernization

// WHITE PAPER

Capturing Business Rules in COBOL Systems

Demystify decades of accumulated business rules buried in legacy apps

CM evolveIT is the single most efficient, effective system of its kind on the market today for analyzing and documenting mainframe systems. Where CM First's CM evolveIT really sets itself apart is in its unique technology and methodology for the capturing the business rules from existing legacy applications.

CM evolveIT's rules capture identifies specific source code paths that are relevant to the 'business rules' while filtering out the non-essential information found in millions of lines of code. We accomplish this by performing interactive backwards slicing on program results that is specifically designed to weed out the noise of the application then we generate convenient diagrams of the resulting logic. We align the application logic with business process documentation and generate system diagrams that visualize the information so it can be effectively shared, improving communication between IT and the business.

CM evolveIT also provides the ability to substitute data aliases for esoteric data field names adding further "business meaning" and consistency to the diagrams for even more clarity. Furthermore, because CM evolveIT uses the source code from the current production system for its analysis, you'll be assured that the results will be 100 percent accurate based on what the system does, not on what someone thinks the system does.

The CM evolveIT product suite has successfully modeled more than one billion lines of COBOL source code and is often deployed by our public and private sector clients. CM evolveIT users include Blue Cross Blue Shield, Crowley Maritime Corporation, Ministry of Transportation Ontario Canada, Ascena Retail, Allstate, American National Insurance, The Commonwealth of Kentucky, and many others.

Contents

What are Business Rules

In a very real sense, an organization's Business Rules are its business – they determine how it responds to external and internal stimuli, and what outcomes it strives to achieve. The combination of suitably chosen Business Rules (there is probably no such thing as a "right" or "wrong" Business Rule) together with appropriate enforcement policies, result in a successful business. Business Processes define the steps which have to be taken to apply the Business Rules.

BUSINESS RULES AND SOURCE CODE

A business rule is not typically just a small set of code that resides in a single program. A business rule is most often implemented through a series of code snippets that are spread out across the multiple components and millions of lines of code that make up your mainframe application. Furthermore, Rules and Logic are often confused. Business Rules– *affect 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 changes over time adds to the 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 and check that your IT systems do what the business expects them to do.

Why is it hard to capture Business Rules with traditional Mainframe analysis approaches?

Legacy systems embody decades of unique and valuable business rules captured in the source code of the application, but capturing this value can be quite difficult and costly, particularly with third generation languages such as COBOL.

In general, COBOL programs were developed quite a while ago using an architecture that primarily focused on the processing considerations with little focus on the organization of the system from a business process perspective. As a result, logic that supports a particular business process or business rule will be spread out across the entire system which can encompass millions of lines of source code. This makes it difficult to find all of the logic that supports a given business rule using typical mainframe scan utilities. These manual approaches necessitate a process of reviewing search results and reading through individual programs. Further complicating the problem is the fact that input and output data that is relevant to the business is renamed, reformatted and reused as it gets processed within the mainframe application. As a result, it takes humans to interpret what is architecture, business logic or an algorithm that is an element of a business rule.

COMPLEXITY MAKES THIS DIFFICULT

Most legacy systems may have hundreds or thousands of changes over 20-30 years which make finding Business Rules 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 members

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. Neither the business nor IT has a great history of doing documentation. 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:

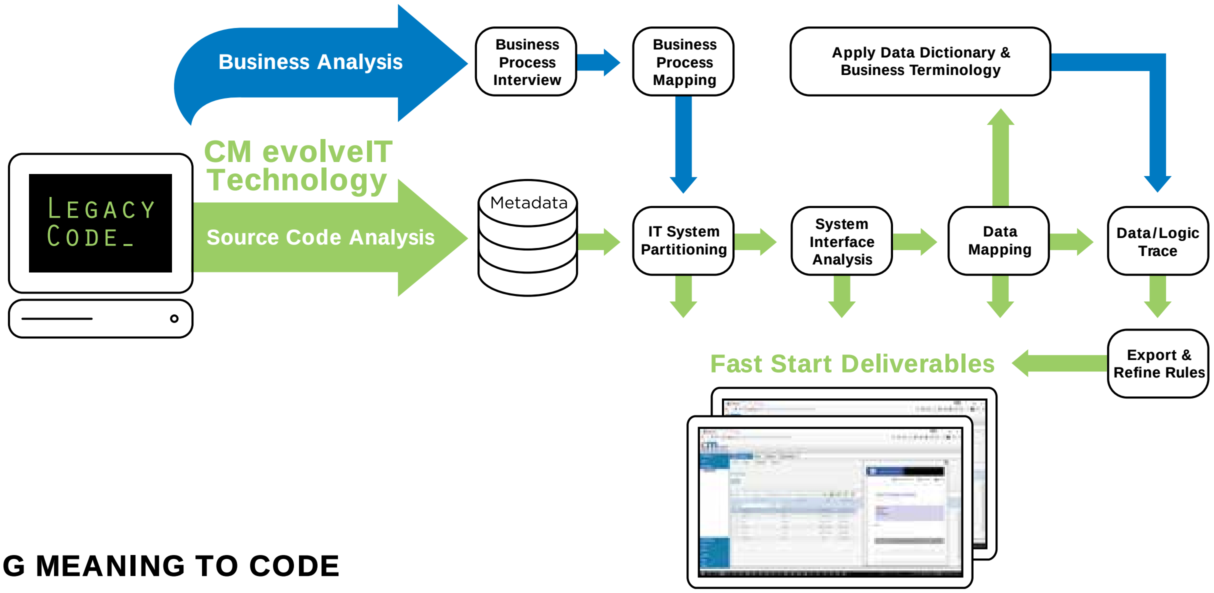
- Lack of common vocabulary between business and IT
- Limited understanding of business functions within applications
- Inability to abstract business understanding out of technical documents.

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. The fact that there isn’t a mechanism to connect various information sources for a more complete view compounds the challenge.

Our Approach

Rules harvesting projects can made be practical with the use of CM evolveIT automation and its proven methodology. Our proprietary methodology on slicing a system is unique in the industry. We have learned how to work with customers to understand how to extract rules in a variety of contexts and situations.



ADDING MEANING TO CODE

To complete a business rule extraction project, one needs strong support from the business analyst. The analyst can then see the programmed influences on the result, and decide which ones are fundamental to the business and which are simply supporting technology. The key notion is to trace system output data backwards through the control and data flow paths that produce or affect it, using automation to handle scale and complexity. This lets the analyst focus on the essence of the computation and its conditions, while the tool handles the problem of navigating through complex mountains of code.

CM FIRST METHODOLOGY PHASES

- **CM evolveIT Automated Modeling of System Source Code**
  Complete process to build all source code into CM evolveIT model-database.
- **System Assessment and Metrics Reporting**
  Generate diagrams and reports which provide context for the overall system.
- **Identify the Business Processes that “Touch” the Application**
  Work with the BA and Business users to identify the use cases for the application and the Business process for each use case.

Benefits

- ✚ Enables building a business vocabulary.
- ✚ Finds unknown rules implemented within the application code.
- ✚ Reduces risk with a phased approach.
- ✚ Filters out non-relevant processing through the rules extraction process.
- ✚ Improves communi-tcations between IT and Business Analysts for as is process.
- ✚ Enhances new system object oriented design by isolating data objects and uncovering methods for new design.

CM FIRST METHODOLOGY PHASES (cont'd)

- **Identify Application Inputs & Outputs**

  Data structures relating to input and output Screens / Reports / Files, databases, etc.

- **Map the Application Inputs/Outputs to the Business Process**

  Connect each data structure to the Business Process that interacts with it.

- **Model the User Interaction with the Application**

  Identify Business Process terminology that will help in interpreting the rules from the trace information.

- **Trace the Input/Output Business Data**

  Using the CM evolveIT Data Logic Trace function, trace the input/output data that is of competitive advantage to the business.

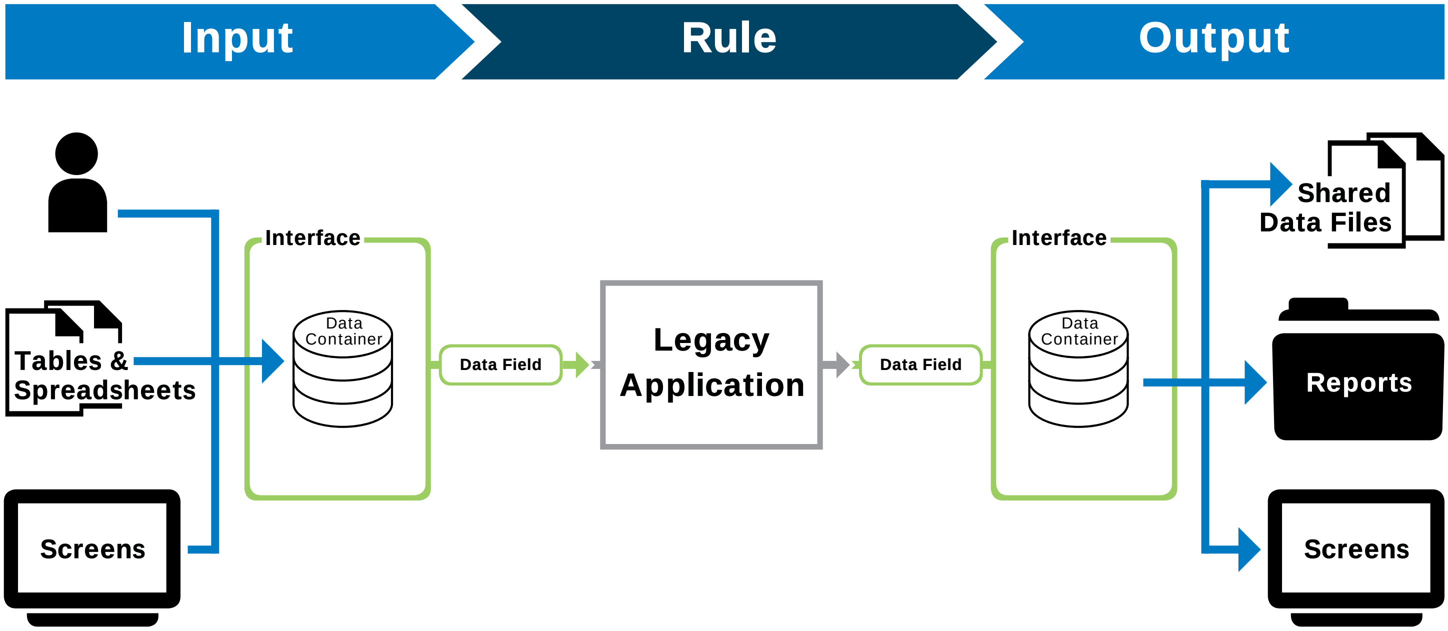
- **Export the Input / Output-Related Processing Paths**

  Export to Snapshot diagrams.

- **Build Business Rules and Supporting Documentation**

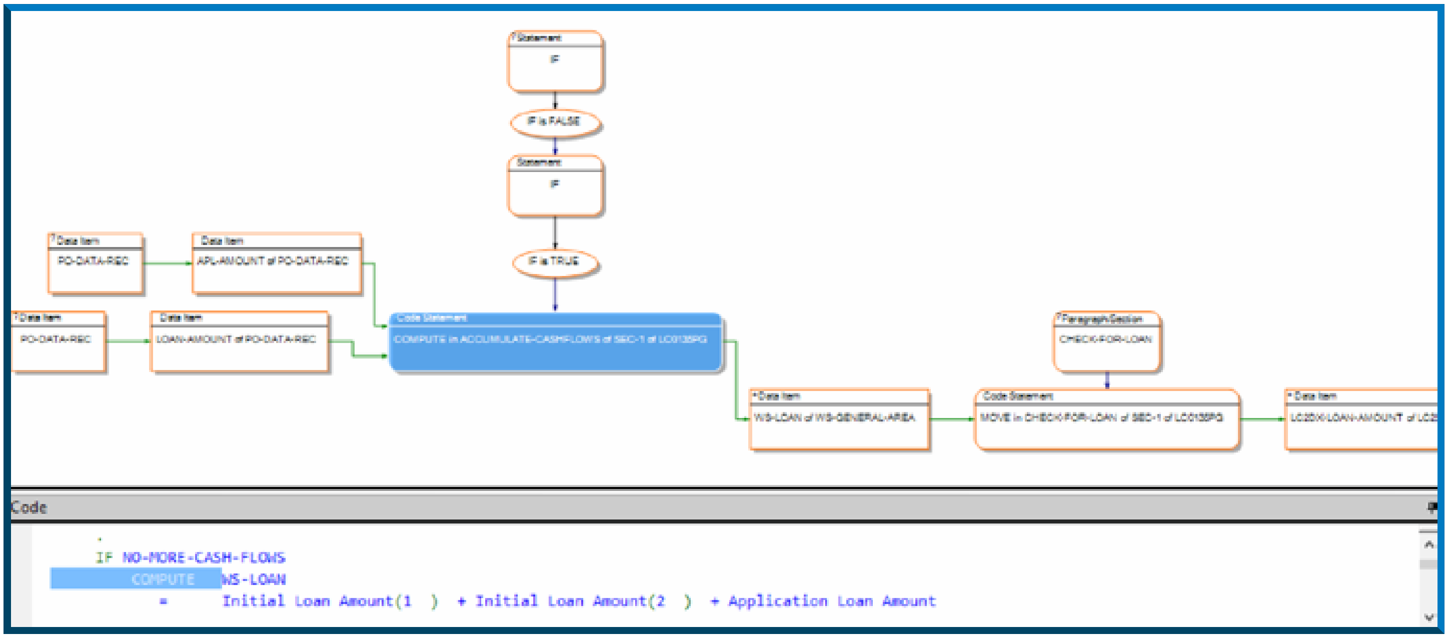
  Combine the business terminology and Business Process understanding with the Data Logic Trace information to build Business Rules.

CONCEPTUAL VIEW OF CM EVOLVEIT DATA LOGIC TRACE



Deliverables

There are two kinds of deliverables in the CM First process. The first are the deliverables that are produced prior to creating business rules and the second are the business rules. The first set are generated using CM evolveIT and provided out of the process in order to speed the process of understanding program logic related to each of the relevant outputs of the system. These first deliverables feed into the business rules generation process by providing the BA/Developer Analyst with the understanding of the code logic relating the rules. When this information is combined with the business context/terminology and a human understanding of the business use cases, then business rules can be produced.


Statement
IF
IF is FALSE
Statement
IF
IF is TRUE
Data Item
PO-DATA-REC
Data Item
APL-AMOUNT of PO-DATA-REC
Data Item
PO-DATA-REC
Data Item
LOAN-AMOUNT of PO-DATA-REC
Code Statement
COMPUTE in ACCUMULATE-CASHFLOWS of SEC-1 of LC0135PG
Paragraph/Section
CHECK-FOR-LOAN
Data Item
WS-LOAN of WS-GENERAL-AREA
Code Statement
MOVE in CHECK-FOR-LOAN of SEC-1 of LC0135PG
Data Item
LC20XI-LOAN-AMOUNT of LC2
Code

```
      .
      IF  NO-MORE-CASH-FLOWS
           COMPUTE  WS-LOAN
               =      Initial Loan Amount(1  )  + Initial Loan Amount(2  )  + Application Loan Amount
```

Figure 1.1 Data Logic Trace Diagram with source code and Business Terminology inserted

Primary Deliverable from BA:

Business Rule:

If all processing of Annuity Loan entries is not completed:

Total Loan Amount = Initial Loan Amount + Second Loan Amount + Application Loan Amount

Output the new Total Loan Amount to the Annuity Letter

SUPPORTING DOCUMENTATION

CM First uses a visual platform to design and construct diagrams to provide additional system information. This progressive approach allows CM First analysts to provide full traceability and change control with flexible documentation options including HTML, RTF, and JSON formatting.
(See Figures 2.1 and 2.2)

Example of Visual Elements Often Included as Parts of Systems Diagramming Packages

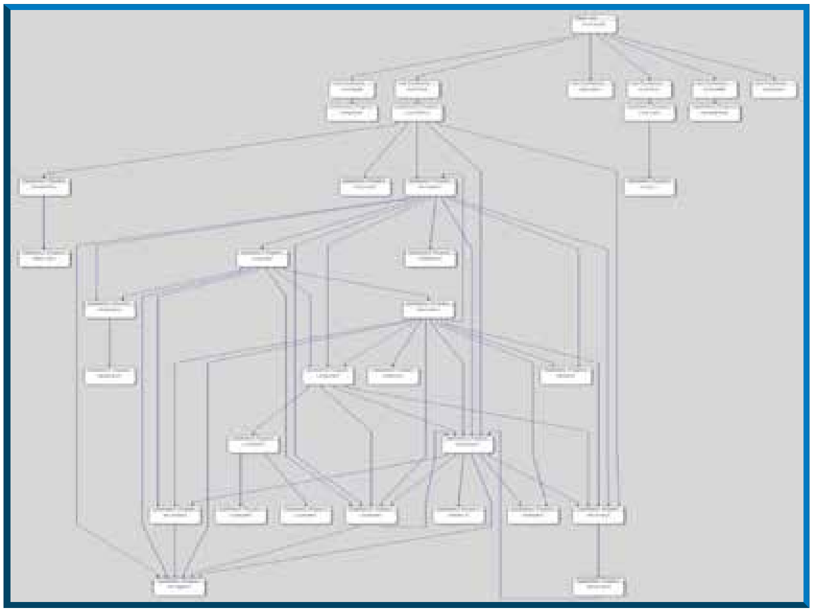


Figure 2.1 Sub-Program Call Tree, All Calls

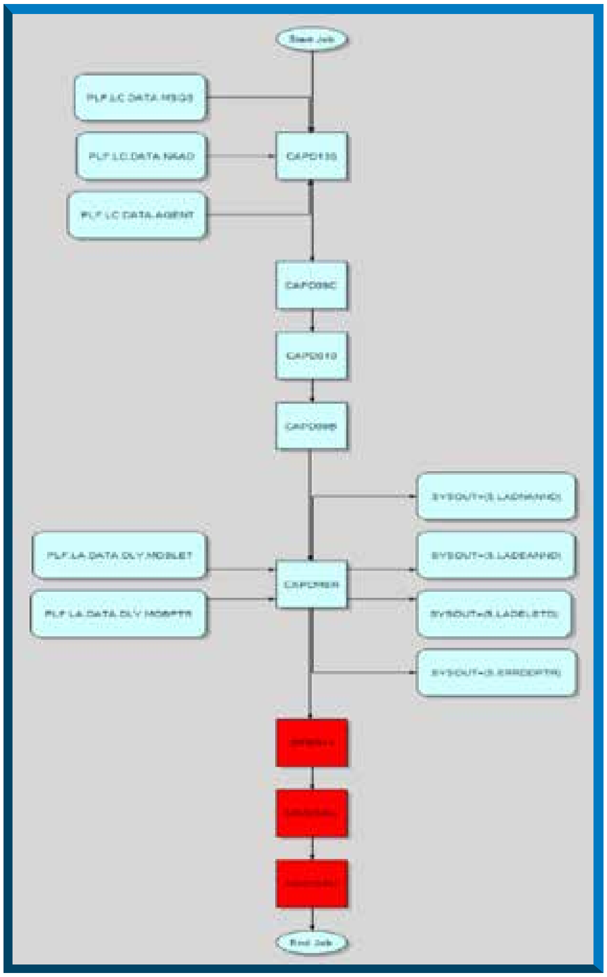


Figure 2.2 Job Flow Diagram Showing Job Steps and Input/Output Files

CM First’s modeling environment can accommodate a variety of user-defined system parameters. Perhaps the most important aspect of CM First’s approach to system documentation is our focus on customer collaboration in order to capture existing expertise and knowledge.

Examples of Possible Source System Diagram Components

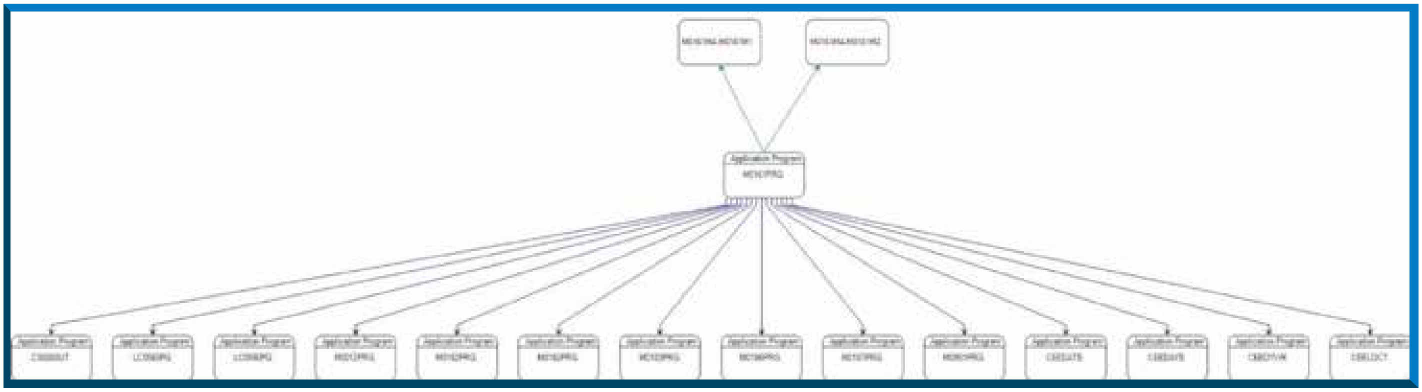


Figure 3.1 Screen-Program – Subprogram Relationships Diagram

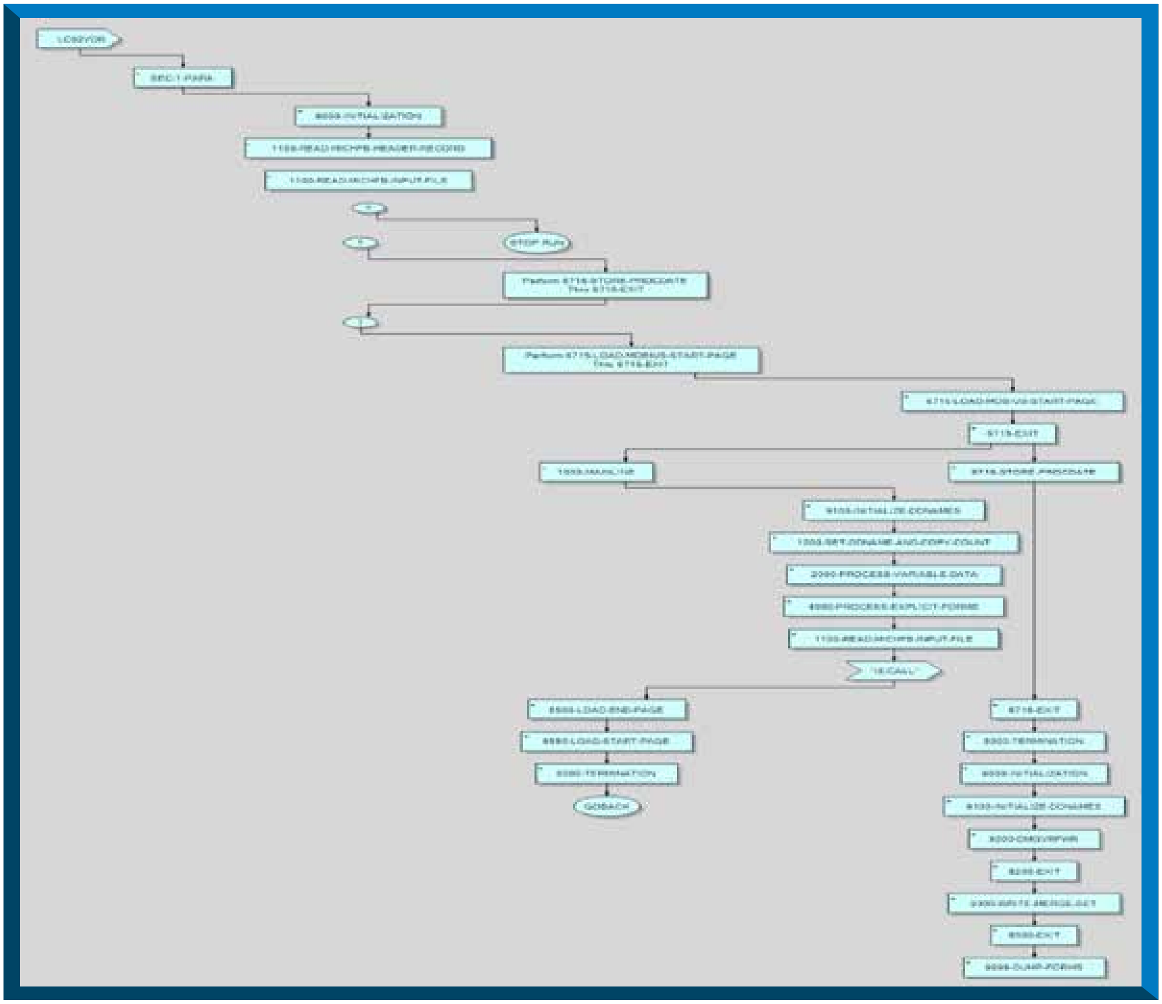


Figure 3.2 Program Control Flow Diagram with Decisions

Examples of Data Flow Diagrams

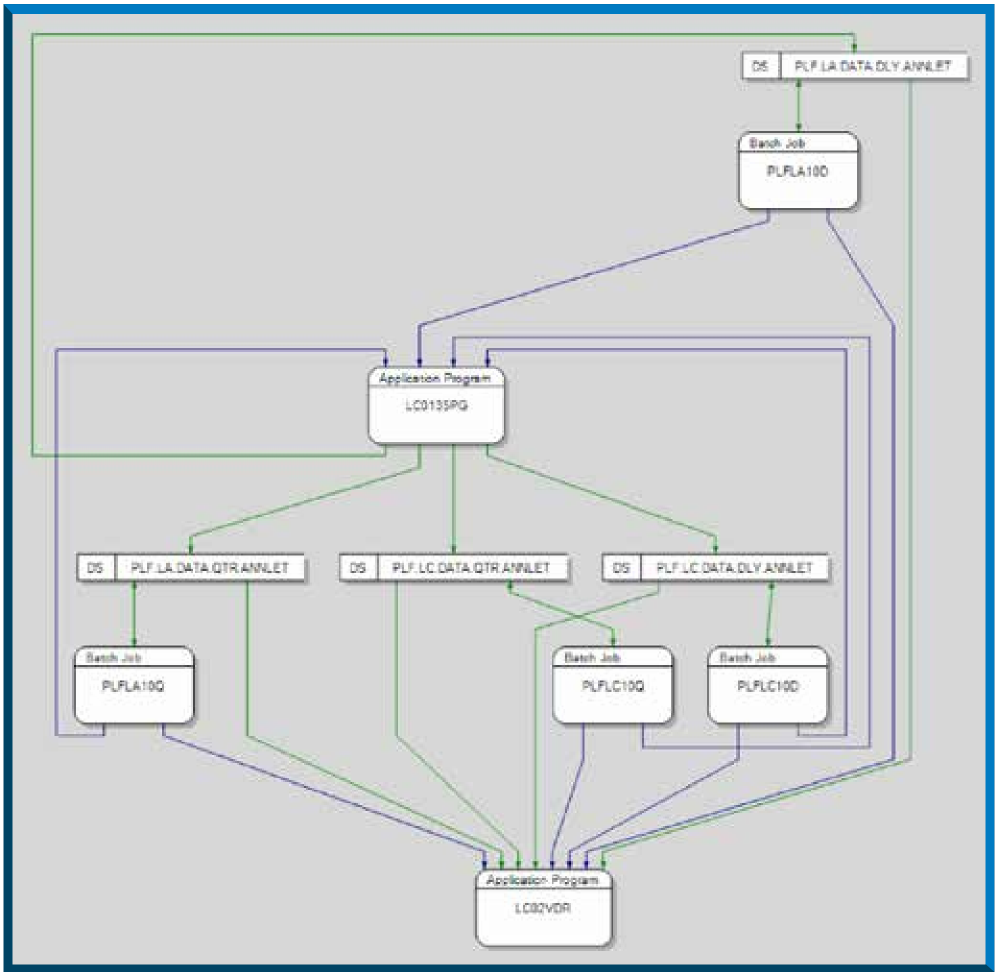


Figure 4.1 Job-Program Physical Data Set Read/Write Diagram

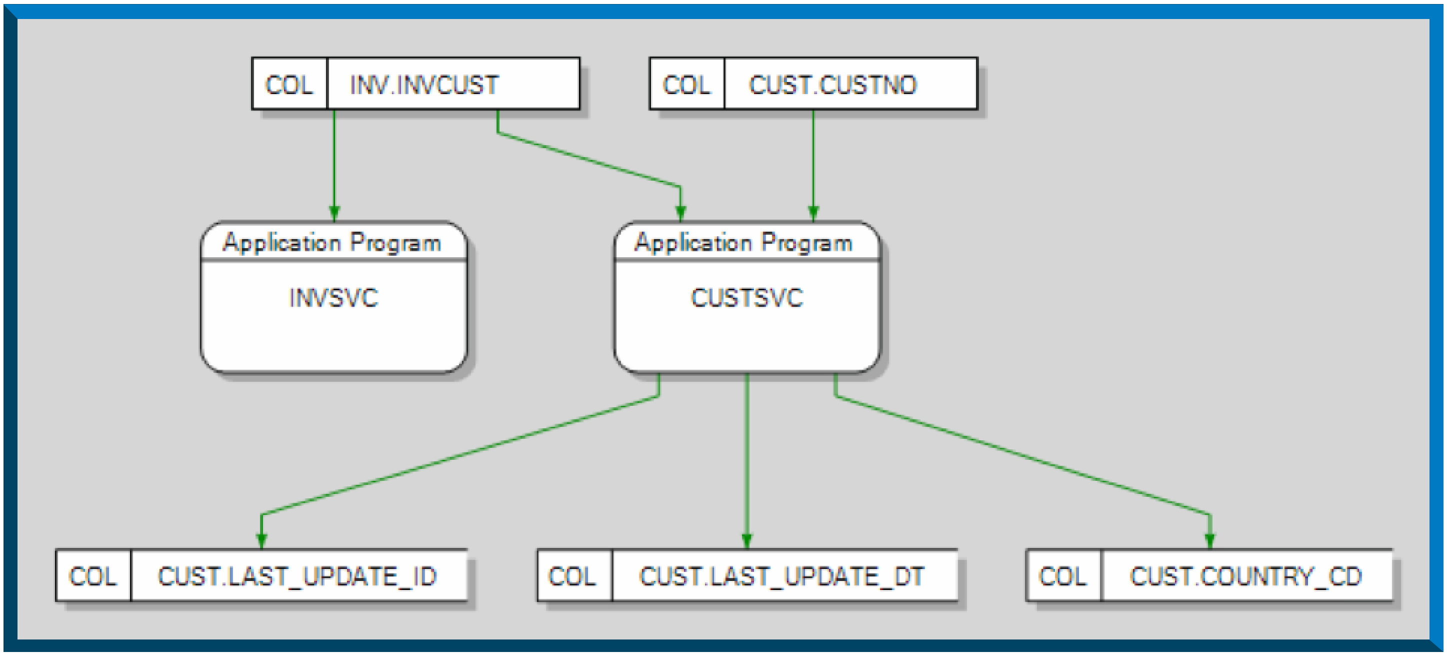
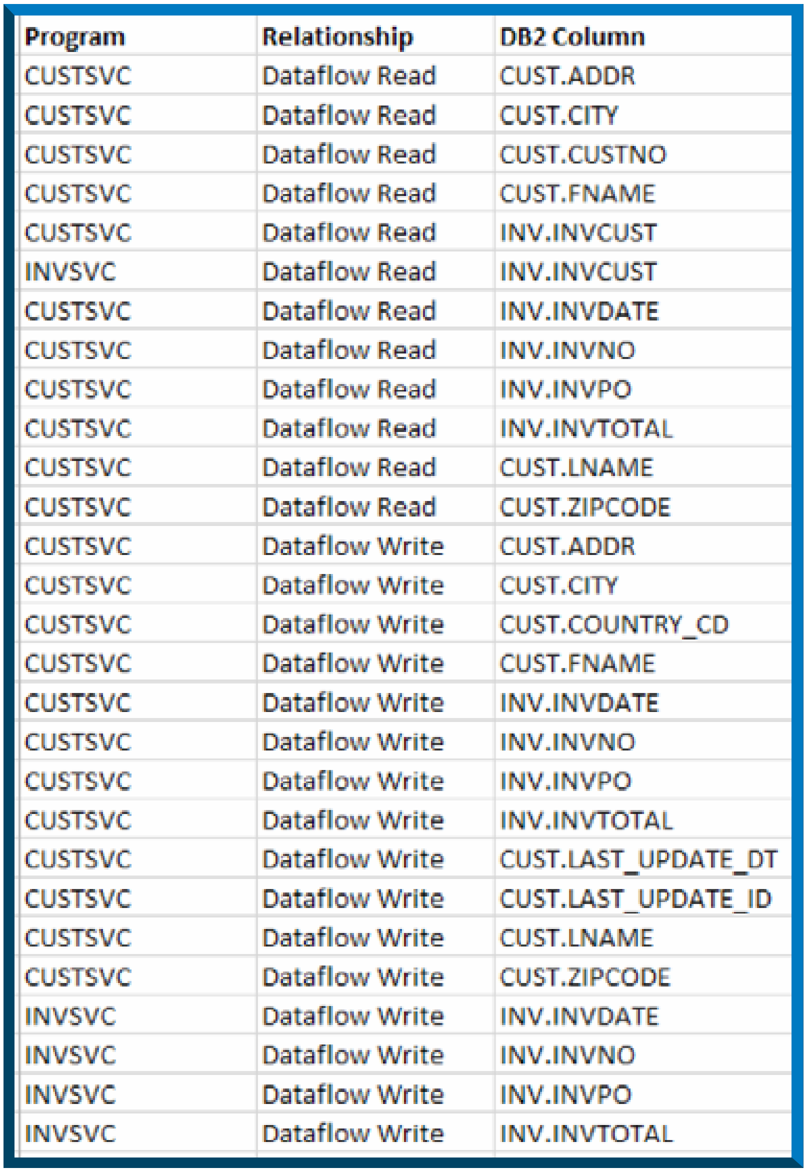


Figure 4.2 DB2 Column Read-Write



Program	Relationship	DB2 Column
CUSTSVC	Dataflow Read	CUST.ADDR
CUSTSVC	Dataflow Read	CUST.CITY
CUSTSVC	Dataflow Read	CUST.CUSTNO
CUSTSVC	Dataflow Read	CUST.FNAME
CUSTSVC	Dataflow Read	INV.INVCUST
INVSVC	Dataflow Read	INV.INVCUST
CUSTSVC	Dataflow Read	INV.INVDATE
CUSTSVC	Dataflow Read	INV.INVNO
CUSTSVC	Dataflow Read	INV.INVPO
CUSTSVC	Dataflow Read	INV.INVTOTAL
CUSTSVC	Dataflow Read	CUST.LNAME
CUSTSVC	Dataflow Read	CUST.ZIPCODE
CUSTSVC	Dataflow Write	CUST.ADDR
CUSTSVC	Dataflow Write	CUST.CITY
CUSTSVC	Dataflow Write	CUST.COUNTRY_CD
CUSTSVC	Dataflow Write	CUST.FNAME
CUSTSVC	Dataflow Write	INV.INVDATE
CUSTSVC	Dataflow Write	INV.INVNO
CUSTSVC	Dataflow Write	INV.INVPO
CUSTSVC	Dataflow Write	INV.INVTOTAL
CUSTSVC	Dataflow Write	CUST.LAST_UPDATE_DT
CUSTSVC	Dataflow Write	CUST.LAST_UPDATE_ID
CUSTSVC	Dataflow Write	CUST.LNAME
CUSTSVC	Dataflow Write	CUST.ZIPCODE
INVSVC	Dataflow Write	INV.INVDATE
INVSVC	Dataflow Write	INV.INVNO
INVSVC	Dataflow Write	INV.INVPO
INVSVC	Dataflow Write	INV.INVTOTAL

Figure 4.3 Example Excel Report – Column Program Read/Write

SUMMARY

At the conclusion of the system assessment phase, CM First will compile and prepare a report with data that illustrates the system statistics. The report will include a source inventory summarizing the number of objects by source and summarizing data source framework.

CM First also recommends discovery to explore additional statistical information of value that may be extrapolated from the system diagram and included in the report.

About CM First

CM First focuses on powering business agility by providing software and services that help customers manage and modernize their enterprise applications. Founded on a mission to manage the complexity of multi-platform application projects, CM First creates innovative application development solutions. A global company, CM First is headquartered in Steinhausen, Switzerland with offices in Austin, Texas and Milan, Italy. With the support of strong global IT partners such as IBM and CA, CM First provides software and services powering development for more than 400 clients in both the private and public sectors, including banking, IT, tourism, and manufacturing. CM First is an IBM Advanced Business Partner and a CA Technologies technology partner and global reseller offering complementary tolling and services to CA customers.

About 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.

For more information, visit **cmfirstgroup.com/offerings/products/cm-evolveit**

Request a Demo Today

Contact us for more information or to schedule a demo. Call 888-866-6179 or email us: **info@cmfirstgroup.com**

CM First Group
888-866-6179
cmfirstgroup.com

7000 North Mopac Expressway
Plaza 7000, 2nd Floor
Austin, Texas 78731