SIBRA

Ingenieur
Gesellschaft für
Datentechnik mbH

LMR Load Module Repository, a short Overview

"The truth only exists in the code that goes into production every night"

- Improvement of your Application Development, program quality and performance
- Analysis of currently running Load Module Libraries for Enterprise, Application Portfolio, and Program Management
- Building up a DB2 based Load Module Repository LMR from currently running Load Module Libraries for deployment as a Configuration Management System
- LMR answers every question about your applications stored in the load module libraries, it offers an analysis and repository quality not achieved until now.
- LMR supports you in every modification, further development and migration of your software, it saves considerably time and money, reduces the project risks and increases the quality.
- LMR is available on z/OS Mainframe and on Windows PC and Server (On UNIX on request)

LMR can be used for the following tasks for example:

- Application Development and Computer Center Administration are supported by a comprehensive Configuration Management System, e. g. to identify which system components and third party software are used by applications to inhibit unused software licenses.
- Doing Quality Assurance QA by finding e. g. not approved or performance disadvantageous attributes, compiler and run-time options, compiler and module versions, etc., which means verification of z/OS and programming standards (see Program Management)
- Improving performance of certain load modules simply by recompiling them with different options.
- Verification of LE/390 conformity of all load module libraries with their load modules
- Interactive query or reporting of all types of cross references for all technical attributes, internal structure, and included modules/CSECTs of all load modules.
- Reporting instances of the same module/CSECT name with multiple compile dates or sizes. This inconsistency check can indicate a flaw in the process used for change control, especially when there is only one source program to create the module/CSECT
- Supporting error diagnosis by tracking the usage of all Load Modul Libraries with their load modules with included programs and subprograms and their options and attributes
- Redesigning and renovating existing applications
- Compiler migrations, e. g. to IBM Enterprise Cobol and IBM Enterprise PL/I

SIBRA
Zentralbüro

Leiblstraße 14
82194 Gröbenzell
Ruf (08142) 9025
Fax (08142) 9028

SIBRA
Projektbüro

Industriestraße 35
82194 Gröbenzell
Ruf (08142) 57264
Fax (08142) 57265

Internet:
www.SibraGmbh.com
email:
SibraGmbh@t-online.de

Mobilfunk:
0172 8409119

Sitz der
Gesellschaft:
Gröbenzell

Registergericht:
München
HRB 62043

Geschäftsführer:
Eberhard Ramm

Sparkasse
Fürstenfeldbruck
Konto: 4911558
BLZ: 700 530 70

Analysis of Load Module Libraries

Effective management requires a big picture view of the portfolio
The concerns at this level are with the entire portfolio

- Enterprise Management Perspective
- Application Portfolio Management
- Program Management

Enterprise Management Perspective

Managers can plan the most effective way of migrating to a new technology or implementing a set of standards

- How many programs are there ?
- What languages and technologies are deployed ?
- Are appropriate quality standards being met ?

  Ensure that production programs conform to the installation standards of the compiler and linkage editor options.
- How fast is migration to a new technology taking place ?

Application Portfolio Management

Portfolio management looks at a single application or possibly a family of programs

- What programs will be affected by a change of a subroutine ?
- What programs are dependent on specific vendor run-time routines ?
- Which programs are "easy to migrate" and which contain specific migration inhibitors ?
- Project leaders can be most/utterly productive by understanding the complete application portfolio. They can avoid many otherwise unexpected situations that often arise when migrating or implementing other changes to existing applications.

Program Management

Developers migrating, modifying or enhancing particular programs also need portfolio information

- What components make up a particular program ? And, are those components also used in other applications ?
- What level of the compiler was used for each component ? Will the compiler need to be changed as part of the application update ?
- Are there compiled CSECTs from a program source with different compile dates and attributes ? These CSECTs could have different functionality !

- What compiler options were specified when the program was last compiled ? In some cases, changing compiler options can change the results produced by a program. Maintaining consistent compiler options across the installation should be part of any configuration management plan.

  A summary of the more critical compiler options used by applications in the specified load library or libraries is provided. LMR produces a single summary report indicating the usage of a given option. The counts / amounts for each option are listed by compiler group.
- Did the source management tool leave an "eye-catcher" in the program to help identifying the program or changes made to the program ?
- Are there OS/VS COBOL applications currently using CICS that will no longer run under CICS TS V3.1 or later ? (These applications must be migrated to a newer COBOL compiler if they are to execute in the future)

  IBM has also removed support for applications compiled with OS PL/I and C/370 Versions 1 and 2. While these programs may not fail at the present time, they are at risk with future releases.
- Which compiler run-time routines and run-time options overrides are included as part of the application load module ?
- What is the release level of the run-time routines used or link edited into the application ?

  Do the routines need to be refreshed to the latest level migrating the application ?
- Does the COBOL module use ACCEPT FROM DATE, or the PL/I module use DATE or DATETIME Built-in Functions to request the system date?
- Similar questions might be asked about the use of SORT, DB2, IMS, CICS or other LE or system functions if those components or subsystems are being upgraded
- Does the application make use of some Language Environment Callable Service that has changed with the latest release of LE ? This happened in OS/390 V2.9 and V2.10
- Does a COBOL program use dynamic CALLs to load an independently linked module for execution ?
  Does a PL/I program use the FETCH/RELEASE statements for dynamic loading routines ?
- Does an Assembler routine issue a LINK or LOAD macro to dynamically invoke a separately link edited module ?
- Do Assembler routines use other SVCs that may require attention when this application is modified ?
- Are there some components in a load module that require updating or relinking to allow the program to run with LE or to run above the 16MB line ?
- and many more attributes . . .

Conclusion of Load Module Repository

Having the right information readily available makes the developer more productive. Problem areas can be addressed immediately, avoiding a lengthy trial-and-error process.

License and Demonstration

For LMR we offer to you use licenses and also complete analyses of load module libraries as professional service.

We also can perform for you complete LE/390, programming language, database and system platform migrations with project management.

We hope, to have gained your interest and would be pleased with being allowed to demonstrate you this tool. You can request further detailed information and examples from us by email to SibraGmbh@t-online.de . For questions we are available with pleasure.

This happens when you do not use LMR: