

# MAJOR NEW VERSION OF THE PRIMA MIGRATION TOOLSET

VERSION 3.1.0.14 - 1<sup>ST</sup> JULY, 2009

## RELEASE NOTES

### FUNCTIONAL

Coinciding with the launch of the COBOL21 Web site, intensive review and enhancement of the Toolset has been carried out and will be continuing. Enquiries from the Web site have resulted in “strengthening” of the Toolset, and some interesting points and considerations being added to the “mix”. As a result, the overall architecture has been reviewed and made more consistent, the user interface has been completely overhauled with better information for users, better wording, and clearer layout. A deeper level of error trapping has been incorporated also.

Thanks to your feedback, the Toolset is improving and evolving.

HERE ARE THE FUNCTIONS WHICH THE TOOLSET CURRENTLY SUPPORTS, IN THE ORDER IN WHICH THEY APPEAR ON THE TOOLSET TABS:

1. The setting up of a Migration Environment (ME) on the local machine where the Toolset is running, or across the Network. This enables centralized administration of the Migration conversion process for each workstation running the Toolset, viewing of pertinent files and databases, setting required Environment Variables, managing ODBC, viewing and editing generated code, searching for possible date fields, and a number of other functions which are described elsewhere.
2. The creation of Relational Database Tables from COBOL record definitions ([Source Sets](#)). (This tool is called TABLECREATE). TABLECREATE does not just create a table, it creates a [Tableset](#). This is a base table normalized to 2NF, with repeating groups attached via Foreign Keys, and supporting cascade delete.
3. The Generation of a Batch program to load the created Tableset. (This tool is called LMG – Load Module Generation.) LMG reads the current existing Indexed File and writes data on it to the Tableset. This program is currently generated in COBOL. COBOL compilation can be easily controlled from the Toolset and any number of programs can be batch compiled. The Toolset generates the necessary compiler support for each program.)

4. The actual load run (execution of the LMG programs) is initiated and controlled by the Toolset. Any number of Tablesets can be loaded in a single run.
5. A Host Variable and COBOL record layout generator. (This tool is called DECLGEN [Declaration Generator], after the IBM Mainframe tool of the same name.) DECLGEN is used “under the covers” by the Toolset, but is also a useful tool for people writing COBOL code to access RDBMS. These are programs which contain Embedded SQL (ESQL), and ESQL is virtually obsolete and deprecated. The Toolset supports the generation of a tiered architecture with database access being carried out by a layer of Objects, rather than ESQL embedded into application programs. This is referred to as the DAL (Database Access Layer).
6. The generation of DAL objects. (The tool that does this is called MOSTGEN – Maintenance Object Server Template Generator). These are general purpose maintenance objects which manage a given Tableset. DAL objects can do all possible actions that could be done against an indexed file, including random, sequential, and skip-sequential processing for both GETting and PUTting . They provide separate instance threads for these streams automatically, and manage all database connections and housekeeping. Data is provided back to the applications in the same record buffer it would have been received from an indexed file, and the DAL object manages the construction and deconstruction of the COBOL buffer used as an interface, completely transparently of the application which invokes it. DAL objects are currently generated in COBOL. (Because they are objects, the source language is really irrelevant, however, advanced database functions like LINQ and Lambdas, provided through C#, or even result and dynaset processing through OLEDB, or ADO .NET, are better options for the future than the limited ODBC and ESQL provided through COBOL. Using the DAL keeps options open and DAL objects may be generated in C# at a future point.)
7. Having generated DAL objects for manipulating data, the existing COBOL applications must be changed, so that the existing calls to indexed files are transformed into DAL invokes. (This tool is called MOSTAMD, and the process is called TRANSFORMATION). The applications are generally procedural COBOL and the DAL objects are OO components, so integrating them could be a tricky piece of software engineering. TRANSFORMATION does it quickly and easily. Currently, it supports Fujitsu PowerCOBOL, and provides a complete breakdown of the PowerCOBOL structure, showing all indexed file definition and access, and all the component “scriptlets” which comprise the PowerCOBOL run unit. MOSTAMD can comment out or delete the existing COBOL verb access and replace it with appropriate method invokes on the DAL object.
8. In addition to the above, this release has added two new functional features:
  - a. A new field on TAB 1 which causes your installation name to be included into all generated listings.
  - b. A new field on TAB 1 which lets you provide support contact information if you are supporting a site where code has been generated by the Toolset.

Obviously, the above is just the “tip of the iceberg” as far as describing each of the functions in the Toolset goes, but it serves to document the current Toolset functionality, as of this release.

## TECHNICAL

### HERE ARE THE TECHNICAL CHANGES IMPLEMENTED INTO THIS RELEASE:

- 1. Support for compiles of LMG programs and DAL objects using Fujitsu NetCOBOL Version 9.**  
The Toolset now supports Versions 6 thru 9, running locally on the workstation, or on a single COBOL server somewhere on the Network. This release has seen the replacement of some of the previous scripting to support remote compilation. It has been condensed and re-organized. The current scripting is tighter and more robust than the previous. Ancillaries for Versions 7 and 9 are different from version 6... The whole lot has now been amalgamated into a single PRIMA Ancillary generation that supports any of these compilers. Assignment of COM Class GUIDs for DAL components during compilation has been implemented.
- 2. Support for more databases than Access 2000/2003.** The Toolset now fully supports Access 2007 and SQL Server 2005. A framework to support other future target platforms has been designed, developed and installed in this release. The design allows customers to “piggyback” to their desired target platform using SQL Server, or to buy an upgrade for their desired target from PRIMA. The model involves use of a [“working set”](#) and a [“target platform”](#). The working set is the Database that will be used by the Toolset tools. (It is not intended to deploy this to your customers) Originally this was Access 2000/3. Now it can be ANY of the three RDBMS mentioned earlier. The target platform (IS what you intend to deploy to your customers...) can be any of these, or any other ANSI compliant RDBMS. If you **don’t** want to buy a working set upgrade (the upgrade would allow you to use your target platform as your working set, which means you don’t need Access or SQL Server and it simplifies a lot of things for your development), you can use SQL Server as your working set (SQL Server EXPRESS is a free download) and the Toolset will “post-process” the SQL Server DAL code to match your desired target (if this is technically feasible; not ALL target platforms have been checked out yet...). You will only deploy the target; the SQL Server working set would be used only for development. Full details, along with the possible target platforms, are provided by clicking the ‘Select target RDBMS’ on TAB 1 of the Toolset.
- 3. DECLGEN has been upgraded to support types used in ANSI as well as types used in Access.** Specifically, Access types: “Text(n)”, “Currency”, “Date”, and “Memo”, are replaced in ANSI by ‘Char/Varchar(n)’, ‘Decimal (14,4)’, ‘Datetime’, and ‘Varchar(n)’. The generation of DDL by TABLECREATE has been updated to reflect this.
- 4. Support for all possible 8 digit date formats has been included in this release.** Previously, 6, 8, and 10 digit dates were supported, but the 8 digit format could only be DDMMYYYY. Now it can be: YYYYMMDD, DD/MM/YY, or DDMMYYYY. Note that DECLGEN generates a COMP-2 (64 bit float) for date fields. This was necessary to support old VB format dates, as used on Access 97. The Toolset converts this to a PIC X(20) field for Host Variables, so that the complete date, including time and relative hours + or – Zulu time (GMT) can be

accommodated. DAL objects strip out whatever is needed by the COBOL buffer, so everything works.

5. **The generated Access “IIF” and VB “IsNull” and “Cstr()” functions which avoid the need for indicator variables in ESQL, by testing fields for NULL before returning them to the COBOL buffer, have been replaced with ANSI compliant “COALESCE” and “CONVERT” SQL functions** (except where the working set is Access, of course... 😊)

## THE FUTURE

### SHORT TERM

#### HERE ARE THE AREAS WHICH ARE CURRENTLY UNDER DEVELOPMENT

1. **A Non-COBOL related facility to analyse RDB tables NOT generated by the Toolset, and generate DAL management objects for them.** This will extend DAL to non-COBOL applications. It will also allow COBOL sites to access Tablesets on RDBs which they may have acquired and did not build themselves.
2. **Implementation of “piggyback” target platform processing. This will be initially in support of PostgreSQL.** Theoretically, it shouldn't be too difficult to modify SQL Server DDL and generated DAL code, so it can support PostgreSQL. It is intended for this to be the model for future “post processing” options.
3. **A User Guide for the Toolset, probably located on the cobol21 Web site.** This will include screen shots, background concepts, tips, and general assistance.

### LONGER TERM

1. **Tools to assist with the currently mainly manual process of refactoring existing COBOL application code into components.** This is part of the implementation of getting the Business Logic migrated to objects, in accordance with the PRIMA Migration Model. <http://primacomputing.co.nz/cobol21/ShowMe.aspx?ptitle=Process%20flow%20diagram..&pcontent=images/MigrationProcessOverview.jpg>
2. **Option of different language and database access support for generated DAL.** Although the current dynamic SQL used by DAL, along with the ability to create multiple instances and threads for itself at run time, makes DAL performance as good as it can be, it is still limited by COBOL ESQL. It would be good to have DAL using LINQ or resultset processing.

## GENERAL OBSERVATIONS AROUND THIS RELEASE

1. The concept of using objects with Procedural code has generated quite a bit of comment in both Usenet forums and private mail. The OO people see immediately how useful this is; the old-timers are suspicious of objects and don't see how they can be mixed with non-OO code. Most of the queries I have had have been persuaded, once they can see the concept. As far as I know, the PRIMA solution is the only one that offers an OO DAL layer, based on the Component Object Model, but I know at least one well known COBOL vendor has shown interest in the concept (they are frequent visitors to the cobol21 site and downloaded the full source package for the COBDATA tool, as soon as it was available...) The attached picture is intended to clarify how DAL is supposed to work.

FINALLY...

Thank you for being a PRIMA Toolset "early adopter". It would be impossible to bring this toolset to market without people who are prepared to support something in its early stages. The feedback and ideas from people using the tool is invaluable. And your support and encouragement is appreciated.

Regards,

Pete.

## GLOSSARY

**SOURCE SET:** The basis for everything. It consists of 2 parts:

1. The COBOL SELECT... ASSIGN statements. The Toolset defaults this file extension to ".CBS" but you can use any extension that suits you.
2. The COBOL Record definition. FD/01 and subsequent structure definition. The Toolset defaults this file extension to ".CBF", but you can use any extension that suits you.

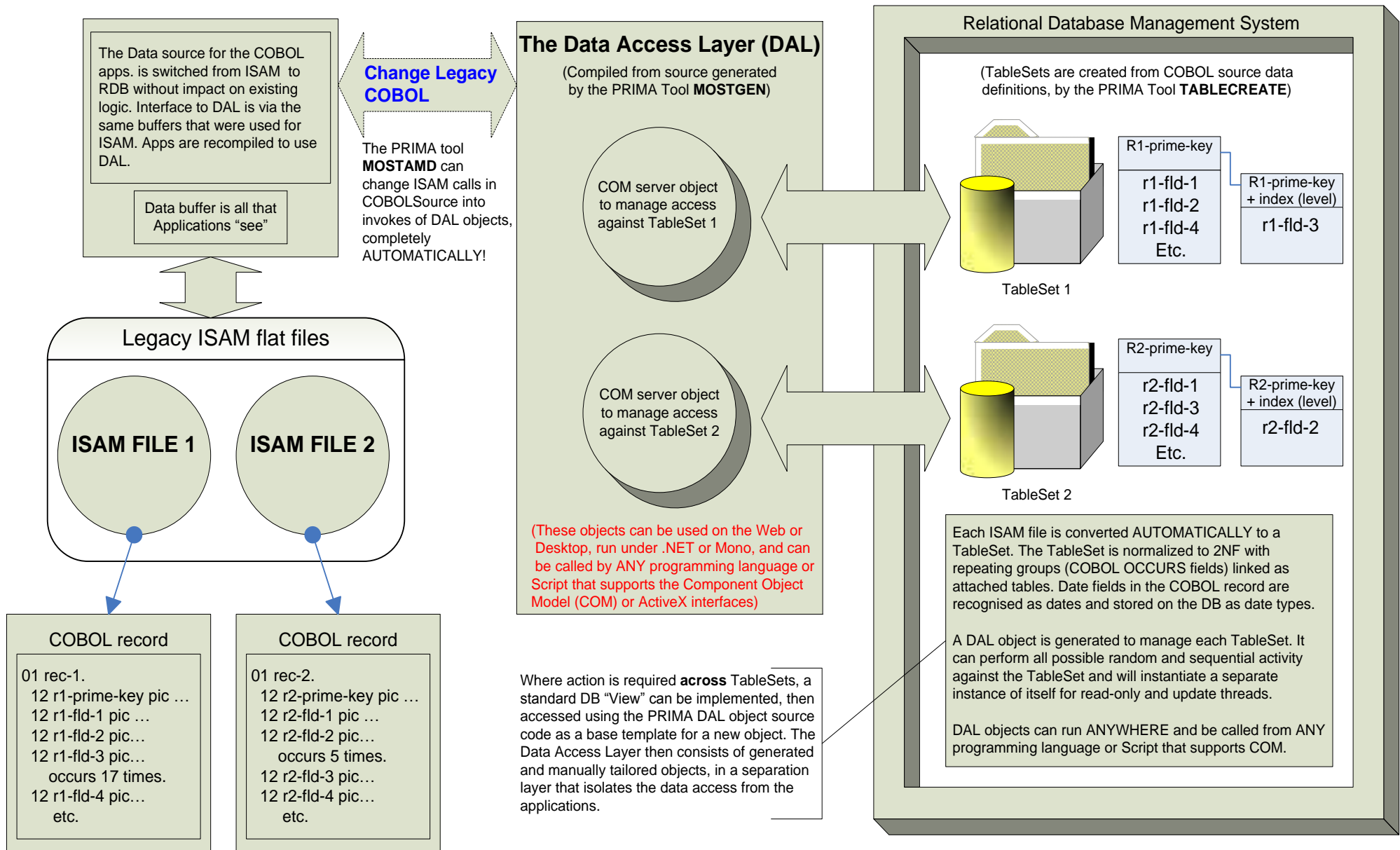
The CREATETABLE tool uses the source set to create the Tableset on the DB, and the metadata which describes each field added to the RDB table.

**TABLESET:** The group of associated tables (Base + attached repeating groups), generated from a single COBOL 01 level definition.

**WORKING SET:** The RDBMS that is used by the Toolset to create and load tables on, and it is accessed by the DAL objects. It will be used for development and need not necessarily be the same as the RDBMS that you will deploy to support your application.

**TARGET PLATFORM:** The RDBMS that you will deploy to your customers. This is your application database. If it is the same as your working set, you do not need any other staging RDBMS. If it is NOT the same, then you must either use SQL Server for your working set or buy an upgrade.

**ISAM to RDB conversion using the PRIMA Toolset**      **May 2009 (Copyright Prima Computing, (NZ) Ltd)**  
 (ALL rights, including intellectual property, reserved. Do not copy or circulate without permission.)



Where action is required **across** TableSets, a standard DB "View" can be implemented, then accessed using the PRIMA DAL object source code as a base template for a new object. The Data Access Layer then consists of generated and manually tailored objects, in a separation layer that isolates the data access from the applications.