Technical Overview

Information Balance, Inc.

Application Road Map (ARM)

Version 4.5.4

PREFACE

ACKNOWLEDGEMENT AND COPYRIGHT

Fifth Edition: September 2015

IB-ARM Version 4.5.4

Trademarks

All product and company names are trademarks or registered trademarks of their owners.

TABLE OF CONTENTS

INTRODUCTION

Application Road Map (ARM) is a comprehensive, bottom-up Application Portfolio Management (APM) solution that provides IT staff with a repository of institutionalized application knowledge that helps reduce everyday task cycle time. It is essentially an automated systems documentation solution, generating a repository of application artifacts and their relationships from parsed source code. The tool provides a variety of functionality benefiting development and support staff as well as application managers.

The focus of IB-ARM is the presentation of intelligence related to application objects, including the dependencies between the various objects. IB-ARM integrates this large volume of 'detail level' information to give higher-level views showing the object interactions in the context of each application and how they relate to the broader system. These views can significantly aid the understanding of IT staff and reduce the learning curve dramatically.

The information is captured and organized in a centralized database repository and presented via a user-friendly web-based interface.

This document describes the overall architecture and structure of the tool, underlying software and installation requirements.

IB-ARM TECHNICAL OVERVIEW

ARM's architecture contains several processes that are performed to prepare a client's repository content. These processes are performed initially to populate the repository and then repeated on an ongoing basis, using the Refresh process, to keep the information current and accurate. The IB-ARM processes are depicted in the diagram below:

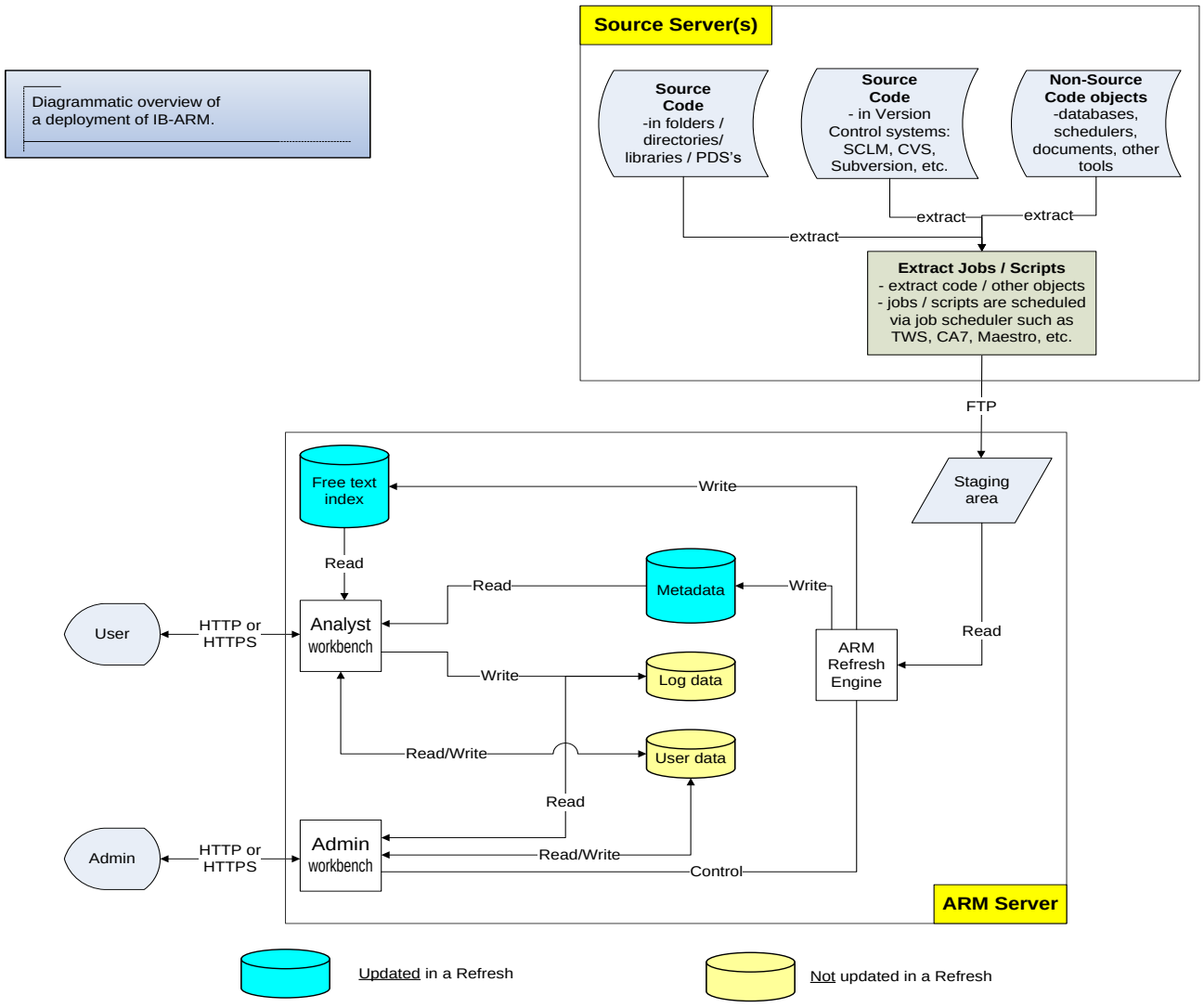

Figure 1: IB-ARM Architecture Overview

SOURCE SERVER(S)

SOURCE CODE EXTRACTION

Source code is extracted from the client's various source server platforms (Mainframe, Unix, Windows, etc.) and transferred to the ARM Server's staging area.

Source code can reside in any number of locations including version control systems (VCS), libraries, folders, etc. Source code collection instructions and/or pre-written extraction jobs, scripts are provided so the data can be extracted in the formats expected by ARM. After initial configuration extraction jobs and scripts are bundled for repeatable execution.

ARM SERVER

The ARM Server houses several processes and databases.

The **ARM Refresh Engine** creates repository content. It is the center-piece of the tool and consists of language parsers, relationship mapping software and database load processors. The Refresh Engine is configured for each installation based on the technology stack that makes up the portfolio.

The **ARM Data Repository** is a collection of data stores that contain user and operational data.

The **Analyst Workbench** is a browser application that provides end-users with functionality to view, navigate, print and/or export repository content. The **Admin Workbench** provides the interface for administrators to generate content (operate the Refresh Engine) and to perform various house-keeping functions.

ARM REFRESH ENGINE

The ARM Refresh Engine is a collection of processes that collaborate to generate the content of the ARM repository. Most of the software is written in PERL, with some parsers coded in JAVA.

INVENTORY

The Inventory process extracts and types input files from the staging area. It works from the installation's Manifest (an XML file configured for the install to describe data to be received) and prepares the collection for the subsequent Parsing process. It also produces a variety of reports in PDF, Excel and XML formats.

PARSERS

The Refresh Engine is configured with the requisite Parsers for the collection. Each language, technology, tool output, database, etc. has its own dedicated parser (see the Appendix for a current list of parsers).

ARM's parsers have all been developed by and are proprietary to IB. Parsers for simple languages are mostly written in PERL; parsers for more complex languages (COBOL, PL/1, Assembler, JCL, JAVA, .Net, etc.) are written in JAVA, using JFlex and JCUP.

INDEXER

The Indexing process works from parser output and creates the Free Text Index, the foundation of ARM's search facility. It also prepares component metadata for the subsequent database load.

LOADER

This process loads the ARM component database (Metadata) using the bulk load facility of the underlying database engine (MySQL, or SQL Server). The tool is typically configured to allow several parallel load processes.

MAPPING

The Mapping engine is made up of a series of mapping rules to resolve relations both within a technology (for example, when one program calls another) and between technologies (for example, when a program accesses a database table). Mapping rules are written in SQL and perform updates to the pre-loaded repository database. Each installation is configured with the mapping rules relevant for the technologies at hand.

TABLE OF CONTENTS (TOC) GENERATOR

This process assembles the table of contents, using a definition file and the data in the metadata tables. The table of contents is essentially an inventory of the applications and components that are present in the repository.

ARM DATA REPOSITORY

ARM data resides in several data stores, implemented – with the exception of the Free Text Index - as a series of MySQL databases[1].

METADATA

Metadata is made up of parsed language components and their relationships. ARM's metadata database is a fully extensible relational database.

FREE TEXT INDEX

The Free Text Index is a special inverted file that contains all the tokenized source code to support source code display, manipulation and search.

LOG DATA

All ARM user activity is logged into this relational database. The Log database is not modified during a Refresh so statistics are retained indefinitely.

USER DATA

Information about ARM users, their selected settings and their ARM activity are recorded in this relational database.

[1] IB-ARM can also be installed on SQL Server.

ARM FRONT-END

ARM's Front-End provides browser based user interface for the **Analyst Workbench** and the **Administrator Workbench**.

MODEL VIEW CONTROLLER (MVC) FRAMEWORK

IB-ARM's content delivery views display repository content. These views include tabular displays, list views, chart views, graphical diagram views, source code views and document views.

IB-ARM's content delivery views are built with Catalyst, a PERL MVC Framework.

BROWSER SUPPORT

IB-ARM supports Internet Explorer 6 and higher as well as Firefox 5.0 and higher.

SEARCH

IB-ARM's search engine is a custom adaptation of KinoSearch (a search engine written in PERL and C and a loose port of the Java search engine library Apache Lucene). ARM's version enables uniform search capability spanning documents and program code written in multiple languages.

DIAGRAMMING

IB-ARM generates a variety of node-and-link graphical diagrams to help visualize application structure. The diagram engine is fully proprietary to IB and was built using open source tools – GraphViz, Dojo, ImageMagick and Inkscape.

OTHER COMPONENTS

Others components used in IB-ARM site construction, display and installation:

- **PERL** – ARM has its own distribution of PERL, which bundles all the dependent modules needed by the parsers and the core framework.
- **Installers** - the installers bundle together the code, with some configuration provided automatically. The installers also encrypt the code, blocking certain PERL modules.

ARM SOFTWARE SUMMARY

The following software are utilized in the development of the ARM product and partly bundled with each install:

PERL 5.10.1	(parsers, front-end, all ARM functionality)
Catalyst 5.8	(front-end)
JAVA 1.6.0	(parsers)
MySQL 5.5	(repository)
SQLlite 3	(inventory, parsers)
Dojo 2.1	(front-end)
Graphviz 2.8	(front-end, diagramming)
ImageMagick 6.7.4-9	(diagram management)
Inkscape 0.48.2	(diagram management)
7Zip	(install)
Unzip	(inventory)
Pdftk 1.44	(report generator)