

Software Product Description

PRODUCT NAME: RuleWorks Version 2.0a

SPD 64.31.01

DESCRIPTION

RuleWorks™ is the next generation rules-based, application development tool from DIGITAL. It provides the convenience of cross-platform development combined with one of the industry's most powerful high level languages and fastest inference engines.

RuleWorks is a language, compiler (with debugger), and Run-Time Library for constructing high performance, modular, object-oriented, forward-chaining, rules-based applications. It provides portability by producing ANSI C sources which can then be compiled with any of the leading C or C++ compilers. There is a Run-Time Library for each supported platform and C compiler.

Rules-based approaches have been successful in numerous applications in the manufacturing industries and wherever decision support or rapid prototyping is required. Other domains include configuration, selection, data acquisition and process control. Health care firms and others who require interactive diagnostic programs have also used expert systems to great advantage.

A rule (or production) is defined as an "if-then" or "when-do" statement consisting of a conditional part, called the Left Hand Side (LHS), and an action sequence, called the Right Hand Side (RHS). Rules operate on in-memory objects which represent concepts or entities in the real world. These objects, called Working Memory Objects, are composed of a set of attribute value pairs.

During each recognize-act cycle, RuleWorks examines the LHS of all active rules to determine which rules' LHS conditions are satisfied given the current state of Working Memory Objects. RuleWorks then applies conflict resolution criteria to determine which of these rules should have its action sequence executed. Actions in turn, can modify the state of Working Memory Objects. This cycle is repeated until no more rules are satisfied or until a rule explicitly halts processing.

A rules-based language differs from a conventional programming language in that rules are not processed in sequential order; rather, the order of execution is driven by the current state of working memory data.

RuleWorks is an evolutionary step beyond its predecessor, DEC OPS5. Most applications written in DEC OPS5 V4.0 are trivial to migrate to RuleWorks.

Portability

The RuleWorks compiler and the run-time systems are written in ANSI C and the compiler generates ANSI C output files. RuleWorks is currently supported on the following platforms:

Processor	Operating System
Intel	Windows 95
Intel	Windows NT
Alpha	Windows NT
Alpha	DIGITAL UNIX
Alpha	OpenVMS
VAX	OpenVMS

Modularity

RuleWorks provides features designed to support the software engineering concepts of modularity and information hiding. For example the programmer can write multiple independent rules-based subsystems which do not interfere with each other. It is also possible for multiple rules-based subsystems to share specific sets of information with each other, or to share all of the information to which each has access. Programmers can also control all of the functional and all of the matchable information subsystem interfaces.

Rules-based subsystems can be called from other rules-based programs or from programs written in other languages. From a rules-based module, calls can be made to any other subprogram (including those which are rules-based), even recursively. Function arguments are passed via standard calling mechanisms with automatic data type coercions.

Match Extensions

The LHS is the "if" or "when" part of a rule. It specifies the conditions in working memory which must be true before the rule can execute. The LHS is composed of condition elements (CEs), each of which can match objects of a particular class, and its subclasses, if any. RuleWorks performs a logical AND operation on all the CEs on the LHS. A rule is eligible to execute when there are objects which match all of its positive CEs, but there are no objects which match any of its negative CEs. The RHS is the "then/do" part of a rule. It consists of one or more

actions. The actions on the RHS of a rule are executed only when the LHS matches working memory and the resulting particular instance is picked during conflict resolution.

RuleWorks provides the following new matching features:

- Variables are evaluated within value disjunctions, including variables bound in other condition elements.
- Relational predicates (>, <, >=, =) can compare integers to floats, and symbols with symbols using localizable lexicographic ordering.
- New similarity predicate, ~=, which for numbers tests for equality within a 1% margin, and which for symbols uses the SOUNDEX algorithm to compare phonetic (in English) similarity.
- New dissimilarity predicate, -~=, which matches when the similarity test fails.
- Comparisons of compound attributes against function return values.
- The containment and non-containment predicates can be applied to scalar attributes, as long as the argument to the predicate is a compound value (e.g. ^name [+] <name-list>).
- The containment predicate can also be applied in conjunction with a scalar predicate to search the compound attribute for a value which satisfies the scalar predicate.

Applications

RuleWorks is the preferred tool for developing high-performance, commercial quality rules-based systems. Systems developed with RuleWorks are well suited to solve problems in:

- Configuration
- Selection
- Diagnosis
- Process monitoring and control
- Scheduling
- Planning
- Decision support
- Data mining
- Rapid prototyping

Applications involving these problems are found in such industries as discrete manufacturing, petrochemicals, banking, insurance, transportation, aerospace, education, health care, and government.

Features

- Support for an object-oriented data model with a single inheritance class hierarchy, with matching at arbitrary levels in the hierarchy, and unique immutable instance identifiers

- Support for multi-valued, or compound, attributes. Provides match primitives for comparing either entire compound values, or a specific element within a compound value, and for searching an entire compound value for some specific value.
- Optional attribute data types allow the user to restrict the domain of an attribute to a specific type of value (or values in the case of a compound attribute).
- Built-in support for SQL access to Oracle Rdb databases on OpenVMS. This includes a simple one-to-one mapping between tuples in the database and the corresponding matchable objects, as well as support for arbitrarily complex mappings
- API allows the direct creation, modification, and deletion of objects in working memory

Application Development Process

Developing applications for any supported platform is straightforward:

- Applications (including client server, object oriented programs) are composed in ASCII text using any suitable editor and the RuleWorks language.
- The RuleWorks compiler is called, either independently from the system prompt or from within another development environment such as Visual C++, and utilizes the ASCII source as its input.
- The RuleWorks compiler produces ANSI C source code portable to the platform in which the application will ultimately be run.
- The RuleWorks-produced C code is compiled and linked with the RuleWorks run-time library utilizing the target platform C compiler. Note that this compiler could be the very same as was in use in the environment from which RuleWorks was called. For example the RuleWorks-produced C can be compiled simultaneously with the Visual C++ sources.
- The resulting object code will execute on the platform corresponding to the C compiler utilized.
- Programs produced with RuleWorks can be created in modules which can be individually tested, debugged and used with new applications as required.

Compiler

RuleWorks achieves high performance through a compiled language implementation, and utilizes a variant of the Rete match algorithm to represent the interdependencies of rules and data in a highly efficient way.

The Rete network greatly speeds up the pattern-matching (inference) operation by eliminating the need for exhaustive redundant tests at execution time.

SQL Interface

RuleWorks provides a built-in interface to SQL (Structured Query Language) which allows users on OpenVMS to easily read data from an Oracle Rdb database into RuleWorks working memory, and values from working memory into an Oracle Rdb database. The SQL interface consists of a set of RHS actions which allow fetch, insert, and update operations with either a simple, one-to-one mapping between object class names and database table names, and a corresponding one-to-one mapping between objects and tuples, or a more flexible form which supports arbitrary mappings.

Run-Time System

The RuleWorks run-time system includes a command interpreter which lets the user control the running of RuleWorks programs and issue debugging commands.

The RuleWorks debugging commands can be used to find errors in the program and to interact with it while it is running. The DEBUG action and the DEBUG qualifier together allow the user to control the invocation of the RuleWorks command interpreter with no changes to the source code. If your program calls routines written in another language, you use that language's debugger to find errors in those routines.

The RuleWorks run-time system also provides an application programming interface for use from other languages.

HARDWARE REQUIREMENTS

Processors Supported

Alpha : All models
VAX: All models supported by OpenVMS and DEC C
Intel: 486 or better

Disk Space Requirements (Block Cluster Size = 1):

These counts refer to the disk space required on the system disk. The sizes are approximate; actual sizes may vary depending on the user's system environment, configuration, and software options.

For OpenVMS VAX Systems

Disk space required for installation: 8,611 blocks
(4,408 Kbytes)
Disk space required for use (permanent): 5,317 blocks
(2,722 Kbytes)

For all Alpha and Intel Systems

Disk space required for installation: 13,779 blocks
(7,055 Kbytes)
Disk space required for use (permanent): 8,991 blocks
(4,603 Kbytes)

SOFTWARE REQUIREMENTS

The requirements listed below apply to currently available RuleWorks kits. RuleWorks can be ported to run on almost any platform for which an ANSI C compiler is available.

For OpenVMS Alpha Systems

- OpenVMS Alpha Operating System V6.2 or higher
- DEC C or DEC C++ V5.5 or higher

For OpenVMS VAX Systems

- OpenVMS VAX Operating System V6.2 or higher
- DEC C or DEC C++ V5.5 or higher

For Intel Systems

- Windows 95 Operating System, or
- Windows NT Operating System V4.0
- WATCOM C/C++ V10.0, or
- Borland C++ V4.5, or
- Borland C++ Builder V1.0
- Microsoft Visual C++ V5.0

For Alpha DIGITAL UNIX Systems

- DIGITAL UNIX Operating System V4.0 or higher
- DEC C V5.2 or higher, or
- DEC C++ V5.5 or higher

For Alpha Windows NT Systems

- Windows NT Operating System V4.0 or higher
- Microsoft Win32 SDK for Windows NT, or
- Microsoft Visual C++ V5.0, or
- Microsoft CL AXP compiler V8.00 or higher

OpenVMS Tailoring

The following OpenVMS classes are required for full functionality of this layered product:

- OpenVMS Required Saveset
- Programming Support
- Utilities

GROWTH CONSIDERATIONS

The minimum hardware/software requirements for any future version of this product may be different from the requirements for the current version.

DISTRIBUTION MEDIA

CD-ROM

ORDERING INFORMATION

Standard Packages:

A RuleWorks™ DIGITAL Custom Application (DCA) Package contains;

- a specific (CPU + operating system) compiler (language translator),
- all currently available standard Run-time libraries,
- a single developer license for the compiler,
- a multiple license for use of the Run-time libraries within your company, with applications you develop,
- the User / Language Reference Guide, Installation Instructions and Release Notes.

Part No.	Media	Description
Q2-00HAZ-69	CD-ROM	RuleWorks DIGITAL Custom Application (DCA) Package
Q2-00HAZ-79	n/a	RuleWorks DCA Warranty
Q2-00HA9-A9	n/a	RuleWorks DCA Services

Customised Packages:

A customised RuleWorks™ DCA Package consists of everything in a standard package, plus Run-time libraries for additional platforms and / or ANSI C compilers.

The RuleWorks compiler supplied may be one from a standard package, in which case the customised package allows cross-platform development for additional target platforms and / or compilers.

Alternatively, the RuleWorks compiler may be ported to run on another platform, so that RuleWorks applications can be developed and run on that platform.

A single Part number covers all standard and customised packages. Customised packages are individually priced.

Evaluation Copies

Customers who make special requests to evaluate RuleWorks may be supplied with a customised, time-limited RuleWorks DCA Package.

Resale of Applications using Run-time Library

For re-selling an application developed with RuleWorks, multiple license packages, or single application run-time licenses for use with one run-time library, are available.

SOFTWARE PRODUCT SERVICES

Contact your local DIGITAL Sales office for support and consulting related information.

Alternatively e-mail RuleWorks@digital.com.

Or visit our Web site at :

<http://www.digital.com/info/ruleworks>
<http://www.europe.digital.com/info/ruleworks>
<http://www.asia-pacific.digital.com/info/ruleworks>

SOFTWARE WARRANTY

Warranty for this software product is provided by DIGITAL with the purchase of a license for the product as defined in the Software Warranty Addendum of this SPD.

© 1997 Digital Equipment Corporation. All rights reserved.

® UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd..

™ The DIGITAL Logo, Alpha AXP, AXP, CI, DEC, DIGITAL, MicroVAX, OpenVMS, RuleWorks, VAX, DEC OPS5 and VMS are trademarks of Digital Equipment Corporation.

® MS-DOS and Microsoft are registered trademarks of Microsoft Corporation.

™ Windows and Windows NT are trademarks of Microsoft Corporation.

™ Pentium is a trademark of Intel Corporation.

™ Oracle Rdb is a trademark of Oracle Corporation.

All other trademarks and registered trademarks are the property of their holders.