



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

XrulesTM

rule based programming extensions for HyperCard®

Demo

Version 1.0.1



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

Copyrights

All materials included with the Xrules product are copyrighted by integrated lab solutions inc. with all rights reserved.

The "Xrules™ Tutorial" and "Xrules™ Demo" stacks may be freely distributed, if they are distributed in their entirety and without modification. No person or entity may distribute parts of these stacks or sell these stacks or any part of these stacks without the express written permission of integrated lab solutions inc.

No person or entity shall distribute or sell the "Xrules™" stack or any part of the "Xrules™" stack without the express written permission of integrated lab solutions inc. unless all of the following conditions are met:

- 1) The person or entity distributing parts of the "Xrules™" stack must be a registered user of Xrules and the parts must be distributed as part of that person or entity's application.
- 2) The distributed application may not contain the "compXrules" XCMD in any form nor may it contain the "LEXT" resource in any form. The names of other Xrules resources may not be changed.
- 3) The distributed application may not contain the Xrules card (the top card), the "notice" card, the "compiler" card, or the "monitor" card.
- 4) If the application contains a copyright notice, the following notice must be included with that copyright notice. If the application does not contain a copyright notice, the following notice must be included in a place visible to the user.

"The XCMDs CHAIN, DELECOMP, INITXRULES, PUSHONSTACK, PUTINTOFACT, and SCROLLTOBOTTOM, the XFCNs FACT, FINDSTR, and POPFROMSTACK, and the hourglass cursor are copyrighted by integrated lab solutions inc. and may not be distributed apart from this stack without the express written permission of integrated lab solutions inc."

The Xrules manuals may not be copied.

L I C E N S E

Registered users of Xrules are licensed to use the product with the XCMD "compXrules", the "LEXT" resource, and/or the Xrules cards intact on one computer at a time. Xrules may not be installed on a file server to be used by multiple users.

T R A D E M A R K S

Xrules™ and the ils logo are trademarks of integrated lab solutions inc.

Macintosh® is a registered trademark of Apple Computer Inc.

HyperCard® and HyperTalk® are registered trademarks of Apple Computer Inc., licensed to Claris Corp.

SuperCard™ is a trademark of Silicon Beach Software, Inc.

All other trademarks or registered trademarks are the property of their respective holders.

W A R R A N T I E S

Integrated lab solutions inc. makes no warranties, express or implied, as to the correctness, fitness for a particular purpose, quality, or performance of this software. Integrated lab solutions inc. assumes no liabilities for direct, indirect, special, incidental, or consequential damages resulting from any defect in this software or its documentation. The user assumes all risks for the quality and performance of this software. If you purchase the product and receive any part of it in an unusable condition or if the distribution disk fails within 90 days of purchase, contact integrated lab solutions inc. for instructions on obtaining a free replacement.

©1991 integrated lab solutions inc.

This document may be copied if it is copied in its entirety and without modification.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

Xrules™

VERSION 1.0.1

Xrules enables you to develop diagnostics, tutorials, "expert systems", and other rule based applications quickly and inexpensively using HyperCard. Xrules provides the inference engine and rule base compiler, HyperCard provides the user interface, and you provide the knowledge. Unlike some shells, Xrules allows you to control the inference process, giving you greater flexibility in developing the user interface. Xrules is a tool for experts and novices alike and includes a tutorial for those not familiar with rule based programming. If you know HyperTalk, you can use Xrules. Features include:

- Compatibility with HyperCard versions 1.2 and later (including 2.0) and SuperCard 1.5
- Allows full use of HyperCard capabilities for user interface
- Implemented as XCMDs for speed and flexibility
- HyperTalk-like syntax for rule base
- Rule base compiled and stored as resources to protect rule base and enhance speed
- Backward/forward chaining and daemons
- No royalties for distributed stacks
- Trace/debug facilities to aid development
- Interactive tutorial explaining the fundamentals of rule based systems
- Documentation
- Telephone support for registered user
- \$99

Xrules provides capabilities required by serious applications while removing the mystique, complexity, and expense often associated with "expert systems".



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

DEMO OVERVIEW

The demonstration disk includes a tutorial stack ("Xrules Tutorial") and a demo version of the development stack ("Xrules Demo").

The Xrules product assumes that you are already familiar with HyperCard and HyperTalk. However, you do not have to be conversant in HyperTalk to use the tutorial.

If you are not already familiar with developing rule based systems, you should read the "Xrules Tutorial" section of the "Xrules Tutorial" stack. This tutorial will introduce you to the fundamentals of rule based programming. It addresses the components and operation of rule based systems and leads you through development and execution of a simple system. The "Tutorial Shell" card in the "Xrules Tutorial" stack allows you to interactively build and execute the examples in the tutorial.

If you are already familiar with HyperCard, HyperTalk, and rule based systems, you can skip the tutorial and proceed to the "Xrules User's Guide" in the "Xrules Demo" stack. The user guide includes a "Sample Application" section which leads you through developing a sample application. You should read the "Xrules Introduction" section and then develop the sample application. The rest of the stack gives detailed information on the development of an Xrules rule base, the operation of the Xrules inference engine, the uses of the various XCMDs and XFCNs, and the use and structure of the development stack.

The "Xrules Demo" stack has the same capabilities as the full-featured Xrules product except that the rule base is limited to 10 rules and daemons. The full-featured product limits the rule base to 16 HyperCard fields.

You may freely distribute the "Xrules Demo" and "Xrules Tutorial" stacks if you conform to the conditions specified in the copyright notice.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

THE XRULES DEVELOPMENT STACK

Xrules provides rule based programming extensions for HyperCard. Xrules' HyperTalk-like syntax and its development environment make it very easy to use. Like many shells, Xrules uses propositional logic with forward and backward chaining. Unlike most shells, Xrules behaves as an extension to HyperCard. As such, you control it from your application rather than having it control your application. Also, unlike most shells using HyperCard, Xrules compiles the rule base and executes via XCMDs. The "Xrules" stack contains the XCMDs, XFCNs, and cards to enable you to develop your applications.

You develop applications for Xrules by beginning with the development stack and modifying it to suit your application. Figure 1 illustrates the layout of the development stack.

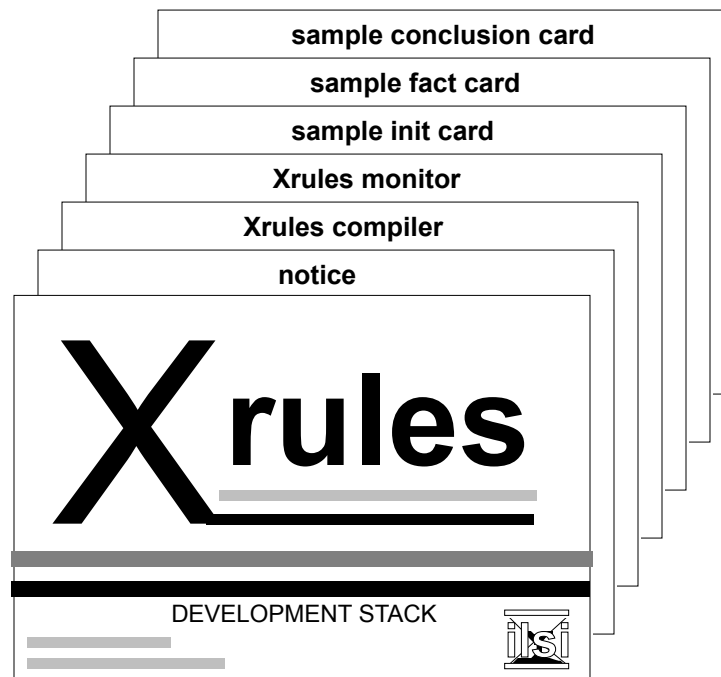
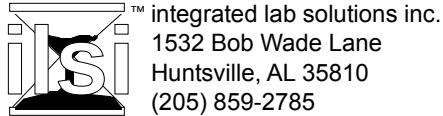


Figure 1 - The Development Stack

Xrules allows full use of HyperCard and HyperTalk for developing the user interface. Cards and handlers developed by you provide the interface between the user and the inference engine. You control Xrules from scripts via XCMDs and XFCNs.

The XCMDs and XFCNs provided with Xrules allow you to control compilation of the rule base, initialization of the system, inference processing, fact assertion, and other miscellaneous functions. These XCMDs and XFCNs are in the resource fork of the Xrules development stack.



The development stack also contains cards to assist you in developing your application. The compiler card is simply a card containing a field for you to enter the rule base and containing buttons for compile, find string, cleanup, and delete compiler functions. The monitor card allows you to execute the system while tracing the steps taken by the system. Sample init, fact, and conclusion cards are included to help you get started.

If you are already familiar with HyperCard, HyperTalk, and rule based systems, you can very quickly begin developing applications using Xrules.

BUILDING AN APPLICATION

The following outline enumerates the steps required to implement a rule based system using Xrules.

- 1) Copy the Xrules development stack from the distribution disk. If you are using HyperCard 2.0, convert the copy to a HyperCard 2.0 stack. Rename the stack to fit your application.
- 2) Open the copy and enter the rules in the rule base field of the compiler card. You may use up to 16 HyperCard fields to contain the rule base. Figure 2 shows the compiler card with a rule base.

Xrules
Xrules Compiler delete compiler

Rule Base

```
fact
passengers, wheels, "used for cargo", make

rule Corvette
if
  fact car and
  fact make = Chevy and
  fact passengers = 2
then
  put Corvette into fact model
  conclude "The car is a Corvette."

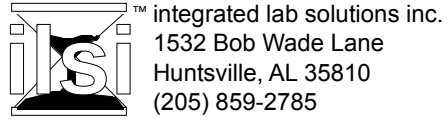
rule Camaro
if
  fact car and
  fact make = Chevy and
  fact passengers = 4
```

String to find
-+-

↩ → find cleanUp ↪
findNext compile

Figure 2 - The Compiler card

- 3) Create any handlers, "fact" cards, and "conclusion" cards required. These handlers and cards contain scripts to communicate with the user, assert facts, and control the operation



of the inference engine. Figure 3 illustrates the process.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

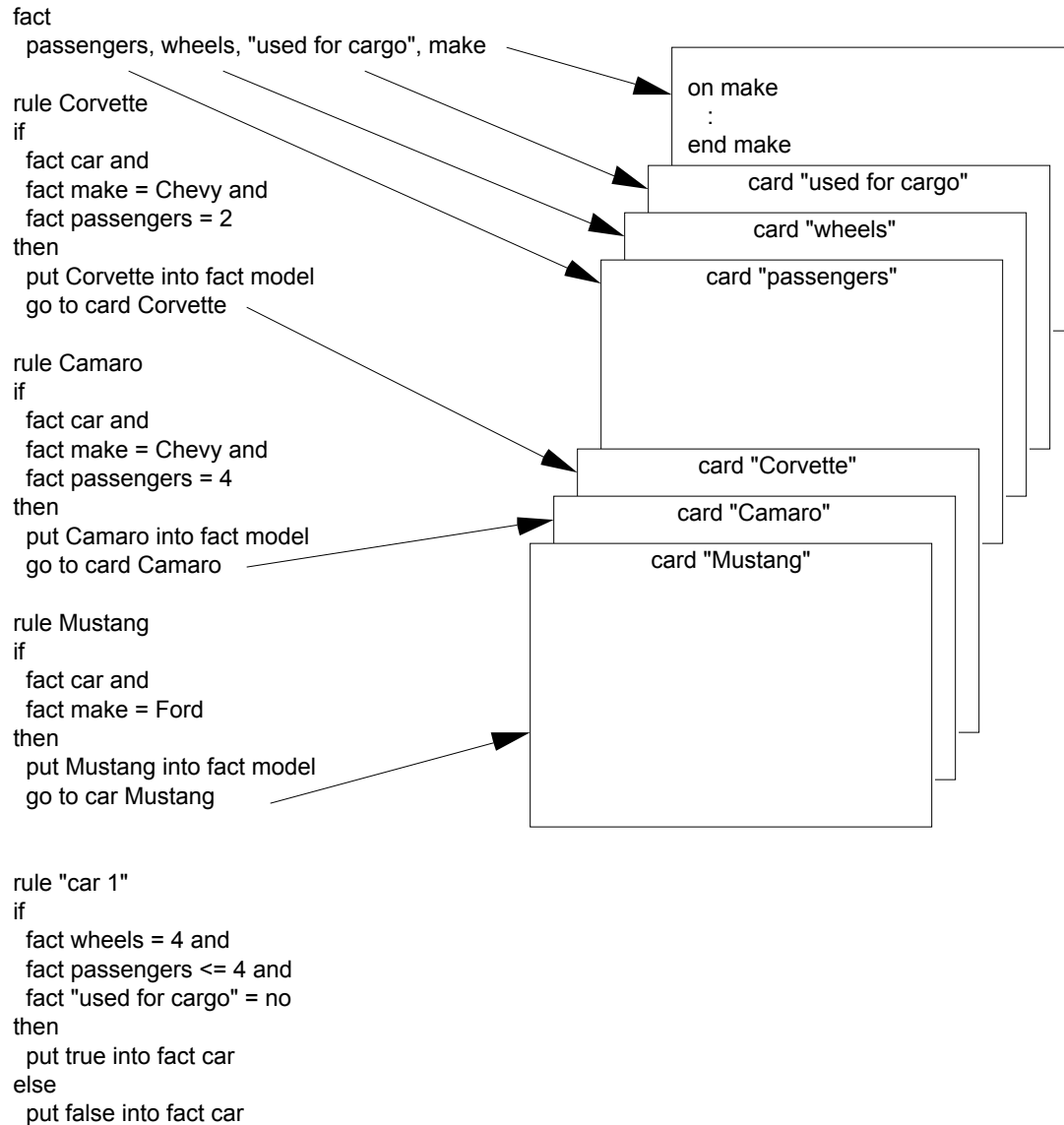
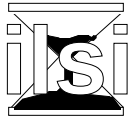


Figure 3 - Building the Stack

- 4) Compile the rule base, fix any errors, and recompile until the rule base compiles without errors.
- 5) Debug and test your application. If needed, execute your application from the monitor card to examine the inference process. Figure 4 shows the monitor card. Correct any errors in the cards or scripts and continue. Correct any errors in the rule base on the compiler card and repeat steps 4 and 5 until the system works correctly.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

☐ trace ☐ step

Xrules
Xrules monitor **Backward Chain**

Facts

Stack

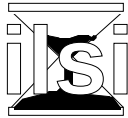
Goal

Figure 4 - Monitor Card

- 6) Copy the stack, delete the compiler (via the "delete compiler" button on the compiler card), the Xrules cards (the logo card, the copyright notice card, the compiler card, and the monitor card), and any extraneous cards from the copy. Add your copyright notice with the following text included:

"The XCMDs CHAIN, DELECOMP, INITXRULES, PUSHONSTACK, PUTINTOFACT, and SCROLLTOBOTTOM, the XFCNs FACT, FINDSTR, and POPFROMSTACK, and the hourglass cursor are copyrighted by integrated lab solutions inc. and may not be distributed apart from this stack without the express written permission of integrated lab solutions inc."

Figure 5 illustrates this process.



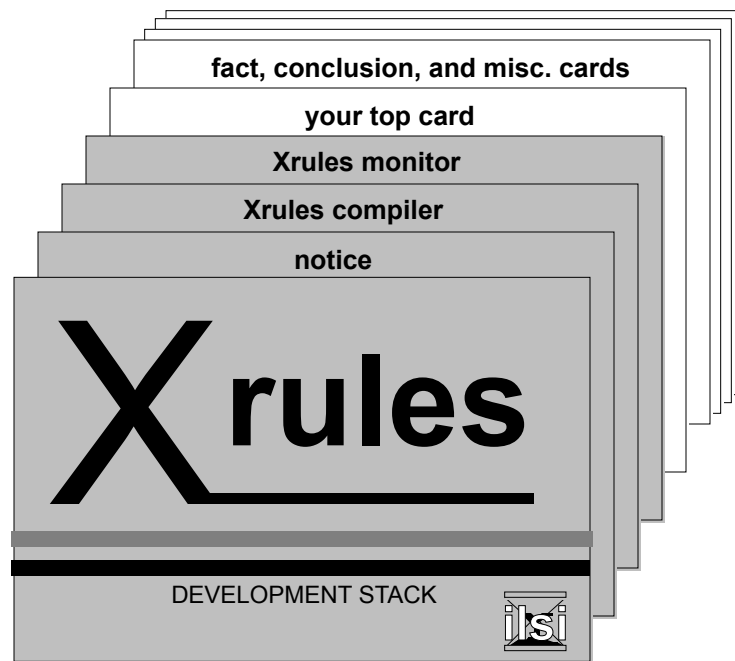
integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

Copy the original stack.



original stack

Delete the Xrules compiler and Xrules specific cards from the copy. Add your copyright notice and include the Xrules copyright notice.



Make copies for distribution.

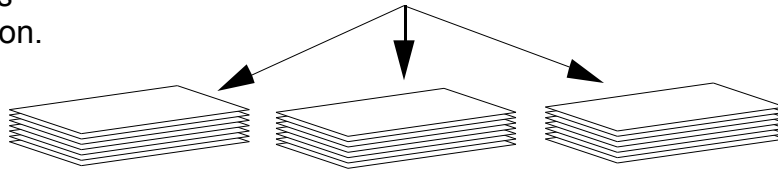


Figure 5 - Preparing the Stack for Distribution

- 7) Apply appropriate protection to the copy and distribute copies of the copy. Retain the original stack with compiler and Xrules cards intact so that you can make updates to your application.

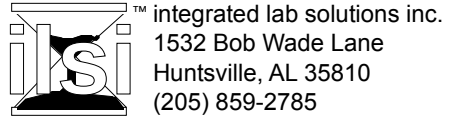
NOTE: THE COMPILER XCMD AND XRULES CARDS MUST BE DELETED FROM AND THE COPYRIGHT NOTICE ADDED TO DISTRIBUTED COPIES!

(Stacks may be distributed without royalty fees only if they do not contain the compiler XCMD and the Xrules cards and only if they contain the copyright notice).

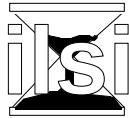
XRULES XCMDs AND XFCNs

The following XCMDs are supplied with Xrules:

CHAIN direction



The "chain" XCMD invokes the inference engine.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

COMPXRULES field_spec {,field_spec}

The "compXrules" XCMD compiles the rule base.

DELECOMP

This XCMD deletes the rule base compiler.

INITXRULES [goal]

The "initXrules" XCMD initializes the system.

PUSHONSTACK type, id, [use]

The "pushOnStack" XCMD pushes an item on the stack.

PUTINTOFACT value, fact_id

The "putIntoFact" XCMD assigns a value to a fact.

SCROLLTOBOTTOM field

The "scrollToBottom" XCMD scrolls to the bottom of a field.

The following XFCNs are included with Xrules:

FACT(fact_id)

The "fact" XFCN returns the value of a fact.

FINDSTR(starting_position, string_to_find, container_to_search)

The "findStr" XFCN locates a string and returns its position.

POPFROMSTACK ()

The "popFromStack" XFCN pops the top item from the stack and returns information about that item.

C O N T R O L L I N G X R U L E S

As indicated earlier, Xrules does not attempt to control your application the way many shells do. You decide when to invoke the Xrules functions. Figure 6 illustrates invoking the various Xrules XCMDs.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

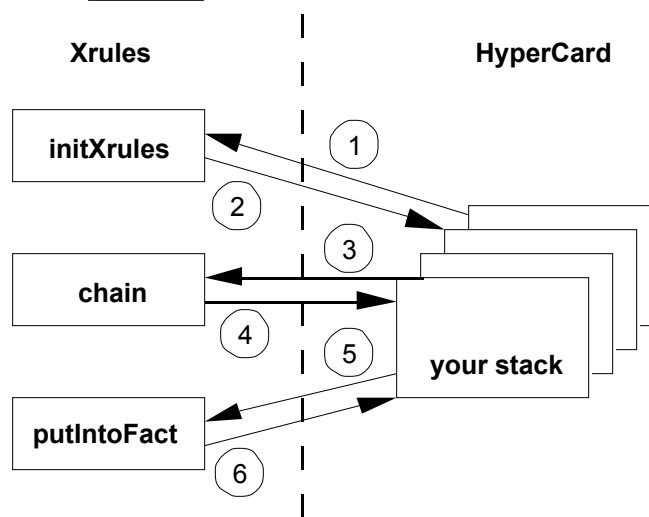


Figure 6 - Controlling Xrules

- 1) Your application initializes Xrules via the "initXrules" XCMD when appropriate. You must initialize the system to begin a consultation session.
- 2) After initializing the system, Xrules returns control to your application at the point where the "initXrules" XCMD was invoked. After Xrules returns control, you may perform any processing appropriate for your application, including reinitializing the system, invoking the inference engine, or asserting facts.
- 3) Your application invokes the inference engine via the "chain" XCMD when appropriate. Generally, you invoke the inference engine after you have initialized the system, to start the inference engine; after you have asserted facts, to resume the inference engine; and after the inference engine has announced a conclusion (via a "go to" statement), to resume the search for other solutions.
- 4) The inference engine returns control to your application under the following conditions: (1) it discovers a fact that has not been asserted for which there is a card having the same name, (2) it reaches a conclusion for which there is a "go to" statement, or (3) the inference process can go no further. When returning after completing processing for the session, the inference engine will display the card from which the "initXrules" XCMD was invoked. After Xrules returns control, you can perform any processing appropriate for your application, including reinitializing the system, invoking the inference engine, or asserting facts.
- 5) Your application asserts facts via the "putIntoFact" XCMD when appropriate. Generally, you assert facts when the inference engine has summoned a fact card and returned control to HyperCard.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

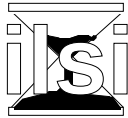
- 6) After asserting the fact, Xrules returns control to your application at the point where the "putIntoFact" XCMD was invoked. After Xrules returns control, you can perform any processing appropriate for your application, including reinitializing the system, invoking the inference engine, or asserting facts.

NOTE: When Xrules returns control to HyperCard, you may visit other stacks or other applications and Xrules will maintain its state. However, when you need to invoke one of the Xrules XCMDs or XFCNs, you must return to the development stack (which becomes your application stack) since these XCMDs and XFCNs and their data are contained only in the development stack.

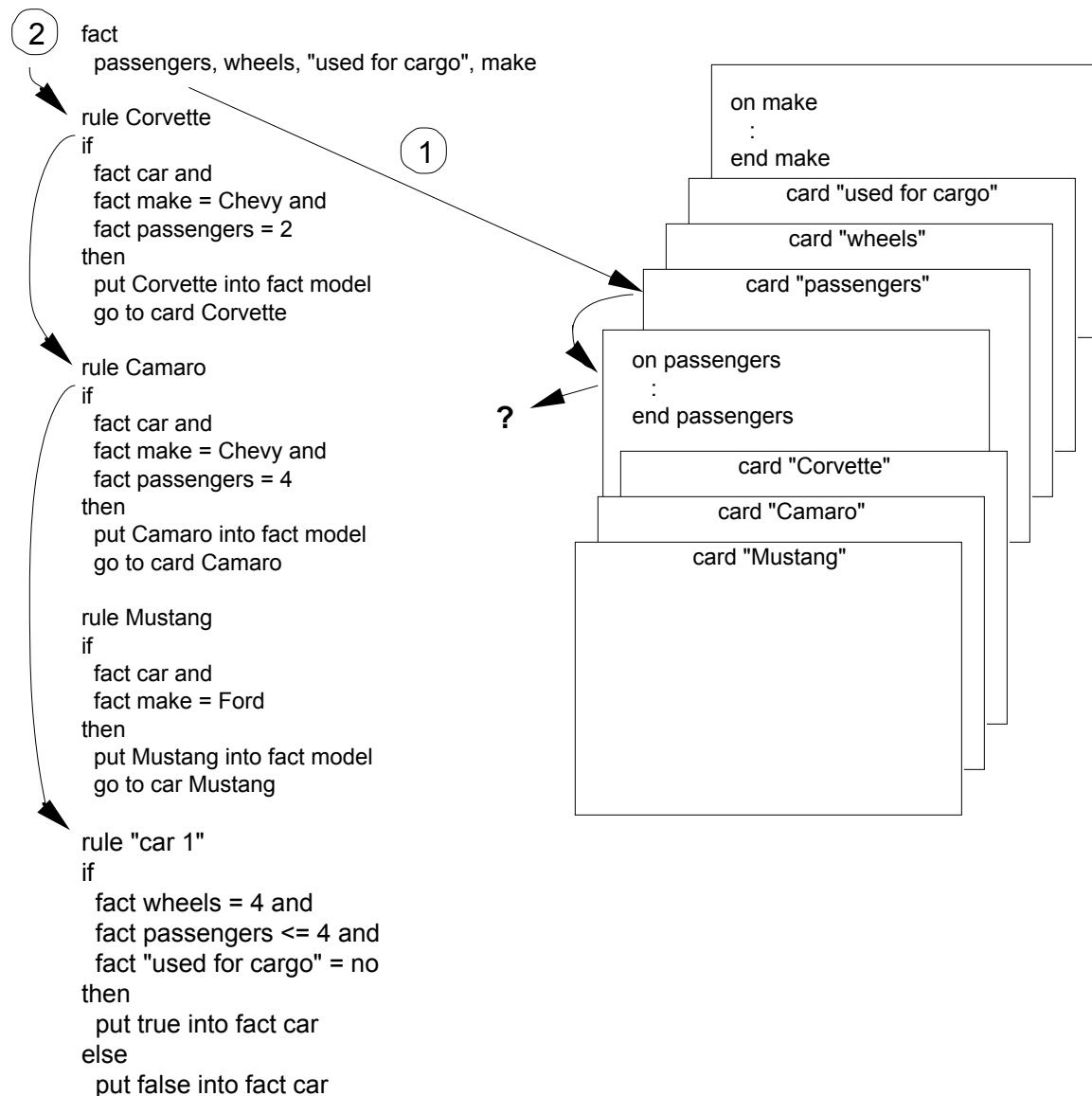
F O R W A R D C H A I N I N G

When forward chaining, the system examines facts in the order that they were declared in the fact declaration section of the rule base. It will first attempt to go to a card having the same name as the fact. If no card exists or if the fact is still not asserted when the inference engine is restarted, it will attempt to execute a handler with the same name as the fact. If the fact is still not asserted after attempting to execute a handler, it will set the fact to unknown ("?"). Once a fact card and handler have been visited, they will never again be visited for the fact.

Figure 3 illustrates the forward chaining process. After initializing the system (via the "initXrules" XCMD), you invoke the inference engine via the "chain" XCMD. The first fact to be examined in our sample is "passengers" because it is the first fact in the fact declarations. Since there is a card for the fact, the inference engine will go to that card and return control to HyperCard (1). If you assert the fact (via the "putIntoFact" XCMD) and restart the inference engine (via the "chain" XCMD), the inference engine will begin examining those rule using fact "passengers" (2). If you did not assert the fact, the inference engine will execute the handler named "passengers." If you assert the fact there, the inference engine will then begin examining those rules using fact "passengers." Note that handlers are treated as subroutines; the inference engine does not return control to HyperCard. If you did not assert the fact in either case, the inference engine will set the fact to unknown ("?") and examine the rules using fact "passengers." After processing fact "passengers," it will begin processing fact "wheels."



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785



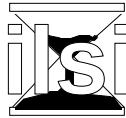
Figure

7 - Forward Chaining

BACKWARD CHAINING

When backward chaining, if the system needs a fact that has not been defined it will first check for a card having the same name as that fact. If there is no card or if the fact is still not asserted when the inference engine is restarted, the system will attempt to execute a handler having the same name as the fact. If the the fact is still not asserted after attempting to execute a handler, the system will process rules capable of asserting the fact, in the order that they appear in the rule base, until the fact is asserted. If the fact is not asserted after processing the rules, it will set the fact unknown ("?"). Once a fact card and handler have been visited, they will never again be visited for the fact.

Figure 8 illustrates backward chaining. When backward chaining, you first initialize the system



™ integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

specifying a goal (via the "initXrules" XCMD). "Model"



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

is the goal in our example (1). Next, you invoke the inference engine via the "chain" XCMD. The inference will search for rules capable of setting goal "model" and will find rule "Corvette." In rule "Corvette", the first unknown fact encountered is "car." The inference engine will go to the card named "car" and return control to HyperCard (2). If you assert the fact (via the "putIntoFact" XCMD) and restart the inference engine (via the "chain" XCMD), it will continue processing rule "Corvette." If you elect not to assert fact car, when you restart the inference engine it will execute the handler named "car." If you do not assert fact "car" in the handler, the inference engine will begin examining rules capable of asserting fact "car." If fact car is not asserted after exhausting all possibilities, the inference engine will set the fact to unknown ("?") and continue processing rule "Corvette."

PROCESSING DAEMONS

Daemons are processed as a result of asserting facts. The system handles daemons in a manner similar to the way that it handles rules when forward chaining. When a fact is asserted, all daemons using that fact are examined whether the fact was asserted by a rule, by the "putIntoFact" XCMD, or by another daemon. The daemons are processed in a depth first manner in the order that they appear in the rule base.

THANKS

Thank you for your interest the Xrules product. If you would like more information about the product, contact us. We encourage you to let us know how you feel about the product, what changes you would recommend, what additional features you would like to see, and, of course, any bugs you may discover.

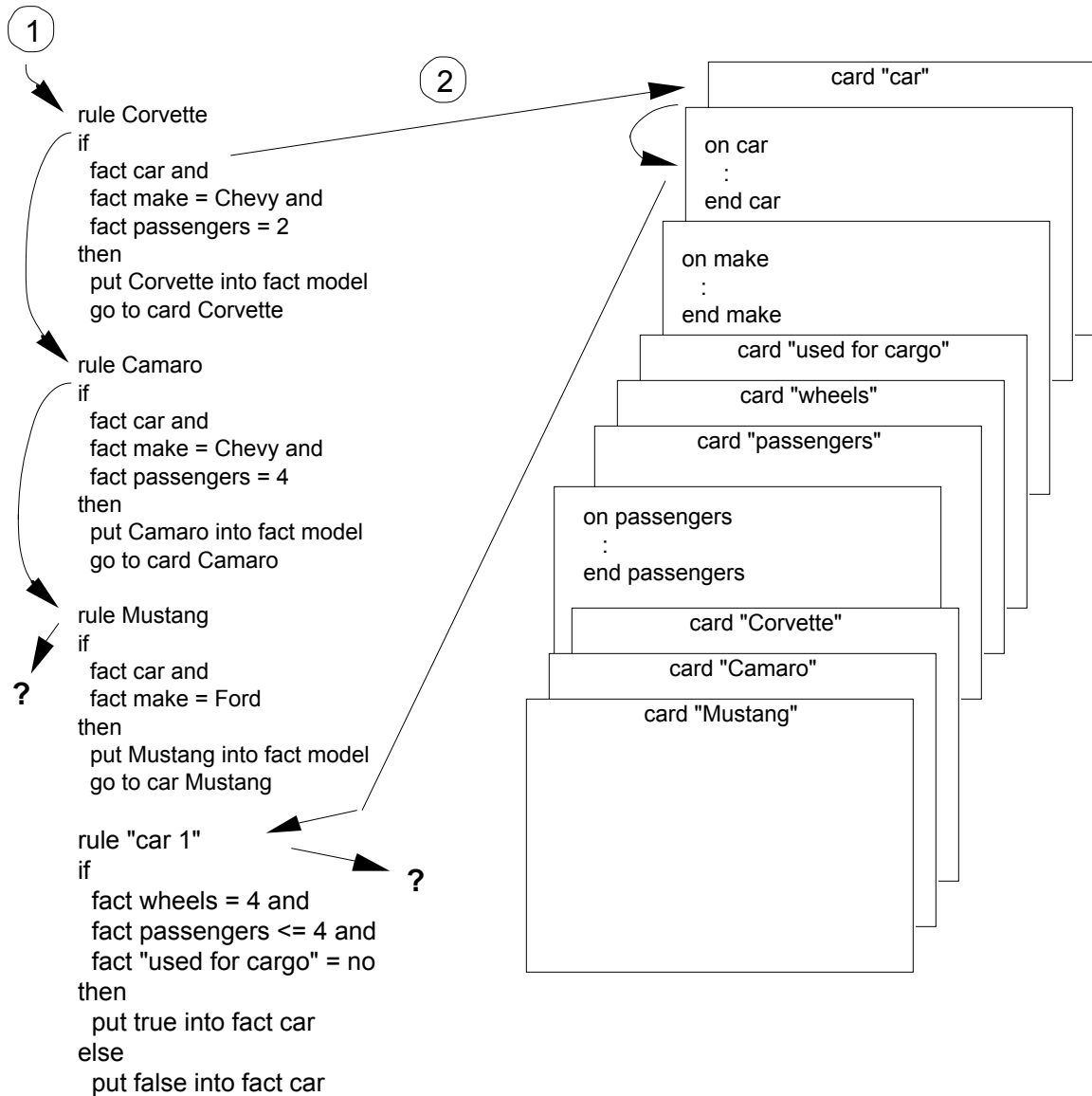
Write or call us at:

integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785

We also thank Dr. Daniel Rochowiak, of the University of Alabama in Huntsville, for his guidance and assistance in developing the product.



integrated lab solutions inc.
1532 Bob Wade Lane
Huntsville, AL 35810
(205) 859-2785



8 - Backward Chaining