11:29:27 OCA PAD AMENDMENT - PROJECT HEADER INFORMATION 05/14/92

Active

Project #: E-15-605 Cost share #: Rev #: 1
Center # : 10/24-6-R7390-0A0 Center shr #: OCA file #:
Work type : RES
Contract#: RK55038 Mod #: CHANGE #1 Document : PO
Prime #: Contract entity: GTRC

Subprojects ? : Y CFDA:
Main project #: PE #:

Project unit: ENGR COLL Unit code: 02.010.108
Project director(s):
GOURISANKAR V ENGR COLL (404)894-7772

Sponsor/division names: LOCKHEED AERONAUT SYS CO-GA /
Sponsor/division codes: 261 / 007

Award period: 911219 to 920403 (performance) 920403 (reports)

Sponsor amount	New this change	Total to date
Contract value	0.00	14,492.50
Funded	0.00	14,492.50
Cost sharing amount		0.00

Does subcontracting plan apply ?: N

Title: PLANNING AND ORGANIZATION OF IRAD ACTIVITIES IN COMPUTER AIDED LOGISTIC.....

PROJECT ADMINISTRATION DATA

OCA contact: E. Faith Gleason 894-4820

Sponsor technical contact

TED RICK
(404)494-8650

Sponsor issuing office

JEAN BELL
(404)494-8361

LOCKHEED
D/52-34 Z/0510
86 SOUTH COBB DRIVE
MARIETTA, GA 30063

Security class (U,C,S,TS) : U ONR resident rep. is ACO (Y/N)
Defense priority rating : N/A N/A supplemental sheet
Equipment title vests with: Sponsor GIT X

Administrative comments -
* CHANGE ORDER EXTENDS DUE DATE TO APRIL 3, 1992.

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 07/27/92

Project No. E-15-605 Center No. 10/24-6-R7390-0A0

Project Director GOURISANKAR V School/Lab ENGR COLL

Sponsor LOCKHEED AERONAUT SYS CO-GA/

Contract/Grant No. RK55038 Contract Entity GTRC

Prime Contract No.

Title PLANNING AND ORGANIZATION OF IRAD ACTIVITIES IN COMPUTER AIDED LOGISTIC..

Effective Completion Date 920403 (Performance) 920403 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	
Final Report of Inventions and/or Subcontracts	Y	
Government Property Inventory & Related Certificate	N	
Classified Material Certificate	N	
Release and Assignment	N	
Other	N	

Comments

Subproject Under Main Project No.

Continues Project No.

Distribution Required:

Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Managment	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
Other	N
	N

NOTE: Final Patent Questionnaire sent to PDPI.

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT (SUBPROJECTS)

Closeout Notice Date 07/27/92

Project No. E-15-605 Center No. 10/24-6-R7390-0A0_

Project Director GOURISANKAR V_____ School/Lab ENGR COLL_____

Sponsor LOCKHEED AERONAUT SYS CO-GA/_____

Project # E-25-M95 PD FADEL G M Unit 02.010.126 T
PO # RK55038 MOD# CHANGE #1 MECH ENGR *
Ctr # 10/24-6-R7390-0A1 Main proj # E-15-605 OCA CO EFG
Sponsor-LOCKHEED AERONAUT SY / 261/007
PLANNING AND ORGANIZ
Start 911219 End 920402 Funded 7,246.25 Contract 7,246.25

LEGEND
1. * indicates the project is a subproject.
2. I indicates the project is active and being updated.
3. A indicates the project is currently active.
4. T indicates the project has been terminated.
5. R indicates a terminated project that is being modified.

E15-605

PLANNING AND ORGANIZATION OF IRAD ACTIVITIES IN CALSTAR

Final Report
Project E-15-605/E-25-M95
RFQ # DJB-91-0007

Presented to
The Product Support Division
LOCKHEED AERONAUTICAL SYSTEMS INC.

by

Georges Fadel
Vellapillil Gourisankar
Karen Hotz

College of Engineering
Georgia Institute of Technology
March 1992

Abstract:

Present and future computers are responsible for a change in the approaches to rapid prototyping. Rapid prototyping in this study, is the ability to modify parameters interactively in a simulation, and observe on a computer screen the results of such modifications. Large computer programs that simulate systems are still and will always be used, however, a new generation of tools from symbolic manipulators to expert systems and genetic algorithms are now used daily by managers, analysts and engineers. These tools can operate concurrently with computer simulations and users would like to interact directly with such programs.

This report summarizes investigations of programming languages, programming environments and generally available tools for logistics support. It characterizes the languages or environments in terms of their Graphic User Interface capabilities (GUIs), suitability for "rapid prototyping", and amounts of procedural programming.

Introduction:

The personal computer and the workstation radically changed the way analysts, engineers and managers can analyze, design and manage their particular tasks. Most of the manageable computer simulations can now run on desktop computers, and new software tools are reducing the time it takes to perform an analysis and interpret the results. Probably one of the most significant contribution of the desktop computer is the ability to visually display parameters, objects, and behaviors. Instead of the stacks of paper with numbers that had to be digested, a graph can convey the information practically instantaneously to the user. Since this capability now exists, the user would like the ability to tweak parameters on the fly, i.e. use some input device and modify one or more variables, displaying the magnitude of this or these variables on the computer screen, and instantaneously, generate response graphs. In order to achieve this high level of interaction, the user needs to have access to non-procedural languages, GUIs, symbolic tools and the like.

The second important requirement for the user is the capability to port the software to other sites for remote use and to other hardware platforms. In this report, we do however stress the need to have the software run on IBM PC type computers running under a widely available operating system like DOS and Windows.

Identification of software:

The first task of this study is to identify available candidate software for rapid prototyping that runs on desktop computers. Literature searches, contacts with industry, and discussions with the Lockheed technical contacts allowed us to identify a number of products. The first cut was to reject programming languages since extensive code development would be needed to develop the environment and the following list of ten products was selected:

1. ASE by Software Artistry
2. Exsys Professional by Exsys

3. ART-IM by Inference
4. ADS 6.0 by Aion
5. Kappa-PC by Intellicorp
6. Nexpert Object by Neuron Data
7. 1st Class by AICorp
8. PDC Prolog by Prolog Dev. Corp.
9. Smalltalk V/win by Digitalk
10. Smalltalk 80 by ParkPlace

The vendors of the software were contacted to identify the capabilities of their products, and tables 1 and 2 were built. Note that the capabilities illustrated in these tables are as identified by the promotional literature or by phone conversations with the vendors. Subsequent testing showed that many available capabilities were in fact not readily accessible or were part of future releases of the software. The tables were generated to allow the comparison of the products on paper and perform a first cut to select a minimum of promising packages.

Four packages were selected for further testing. These are:

1. **Exsys** Professional by Exsys
2. **Kappa PC** by Intellicorp
3. **PDC Prolog** by Prolog Dev. Corp.
4. **Smalltalk** V/Win by Digitalk

The selected programs were acquired, and a set of tests established to test for rapid prototyping capabilities. These tests included the following:

1. Ability to generate non-procedural code
2. Availability of windowing capabilities
3. Availability of GUIs for graphs
4. Speed
5. Connectivity with other programs
6. Expert capabilities
7. Symbolic capabilities
8. Assessment of future potential

A further breakdown of the testing is illustrated in the following section on hands-on evaluation:

HANDS-ON TEST PROCEDURE

This section outlines the hands-on tests that were planned to be conducted on each candidate product:

(1) SQL Transactions.

A table with numerical and character data will be defined in a relational data base system supporting SQL. The table resident data will be tested for extraction and use within the rapid

prototyping environment. The method of extraction (flat file based or peer to peer), easiness of extraction, and sufficiency of documentation will be evaluated.

(2) Spreadsheet Transactions.

A spreadsheet with numerical and character data will be created. This data will be tested for extraction and availability within the rapid prototyping environment. The method, easiness of extraction and sufficiency of documentation will be evaluated. The availability of dynamic data link capability with external applications will be checked.

(3) Mathematical Modeling, Analysis.

A set of algebraic equations will be modeled when possible (symbolically, procedurally, or otherwise) within the rapid prototyping environment. Various avenues available for solution will be examined.

(4) Data Presentation.

A file containing a list of 2-D and 3-D data will be prepared. This data will be displayed within the rapid prototyping environment. The method, flexibility, and easiness of displaying data graphically will be examined.

(5) GUI Development.

We will implement a directory listing capability within the rapid prototyping environment to examine the various GUI development features.

(6) Interfacing With External Modules.

We will develop the capability to utilize an external module from within the rapid prototyping environment. Two different external modules will be used: a function developed via a procedural language and a non-procedural module such as an expert system.

(7) Run Time Support.

An executable file will be developed to examine the size, mobility, and resource requirements.

(8) Documentation.

The quality and extend of availability of documentation and technical support will be investigated. The ease of including help files and running without a manual will be assessed.

(9) Expert system development.

The time required to implement a short application (5 rules) will be assessed. Measures of performance will include time to write, time to compile (if needed), time to get running, number of lines of code. The ease of change will also be quantified.

Table 1

Company	Intellicorp	Neuron Data	Aion	Software Artistry	Exsys Inc.
Name	Dave Saslav	Kelly Skarakis		Beth Lowder	Richard Bright
Phone	415-965-5648	415-688-3514	404-551-8270	317-876-3042	505-256-8356
Product	Kappa-PC	Nexpert Object	ADS 6.0	ASE	Exsys Prof.3.0
Dev RAM	640K-2M	3-4+ M	2+ M	640K	640K
Run RAM	640K-2M	640-3M	2M	640K	640K
Disk space	2M	4-12M	5M	5M	2M
Software	Dos 3.0	Dos 3.0		Dos 3.1	Dos 2.0+
	Windows 3.0	Windows 3.0			Windows 3.0
Price dev	3,500	5,000-12,000	8,500	2,000	2,500-5,995
Price exe	450/copy	750-1000/copy	900/copy	250/copy	1,495/dev lic
Site Lic	Y	Y	Y	N	Y
Tech sup	Y	Y	Y	Y	Y
Document	Y	Y	Y	Y	Y
Compile	6/31/92	C	C	C	C INTERFACE
Prog inter	Y	open interface	Y	Y	Y
MS Windows	Y	Y	Y	N	Y
SQL	Sequelink	Y	Y	Y	N DOS ver.
Lotus	Y	Y	Y	N	Y
ASCII	Y	Y	Y	Y	Y
Languages	C;PAS;FOR	C;COB;FOR	PAS;FOR;COB PL1,C,ASSM	C	C
Hyper	LIMITED	Y	Y	Y	Y
Math fn	Y	Y	Y	Y	Y
Symbolic	N	N	N	N	N
Pattern	Y	Y	Y	N	Y
Lists	Y	Y	Y	Y	N
Inherit	Y	?	Y	N	Y
Dates	LIMITED	Y	Y	Y	y
Other	demons				confidence/BI.B.
GUI	6/31/92	open interface	N	Knowledge Eng.	Y
icons	N			N	
menu bars	N			Y	
pop-up	N			Y	
dialog b	Y			Y	
windows	Y			Y	
dials	Y			Y	
scroll bar	Y			Y	
pointing	Y			Y	
selection	Y			Y	
Dev aid	Y	Y	Y	Y	Y
Debugging	Y	Y	Y	Y	Y
Cr help	N	Y	Y	Y	Y
Cr scrn	Y	Y	Y	Y	Y
Graphs	only line	open interface	Y	N	N
Graph lib	Excel	open interface	Y	N	N
Visual aid	Excel	open interface	Y	N	N

Table 2

Company	AICorp	Prolog Dev. Ctr	Digitalk	Parkplace	Inference
Name	Linda Ubertini			Scott Holt	Tech Info
Phone	617-891-6500	404-873-1366	310-645-1082	894-6168	800-322-5590
Product	1st Class	PDC PROLOG	Smalltalk V	Smalltalk 80	ART-IM/Window
Dev RAM	640K	640K	3-5 M	4M	4M
Run RAM	300K	?	3-5 M		4M
Disk sp		720K?	3.2M	10M(+6M)	8M
Software	Dos 2.0+	Dos 3.0+ Window	Windows 3.0	Dos, Unix, Mac	Windows 3.0
Price dev	1995	599	499.95	3,500-5,300	8,000
Price exe	0	0	N		1,600/copy
Site Lic			N	Y	Y
Tech sup	Y	Y	Y	Y	Y
Document	Y	Y	Y	Y	Y
Compile	C;PAS	PROLOG	Y	Y	C
Prog inter	Y	Y	N	Y	Y
MS windows	in 3 months	Y	Y	Y	Y
SQL	?	Qelib	N/ Synerg.	N/Synerg.	N
Lotus	Y	Toolkit	N/DDL	N/DDL	N
ASCII	Y	Y	Y	Y	Y
Languages	C;PAS	C;C++;ASSM	St	St	C
Hyper	Y	Y	Y	Y/Facets	Y
Math fn	Y	Y	Y	Y	Y
Symbolic	Y	N	N	N	N
Pattern	N	Y	?	?	Y
Lists	?	Y	?	?	Y
Inherit	N	Y	Y	Y	Y
Dates	Y	limited	Y	Y	Y
Other	confidence	confidence			
GUI	N	Y	Y	Y	Y write code
icons			Y		
menu bars			Y	Y	
pop-up			Y	Y	
dialog b			Y	Y	
windows			Y	Y	
dials			Y	Y	
scroll bar			Y	Y	
pointing			Y	Y	
selection			Y	Y	
Dev aid	Y	Y	Y	Y	Y
Debugging	Y	Y	Y	Y	Y
Cr help	Y	Y	?	Y	Y
Cr scrn	Y	Y	Y	Y	Y
Graphs	N	Y	Y	Y	N
Graph lib	N	limited	Y	Y	N
Visual aid	N	N			N

RESULTS OF HANDS ON TESTING:

The following four sections summarize the results of the hands on testing:

EXSYS for Windows

According to technical support from Exsys (2/7/92), it is **not possible to perform SQL transactions with the windows version** of this product. They do however have a **built in interface to dBase III**. DBase III files can be accessed through a rule, variable, report generator, or the command language. To access a file one selects dBase III from the data acquisition screen and fill in the blanks. On this screen the data base is specified, and which data is to be stored or retrieved. Fields in the database record and Exsys variable names are used in pairs, each field must have a corresponding Exsys name, and **one variable at a time** is retrieved or stored.

Exsys **has** a built in **interface to LOTUS 123** and it can easily display ASCII files. To send or receive data from the LOTUS spreadsheet, just select LOTUS 123 from the data acquisition screen and then fill in the blanks. Here again, each cell has a corresponding Exsys variable name, and **one cell at a time** can be stored or retrieved. To get data from a ASCII file just select table look up from the data acquisition screen and fill in the blanks.(There can only be two columns of data)

Exsys **does not support** differentiation or integration. But it does handle trig functions and Boolean operations.

Exsys **does not have the capability to create graphs**. An external graphical program would have to be called.

To create a GUI type screen, a page of **code has to be written** to display one screen. This code can be written without leaving Exsys, but the editor is so poor that this isn't practical. The external editor or word processor can't add any extra characters to the file. The customs screens are stored in a separate file. Once a user has defined screens for one expert system, it is **easy to modify the screen file to create screens for a new expert system**.

To **run external programs**; "The external program called can be any program that can be run from the operating system. The program can be written **in any language or** it can be **an application**. When Exsys Professional calls an external program, EXSYSP.EXE(or EDITXP.EXE) remains in memory. Thus, there must be **enough memory** for both Professional and the program called to run."(from page H-2 and H-3 in the manual) To do this, "run external program" is selected from the data acquisition screen. The name of the program and the parameters to be passed back and forth are entered into blanks on this screen.

The executable file for a 6 rule expert system is 1500 bytes for the expert system (text and rules) and 900 bytes for one screen. The compiled expert system is 5000 bytes. But the **compiled code can only be used with the Linkable Object Module interface**. According to the manual page L-5, the ASCII files can be transported to UNIX and VAX/VMS with slight modifications.

The **documentation is terrible**. The documentation was written for the DOS version and a supplemental section for the windows version was provided. The section that gives information on the windows version is not included in the index. In order to create something on the screen one has to look in both the DOS and windows section. Half of what is needed is in the DOS section and half is in the windows section. Actually, several calls to Exsys were needed because of missing information. The DOS version has a tutorial, but Exsys couldn't get this to work with the windows version. So there are no examples of how to write a complete expert system. But the **technical support people from Exsys were very knowledgeable and helpful,** and most of the questions were resolved on the phone.

It took **11 minutes to create a six rule backward chaining expert system**. It took **5 minutes to create a custom screen**. *This was done using an editor that wasn't a part of Exsys*. It was **very easy to create the rules** and to modify them. But once a variable was created it could not be deleted from a screen in Exsys. A user would have to exit out of Exsys, and then edit the ASCII file that contains the expert system.

To create frames, a file has to be created with an external editor for each type of frame. Basically, the user has to type in a table of the values that will be used. It can only handle inheritance from 1 frame. To use a frame in a rule, frame is selected from the data acquisition screen and the blanks on the screen are filled in.

Also, this version had a **bug** in it that would kick the user completely out of Exsys to the windows screen.

Kappa-PC

Kappa-PC by itself cannot support SQL calls to a data base. It does have a built in interface to a program called SequeLink by TechGnosis. This program allows Kappa-PC to be linked to an SQL data base. A user can then use SQL commands in the knowledge base. Kappa-PC does however have a **built in interface to DBASE III, and DBase IV**. The data in these files can be accessed by specifying a cell or by using a key word to find a particular row. Once this row or record is found, either data from one field can be retrieved, or the whole record can be written into a slot in an instance. The same can be done for writing data into a data base. Lines of code have to be written to put values into or receive values out of a spreadsheet or data base. First the data file is opened with this command DBOpenFile(filename). Then commands such as DBCellRead (n,n) have to be written. This set of commands could be written as a function, and then the name of the function used to initiate a read or write from a data base.

Kappa-PC **has** a built in **interface to Lotus 123**. The same procedure as the one described earlier for the databases can be used to assess data from spreadsheets.

Kappa-PC **does not support** differentiation or integration. Also, it does not have hyperbolic functions or a rounding function, but it does handle trig functions and Boolean operations.

Kappa-PC **does have the capability to create graphs**. However, only 6 data points can be displayed at a time. Also, up to 6 lines can be displayed at a time.

Creating GUI type **screens is very easy**. Basically, the user selects the session window icon and activates the layout mode. Then from a list of options the type of image that is to be created is selected. Once the image is picked, a screen comes up that displays the image's attributes, and the user fills in the blanks.

To **run external programs** a user would type Execute (programname, <arg1>, <arg2>, <arg3>);. But **only three arguments** can be passed back and forth. When Kappa-PC calls an external program, Kappa-PC remains in memory. Thus, there must be **enough memory** for both Kappa-PC and the program called to run.

The executable file for a 6 rule expert system is 12000 bytes for the ASCII text file. And 99,000 bytes for the binary file. The binary file is a snapshot of what is in memory, not an executable file. In its present release, Kappa-PC does not have the capability to create an executable program. The image file created cannot easily be ported to another platform, or to another computer running a different operating system.

The **documentation is not bad**. It could use more examples. For the most part, the answer to my questions was in the manual. I still had to call IntelliCorp to find answers to some questions. I had a **hard time getting someone at IntelliCorp** to answer my questions (return the calls).

It took **18 minutes to create a six rule backward chaining expert system**, with one custom screen. It was easy to create the rules, but there was **no way to duplicate rules without leaving Kappa-PC**. So, this involved a lot of typing. Also, Kappa-PC is **case sensitive** so Car is completely different from car. And since Kappa-PC doesn't do any parameter checking when a rule is created it can be difficult to debug the code. If the user puts in a capital C instead of a lower case c, there is no error checking that will help find it. **In order to rerun a knowledge base each of the parameters had to be reset**. Some inherited parameters could be reset as a group, but if they weren't inherited they had to be reset individually. Or a function could be written that would be called if a user needed to reset the knowledge base.

Creating frames and slots was very easy. Basically, to create a frame or slot, one just has to fill in the blanks on a screen that pops up. Also inheritance is very easy it is a matter of just toggling a switch. But an object can only get inheritance from one other object.

Note that Kappa-PC cost consists of a basic rate of $950 plus a yearly support rate of $950.

SMALLTALK V/WIN

Smalltalk by itself cannot support SQL calls to a data base but a product called Smalltalk/SQL by Synergistic Solutions can. It also does not support calls to DBase III, DBase IV but there is a product called QNE by Pioneer which does. It **does not support Lotus**

123. Although **it has a DDL** (Dynamic Data Link) ability so it can communicate with many programs once the interface code is written.

Smalltalk **does not support** differentiation or integration. Also, it doesn't have hyperbolic functions. But it does handle trig functions and Boolean operations. There are some public domain math classes that may do matrix operations, differentiation, and integration.

Smalltalk **does have the capability to draw lines** from point to point, rectangles, and bit maps. But any sophisticated graphics would have to be done by an external program.

Creating GUI screens is not very hard. Basically, the user selects the class of windows that they want to create. Then the user makes a specific instance of that class. In this instance the user puts the information about the window that makes it unique.

To run external programs a user has to create a DDL. And then call that DDL from Smalltalk. An alternative would be to open a shell to DOS and execute the external program.

An **expert system was not created** as part of out testing because of the complexity of the product. There wasn't enough time to adequately learn the product. There are however expert system shells that were written in Smalltalk. Some are public domain and offer forward and backward chaining, pattern matching and other functions. Once the shell is obtained, creating the expert system should be easy.

The documentation is not very complete. There are third party manuals which explain this language better. Basically, I couldn't figure out how to create a program from scratch after studying the manual and going through the tutorial. I spent about 26 hours doing this.

Creating classes and instances was very easy. Basically, this is how a Smalltalk program is created. A class is selected and then methods and instances are created for this class. There are rule based expert system class libraries which would help to develop expert systems. These classes allow a user to use both objects and rules together. There is a significant built in list of class libraries but when you deploy a Smalltalk application you have to take most of the environment with it. There is no executable file that stands by itself. **Its development environment is geared towards rapid prototyping.**

PDC PROLOG

PDC Prolog by itself **cannot easily support SQL** calls to a data base. It does have a built in interface to a program called Qelib which sells for $300-400. This program allows PDC Prolog to be linked to an SQL data base. A user can then use SQL commands in the knowledge base. PDC Prolog **does** however **have a built in interface to DBase III, DBase IV, and Lotus 123** through the toolkit.

PDC Prolog **does not support** differentiation or integration. Also, it doesn't have hyperbolic functions. But it does handle trig functions and Boolean operations.

Under DOS, Prolog needs the Toolbox to generate graphs, dials, etc., Under Windows, the Windows API is needed to generate the graphs similarly to the Toolbox. Also, Borland's toolbox can be used as an alternative environment.

Creating GUI type screens is done by using an external program to create the screens. Then back in the Prolog program lines of code would have to be written to handle the message passing between prolog and windows. Whoever creates a program with windows would have to know how to use both Prolog and windows. **Prolog is the most difficult environment out the 4 environments that were looked at to create windows in.**

To run external programs a user would type System(Dos command line). Basically, any command line can be put within the parenthesis.

The executable file for a 6 rule expert system is 1454 bytes for the ASCII text file. And 38700 bytes for the compiled file.

The documentation is very good. Also, there are many third party books and Prolog tools that can help a user find a solution to a particular problem.

It took Mike 15 minutes to created a six rule backward chaining expert system, with no custom screens. Basically, he wrote a program with a data base, predicates, clauses, and goal sections. This program was 61 lines long. This involved a lot of typing. But there are some programs out there that might allow one to reduce the amount of typing. The debugger is good. Basically, writing a prolog program is the same as writing any other computer program except is has a different syntax which makes writing expert systems easier. Prolog is a good language to write expert system shells in, but it is more complicated to write a specific expert system in prolog than in an expert system shell.

Inheritance is basically **done with programming style it is not a built in** feature of the program. PDC Prolog has a DDL ability so it should be able to link up with many programs. Although an interface program would have to be written.

CONCLUSIONS

Several programming environments and programming languages were evaluated as possible candidates for the development of a rapid prototyping environment. Four products appeared to be better suited for our needs: Exsys, Kappa PC, PDC Prolog and Smalltalk. The first two are expert systems shells that make the development of a RPI very fast and easy. Of the two, Kappa PC is the most versatile and powerful. This type of environment does however present problems when one is faced with distributing executable programs. The environment is suited for use at few sites as a test environment, and executable codes are not easily or readily produced. However, the time needed to develop an application is the shortest if starting from scratch. If similar applications were previously developed, then PDC Prolog or Smalltalk could be quickly modified for a variant problem. Both Exsys and Kappa PC had some shortcomings in drawing graphics, executing SQL transactions and getting data from spreadsheets (accessing only one cell at the time, or one record / row at the time respectively). These products are newcomers to the Window environment and should improve significantly with new releases.

The programming languages / environments of PDC Prolog and Smalltalk display some of the same strengths and limitations. Since they are languages, code has to be written for a particular application, and the development time from scratch is significantly larger than for Kappa PC. However, there is more flexibility and customization capabilities. PDC Prolog allows one to create an executable program that can be copied to other sites and run without the Prolog compiler. Smalltalk /V does not allow the creation of executable code, but Park Place Smalltalk does (however not on DOS machines). Smalltalk has an advantage over Prolog since inheritance and objects are inherent to the system. The Smalltalk language is however difficult, and the learning curve could be flat for a longer period than for Prolog. Both environments have user services in form of bulletin boards, technical phone service. PDC Prolog has the advantage of being in town, Smalltalk is more widely used in the US.

The overall conclusion is the lack of an overall winner. There is no perfect environment, shell or language. Considering the time and effort already spent by Lockheed in PDC Prolog, and considering the fact that of the investigated platforms, only Prolog allows generation and free distribution of compiled code, we would recommend continuing with Prolog at least for the foreseeable future. We should mention that, since the selected environment is a programming language, we should have compared it to other programming languages beside Smalltalk. In particular, a language like C++, in spite of its intricacies, could probably provide a powerful and very flexible environment, but at a high initial cost in time and effort.