

RADIO-FLIER ELECTRONICS

PRODUCT REQUIREMENTS DEFINITION COOKBOOK

Synopsis: This describes the content of a meaningful product requirements document. It is meant to serve as a “cookbook” of sorts, for producing requirements documents for real products. This document is based on many years of experience interfacing between Engineering and Marketing.

APPROVALS

<u>Signatory Title</u>	<u>Approving Signature</u>	<u>Signatory</u>	<u>Date</u>
General Manager		G. Barecki	
Director of Engineering		T. Farrand	
Director of Sales		S. Wittinger	
Director of Marketing		J. Michael	
Director of Operations		T. Andersen	
Acceptance Test Manager		G. Buchmeister	

DOCUMENT CONTROL

Revision Number	1.0
Revision Date	December 20, 2001
Principal Author	Thomas Farrand
Assisting Author	-none-
Document Format	Microsoft Word® 2000
Document Status	DRAFT
Name of Document	PRD Cookbook
Type of Document	PERMANENT
Storage Location	Author's PC, 32-bit Drive C:, \My Documents\PRD Cookbook
Associated Project Identity	-none-

This document contains information proprietary to Radio-Flier Electronics, and may not be reproduced, disclosed, or used in whole or in part without the express written consent of RFE.

Table of Contents

1. FROM THE AUTHOR.....	4
2. SYNOPSIS.....	6
3. DOCUMENT HISTORY & DISTRIBUTION.....	6
4. INTRODUCTION.....	7
4.1 Glossary of Terms.....	7
4.2 Background.....	7
4.3 Statement of Benefits.....	8
4.4 Charging Method.....	8
4.5 Usable Product Life.....	8
4.6 Justification.....	9
5. DESCRIPTION OF REQUIREMENTS.....	9
5.1 User Interface.....	10
5.1.1 Context.....	11
5.1.2 Icons and the Tool Bar.....	11
5.2 System Description.....	11
5.2.1 Data Interchange.....	12
5.2.2 System Security.....	12
5.2.3 Customer Supplied Functionality.....	12
5.2.4 Related System Functionality.....	13
5.3 Administrator Functions.....	13
5.3.1 Administrative Security.....	13
5.3.2 Administrative System Access.....	13
5.4 Compatibility with Existing Products.....	13
5.5 Sizing/Scaling Requirements.....	14
5.6 Performance Requirements.....	14
5.7 Graceful Degradation Considerations.....	14
5.8 Physical Characteristics.....	15
5.9 Environmental Requirements.....	15
5.10 Technical Policy Requirements / Agency Compliance.....	15
5.11 Quality Requirements.....	16
5.12 Product Support Considerations.....	16
5.13 Long-term Product Support/Maintenance.....	16

5.14 Milestone Targets 17

6. RISKS..... 17

6.1 Dependencies 17

6.2 Client Objections..... 18

6.3 Export Requirements..... 18

6.4 Resource Availability..... 18

7. OPEN ISSUES 18

7.1 Technical Issues to be Resolved..... 19

7.2 Additional Information to be Provided 19

8. EXECUTIVE SUMMARY 19

8.1 Development Costs 19

8.2 Long-term Maintenance Costs..... 19

8.3 Availability Target..... 20

8.4 Anticipated Revenue 20

8.5 Benefits 20

8.6 Risks 20

8.7 Recommendations..... 20

1. FROM THE AUTHOR

A Product Requirements Definition should be just that: an orderly list of requirements that define what shall be produced. A PRD is authored as a collaborative effort between Marketing and Engineering. Engineering cannot expect Marketing to understand every nuance of the engineering process and must therefore contribute some consulting effort to the production of a PRD. On the flip side, Marketing cannot expect Engineering to write a PRD for them. Defining the *what* that is wanted must be done principally by Marketing, with a bit of help from Engineering. It is the job of Marketing to assess market needs and then specify products and services to fill those needs. It is the job of Engineering to implement those specifications to conform to corporate standards.

According to the doctrines of many U.S. corporations, a PRD is the document that kicks-off a development undertaking. A PRD should make the business case for doing "it" in the first place. And, the PRD must identify in detail exactly what "it" is. Engineering can and often does develop products without any sort of PRD at all. **Q:** Why then is a PRD needed? **A:** Time-to-market:

The steps necessary to progress an idea from concept to a reproducible product span many disciplines. Engineers are problem solvers by nature, and given enough time, they will cover all the bases necessary to devise and launch a product by themselves. The keyword here is *time*. An engineer can only progress through steps in a sequential fashion. Step one must be completed before progressing to subsequent steps. That is the methodology that must be used when no PRD is used to guide product development. That method works, but it represents a poor use of the workforce. With current time-to-market demands, a better solution must be had. When properly implemented, the PRD is that solution.

If engineers are spending time to assess market need, analyzing competition, developing marketing and selling strategies, comparing features, and writing what-to-do documents, then those engineers aren't doing their principal function: engineering. A much more efficient process would have the Marketing group define what is wanted and then work with Engineering to polish the rough edges. This is an efficient approach because Engineering is finishing off the previous development while Marketing has begun work on the next development specification. Remember, if your engineers are working hard to define what they'll do next... they are not doing the engineering job that only they can do. The result is poor designs, bug-laden software, and ill-received products.

Producing a proper PRD is admittedly, a time-consuming effort. But this is time well spent in the long run. Expensive engineering talent is not squandered on minutiae. In the end, you will have a product that accurately reflects what you've asked for. And, if Marketing has done their job responsibly, you will end up with a killer product for the marketplace.

The numbered sections of this document are meant to serve as a guide (or cookbook) in writing a meaningful and relevant PRD. Ultimately, all the questions raised herein will have to be answered. If the questions are not addressed here, when and where do you suppose they will be answered? Without exception, unanswered questions in the specification portion of the development process become customer support questions later on. It is much easier and cheaper to resolve issues before launching a product. Fixing a problem once a product is launched costs between 50 and 10,000 times as much as it would cost to fix when in the specification phase. Think about that.

Does a PRD really have to specify every tiny, insignificant requirement? No. You need only specify those requirements you actually wanted implemented! Look, there are simply two choices to specifying a product: 1) You can verbally describe what you want in gross terms and then let Engineering take off on their own tangent, or 2) You can specify what you want up-front. The former, puts the Company in a very awkward position... you'd better like what Engineering did because you're stuck with it. If you didn't specify exactly what you wanted, you cannot blame the Engineer because "He got it all wrong!" The latter is a much better solution because it makes product development predictable, precise, and with minimal surprises.

Consider also, the use of consultants. We employ consultants on occasion to help even out the workload. If we rely on oral tradition to product specification, how do we communicate our needs to consultants? Shall we pay them \$75.00 - \$150.00 per hour to sit in meetings, while someone stands up and gestures what they want? This situation becomes unworkable if the consultants are geographically distant from us.

A PRD should not contain hype, fluff, puffing, or other advertising gimmicks. A PRD must give "just the facts". Feel free to use your creativity when writing ad copy - just keep the ad copy mindset on the back burner when authoring a PRD. Authoring a PRD is likely to be a collaborative effort between Marketing and Engineering. Members of both parts of the organization must agree on content, as this document is a "contract" of sorts. If some requirement is difficult for Marketing to describe, they should feel free to ask Engineering for help or guidance.

Before I step from my soapbox, there is one last point I'd like to make. Do a PRD properly, or don't do it at all. Writing some vague document that says nothing of substance is a colossal waste of time for the author and an even bigger waste of Engineering time if tasked with reviewing and using it. If the boilerplate of a PRD constitutes a significant portion of the content by volume, throw it away. You have likely not said anything of substance or value.

Herewith, is my idea of what a PRD should contain....

2. SYNOPSIS

This text area is normally used for a brief description of the development. If the development spans more than one phase, the elements pertinent to this phase should be stated. The purpose of a synopsis is to aid someone who is scanning for a particular specification and may not be certain of the title or exact content. Keep it succinct. One to three paragraphs are appropriate. The synopsis is located as the first item. (Don't bury it on page 47 paragraph six.)

3. DOCUMENT HISTORY & DISTRIBUTION

DOCUMENT HISTORY

Version	Description of Version	Date
0.1	Initial stab at requirements - not distributed	06/22/01
0.2	Revision to introduction - not distributed	06/30/01
0.8	Added glossary of terms, sneak preview to gen'l manager	07/12/01
0.9	Draft version for Marketing eyes only review	07/13/01
0.92	Incorporate Mkt'ing changes from review meeting	07/20/01
1.0	First issue to full distribution list	07/27/01
1.1	Incorporation of changes from 1 st full review - not distributed	08/03/01
1.2	Second issue to full distribution list	08/10/01
1.3	More changes resulting from review - add financial data	08/17/01
1.4	Final issue - released to Development to implement	08/27/01

DOCUMENT DISTRIBUTION

Recipient	Title	Location
Person 1	VP & General Manager	Chicago
Person 2	Director of Sales	Addison
Person 3	Marketing Manager	Addison
Person 4	Senior Development Engr.	Addison
Person 5	(Consultant)	Schaumburg

4. INTRODUCTION

State whatever might be appropriate to initiate a person unfamiliar with the content of this document. If this document serves as a replacement for a multiplicity of other documents, this might be a good location to identify what documents are being replaced (and why).

The introduction should establish the "where are you coming from" tone. Not everyone who reads your document will understand how this document fits into the grand scheme. This is where you should identify the context of this document.

If more than one product line is involved, briefly describe how this product-to-be fits in with the existing products. Or, describe why this product stands alone and unrelated to other products.

4.1 Glossary of Terms

Optional: Define the terms used in this document that might otherwise cause ambiguous requirements (especially AKA's). If this specification crosses multiple development groups, a glossary is mandatory. The vernacular of one development group isn't necessarily common to all.

4.2 Background

Describe what problem this development will address or what utility will be provided - in essence, why are we doing this? Describe what will be produced: is it a new software product, a new hardware product, an enhancement to an existing product, or a combination? Describe the tangible deliverables of this specification.

If an historical perspective is needed, this is where it goes. Know your audience. If the audience is diverse or has never worked together, a lot of background information could be vitally important.

4.3 Statement of Benefits

Explain how this development ties-in with the overall Marketing strategy for this family of products or services. List the benefits, as observed by the client - state why the client will want to purchase, lease, or upgrade to it. This allows the Marketing person the opportunity to establish the context of this product in relation to our other products. Engineering will gain a better understanding of how this new product fits in with products he is already familiar.

4.4 Charging Method

Describe how the buyer of this product or service will be charged. Is it an outright sale? Will we lease or rent on a per session basis? Will we lease, rent, or outright sell a group of sessions of this product? Do we sell a base platform and then sell sessions in a block, singly, or in pairs?

Will this product be offered as a promotion? Will it be offered "free" for users who upgrade before a certain time limit? All of these questions inform Engineering as to what sort of system granularity is envisioned. This section also helps Marketing clarify its thinking on ROI details.

4.5 Usable Product Life

When will we start offering this product? When will we cease to offer this product? (If dates cannot be stated, the circumstances governing withdrawal should be specified.) When will we stop supporting this product? (If a date cannot be stated, the circumstances governing support cessation should be specified.)

We are supposed to know up-front, when a product begins and ends its existence. That is merely good planning. Engineering must assess what support level might be needed as a consequence of having such issues addressed.

4.6 Justification

List the specifics as to why it makes financial sense to undertake this development. Explain the anticipated revenue arising from new sales or leases, and upgrades to existing sites. Explain where your revenue data comes from and your confidence factor in those data. Other justification might fall into one of these categories:

- strategic importance: must have because it is a prerequisite for a larger sale
- product obsolescence: xxxx is no longer available and we must have it in order to...
- appeasement: we ticked-off a client and this is an incentive to...
- good will: promote the Corporate image

This information communicates the relative importance of this product to Engineering. By "seeing the numbers", Engineering can early on debunk a specification that has no hope of ever turning a profit due to enormous development costs or effort.

5. DESCRIPTION OF REQUIREMENTS

State the overall objectives in general terms - expand into the specifics in subsequent sections. When defining "features", it is essential they be ranked as to importance. Use three categories for ranking product features:

- Absolute Requirement: must have or else... mission critical
- Real Important: really should have it, but negotiable
- Would Be Nice: beneficial, but not essential

Doubling the cost of a development to include an unimportant feature is senseless. By knowing in advance what features are essential, Engineering can minimize the likelihood of generating inflated development estimates.

This section of the document should be the 'meatiest' of all and will probably span many pages of text. Remember, if it is unsaid here - it is unimplemented later on. Writing this section is likely to be a major undertaking. Most of what Engineering needs to know will be here. Also, most glossary references will come in this section. That is why the glossary must appear before this section.

5.1 User Interface

List the stimuli available to the user and state what will result, case-by-case. Give as much information as to screen layout as possible. The "look and feel" of the product will be defined by the user interface. It is therefore critical that you impart as much information as possible regarding screen layout and content. If the product does not have a "screen" per se, it is important that all aspects of how the user will interact with the system be stated.

State what functionality might be inferred but not actually supplied. (Summarize this also in the RISKS section.) For example, a mouse is usually found on Windows-based products. If there is no support planned for a mouse, you should state this very clearly. Any "expected", implied, or inferred functionality that will not be provided should be specifically listed. If "missing" functionality will be provided at a later date (phased development), state that here as well.

Think about the user interface in a logical sense. When using the system, what is the user likely to do first, second and third? If the user interface is a menued tree, can the user jump entire branches without traversing up the branch he/she is currently on? How many levels deep are the menus? A diagram of the user interface might be appropriate. Even if the specific details have yet to be worked out, describe as much as is practical. Engineering will be happy to work with you in sorting this out.

State the specific functionality that is to be provided. Dealing with error conditions is very much a part of the look and feel of a product. When the user attempts to do things that are erroneous, state how these errors will be dealt with. Again, this should be done on an event-by-event basis. Some common examples of error conditions that should be dealt with are:

- ⇒ numeric input required, alpha entered instead
- ⇒ alpha required, numeric data entered instead
- ⇒ upper/lower case required in certain fields
- ⇒ system capacity exceeded in some way (e.g. insufficient disk space)
- ⇒ corrupted data files, configuration information, or the like
- ⇒ external device not present, not installed, or not responding (e.g. network is down)
- ⇒ field "X" must be complete before filling out field "Y"
- ⇒ invalid names, passwords or PIN numbers

Also, state implied performance limitations and caveats such as: all "important" functions must be available by a single keystroke or single mouse click. (List the "important" functions in that case.)

The following two subsections are meant as examples of introductory verbiage for the user interface on a fictitious product. These are examples only!

5.1.1 Context

The user interface should be graphically represented and pleasing to the eye. For example, our customers will already be familiar with MicroSquish Wurd . The arrangement of the work area and the tool bar is particularly nice and we should maintain a similar, albeit different, look and feel. Color choices should be variable according to operating system limitations. The default color selection should be made the same as the default colors for standard Microsquish Windoughs products.

5.1.2 Icons and the Tool Bar

Experience has shown that icons usually bear no resemblance to the function performed and their inclusion should be minimized. If found necessary, have a pop-up window (balloon help / tool tips) that explains the icon, appear after 1 or 2 seconds of the cursor resting on the icon. In general, the tool bar should be located at the bottom of the screen, rather than the top. People with bifocal glasses are handicapped with toolbars located at the top of the screen. {...and so on...}

5.2 System Description

Fully describe the system in which this product will be used. Describe the context in which this product will operate. A drawing would be very helpful. When including a drawing, please don't mix logical entities with physical devices. For example, if you are explaining how the user interface will work in a menu tree, don't diagram logical (abstract) entities on the same page as physical (tangible) devices. It is also unnecessary to assign computational tasks to specific compute devices. Leave that up to your Engineering department - they are experts at doing this sort of thing.

5.2.1 Data Interchange

Is data interchange a closed or open-loop system? Describe how data will pass from device to device. (This will be done at a very high level in this sort of document. Engineering can sort out the specifics.) Describe what sort of communications pathway is envisioned. For example, show which pathways are networks and which are point-to-point communications. If there is a physical requirement (such as single mode fiber) explain that requirement and where it fits in.

What design elements must there be to warrant error-free data interchange? If there is no specific requirement for data delivery guarantees, say so. Guaranteed deliveries tend to be expensive to implement and complicate the design. However, they are often unavoidable.

5.2.2 System Security

Describe the security measures that are to be present and how they are to be implemented. How large are password fields? Will there be a non-redundant mechanism for passwords? Will we use PIN numbers? How many digits in a PIN number? Will the system generate PIN numbers automatically or will the user be allowed to choose? If the user is allowed to choose a PIN number, how will duplicates be dealt with? And so on.

Describe how security could be breached and what might the consequences be. Are there any liability issues we must confront if system security is breached? (If there are, detail these liabilities in the RISKS section of this document.)

5.2.3 Customer Supplied Functionality

What provisions are there for connection of customer supplied elements? Will the customer be running "their" software concurrently on our compute platforms? What sort of applications will be running and what assurances have we given about the reliability of such an arrangement? This can be a rich source of risk. List these risks in the section: RISKS.

5.2.4 Related System Functionality

Is any of the product functionality provided by other parts of the system, not now in place? Describe how this functionality fits in with the product offering depicted in this document. At the very least, refer the reader of this document to other related documents by title, author, and date.

5.3 Administrator Functions

This section deals with administrative requirements of the system. Be certain to document any interface requirements for the system administrator, in the "User Interface Requirements" section.

5.3.1 Administrative Security

Who is allowed to initiate administrative functions? How will such access be granted and from what sort of system connection, will this access be granted?

5.3.2 Administrative System Access

Will there be remote, dial-in access to administrative functions? On what sort of compute platform will the user gain access to administrative functions? Oftentimes, this question dredges up a whole new set of requirements for a separate product. Be careful to not get too carried away with this area!

5.4 Compatibility with Existing Products

With which versions of peripheral products will this be compatible? What known incompatibilities are anticipated (summarize in "RISKS" section)? This ties in with requirements for customer-supplied equipment, too.

5.5 Sizing/Scaling Requirements

If there are limitations available such as never needing to support more than ten users, there may be a significant savings in development costs. Similarly, if the system is to be expandable to a thousand positions, it is essential to consider that from the outset.

System scalability can have dramatic effect on the end-user cost of the system. Please bear this in mind when deciding on system sizing. Suppose a system were specified that provided from 1 to 10,000 simultaneous users. While such a system could be engineered, it is unlikely in the extreme that such a system would be economically viable, with only a single user.

5.6 Performance Requirements

How fast for how long and with how many concurrent requests, users, or sessions? State the required response times, updates/sec, transactions/sec, etc. What are the worst-case performance requirements? What are "average" performance requirements/expectations? Faster costs more money - keep your requirements realistically in-tune with the actual need.

5.7 Graceful Degradation Considerations

At some point the capacity of the product will be exceeded. This section deals with the performance requirements during "overload" conditions.

- ↳ define absolute limits before "overload" is said to occur
- ↳ state what level of performance is required, once an overload condition has been reached
- ↳ what will happen if multiple parameters are simultaneously exceeded?
- ↳ will recovery (after an overload) be automatic or require manual intervention?

Again, think back to "error conditions" you described in the User Interface section. What should be done if, for example, the user runs out of disk space because the error log was filled to capacity?

5.8 Physical Characteristics

(This generally applies to those developments having a hardware element.) State the maximum outside dimensions and product weight limitations, as applicable. One must state where the product is to be installed. List which solids, liquids, and gases for which the product must be impervious. State what abuse the product might be subjected to, and to what degree of survivability it must maintain. State the required MTBF and by which method the MTBF will be calculated. State the required MTTR (if applicable) and how it will be calculated. State how MTTR will be validated prior to product launch.

5.9 Environmental Requirements

In what sort of environment will this product be used? (Generally, this is a hardware related section but could pertain to software.) Some environmental issues are:

- What distance is involved between interconnected physical units?
- What physical medium of data interchange will be required?
- What method of signaling will be required?
- What bandwidth is required?
- Will (can) this ever be used in a "home" environment? When? How?
- What power requirements are there? Is battery backup a necessity? Power conditioning?
- State the temperature & humidity requirements if different from Corporate Policies
- State ESD survivability requirement: amplitude, duration, events per unit of time, etc.

5.10 Technical Policy Requirements / Agency Compliance

State which Corporate Technical Policies govern the various aspects of this product or are relevant to the development. List, specifically, the Technical Policy affecting the specific requirements.

List all regulatory agencies for which this product must conform such as VDE, CSA, UL, FCC, etc.

5.11 Quality Requirements

This section deals mainly with system reliability and repeatability. Some of the quality attributes of a product we should be concerned with are:

- * If calculations are involved, specify required accuracy
 - define the standard by which the "accuracy" will be measured
- * Product availability ("up" time): be specific as to expected system availability
 - define how this will be tested prior to product launch
- * List other quality requirements as applicable and the method of validation

5.12 Product Support Considerations

This section deals with how the product is to be supported and what maintenance facilities are to be provided. The following four categories should be addressed in appropriate detail:

- ① user maintenance/configuration capabilities
- ② system administrator maintenance/configuration capabilities
- ③ field service group maintenance/configuration capabilities
- ④ development group support requirements

5.13 Long-term Product Support/Maintenance

Long-term product maintenance must be considered from the outset. What financial provision has been made to support this product? Will there be recurrent maintenance fees for continued product support? Why not?

If this product is a custom development, what maintenance fees are anticipated? Were these fees factored-in when the custom development quote was given? How will maintenance be billed? What is the anticipated support requirement (in hours) per annum?

List any other support issues and requirements here.

5.14 Milestone Targets

State which key elements are to be delivered, and at what point in time:

- Detailed Design Specification
- Acceptance Test Specification
- Integration Testing Completion
- Alpha Test Completion
- Beta Test Completion
- General Product Release
- Whatever else is deemed important

Make certain that all personnel required for this development are available for the targets specified. If availability is in question, list such in the **RISKS** section.

6. RISKS

All of the aforementioned areas of risk should be summarized here. Go into as much detail as needed to make the reader understand the risk elements. Some risk elements can be minimized or controlled by careful requirement "pruning". This might be a good place to underscore why certain requirements are worded as they are.

6.1 Dependencies

List any dependencies here such as:

- works only with hardware revision "6" or later of the XYZ product
- requires firmware version greater than 3.0.1
- must use release 3 of the kernel and version 4.XX or better X-Windows

6.2 Client Objections

List any anticipated client objections and how they might be dealt with in relation to the requirements. Some objection categories might be:

- operating costs
- maintainability
- upgradeability: can't do it, must do it, is costly, forklift upgrade
- "missing" features: phased release
- not reusable when "xxx" product is released
- "better and less costly than product XXX that was purchased only 6 months ago"

6.3 Export Requirements

List any export considerations here. Software as well as hardware is controlled by the U.S. Department of Commerce. If all portions of the product are for domestic consumption and sale, this section can be omitted or marked as not applicable. Some export issues are:

- What is the target geographic distribution for this product?
- Is I.V.L. acceptable? GDEST?
- What EEC issues are there?
- List any other export issues here.

6.4 Resource Availability

List all internal and external resources necessary to provide the development in the desired timeframe and within projected costs. Some resources may not be under your control; as such, they present several risk elements.

7. OPEN ISSUES

There should be no open issues when this document is approved. Anything unsaid in this document, will become a costly mistake later on.

7.1 Technical Issues to be Resolved

List the issues, who is actioned to resolve them, and the promised date for resolution.

7.2 Additional Information to be Provided

List what specific information is to be provided, by whom, and the date when the information will be available.

8. EXECUTIVE SUMMARY

Summarize the key points necessary to make a go/no-go decision. Keep it brief.

8.1 Development Costs

This can only be filled-in once Engineering has had time to review the technical content. Costs should probably be listed in man-hours for labor and materials costs in dollars. At the end, hours can be converted to dollars at the prevailing labor rate.

8.2 Long-term Maintenance Costs

Provide a projection of the maintenance costs associated with this development. Consider the cost in man-hours of indirect labor as well as the cost of maintaining the software build platforms. One must also consider the cost of compilers and assemblers going “stale”. It is a common practice by software tool manufacturers to obsolete software tools quite often. The “new” software tool might be more costly or contain bugs that preclude compilation of the “old” application code. It is possible that a major re-write might be called for, just to maintain existing functionality.

8.3 Availability Target

The product availability dates should be listed in this section. If the development includes a phased rollout of the product, all availability dates should be listed and so identified.

8.4 Anticipated Revenue

This section contains the all-important revenue estimates for the life of this product. The expression of anticipated revenue should be in the same manner as stated in the section “**Charging Method**”.

8.5 Benefits

8.6 Risks

List all of the identified risk elements and potential circumstances which might lead to a risk situation.

8.7 Recommendations

Short and sweet, please.

End of Document