

# *Project Management*

## NOTES

---

<b>AUTHOR</b>	<b>RANDALL MAAS</b>
<b>OVERVIEW</b>	This is a fascicle to capture my notes on project management.

**RANDALL MAAS** has spent decades in Washington and Minnesota. He consults in embedded systems development, especially medical devices. Before that he did a lot of other things... like everyone else in the software industry. He is also interested in geophysical models, formal semantics, model theory and compilers.

You can contact him at [randym@randym.name](mailto:randym@randym.name).

LinkedIn: <http://www.linkedin.com/pub/randall-maas/9/838/8b1>

<b>PREFACE .....</b>	<b>1</b>
<b>1. ORGANIZATION OF THIS DOCUMENT.....</b>	<b>1</b>
<b><u>CHECKLISTS.....</u></b>	<b><u>2</u></b>
2. CUSTOMER FOCUS .....	2
3. COMMUNICATION .....	2
4. PROJECT.....	3
5. DIAGRAMS.....	3
6. ELECTRONICS DEVELOPMENT CHECKLISTS .....	4
7. SOFTWARE DEVELOPMENT CHECKLISTS .....	4
<b><u>APPENDICES .....</u></b>	<b><u>7</u></b>
<b><u>PROJECT PROPOSAL OUTLINE.....</u></b>	<b><u>9</u></b>
<b><u>PROJECT STATE RUBRIC.....</u></b>	<b><u>10</u></b>
8. PROJECT READINESS .....	10
9. TEAM READINESS .....	13
<b><u>TEAM WORK RUBRIC .....</u></b>	<b><u>15</u></b>
10. TEAM STRUCTURE .....	15
<b><u>PRINCIPAL SYSTEMS ENGINEER .....</u></b>	<b><u>21</u></b>
11. ESSENTIAL FUNCTIONS .....	21
12. SKILLS & EXPERIENCE.....	22
13. ADVOCATE QUALITY .....	23
14. OTHER GENERAL EXPECTATIONS.....	23
<b><u>PRINCIPAL SOFTWARE ENGINEER .....</u></b>	<b><u>25</u></b>
15. ESSENTIAL FUNCTIONS .....	25
16. SKILLS & EXPERIENCE REQUIRED .....	26
17. OTHER GENERAL EXPECTATIONS.....	28

*[This page is intentionally left blank for purposes of double-sided printing]*

# Preface

---

This fascicle is my notebook on project management.

## 1. ORGANIZATION OF THIS DOCUMENT

- CHAPTER 1: PREFACE. This chapter, wherein I describe the other chapters.
- CHAPTER 2: STARTING PROJECT. What to look for and do when starting on a project.
- CHAPTER 3: PROJECT MANAGEMENT. Function and feature identification
- CHAPTER 4: PROJECT & DEVELOPMENT PLAN. Includes work breakdown structure
- CHAPTER 5: PROJECT SCHEDULING. Including schedule estimation.
- CHAPTER 6: PROJECT ROLES. Provides a description of the different roles in a project.
- CHAPTER 7: PROJECT COSTS.
- CHAPTER 8: CHECKLISTS. This chapter provides checklists for projects.
- CHAPTER 9: WORK ASSIGNMENTS.
- CHAPTER 10: PROFESSIONAL DEVELOPMENT. Teams and individuals

APPENDICES: The appendices provides extra material

- APPENDIX A: ABBREVIATIONS, ACRONYMS, & GLOSSARY. This appendix provides a gloss of terms, abbreviations, and acronyms.
- APPENDIX B: PROJECT PROPOSAL OUTLINE. This appendix provides a basic project proposal outline
- APPENDIX C: PROJECT RUBRIC. This appendix provides rubrics relevant in assessing the project and plan.
- APPENDIX D: TEAM RUBRIC. This appendix provides rubrics relevant in assessing the team cohesion and structure.
- APPENDIX E: PRINCIPAL SYSTEMS ENGINEER JOB DESCRIPTION.
- APPENDIX F: PRINCIPAL SOFTWARE ENGINEER JOB DESCRIPTION.

REFERENCES AND RESOURCES. This provides further reading and referenced documents.

## CHAPTER 8

# Checklists

---

This chapter summarizes the checklists

- Customer focus
- Kickoff checklist
- Reviews checklist
- EE design decision list
- Software development organization checklist
- Software revision control setup checklist
- Software Release checklist

## 2. CUSTOMER FOCUS

Customer focus

- Is each element/requirement traceable back to the customer/operator?
- How is each element/requirement traceable back to the customer/operator?
- Can each functional team/role explain its parts, and the majority of others?
- Is there a walkthrough explicitly tracing back to the customer?

## 3. COMMUNICATION

- Does your team understand your company's strategy?
- Does your team understand engineering's roadmap?
- Does your team understand why the roadmap meets the goals of the strategy?
- Do you have regular communication meetings and e-mail with your team?
- Are people on your team willing to tell you bad news?
- Do you hear information about your team from your team before you hear it from others?
- Do members of your team communicate with each other and the rest of the company in a respectful manner?
- Do you provide information to your boss before he or she has to ask for it?
- Do other people in the company know what your team is doing and accomplishing?
- Do you communicate in a positive fashion?

## 4. PROJECT

### 4.1. EXECUTION CHECKLIST

- Do you have your customer's requirements?
- Do you have an approved budget?
- Do you have an approved roadmap?
- Do you have an approved schedule?
- Are you delivering the product on time?
- Do you hire developers in a timely fashion?
- Is your team capable of dealing with change?
- Are you capable of keeping your team focused and resisting change?
- Do your customers encounter a lot of quality issues with released products?
- Do you and your team measure how well you do your work on a regular basis to find ways to improve?

### 4.2. KICKOFF CHECKLIST

- Create software revision control
- Product cost targets
- Hardware cost model / targets
- EE cost model/targets

### 4.3. REVIEWS CHECKLIST

- project plan review
- design review
- implementation review
- EE review performed
- Software (code) review performed

## 5. DIAGRAMS

- 3/4 view of product
- 3/4 view of removable elements
- 3/4 view of major subassemblies
- Exploded view of subassemblies
- Blue prints
- Electrical schematics
- Harness wiring diagram

## 6. ELECTRONICS DEVELOPMENT CHECKLISTS

### 6.1. EE DESIGN DECISION CHECKLIST

Is the subsystem / area ready to have a committed design? Is its design technically ready?

Is the hardware design document sufficiently complete?

Sufficiently detailed to allow functional safety to be assessed.

Is there missing detail? Is it being hidden?

### 6.2. SCHEMATIC REVIEW CHECKLIST

Style check

- Are the names clear?
- Do they use a good group / naming convention (e.g. related signals should be grouped by name)
- Is the name format consistent?
- Names only employ alphanumeric characteristics?
- Are the directions of the signal arrows following the logical direction of signal flow?
- The roles for each connector signal are defined?
- There are bypass and decoupling capacitors for power-supply inputs?
- The capacitor sizing and topology matches the datasheet(s) and application notes?

Test points

- Are there enough test points?
- Dog-bones to allow development & test isolation?

## 7. SOFTWARE DEVELOPMENT CHECKLISTS

### 7.1. SOFTWARE DEVELOPMENT ORGANIZATION

Check that the software development organization has:

- Software plan(s)
- Coding standards
- Version control
- Static analysis tools/testing
- C-function unit testing & software integration testing
- Peer review
- Test strategy
- Function test strategy
- Tool qualification
- Field trials

## 7.2. SOFTWARE REVISION CONTROL SETUP

(At start) Rep/change-control checklist

- Select identifier for the project:
- Create SVN repository with that identifier
- Create trouble ticket data base with that identifier
- Perform initial check-in
- Create build instructions
- Create SDP labeling plan
- Release identifiers plan is documented
- Branching plan is documented
- Release note template

## 7.3. REVIEW CHECKLIST

- Initialization
  - Are the clocks set correctly? i.e., no over-clocking at the voltage and/or temperature
  - Initialization order
- Check prioritization
  - Rate Monotonic Analysis (RMA) and dead-line analysis performed
  - Task/thread prioritization based on the analysis
  - Mutex prioritization based on the analysis
  - Interrupt prioritization based on the analysis
  - DMA channel assignments match hardware function constraints
  - DMA channel prioritization based on the analysis
  - CAN message priorities based on the analysis
  - ADC priorities based on the analysis
  - BLE notification/indication priorities

Has the memory been partitioned in a manner suitable for Class B? ie, Does isolate and check the regions?

## 7.4. RELEASE CHECKLIST

- Bug tracking tickets have been updated and dispositioned
- Built the project object-file/executable/etc
- Placed the object file in the release folder, with name to match standards
- Release notes updated
- Release notes distributed to Test (who may distribute them to the field test)
- Release notes distributed to Technical Publications (who may distribute them to the field)
- Release notes distributed to TBD (who may distribute the information to call centers)

- Release notes distributed to Vendors/Contractors/Sub-contractors

# Appendices

---

- ABBREVIATIONS, ACRONYMS, & GLOSSARY. This appendix provides a gloss of terms, abbreviations, and acronyms.
- PROJECT PROPOSAL OUTLINE. This appendix provides a basic project proposal outline
- PROJECT RUBRIC. This appendix provides rubrics relevant in assessing the project and plan.
- TEAM RUBRIC. This appendix provides rubrics relevant in assessing the team cohesion and structure.
- PRINCIPAL SYSTEMS ENGINEER JOB DESCRIPTION.
- PRINCIPAL SOFTWARE ENGINEER JOB DESCRIPTION.

*[This page is intentionally left blank for purposes of double-sided printing]*

## APPENDIX B

# Project Proposal

## Outline

---

Outline for a proposal:

- Cover letter
- Cover / Title Page
- Proprietary Notice
- Table of Contents
  1. Introduction
  2. Technical Approach, objectives and methodology
  3. Project Team: Corporate Team, Project Team
  4. Relevant Experience
  5. Budget
  6. Schedule
  7. Certifications
  8. Evaluation Criteria
- Appendices
  - Corporate Description
  - Project Descriptions
  - Resumes of Key Staff

## APPENDIX C

# Project State Rubric

This appendix describes the rating of project state

## 8. PROJECT READINESS

### 8.1. BASIC PROJECT PLAN

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement
<i>scope of work</i>	A clear, complete written scope of work, that all parties can articulate and agree on. It includes the deliverables, tasks, and criteria; it is clear what is in scope and what is out of scope.	The scope of work covers the overall tasks, and parties agree to it.	The scope is unwritten, incomplete, vague, ignored or unknown by many parties	There is no defined scope of work.
<i>objectives</i>	The objectives are clear, well defined, specific, enumerated, and concrete. They are free of negative clauses, actionable, realistically achievable, and have measurable criteria. They have time bounds.	The objectives are clear, often specific. But may be lacking an detail, measures, time bounds. It isn't clear if all of the objectives are achievable or realistic.	The objectives are vague generalities, unrealistic, lacking clear definition of knowing they were achieved	none/unclear
<i>roadmap</i>	Has a clear roadmap in accessible language, with milestones, path finding development, an product generations. The roadmap has been approved.	There is a road map with milestones, research areas, and some product generations. There isn't a roadmap approval process.		There is no roadmap
<i>milestones</i>	Specific, attainable, significant, progressive. It is clear what the milestones achieve for the project and reduce risk.		There are few milestones, and several little useful purpose for the project	There are no milestones
<i>schedule</i>	There is a believable,		There is a loose schedule. It is vague,	There is no schedule

*Table 1: Basic project plan rubric*

---

approved schedule

not believed or unapproved.

---

## 8.2. THE ENGINEERING BITS

<b>Trait</b>	<b>Exceptional</b>	<b>Acceptable</b>	<b>Unsatisfactory</b>	<b>Needs improvement</b>
<i>Specification</i>	The specification is a rigorous; the vocabulary is defined and used consistently. It is straightforward to define test cases	The specification is detailed, but omits many areas, and is often vague. The vocabulary may be inconsistent, or undefined	There is a specification but it is unclear, or incomprehensible	no specification
<i>Requirements</i>	The requirements are complete, and consistently follow the EARS requirements syntax. All are testable and flow. There is a requirements style guide. Straightforward to go to realization and tests	Has a clear set of customer requirements, and team has developed a more expansive set of requirements to achieve the objectives well.	There are few requirements; or they are inconsistent, very incomplete, or unclear	Neither the customer nor the team have requirements
<i>Nomenclature</i>	There is a clear nomenclature used in naming of things. It is used consistently. It is explicitly and clearly defined.	A nomenclature is used, but may be implicit. The naming is mostly consistent, different forms of naming may occasionally used in different places.	A nomenclature is not used, or is implicit. The naming is not consistent, different forms used in different places.	A nomenclature is not used. The means for referring to things lacks consistency, clarity; shows not sign of having been though thru
<i>Review</i>	Principals/Team Leads have reviewed requirements for clarity, consistency, thoroughness / completeness, and realizability.			The requirements have not been or are not reviewed for style, content, and consistency
<i>Technology readiness</i>	Actual application in final form under expected or mission conditions. Integrated into existing system	Prototype at near planned operational system or representative conditions.	Proof of concept performed, showing that basic technology integration will work together.	Lab studies, basic properties identified. Advertising material used as source of info
<i>Technical feasibility</i>	Relatively simple project, no or little outsourcing required	Somewhat complex project with a degree of outsourcing involved	Complex project involving a lot of external expertise	Infeasible, or not clear if it is doable

**Table 2:** Basic project plan rubric

### 8.3. RESOURCING

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement
<i>funding</i>				unfunded, and may never be funded
<i>budgeting</i>	The budget is approved. clear cost/budget model: simple & useful calculation templates; clear guidance to requirements and design choices.		some rules of thumb. The budget may not be approved.	unclear cost model, unrealistic budget, expects vendors (e.g. FAEs) to do a significant amount or critical work, for free. The budget may not be approved.
<i>staffing</i>	appropriate staffing with lead roles experienced in the appropriate domains		Inadequate staffing. Vendors (e.g. FAEs) to do significant or critical work, unpaid.	expects vendors (e.g. FAEs) to do a significant amount or critical work, for free.
<i>tools</i>	The best development tools available. Every desk has a DVM, power supply, and oscilloscope		Tools are cheap but low cost. E.g. OpenOCD, GNU Compiler, Eclipse editor. Computers are slow, small monitors.	Lack of tools. Interpretation is required

**Table 3: Resourcing rubric**

### 8.4. PROCESS

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement
<i>process</i>	process, structured adaption to each project	follows process, adaption is a bit adhoc per project	has process, but it is seldom followed; adhoc	no process
<i>maturity</i>	The process is managed with metrics	The processes are defined as standard business processes.	There is some process discipline. Not too rigorous. There may be consistent results.	Chaotic, adhoc, individual heroics. The characteristics /process are undocumented. Changes are ad hoc, uncontrolled and are reactions by users or events.

## 9. TEAM READINESS

### 9.1. ROLES & RESPONSIBILITIES RUBRIC

<b>Trait</b>	<b>Exceptional</b>	<b>Acceptable</b>	<b>Unsatisfactory</b>	<b>Needs improvement</b>	<i>Table 4: Roles &amp; Responsibility rubric</i>
<i>Roles</i>	Clearly defined roles	Defined roles	Loose role assignments	No clearly-defined roles	
<i>Responsibilities</i>	Clearly defined responsibilities	Defined responsibilities; some are implicit	Loose responsibility assignment	No clearly-defined responsibilities	
<i>Accountable for</i>	It is clearly listed what each member is accountable for	Some of the things a member is accountable for is listed	The things a member is accountable for are occasionally spelled out; often implicit in the responsibilities or roles.	It is not clearly defined what each member is accountable for.	
<i>Delegation</i>	Tasks are regularly delegated, and proceed without interference	Tasks are delegated, and often proceed without interference	Tasks are occasionally delegated, or have micromanagement or other interference.	Tasks are not delegated, or often proceed with micromanagement or other interference	
	Workload is distributed fairly and team members understand each other's roles	Work is distributed fairly, but with individual focus only	Uneven work distribution, or assignments change frequently	Not clear who completed which tasks and/or very uneven distribution of work, or assignments change frequently	
<i>Collaboration</i>	Team members (happily!) fill each other's roles, if needed	Team members assist each other without being asked	Team members will help each other, if asked	Team members not collaborative	

### 9.2. SCHEDULE

<b>Trait</b>	<b>Exceptional</b>	<b>Acceptable</b>	<b>Unsatisfactory</b>	<b>Needs improvement</b>	<i>Table 5: Schedule rubric</i>
<i>buy in</i>	Team develops and buys into their schedules	The team partly develops their own schedule, but has confidence in it.	The team is handed a schedule, and/or has little confidence in it.	There are no schedules and/or the team does not believe the schedule	
<i>weekly accomplishments</i>	All team members know what they are to accomplish before the weekend.	Team members have some expectations about weekly accomplishments.	Team members know little (or none) of expected accomplishments, but work on things as they occur or available.	Team members do not know, nor care, what they are to accomplish each week.	
<i>time management</i>	Team members give concrete examples of learning time management	Team mentions learning time management	Time management skills are weak	Time management is poor or purely directed by the manager	

### 9.3. PROBLEM SOLVING

<b>Trait</b>	<b>Exceptional</b>	<b>Acceptable</b>	<b>Unsatisfactory</b>	<b>Needs improvement</b>
<i>problem solving</i>	A problem was identified and the team worked together to find a solution	A problem was identified and there is compromise evident in the solution	A problem was identified, but the chosen solution was inadequate to some team members	A problem was identified, but no steps were taken to identify a solution
	Various solutions were tested and then incorporated	Team tested various solutions to solve the problem	Some team members didn't accept the solution	One team member used power to reach their desired outcome
<i>timeliness</i>	Team has a keen sense of urgency in addressing issues	Issues are addressed	Issues may be ignored or addressed very slowly.	Issues are often ignored

**Table 6:** Problem solving rubric

## APPENDIX D

# Team Work Rubric

This appendix describes the rating of teams

## 10. TEAM STRUCTURE

### 10.1. PROFESSIONALISM

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement	<i>Table 7: Gracious professionalism rubric</i>
<i>respect</i>	Team members give concrete examples of respect for teammate	Team members show respect for teammates	Team members show limited respect for each other	Team members show little/no respect for each other	
<i>community &amp; institution</i>	Team members show increased awareness of their business/community issues, including concrete examples	Team members imply increased awareness of business/community	Team members show limited awareness of business/community issues	Team members show no awareness of business/community issues	
<i>impact on lives</i>	Team members clearly discuss how this increased awareness translates into other areas of their lives	Team members are vague about how this awareness translates into other aspects of their lives	Team is aware of Gracious Professionalism, but gives no concrete examples of what they have done to help others	Team members compete with each other to be heard	
<i>helping others</i>	Team members give concrete examples of how they have helped each other/ others	Team implies that they have helped each other/other teams	Team did not help each other/ other teams	Team doesn't understand the concept of Gracious Professionalism	

### 10.2. DYNAMICS

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement	<i>Table 8: Team dynamics rubric</i>
<i>input</i>	Team accepts input from all and sees the big picture in their overall goals	Cooperation is a dominant theme	Simple majority had input at meetings	one person's ideas are used	
<i>decision making</i>	Team members show equality and value each other's roles by entire team making decisions	Decisions made by most of the team, however focuses on individual tasks	Decisions made by simple majority without collaborative discussion	Team members working against each other	
<i>collaboration</i>	Collaboration and co-	Team collaborates	Team coexists	Coercion and/or	

	ownership are dominant themes with the members recognizing interdependence	well	peacefully	confrontation dominate
<i>micromanagement</i>	Team members and leadership do not engage in or allow micromanagement. Accountability and delegation are done maturely. The distinction between micromanagement and accountability is clearly understood.	Micromanagement is infrequent. Leaders are responsible for recognizing it and often address it.	Micromanagement occurs. The difference between accountability and micromanagement is poorly understood or ignored.	Micromanagement is permitted. The difference between accountability and micromanagement is not understood or recognized.

### 10.3. MORALE

<b>Trait</b>	<b>Exceptional</b>	<b>Acceptable</b>	<b>Unsatisfactory</b>	<b>Needs improvement</b>
<i>confidence</i>	All team members spoke, showing confidence in themselves as well as the team	Everyone was ready to answer at least one question	About 1/2, the team spoke	Only one team member spoke
<i>interest</i>	Team members show equal investment in team	Most of the team appears excited and interested	About 1/2 the team seems interested	Some team members seem disinterested
<i>enthusiasm</i>	Members enthusiastically work together to include each other	Members are enthusiastic, but talk over one another	Members are not paying attention to one another	Most team members are disengaged
<i>engagement environment</i>	Team members find the organization to be a positive work environment.	The work environment has its ups and downs, but overall positive enough.	The work environment is acceptable, but nothing to brag about	Team members find the work environment hostile
<i>engagement</i>	Highly engaged <ul style="list-style-type: none"> <li>▪ What can I do for others?</li> <li>▪ I inspire others to do their best</li> <li>▪ I love it working here</li> </ul>	Engaged <ul style="list-style-type: none"> <li>▪ I'm a vital part of the business</li> <li>▪ I feel important at work</li> <li>▪ I'm really busy &amp; very likely I'm highly stressed</li> <li>▪ I'll leave if something better comes along</li> </ul>	Almost engaged <ul style="list-style-type: none"> <li>▪ I know I'm part of something bigger</li> <li>▪ I'm almost engaged but there are times when I am not</li> <li>▪ I'm proud to work here, but I wouldn't necessarily shout it from the rooftops</li> <li>▪ I might leave if I'm tempted</li> <li>▪ There are no career</li> </ul>	disengaged <ul style="list-style-type: none"> <li>▪ I have poor working conditions</li> <li>▪ dislikes manager or working in my team</li> <li>▪ dislikes job; reads job ads</li> <li>▪ I'm here for the money, leaving when I can</li> <li>○ I'm not satisfied with the job that I do</li> <li>○ My work doesn't excite me</li> </ul>

Table 9: Morale rubric

- I'm an achiever
- I'm a clock watcher

development prospects here

## 10.4. CORE VALUES

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement
	Group articulates a clear understanding of the experience	Team shows a keen interest in subject matter, but limited use of concrete examples	Some members show an interest in science, engineering or technology	No clear enthusiasm for science, engineering or technology
	Team gives concrete examples of new skills acquired and their interest in the subject area	Team implies new skills acquired	Limited attention paid to new skills acquired	Team doesn't mention new skills acquired
	There are regular leadership opportunities, and it is clearly communicated how to anticipate them in the future.	There are some leadership opportunities	There are occasionally leadership opportunities, but it is unclear when they will be.	There are no leadership opportunities.
	The team measures work performance systematically, and regularly seeks ways to improve.	The team has awareness of performance, and performs some improvements	There are no metrics, the team occasionally finds way to improve or methods to improvement are ad hoc and ineffective.	There are no metrics, and the team does not find ways to improve.

Table 10: Core values

## 10.5. RIGHT FOCUS

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement
<i>Customer focus</i>	Team members show keen interest in users, their points of view, and considerations. Concrete actions taken to meet user considerations.	Team members show awareness how this translates into the users life. Can articulate some considerations on behalf of user.	Team members' show limited respect for users. Limited awareness of user issues.	Team members primarily disparaging or negative comments about customers/users
<i>QA focus</i>	Team members show keen interest in quality, test & QA points of view, and considerations. Concrete actions taken to meet quality considerations.	Team members show awareness how this translates into the quality. Can articulate some considerations on behalf of quality and test.	Team members show limited respect for QA and test. Limited awareness of quality and test issues. Product is "thrown over the wall" to test	Team members do not see value in quality system; primarily disparaging or negative comments about testing, quality systems

Table 11: Right focus

f

## 10.6. TEAM COMPOSITION

Trait	Exceptional	Acceptable	Unsatisfactory	Needs improvement
<i>stability</i>	Team composition is stable with time. Team has developed a mature, long-term committed relationship	A core set of the team has been together for some time, and trusts each other	Members of the team come and go; there are short term assignments to the team.	There is frequent turn over, members bungee in for a project and soon leave
<i>recruiting</i>	Identifies important skills and experiences to advance the team. Seeks out and hires members with explicit history and skills in target areas	recruits new members without regard to skillset / experience	Rarely recruits new members, and only if desperate	avoids actively (or with passive resistance) potential members with a history in critical skillset areas
<i>hiring</i>	Team hires well and systematically. There is a ready reserve of talent without idleness.	Team hires, but in an ad hoc manner	Team hires slowly, in an ad hoc manner. Hires may prove to be poor.	Hiring doesn't occur. Team is persistently short-staffed.
<i>change</i>	Team engages in change maturely. It accepts change as well as initiates it, and is selective in which changes it implements.	Team grudgingly accepts change. Rarely initiates it.	Team resists change.	Team is unable to initiate or accept change. May have hostile reactions.
<i>focus</i>	Leadership is able to keep team focused, make meaningful changes, and keep projects on track.	Are you capable of keeping your team focused and resisting whims?		Leadership and team lack focus. Changes may be made arbitrarily and frequently.
<i>training</i>	The Program Manager ensures that developers receive regular training on design and coding practices  The Program Manager ensures that testers receive annual training	Some developers receive training; often by exception. The training may be vendor marketing.	Some testers receive training; often by exception. The training may be vendor marketing.	Developers receive irregular or no training  Testers receive irregular or no training

**Table 12: Team composition rubric**

## 10.7. COMMUNICAITON

<b>Trait</b>	<b>Exceptional</b>	<b>Acceptable</b>	<b>Unsatisfactory</b>	<b>Needs improvement</b>
<i>company strategy</i>	Team understands the company strategy, and works to carry it out or be compatible with it.	Team is aware of or understands the company strategy	Team is aware of but indifferent to company's strategy; company's strategy may be vague	Team is unaware of or indifferent to company's strategy; company doesn't have an identifiable strategy
<i>roadmap</i>	Team understands the engineering roadmap, its relationship to the company strategy, and works to plan and schedule products to it.	Team is aware of or understands how the roadmap meets the strategy's goals	Team is aware of but indifferent to how the roadmap meets the strategy's goals.	Team is unaware of or indifferent to the roadmap's relationship to the strategy.
<i>communication</i>	There is regular communication with the team in meetings, with email follow-ups.	There is communication with the team; via email or meetings. Information flows thru hallway conversation and rumors	There is infrequent communication with the team; email may be unclear; there may be inadequate meetings. Information flows thru hallway conversation and rumors	There is little communication with the team; email may be unclear; there may be inadequate meetings
<i>bad news</i>	Team members are comfortable bringing up bad news at both regular meetings, in heads up emails and phone calls.	Team members will admit bad news if asked	Team members will admit bad news if asked	People hide bad news
<i>communicating up</i>	The lead/boss always hear information about the team from the team before hearing it from others.	The lead/boss hears most news about team from the team before hearing it from others.	Often hear news about team from others before hearing it from team	Hear news about team from others before hearing it from team, or not at all from team.
<i>initiation</i>	Each member provides information to their boss before he or she has to ask for it.		Provides information to boss, but only if he asks for it.	Never say anything.
<i>respect</i>	The team members communicate with each other and the rest of the company in a respectful manner.			Dominance displays and disrespect are the norm
<i>positive</i>	Communicates in a positive, optimistic tone			Frequently has a negative tone, can't do attitude and/or uses negative syntax.
<i>spreading the word</i>	Other people in the company know what your team is doing and has		Few in the company know what the team has accomplished, or is working on.	No one in the company knows what the team has accomplished, or is

**Table 13: Team communication rubric**

---

accomplished.

Awareness is  
incidental, or based on  
rumors

working on

---

## APPENDIX E

# Principal Systems Engineer

---

The Principal Software Engineer will be a member of a team working on the development and support of a line of products. The Principal Software Engineer will apply advanced knowledge, expertise and leadership to embedded platform, infrastructure, and integration software.

The Systems Engineer will be a member of a team working to development and support [TBD] of a line of products.

As a Systems Engineer, you will be responsible for providing technical leadership across functional groups. You will be the go-to person for articulating the products technical definition and use environment. You will synthesize market needs into functional and quality requirements. You will work closely with marketing, program managers, project managers, mechanical engineers, software developers, test engineers, technical publications and others in a team environment to meet business and quality objectives.

This includes the definition and documentation of electro-mechanical (and software) modules to add new or modified functionality to current productions, as well as the development of new, complex products.

### 11. ESSENTIAL FUNCTIONS

Lead the definition of the product and requirements:

- Analyze, update, and maintain systems and features that meet the expectations and uses of the customer, standards certification agencies, and the business unit.
- Partner with new product introduction teams to move new features and requirements into product development.
- Identify and solve system issues for released systems.
- Lead and participate in product definition / requirements reviews.
- Ensure that the requirements and product definition are adequate for team leads to develop their respective portions of the product.
- Provide technical leadership in the evaluation and maintenance of system designs across functional groups.
- Perform architectural design activities involving physical and functional partitioning of system functionality into design elements.

- Understand the project constraints – the milestones, deadlines, spending limits, etc. Liaise with the product manager and technical leads to explore appropriate solutions to achieve the required product features. This will require managing scope creep. If the project owner wishes to add features or a change in methodology, the leads will evaluate the amount of work it will require to accomplish. Help the product owner decide whether or not the new idea is truly a 'must have.' Make sure that they identify what other feature(s) they're willing to sacrifice in order to get the new one in.
- Ensure that the engineers are appropriately contributing to the risk management efforts.
- Facilitate technical discussions among the team(s) and ensure that decisions are made in a timely manner and documented.
- Participate in Risk Analysis and Hazard Analysis activities for systems, features, and implementations with the intent to eliminate or mitigate identified risks.

### 11.1. WRITING

- Document trade-offs, rationale, and potential solutions.
- Develop of the required technical project documentation.
- Translate user needs into formal requirements

### 11.2. COMMUNICATION

- Lead meetings to communicate the product definition, it's functions, use environments, technical features / parameters, and the product requirements to the team.
- Communicate the customer profile and perspective to the team.
- Communicate the relevant legal, industry, and corporate standards to the team
- Represent the company in meetings with vendors, distributors and customers
- Assist the sales organization in demonstrations, presentations and engineering estimates

### 11.3. CRITICAL THINKING SKILLS

- Break problems down into solvable ones, identify relevant criteria, apply knowledge and sound judgement to form well-reasoned system design decisions.
- Review technical literature and manuals. Research the relevant legal, industry, and corporate standards to apply.
- Identify & resolve important unknowns, uncertain or outdated data.
- Identify & resolve trade-offs, contradictions;
- Lead design of experiments, characterization studies and other activities to objectively ground discussions.
- Understand the relevant mechanics, model, theories, etc behind the key technologies

## 12. SKILLS & EXPERIENCE

- Analytical thinking - you simplify complex problems, processes or projects into component parts to explore and evaluate them systematically.

- Strategic thinking – you identify a vision and the plans needed to achieve the end goal; you evaluate situations, decisions and issues in the short, medium and long term.
- Influencing and persuading - you present sound, well-reasoned arguments to convince others. You draw from a range of strategies to persuade people in a way that results in agreement or behaviour change.
- Communication — your message is understood clearly, and are able to adopt a range of styles, tools and techniques appropriate to the audience and the nature of the information.
- Managing relationships and team working - you build and maintain effective working relationships with a range of people. You work co-cooperatively with others to be part of a team (as opposed to competitively), and can accomplish projects working separately or alone.
- Experience in System Engineering discipline or use of system engineering methodologies. You'll be called upon to help identify the methods that work for a particular project or team — and to help avoid habits that do not work.
- Knowledge of UL, ISO, Quality System, and other requirements. Again you'll be called upon to help identify the methods that work for a particular project or division — and to help avoid those that are not a good fit.
- Experience generating technical documentation (examples include specifications, engineering reports, test plans & procedures, validation plans, or validation reports)It's nice to have experience in requirements development, analysis, allocation, review, and tracing

### **13. ADVOCATE QUALITY**

- Ensure personal understanding of all quality policy/system items that are personally applicable.
- Follow all work/quality procedures to ensure quality system compliance and high quality work.
- Define and perform system validation activities ensuring the behavior of the system meets user needs and intended use. The Test team verifies that the product meets the requirements you set... but are the requirements the right requirements for the customer? You will develop plans to help answer that question. Review and introduce new test cases.

### **14. OTHER GENERAL EXPECTATIONS**

It is expected, or strongly desired that you are:

- A remarkable listener. A system engineer is remembered for clear articulation of the product vision and it's details... and this takes a skilled, patient listener to hear what these really are.
- A negotiator. You will be most successful by getting clear, unambiguous, concrete commitments from each of the stake holders and team members... none of which work for you (and probably not for your boss... or his).
- Good at clearly defining all sorts of goals & problems. You'll be working with lots of people who enjoy solving problems and want to know which problems are the right ones

to solve for our customers. You use analytical thinking to decompose the “big picture” problem to its essentials and communicate it to the whole team.

- Good at bringing people together. This role is inherently one that requires bringing different disciplines together; this will require tact, flexibility and firmness in equal measure.
- Self-motivated. You’re always seeking to exceed the customers expectations. You’re always thinking of what else needs to be done. You learn what you need to succeed.
- Reliable. You keep the promises that you make, both to the customer and to you colleagues.
- Independent. You’re able to produce results without requiring constant supervision, or support.
- Helpful. You’re willing to take time out of your day to teach a colleague something new, or to help someone fit their work into the teams.
- Responsive and flexible. You’re willing to do things not explicitly listed here – or anywhere else.

f

## APPENDIX F

# Principal Software Engineer

---

The Principal Software Engineer will be a member of a team working on the development and support of a line of products. The Principal Software Engineer will apply advanced knowledge, expertise and leadership to embedded platform, infrastructure, and integration software.

As a Principal Software Engineer, you will be responsible for providing technical leadership in developing new and updating existing embedded real-time software. You will need to understand the product function(s) and product use environments, as well as the detailed design of the product and the day-to-day running of a software development team. You will also be contributing to the implementation and delivery of software and hardware components. This includes the development and documentation of software to add new and modified functionality to current and new applications, troubleshooting and fixing defects in the software, and applying best practices within the team.

The Principal Software Engineer will work with marketing, other software developers, electrical engineers, clinical, service, operations, and verification engineers in a team environment to meet business and quality objectives, and synthesize market needs into quality software requirements. You will be expected to work closely with other senior technical team members, particularly in software, to ensure that software design & implementation meets these overall requirements.

### 15. ESSENTIAL FUNCTIONS

Lead Software team to architect, design and implement platform software:

- Communicate the customer profile, product function(s) and product use environments to the team
- Communicate the relevant industry, and corporate standards to the team
- Ensure that the software requirement and architecture are adequate for software engineers to implement their software.
- Understand the project constraints – the milestones, deadlines, spending limits, etc.
- Liaise with the product manager and technical architect to explore and suggest appropriate technical solutions to achieve the required product features as well as monitoring technical progress against plans. This will require managing scope creep: If the project owner wishes to add features or a change in methodology, evaluate the amount of work it will require to implement and help them decide whether or not the

new idea is truly a 'must have.' Make sure that they identify what other feature(s) and timeline they're willing to sacrifice in order to get the new one in. Work with the project manager to determine whether it's possible to add resources to the project.

- Perform work breakdowns and estimation for scheduling. Help avoid missing deadlines by clearly defining what's expected well in advance. Review the project plan with the project management team to set expectations as to the basic sequence and timing of the project work (if the the team needs Feature X by Date Y, then this plan must account for it), as well as who does what.
- Communicate the schedule to the team
- Ensure that the design is robust, meets requirements; that the architecture and documentation is complete and accurate.
- Ensure that the code is robust, complete, and meets the design.
- Monitor work against the production schedule closely, providing progress updates and report any issues or technical difficulties to the senior developers on a regular basis. Track deliverables and keep keep the staff informed about the actual progress made; provide reminders as the deadline approaches. It's inevitable you will not execute the project exactly as planned, so communicate the adjustments that need to be made, and consequences of the delay on the project as a whole, ideally with suggestions for how to mitigate the impact.
- Ensure code testability through unit testing and definition of integration tests
- Ensure that the developers are appropriately contributing to the Software Risk management efforts.
- Facilitate technical discussions amongst the software team(s) and ensure that decisions are made in a timely manner and documented.
- Lead and participate in design and code reviews
- To be responsible for work quality, ensuring it meets the technical standards for all embedded and software products (e.g. coding standards, MISRA checks, etc.)
- Mentor software engineers on software development in a medical, safety-critical, and/or regulated environment.

## 15.1. WRITING

- Assist in defining the team's technical strategy and advising on product roadmaps and migration plans to achieve strategic goals
- Ensure clear, comprehensive technical documentation

## 16. SKILLS & EXPERIENCE REQUIRED

- Analytical thinking – you simplify complex problems, processes or projects into component parts to explore and evaluate them systematically
- Strategic thinking – you identify a vision and the plans needed to achieve the end goal; you evaluate situations, decisions and issues in the short, medium and long term.

- Influencing and persuading – you present sound, well-reasoned arguments to convince others. You draw from a range of strategies to persuade people in a way that results in agreement or behaviour change
- Communication – your message is understood clearly, and are able to adopt a range of styles, tools and techniques appropriate to the audience and the nature of the information
- Managing relationships and team working - you build and maintain effective working relationships with a range of people. You work co-cooperatively with others to be part of a team (as opposed to competitively), and can accomplish projects working separately or alone.

## 16.1. DEVELOPMENT PROCESS EXPERIENCE

- Minimum of 10 years of software development experience, from concept to production.
- Expert knowledge in software analysis/architecture, technical requirements, design and development.
- Practical programming experience developing software for a commercial product, ideally involving significant capital and medical equipment.
- Experience working with other technical leads in a complex environment with multiple concurrent projects is a strong plus.
- Experience leading a software team to design and develop sophisticated software.
- Experience integrating codebase(s) with multiple provenance and different features, characteristics, and deployment platforms.
- Hands-on Experience with a professional software development process, such as FDA regulated (IEC 62304) Software Development. Expert knowledge of FDA regulations strongly preferred.
- Well-versed in R&D, product development, and clinical/regulatory process
- ISO 60601 (safety of medical devices), ISO 14971 (risk management) TIR45 (“agile” methods), ISO 13485 (quality management)

## 16.2. SOFTWARE DEVELOPMENT TECHNICAL EXPERIENCE

- Expert-level knowledge of embedded, and multi-platform software systems.
- Expert knowledge of distributed systems, interprocessor/interconnect schemes and protocols, and network programming (IP, TCP, UDP, etc).
- Hands-on experience developing real-time software, understanding bounded behaviour, rate-monotonic scheduling, etc. (expert knowledge of one or more real-time OS strongly preferred)
- Expert-level knowledge of one or more operating systems (multiple is a plus) and operating system features in general
- Experienced with build, deployment, and configuration management systems for heterogeneous, multi-platform software. Must have expert-level skills with software development tools (CM, bug tracking, source control etc.) and development process. Experience implementing such systems is a strong plus.

- Expert knowledge of software applications and utility frameworks.
- Hands-on knowledge of C/C++ is required..

### 16.3. EXPERIENCE WITH ELECTRONICS

Expert in embedded software development. Strong working knowledge of integrating software and electronics:

- Track record with multiple deployment platforms, from high-end microprocessors running full-featured operating systems, down to bare-metal embedded code deployed on lightweight microcontrollers.
- Communicating with physical devices over low-level data links (e.g. SPI, I2C, serial/UART)
- Debugging hardware/software challenges using standard tools and equipment
- Design and architecture of device drivers (both bare-metal and OS)

## 17. OTHER GENERAL EXPECTATIONS

It is expected, or strongly desired that you are:

- Good at solving all sorts of problems. New problems don't intimidate you; you use analytical thinking to decompose the problem, solving it quickly.
- Self-motivated. You're always seeking to exceed the customers expectations. When you've completed your tasks, you start thinking of what else needs to be done. You learn what you need to succeed.
- Reliable. You keep the promises that you make, both to the customer and to you colleagues.
- Independent. You're able to produce results without requiring constant supervision, or support.
- Helpful. You're willing to take time out of your day to teach a colleague something new, or to help someone work thru a particularly hard problem.
- Responsive and flexible. You're willing to do things not explicitly listed here – or anywhere else