## System Requirements Specification (SRS) PV wireless monitoring

|  |
| --- |
| Using this Document - Business Benefits |
| The System Requirements Specification (SRS) document describes all data, functional and behavioral requirements of the software under production or development.  This 10-section template covers the overall description of the system/software to be implemented, use cases and scenarios, data model, functional and non-functional requirements, interface and behavioral models, as well as restrictions and validation criteria to be used for the software. The Appendices may include business rules, glossary, traceability matrices and other necessary supplementary information that are specific to the system. |

**Table of Contents**

System Requirements Specification (SRS) 1

Using this Document - Business Benefits 2

Introduction 4

Goals and objectives 4

Statement of scope 4

Software context 4

Major constraints 4

Document Conventions 5

Intended audience and reading suggestions 5

References 5

Overall Description 6

Product perspective 6

Product features 6

User classes and characteristics 6

Operating environment 6

Design and implementation constraints 6

User documentation requirements 6

Assumptions and dependencies 7

System requirements 7

Usage Cases (Usage Scenario) 8

User profiles 8

Use cases 8

Special usage considerations 8

Data Model and Description 11

Data description 11

Functional Requirements 12

Functional model and description 12

Description for function n 12

Non-functional Requirements 13

Performance requirements 13

Safety requirements 13

Security requirements 13

Software quality attributes 13

Other requirements 13

Interface Requirements 14

External machine interfaces 14

Hardware Interfaces 14

Communications interfaces 14

Control flow description 14

Behavioral Model and Description 15

Description for software behavior 15

State transition diagrams 15

Control specification (CSPEC) 15

Restrictions, Limitations, and Constraints 16

Validation Criteria 17

Classes of tests 17

Expected software response 17

Performance bounds 17

Appendices 18

Glossary 18

Business rules 18

System traceability matrix 18

Analysis models and metrics 18

Product strategies 18

Issues list 18

Supplementary information (as required) 18

Crosscheck 19

# Introduction

This section presents an overview of the SRS to help the reader understand how the document is organized and how to read and interpret it. The System Requirements Specification (SRS) document describes all data, functional and behavioral requirements of the software.

## Goals and objectives

Describe here overall goals and software objectives.

## Statement of scope

Present a description of the software. Major inputs, processing functionality and outputs are described without regard to implementation detail.

## Software context

Place the software in a business or product line context with the aim for the reader to understand the 'big picture'. Strategic issues relevant to context are discussed.

## Major constraints

Note here any business or product line constraints that will impact the manner in which the software is to be specified, designed, implemented or tested.

# Document Conventions

Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.

For example

This section describes the notational conventions used in this document, as follows:

|  |  |
| --- | --- |
| **Indicator** | Description |
| Italics | Text in italics is meant to help the reader understand the content and organization of the document. It does not contain requirements information. |
| [Brackets] | Text in [brackets] represents variables. |
| Pseudo-code | Pseudo-code is a code stub to illustrate part of the implementation. |

## Intended audience and reading suggestions

Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.

For example

This document is intended for the following audience:

|  |  |  |
| --- | --- | --- |
| **Audience** | **Reading Suggestions** | **Sign-Off Required** |
| Product Management | Sections 2 & 4 | Yes |
| Product Development | Sections 2 to 5 | Yes |
| Quality Assurance | Sections 4 & 5 | No |
| Documentation | Sections 2 & 4 | No |

## References

List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.

# Overall Description

Present here a high-level overview of the product being specified and the environment in which it will be used, the anticipated users of the product, and the known constraints, assumptions, and dependencies.

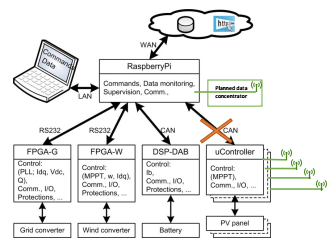
The system to be developed is a wireless monitoring system capable of acquiring data from several PV converters, supporting the mechanical and EMI influences from the environment and being integrated to the current monitoring infrastructure.

## Product perspective

Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.

The wireless feature is intended to be added to the monitoring infrastructure supported in a System-on-Chip computer – namely a Raspberry Pi 2 – and the PV power converters.

The diagram in the following figure presents the overall perspective of the development expected by this SRS



The major requirements of the microgeneration monitoring system are the data acquisition and storage, having a website interface that might be easily accessed from the internet (with all the security constrains guaranteed). The relevant data must be identified, collected and transmitted using regular communication channels (RS232, SPI, CAN, etc).

## Product features

Summarize the major features the product contains or the significant functions that it performs or lets the user perform. Details will be provided in Section 3, so only a high level summary is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or a class diagram, is often effective.

This addition to the microgeneration project has two main objectives:

1. Extend the monitoring feature to the PV power converters
2. Implement and demonstrate a feasible wireless communication among the PV power converters and the head of the system

## User classes and characteristics

Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the favored user classes from those who are less important to satisfy.

No user classes are expected in this communication module.

## Operating environment

Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.

This system will be implemented in a cabinet with EMI constrains caused by the power converters. This SRS will include the specification of the hardware platform.

## Design and implementation constraints

Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).

## User documentation requirements

List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.

## Assumptions and dependencies

List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change.

Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).

## System requirements

Describe the minimum system requirements for this product (e.g. platform version, disk space, RAM, etc.)

# Usage Cases (Usage Scenario)

Provide a usage scenario for the software. It organized information collected during requirements elicitation into use-cases.

## User profiles

Describe here the profiles of all user categories.

## Use cases

Present all use cases for the software.

## Special usage considerations

List all special requirements associated with the use of the software.

For example, a typical use case for a cafeteria ordering system (COS) may include the following actors and the use cases relevant to their roles:

|  |  |
| --- | --- |
| Primary Actor | Use Cases |
| Patron | 1. Order Meal 2. Change Meal Order 3. Cancel Meal Order 4. View Menu 5. Register for Payroll Deduction 6. Unregister for Payroll Deduction 7. Subscribe to Standard Meal 8. Modify Meal Subscription 9. Override Meal Subscription |
| Menu Manager | 1. Create Menu 2. Modify Menu 3. Define Meal Special |
| Cafeteria Staff | 1. Prepare Meal 2. Generate Payment Request 3. Request Delivery 4. Generate System Usage Reports |
| Meal Deliverer | 1. Deliver Meal 2. Record Meal Delivery 3. Print Delivery Instructions |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case ID: | 1 | | |
| Use Case Name: | Order Meal | | |
| Created By: |  | Last Updated By: |  |
| Date Created: |  | Date Last Updated: |  |
| Actors: | Patron | | |
| Description: | A Patron accesses the Cafeteria Ordering System from the corporate intranet or from home, optionally views the menu for a specific date, selects food items, and places an order for a meal to be delivered to a specified location within a specified 15-minute time window. | | |
| Preconditions: | 1. Patron is logged into COS. 2. Patron is registered for meal payments by payroll deduction. | | |
| Postconditions: | 1. Meal order is stored in COS with a status of “accepted”. 2. Inventory of available food items is updated to reflect items in this order. 3. Remaining delivery capacity for the requested time window is updated to reflect this delivery request. | | |
| Normal Flow: | **1.0 Order a Single Meal**   1. Patron asks to view menu for a specified date. 2. System displays menu of available food items and the daily special. 3. Patron selects one or more food items from menu. 4. Patron indicates that meal order is complete. 5. System displays ordered menu items, individual prices, and total price, including any taxes and delivery charge. 6. Patron confirms meal order or requests to modify meal order (back to step 3). 7. System displays available delivery times for the delivery date. 8. Patron selects a delivery time and specifies the delivery location. 9. Patron specifies payment method. 10. System confirms acceptance of the order. 11. System sends Patron an e-mail confirming order details, price, and delivery instructions. 12. System stores order in database, sends e-mail to notify Cafeteria Staff, sends food item information to Cafeteria Inventory System, and updates available delivery times. | | |
| Alternative Flows: | **1.1 Order multiple meals** (branch after step 4)   1. Patron asks to order another meal. 2. Return to step 2.   **1.2 Order multiple identical meals** (after step 3)   1. Patron requests a specified number of identical meals. 2. Return to step 4.   **1.3. Order the daily special** (after step 2)   1. Patron orders the daily special from the menu. 2. Return to step 5. | | |
| Exceptions: | **1.0.E.1 Current time is after order cutoff time** (at step 1)  1. System informs Patron that it’s too late to place an order for today.  2a. Patron cancels the meal order.  2b. System terminates use case.  3a. Patron requests to select another date.  3b. System restarts use case.  **1.0.E.2 No delivery times left** (at step 1)  1. System informs Patron that no delivery times are available for the meal date.  2a. Patron cancels the meal order.  2b. System terminates use case.  3. Patron requests to pick the order up at the cafeteria (skip steps 7-8).  **1.2.E.1 Can’t fulfill specified number of identical meals** (at step 1)   1. System informs Patron of the maximum number of identical meals it can supply. 2. Patron changes number of identical meals ordered or cancels meal order. | | |
| Includes: | None | | |
| Priority: | High | | |
| Frequency of Use: | Approximately 400 users, average of one usage per day | | |
| Business Rules: | BR-1, BR-2, BR-3, BR-4, BR-8, BR-11, BR-12, BR-33 (As specified in the Business Rules section of the document, which may also be placed as part of the Appendix section.) | | |
| Special Requirements: | 1. Patron shall be able to cancel the meal order at any time prior to confirming the order. 2. Patron shall be able to view all meals he ordered within the previous six months and repeat one of those meals as the new order, provided that all food items are available on the menu for the requested delivery date. (Priority = medium) | | |
| Assumptions: | 1. Assume that 30 percent of Patrons will order the daily special (source: previous six months of cafeteria data). | | |
| Notes and Issues: | 1. The default date is the current date if the Patron is using the system before today’s order cutoff time. Otherwise, the default date is the next day that the cafeteria is open. 2. If Patron doesn’t want to have the meal delivered, the precondition requiring registration for payroll deduction is not applicable. 3. Peak usage load for this use case is between 8:00am and 10:00am local time. | | |

# Data Model and Description

Describe the information domain for the software.

## Data description

Describe here data objects that will be managed/manipulated by the software.

### Data objects

List the data objects and their major attributes.

### Relationships

Describe the relationships among data objects using an ERD- like form. Do not attempt to provide detail at this stage.

### Complete data model

Develop an ERD for the software.

### Data dictionary

Provide a reference to the data dictionary maintained in electronic form.

# Functional Requirements

## Functional model and description

Describe each major software function, along with data flow or class hierarchy (OO).

## Description for function n

Present a detailed description of each software function. Section 4.1 is repeated for each of n functions.

### Processing narrative (PSPEC) for function n

Present a processing narrative for function n.

### Function n flow diagram

A diagram showing the flow of information through the function and the transformation it undergoes is presented.

### Function n interface description

Describe in detail the input and output interfaces for the function.

### Function n transforms

Present a detailed description for each transform (subfunction) for function n. The following section is repeated for each of k transforms.

#### *Transform k description (processing narrative, PSPEC)*

#### *Transform k interface description*

#### *Transform k lower level flow diagrams*

#### *Transform k interface description*

### Performance issues

Specify special performance required for the subsystem.

### Design constraints

Note any design constraints that will impact the subsystem.

# Non-functional Requirements

## 

## Performance requirements

If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.

## Safety requirements

Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.

## Security requirements

Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.

## Software quality attributes

Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.

## Other requirements

Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.

# Interface Requirements

Describe the software interface(s) to the outside world. Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.

## External machine interfaces

Describe interfaces to other machines (computers or devices).

### External system interfaces

Illustrate interfaces to other systems, products or networks.

### Human interface

Provide an overview of any human interfaces to be designed for the software.

## Hardware Interfaces

Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.

## Communications interfaces

Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

## Control flow description

Present the control flow for the system with reference to the following section of this document.

# Behavioral Model and Description

Describe the behavior of the software.

## Description for software behavior

Present a detailed description of major events and states.

### Events

List all events (control, items) that will cause behavioral change within the system.

### States

List all states (modes of behavior) that will result as a consequence of events.

## State transition diagrams

Depict the overall behavior of the system.

## Control specification (CSPEC)

Describe the manner in which control is managed by the software.

# Restrictions, Limitations, and Constraints

Note all special issues which impact the specification, design, or implementation of the software.

# Validation Criteria

Describe the approach to software validation.

## Classes of tests

Specify the types of tests to be conducted, including as much detail as is possible at this stage. Emphasis here is on black- box testing.

## Expected software response

Describe the expected results from testing.

## Performance bounds

Specify all special performance requirements.

# Appendices

Present all information that supplements the Requirements Specification.

## Glossary

Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.

## Business rules

Include all relevant business cases that impact the system/software, using the following sample format:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Rule Definition | Type of Rule | Static or Dynamic | Owner |
| BR-1 | Delivery time windows are 15 minutes, beginning on each quarter hour. | Fact | Static | Project Manager |
| BR-2 | Deliveries must be completed between 10:00am and 2:00pm local time. | Constraint | Dynamic | Project Manager |

## System traceability matrix

Provide a matrix that traces stated software requirements back to the system specification.

## Analysis models and metrics

Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.

## Product strategies

If the specification is developed for a product, a description of relevant product strategy is presented here.

## Issues list

This is a dynamic list of the open requirements issues that remain to be resolved, including TBDs, pending decisions, information that is needed, conflicts awaiting resolution, and the like.

## Supplementary information (as required)

|  |
| --- |
| Crosscheck |
| Ask yourself the following questions:   * Do the goals and objectives in the introduction section of this document agree with the business requirements of the project? Do they agree with the project charter? Are there any elements of the project charter that are unsupported by the system? Why? * Have you verified the use cases with end users who did not participate in the development of the original user cases? * Have you asked those users, “what’s missing”? Since missing elements are often outside of the core system scope, are the elements satisfied by outside systems that interface with this one? Are those outside systems captured in the interface requirements? If not, have you captured those missing elements under restrictions, limitations and constraints? * Does the documentation you have identified as necessary, cover all user classes? (end user, power user, technical user) * Have you fully described all data, functional and behavioral requirements of the software/system as laid out in this template? * Have you completed the overall description of the system/software to be implemented, use cases and scenarios, data model, functional and non-functional requirements, interface and behavioral models, as well as restrictions and validation criteria to be used for the software? * Have you included all relevant business rules, glossary, traceability matrices and other necessary supplementary information in the Appendix section? |