"Why would I even need a software requirement specification for elearning system?" is a frequently asked question among the people who hire a custom elearning software company for the first time. In this article, we will explain why having an lms requirements document is beneficial to any eLearning project and show an example of one.

Contents

SRS vs. V&S

There is a document that can be confused with SRS - a **Vision and Scope (V&S) document**. However, they have major differences. **V&S contains high-level descriptions** of what the finished product means to be (vision) and which features it should include to bring this vision to life.

SRS, on the other hand, describes the project in detail and serves as a foundation and a roadmap for the project.

Both documents have their place in the development process: V&S is created at the early stage to clarify the project's intention and locate the areas which require in-depth analysis. SRS comes later and gives the customer a realistic estimate of the time and money needed to complete the project, as well as extended descriptions of each main feature.

SRS and V&S are traditionally written by a business analyst.

How Belitsoft Can Help

Given that we have over 15 years of experience in creating products for both SMEs and multinational corporations, here's what we can do for you:

- **SRS and V&S preparation.** We can help you structure your idea and make it understandable by any dedicated developer anywhere in the world. After you get your SRS from us, you'll be able to get quotes from multiple potential contractors knowing full well that all the estimates are based on the same feature set.
- **Custom software development.** If you like what you see in your documents you can continue working with us until the project release and beyond.

Example of SRS for eLearning project

There are many ways to create a software requirements document, as there are no regulations governing its use. Moreover, the actual documents are the confidential property of the customers and are protected as such (see our IP protection policy). We will use a mock SRS for a corporate learning management system (CLMS) as an example.

1. Purpose

This section describes the purpose and possibly the target audience of the document.

Example

The purpose of the Corporate Learning Management System Software Requirements Specification is to describe the main functionality of CLMS in a clear and concise way.

The document is to be continuously updated to reflect the changes to requirements during the implementation and utilization of the Corporate Learning Management System so that an accurate baseline of actual requirements is available at any time.

2. Scope

This section includes the following:

- The name of the software being developed;
- Functions and purpose of this software;
- The goals of this software and the benefits it will bring;

The name, goals, and functions should be consistent with other documents used in the development to avoid confusion.

Example

The purpose of the Corporate Learning Management System is to improve the learning process for Employees and deliver the most relevant training content.

Below see is a list of all major features of the solution.

Feature	Name	Description
HR-manager		
FT-01	**User Management**	1. Authorization/ Registration 2. Add User 3. View the list of Users 4. View User profile information 5. Edit User profile information 6. Delete User 7. Archive/ Unarchive User 8. Manage User's rights 9. View User dashboard
FT-02	**Administration**	1. Create Company Organizational Structure 2. Create Skill Matrix
Learner		
FT-03	**Onboarding**	1. Start/Stop/Skip/Complete System Onboarding 2. Start/Stop/Complete Company Onboarding

Feature	Name	Description
FT-04	**Professional development**	1. Add Skill 2. Set Deadline 3. Delete Skill 4. Archive Skill 5. View the Skill completion progress 6. View the list of Skills 7. View information about Skill 8. View the list of completed Skills 9. View the list of Interests (favorite Skills) 10. Search for Skills
FT-05	**Course** **(All Courses are stored in LMS)**	1. Start the Cours 2. Stop/Continue passing the Course 3. Complete the Course
FT-06	**Certificates**	1. Upload the Certificate 2. View the list of Certificates
FT-07	**Badges**	1. Get Badges 2. View the list of Badges
FT-08	**Events**	1. View the list of Events 2. Create new Event 3. Edit Event 4. Delete Event
FT-09	**Widget**	1. Add new Widget 2. Hide widget
FT-10	**Calendar**	1. View upcoming Events
FT-11	**Chat**	1. Online chat with other Users 2. Online chat with the Support team 3. View the list of dialogues

Feature	Name	Description
FT-12	**Notifications**	1. Enable/Disable Notifications 2. View the list of Notifications
Mentor		
FT-13	**Professional development**	1. Add Skill to Learner Professional development plan 2. Set Deadline 3. Create Skill package 4. Delete from Skill Professional development plan 5. View the Skill completion progress 6. View the list of Skills 7. View information about Skill 8. View Learner dashboard

3. Product Perspective

This section describes the relations this product has to other software (if any). If the product is a part of a larger suite, this is where the connections between different elements should be described (e.g. with a block diagram).

Here are the most important points that need to be included:

- System interfaces;
- User interfaces;
- Hardware interfaces;”
- Software interfaces;
- Communications interfaces;
- Site adaptation requirements.

Example

The CLMS requires an internet connection to display content and allow users to interact with it. All system information is stored in a database, which is located on a web-server. The CLMS requires an internet browser (Chrome, Firefox, or Edge) to be accessed.

3.1 System Interfaces

This is the place to describe all the interfaces with other systems and the requirements for the implementation of each one.

3.2 User Interfaces

This is what needs to be specified:

- **The logical characteristics** of each interface between the software product and its users. This includes the required screen formats, page or window layouts, the content of any reports or menus, or availability of programmable function keys necessary to accomplish the software requirements.
- **Interface optimization methods** and the people responsible for them. A simple list of dos and don’ts will do the trick.

This section can also contain a link to the style guide for the project.

Example:

A first-time user of the CLMS should see the login page when they go to the system's website. The users are added by an HR-manager, no registration option is required. If the user is logged in they should be directed to the course selection page. Here they can see the courses that are available to them, including the onboarding program.

3.3 Hardware Interfaces

Specify the logical characteristics of each interface between the software product and the hardware elements of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.

3.4 Software Interfaces

Specify the use of other software that is necessary for the product to function. It can be operating systems, web browsers, data management systems, etc. If the product is modular, describe the relations between each module (e.g. course management and user management).

Each piece of connected software should have its specification number and version number to avoid confusion and possible version conflicts.

Each included interface should have a statement on why it is necessary and a description of the message content and format. If the interface is already well-documented, a link to the documentation will be enough.

3.5 Communication Interfaces

Specify the various interfaces to communications (e.g. LAN protocols used).

3.6 Memory Constraints

List the requirements for internal and external storage devices (if applicable).

3.7 Operations

Specify the operations required by the user such as:

- User-initiated operations (e.g. launching a course);
- Periods of interactive operations and periods of unattended operations (e.g. data integrity check);
- Data processing support functions;t
- Backup and recovery operations.

Sometimes this is included in the User Interfaces section.

3.8 Site Adaptation Requirements

This section describes:

- Definition of the requirements for any data or initialization sequences that are specific to a given site, mission, or operational mode (e.g., grid values, safety limits, etc.);
- Specification of the features specific to a site or the mission that need to be changed to adapt the software to a particular installation.

4. Product Functions

This section contains the general description of the jobs that the product will perform. In the case of CLMS, this would include uploading and playing courses, managing users, and skill matrix creation without mentioning every minor detail of each activity.

If there is a V&S document, some descriptions can be taken directly from it.

The end goal here is clarity. As such, one of the popular benchmarks is that the product functions section is understandable to a person who reads it for the first time. Using visual aids (diagrams, pie charts, etc.) is recommended.

5. User Characteristics

Describe the target audience of the product, primarily from a usability standpoint. Include everything that can affect the interaction with the product, including disability, education level, and technical skills.

Describe the target audience of the product, primarily from a usability standpoint. Include everything that can affect the interaction with the product, including disability, education level, and technical skills.

6. Limitations

This section should include a high-level description of factors that will impose constraints on the development process and the finished product’s operation. They can be:

- Laws and regulations;
- Hardware limitations (e.g., signal timing requirements);
- Interfaces to other applications;
- Parallel operation;
- Audit functions;
- Control functions;
- Higher-order language requirements;
- Signal handshake protocols (e.g., XON-XOFF, ACK-NACK);
- Quality requirements (e.g., reliability);
- The criticality of the application;
- Safety and security considerations;
- Physical/mental considerations

7. Assumptions and Dependencies

This section includes the factors that aren’t design constraints per se, but rather can affect the development process if changed. For example, the team may assume that a specific human resource management (HRM) system will have an open API that will be used for integration with CLMS. If the HRM doesn’t have one, the development scope would have to be changed.

8. Apportioning of Requirements

Describe how the requirements for the product will correspond with its actual elements and features.

A requirement might have several elements fulfilling it. Moreover, some of the elements can be planned to be released later than the core features - this should also be noted.

This can be done best by using a cross-reference table.

9. Specific Requirements

This is the place to describe all the software requirements in detail. The amount of said detail should be sufficient for the designers, developers, and testers to do their respective jobs.

The minimum level includes the description of each possible input and output of the system, as well as all the functions of the product related to them

10. External Interfaces

List all the systems that the product is connected with and the information it either sends to them or receives from them. This information shouldn’t repeat points 3.1 through 3.5, but rather complement them.

Each interface defined should include the following content:

- Name of the item;
- Description of purpose;
- Source of input or destination of output;
- Valid range, accuracy, and/or tolerance;
- Units of measure;
- Relationships to other inputs/outputs;
- Screen formats/organization;
- Window formats/organization;
- Data formats;
- Command formats;
- End messages.

11. Functionss

Define the main activities that have to happen in the product (with the relevant inputs and outputs), including:

- Validity checks on the inputs;
- The exact order of operations;
- Responses to abnormal situations, including error handling and recovery;
- Effect of parameters;
- Relationship of outputs to inputs, including Input/output sequences and Formulas for input to output conversion;
- It may be appropriate to partition the functional requirements into subfunctions or subprocesses. This does not imply that the software design will also be partitioned that way.

Example

Assessment Module

CLMS will use summative assessment in training.

The goal of summative assessment is to evaluate student learning at the end of an instructional unit by comparing it against some standard or benchmark.

It can help determine whether or not the learner is ready to move onto the next section.

Advantages:

Know if students have understood the material

Determine achievement level

Identify weak spots

Measure training success

Basic flow

It’s necessary to pass the Test to complete the skill or to pass the Interview. The system will change Skill status according to Employee results.

12. Usability Requirements

The usability requirements should be just as specific and quantifiable as any other. They should also be supported by including their purpose, e.g. “to reduce waiting time and user frustration, 90% of users should be able to log into the CLMS within 4 seconds.”

13. Performance requirements

This section is meant for the static and dynamic numerical requirements placed on the software as a whole.

Static numerical requirements are the ones that don’t fluctuate (e.g. “1000 simultaneous users”). They are sometimes placed in a separate section called “Capacity”.

Dynamic numerical requirements can be dependent on other factors (e.g. “90% of transactions processed within 1 second during normal hours and within 2 seconds during peak hours”)

14. Verification

This section lists all the approaches and methods used to ensure that the product fulfills the requirements. Traditionally, the requirements and functions are listed as a table with each item corresponding to one or several verification methods.

15. Supporting Information

Additional materials that can be attached to the SRS include sample input-output formats, description of the problems that the product should solve, cost analysis studies or any other information that can help the reader.