Documentation Guidelines 2022/23 - SCRUM

Summary of Documents

Document	M/O	Description
Project plan	M	Presentation slides (+revisions)
Requirements definition	M	Presentation slides (+revisions)
Design description	M	Presentation slides (+revisions)
Product backlog	M	Living document or within a tool
Product burn down chart	O	Living document or within a tool
Sprint backlog	M	Living document or within a tool
Sprint burndown chart	M	Living document or within a tool
Minutes-of-meeting documents (MoM-s)		Living document or within a tool
Sprint planning report	O	Presentation slides
Sprint review report	M	Presentation slides
Sprint retrospective report	M	Presentation slides
Acceptance test plan	M	Presentation slides
Acceptance test report	M	Presentation slides
Final report	M	Document (20 page limit – the same as the SCORE report)
Product technical documentation	O	As agreed with the customer

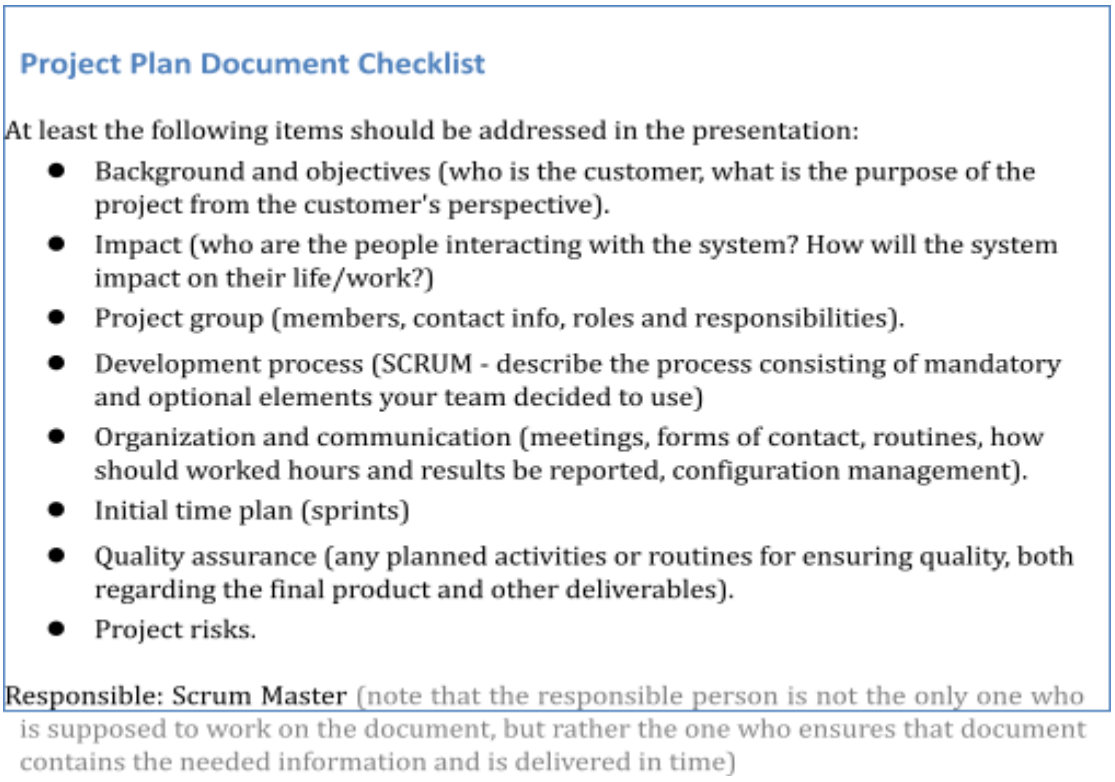
* M = mandatory, O = optional

1. Project plan

From this set of slides, the interested party should quickly get a high-level understanding of what the project is about, and get some basic facts on the organizational scenario at hand, such as who the customer is and who is in the project group. The document should also specify how the work will be organized (how SCRUM is implemented) and present an initial time plan for the project. Before submitting, the content of the slides should be subject to an internal review. Goals of the review should be to understand the following:

- Does the content provide a clear picture of how your project is organized and how you plan to carry out the work?
- If you present the content to another team, would they be able to carry out the project according to it?

The whole project team is involved in producing this content. The Scrum Master is responsible for the content. There should be only an initial and a few more revisions (if needed) of this content. Each revision should be stored as a separate set of slides (i.e., presentation file) and easily accessible throughout the project duration.



Project Plan Document Checklist

At least the following items should be addressed in the presentation:

- Background and objectives (who is the customer, what is the purpose of the project from the customer's perspective).
- Impact (who are the people interacting with the system? How will the system impact on their life/work?)
- Project group (members, contact info, roles and responsibilities).
- Development process (SCRUM - describe the process consisting of mandatory and optional elements your team decided to use)
- Organization and communication (meetings, forms of contact, routines, how should worked hours and results be reported, configuration management).
- Initial time plan (sprints)
- Quality assurance (any planned activities or routines for ensuring quality, both regarding the final product and other deliverables).
- Project risks.

Responsible: Scrum Master (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

2. Requirements Definition

The purpose of this content is to define what you are supposed to develop, viewed from the user perspective. This will be used initially to ensure that you have a common view of the task before you (within the group, and between the group and the customer). It will later be used to guide the development, and it is also one criterion that can be used to measure the quality of the delivered system. Remember that *requirements should focus on what to do, not how to do it*. Avoid describing solutions.

The following items are relevant for most projects:

- High level description of the domain and the problem.
- If the project is about extending or improving something that exists, give a description of the existing system, or the context: any existing or planned entities that the developed system should interact with, but which will not be developed in the project (for example, physical equipment or an existing database).
- Requirements and user stories, as detailed in the next section.

The Product Owner maintains the Requirements Definition content. There can be more than one version of this content, and changes in the content between versions should reflect both the improvement of understanding of what the project team is about to develop, as well as the evolution of requirements from the customer side.
There can be a Requirements Definition document behind the scene (not mandatory), and the slides presented to teaching staff can represent only a *view* of that document.

Requirements and user stories

Requirements are prescriptive assertions that the system should fulfill. Examples of requirements are those shown in the following table.

ID	Name	Description
Ticket Reservation		
CMS-1	Existing customer sign-in	Customer should log in with the existing login and password
CMS-2	New customer sign-up	A new customer should registers its personal information and receive a temporary password
...	...	...
Ticket reservation		
RES-1	Seat reservation	The system should support users in reserving a place on the train upon ticket purchase
RES-2	Reservation response time	The system should respond in max 5 secs to user inputs
...	...	...

Make sure all requirements can be validated, i.e., that there is a way to determine if a requirement is satisfied or not, avoid vague and ambiguous requirements such as "the system should be sufficiently fast". If you have a requirement such as “the system will have 99,95% availability”, be ready to show that you know how to validate it. Requirements should be understandable for both the customer and the group, avoid being too technical. Requirements can be functional as CMS1, CMS2 and RES-1, or non functional as RES-2.

User stories are the primary production unit in the scope of SCRUM-based Agile Software Development. Essentially, a user story identifies a very high-level description of a usage scenario containing just enough information so that the developers can: (a) provide a reasonable estimate of the effort needed to develop a Potentially-Shippable-Product (i.e., a self-contained and readily shippable part of the final end-product) for that story; (b) incrementally refine the details present in the user-story as the development problem becomes more clear; (c) implement the user-story in practice with a reasonable amount of assumptions.

While there are various approaches for defining user stories, we suggest you use them as a tool to clarify how the users can interact with the system in order to achieve the fulfillment of functional requirements. For instance, a user story associated to RES-1 in the table above could contain the following steps:

> Upon selection of the departure and arrival stations, of the train to be used, and of the kind of service (business, economy, first class etc.) and before requiring the payment of the ticket:
> - The system asks the user if he/she wants to select a specific place.
> - If the user answers no, then the system assigns a random place to the ticket being purchased.
> - Otherwise, the system shows to the user a proper GUI to allow him/her to select one of the available places.
> - The user selects a place and confirms the selection.
>
> This description would greatly benefit from the usage of UML sequence diagrams (refer to: http://agilemodeling.com/artifacts/sequenceDiagram.htm).

For traceability reasons, each requirement, besides the ID, the name and a description, includes also a motivation and the source (e.g., from customer or identified by the project group).

User stories should be explicitly associated to corresponding requirements (which might not be a straightforward 1:1 mapping). Moreover, they include the following information:

- Story ID (in case of use case or user story).
- Story name.

Requrements Definition Checklist

- Short background (refer to Project plan document for details).
- High level description of the domain and the problem.
- If the project is about extending, improving or integrating something that exists, a description of the existing systems
- Requirements and user stories: Each requirement/story definition must include the following elements:
  - Unique ID
  - Short and descriptive name
  - Motivation and source
  - Detailed description (text, diagrams, pictures etc)
  - For user stories: references to the corresponding requirements

Responsible: Product Owner (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

Revisions: yes, as more information is provided by customer or development team, each version published, latest version easily identified

3. Design Description

The design description should capture important design decisions you have made in the project and should provide a good basis for understanding the code. A possible reader (in addition to the course staff) would be a new member joining the group, or someone who will maintain or evolve the system further.

The following items are relevant in most projects, but you should think about what is important to capture e.g., maybe some of the items below can be skipped, but possibly other items should be added:

1. Short project background (refer to Project plan document for details).
2. High level description of the system to be developed and the desired functionalities (refer to Requirements definition document for details).
3. System overview, especially if the developed software is part of a larger system (e.g., describing the operational context, e.g., hardware and any equipment controlled by or providing input to the system, and detailed descriptions of interfaces to other systems or existing parts of the system not developed within the project).
4. Software architecture. What does the overall decomposition of the software into modules/units/components look like (from a static and dynamic perspective, i.e., what are the units and how do they interact)? What is the rationale (motivation) for this design? Depending on your project, you might also want to describe concerns such as major system states, persistent data, access control, synchronization and timing, error handling, security, etc. Use appropriate figures and diagrams (for instance suitable UML diagrams) to define and exemplify. Make sure to explicitly show the connection between the requirements and use cases on one hand and the software modules on the other.
5. Graphical user interface. Describe the structure and general layout of the interface. You can show the interface by providing mockups (which can later be substituted with screenshots).
6. Detailed software design. For each (or some) of the architectural units, provide a detailed description of its internal design from a static and dynamic perspective i.e., what are the parts (e.g. classes) and how do they interact. Depending on your project, you can describe your database model, Web page organization, key algorithms, protocols between different components etc. Use appropriate figures and diagrams (for instance suitable UML diagrams) to define and exemplify.

The design description is supposed to be a living document that is updated with your application through the development process. In the initial phases of the project it will contain points 1, 2, 3, 4 and 5, while point 6 will be added later on and updated at the beginning of each sprint.

The Scrum-Master is responsible for the document, but all team members must contribute to its content. All versions of this document must be clearly marked and accessible.

Design Description Document Checklist

At least the following should be addressed in the document:

- High-level system description (text, images)
- System overview, system context (text, images)
- Architecture (text, UML diagrams, images)
- GUI (images, mockups, text)
- Detailed software design for (each) architectural unit(s), static and dynamic perspective, database schema(s), web site /page organization, algorithms, protocols (text, UML diagrams, images)

Responsible: Scrum Master (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

4. Product Backlog

The Product Backlog (Scrum Handbook, page 18) is a simple document (usually in the form of a table) or part of the management tool used, and shall include the following elements as backlog item types:

- Product features to be developed (with included references to requirements and/or user stories, as defined in Requirements Definition). Note that more than one feature can contribute to a specific requirement or to a user story.
- Engineering improvements (improving performance etc.) and corrective actions (e.g. correcting identified bugs, can contain references to tickets in a bug tracking system or other system used by the project team).
- Technical work (such as installing/updating software, setup of development environment etc.).
- Knowledge acquisition (such as researching different libraries/tools etc.)
- Other elements relevant to project work that consume project team resources.
- The product backlog should NOT include activities specific to the DSD course, but not relevant to the product development, such as various presentations (requirements, alpha, beta, final etc.)

We suggest that you use some or all the following attributes for product backlog items:

- Item ID, defined by the Product Owner
- Item name, defined by the Product Owner
- Item definition date, defined by the Product Owner
- Item source, if feasible (requirement ID and/or name, team's internal requirement, bugID from bug tracker system, ...)
- Item priority, defined by the Product Owner
- Status, controlled by the Product Owner
    - DRAFT - not completely refined, cannot be selected for the next sprint.
    - DEFINED - fully defined, can be selected for the next sprint.
    - UNDER DEVELOPMENT - being implemented in the current sprint.
    - COMPLETED - no work needed, work effort remaining reached 0.

    - REFINED – DRAFT item split into two or more new items during the refinement process, working effort set to 0.
    - ABANDONED – will not be worked on, work effort remaining set to 0.
- Estimate of risk, defined by the Development Team
- Estimate of value, defined by the Product Owner
- Initial estimate of work effort, defined by the Development Team
- Estimates of work effort remaining as of sprint x, updated by the Product Owner

The list of backlog items will be sorted according to the current item priority. The content will be updated frequently (initial backlog shall be created from the requirements, its contents redefined, and items reprioritized during product backlog refinement process, sprint planning meetings etc.); this content should be stored within a single living document or within a specialized management tool the team is using. There shall be only one version of this living document – its contents will evolve during the project (if possible, document versioning should be turned on). The Product Owner maintains the document or tool content, but all team members shall be involved (providing work and risk estimates etc.)

Product Backlog Checklist

Backlog contains prioritized list of backlog items with suggested properties defined in the preceding text:

- ID
- Name
- Date of definition
- Source
- Priority
- Status
- Risk estimation
- Value estimation
- Estimate of work effort (points)
- Estimate of work remaining as of sprint *x*

Responsible: **Product Owner** (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

Revisions: living document or within a tool

5. Product Burndown Chart

Product burndown chart (Scrum Handbook, page 29) is created from the product backlog and is updated after each sprint is finished. The Product Owner maintains it. It can be a in a form of a standalone living document, part of the Product Backlog document or part of a management tool used.

Product Burndown Chart Checklist

Responsible: Product Owner (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

Revisions: living document or within a tool

6. Sprint Backlog

There will be one Sprint backlog document for each sprint conducted. Sprint Backlog (Scrum Handbook, page 22) document should be a relatively short living document (or part of the management tool used) containing only basic information on current Sprint Backlog status; list of backlog items and additional data. Backlog items are development tasks usually derived from the selected product backlog items during the sprint-planning meeting, but their list can be updated during the sprint. The first version of this document should be the main outcome of the sprint planning meeting. This document contains all the data for the sprint burndown chart. We suggest the following data should be present in the Sprint backlog document:

- Task ID
- Task name
- Product backlog item and/or itemID the task belongs to (if applicable)
- Team member(s) assigned to the task, filled in by the volunteering team member(s)
- Initial estimate of work effort in hours, estimated by the whole team during sprint planning meeting
- Estimates of effort remaining as of sprint day *x*, in hours, estimated by the team member assigned to the task until it reaches 0

The Scrum Master maintains this fast-changing content, within a living document or a specialized tool. All team members will update it daily.

Sprint Backlog Checklist

The document contains the list of sprint backlog items, the following properties are suggested:

- ID
- Name
- Originating product backlog item (ID and/or name)
- initial estimate of work hours
- assigned team member(s)
- estimate of remaining work hours as of sprint day *x*

Responsible: Scrum Master (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

7. Sprint Burndown Chart

The Scrum Master maintains the Sprint Burndown chart (Scrum Handbook, page 25). There will be one Sprint burndown chart for each sprint conducted. It can be a standalone document, part of the tool, or part of the Sprint Report document.

Sprint Burndown Chart Checklist

The document contains the graphical representation of the sprint progress.

Responsible: Scrum Master (note that the responsible person is not the only one who is supposed to work on the document, but rather the one who ensures that document contains the needed information and is delivered in time)

8. Minutes-of-Meeting

Minutes of meeting (MoM) documents capture the most important data on meetings (attendance, discussion topics, conclusions, actions resulting from conclusions). Separate documents can be produced for each such meeting (longer or team meetings or short meetings such as stand-up meetings …) or there can be only one living document to which new MoMs are appended.

Scrum Master is responsible for MoM documents but should delegate creation of such documents to other team members.

9. Sprint planning report

This is a type of MoM document (Scrum Handbook, page 20) that captures the most important content of a sprint planning meeting. It should be in a form of a set of slides and included in the sprint report presentations. Sprint planning report can contain or

point to the versions of product backlog and sprint backlog, as they were defined at the sprint planning meeting.

10. Sprint review report

This is a type of a MoM document (Scrum Handbook, page 28) that should be in a form of a set of slides. It should contain the review of the current state of the *product* (is it potentially shippable, what was the outcome of the testing?) and requirements (sprint backlog, sprint burndown chart, product backlog), conclusions drawn from the discussion among team members and with the product customer.

11. Sprint retrospective report

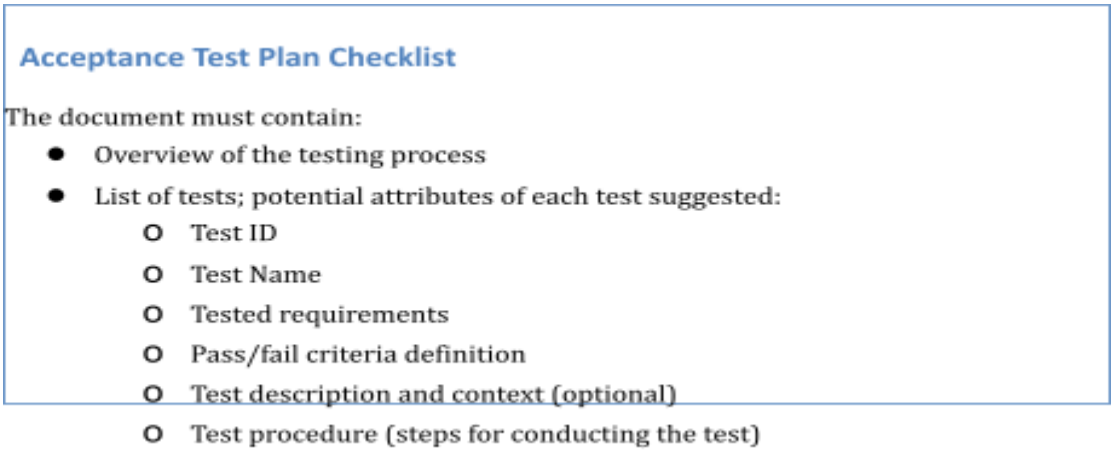
This is a type of a MoM document (Scrum Handbook, page 28) that should be in a form of a set of slides. It should document conclusions drawn from the discussion among team members, and changes to the *process* that should be introduced in the next sprint.

12. Acceptance Test Plan

Acceptance testing is done to determine if the requirements are met. The acceptance test plan should outline the plan for test activities to be performed by the customer (or together with the customer), motivated by the requirements. Each test should be described detailed enough so that someone else could carry out the test based on these instructions. The test specifications will differ a lot between different projects, and you have to decide what tests are relevant in your particular project and how to describe them in the best way. The following items give some ideas of what you could address:

- As always, you need some background and references to other related documents.
- Give a high level overview of the test activities, and motivations why these are relevant and sufficient.
- For each test, define detailed instructions how to carry out the test and how test results are measured (pass/fail and what is the criteria, letting the test subject answer a set of questions, etc.). Remember to relate the tests to the requirements (use requirement ID and/or name to reference the particular requirement).
- Document and prove that each requirement / use case / user story is covered by at least one test, so that the set of used tests presents a thorough and complete test of all the defined requirements

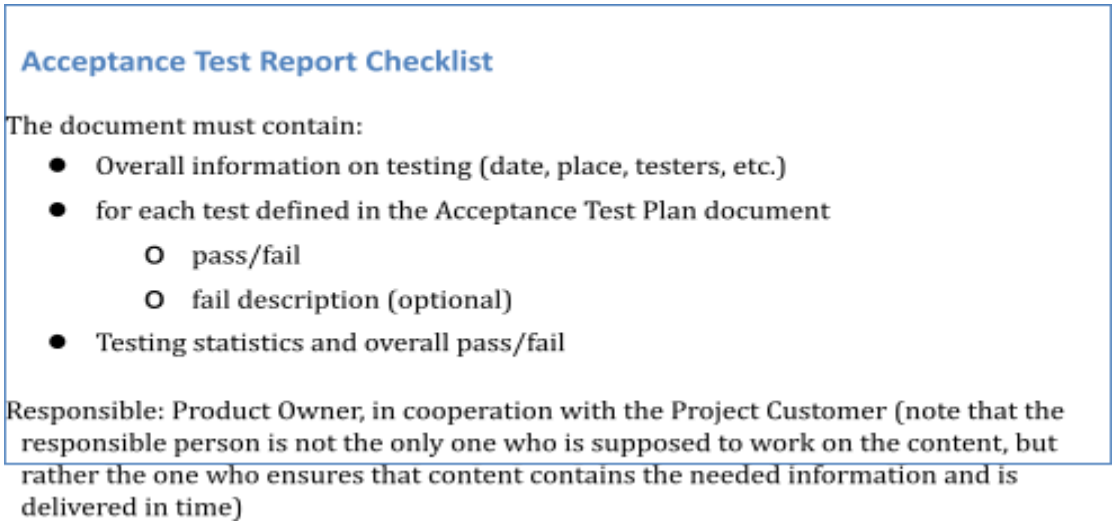
Product Owner is responsible for the set of slides, but the Project Customer and all team members should be involved in its creation. There can be a real document used by the



team behind the scene (not mandatory), and set of slides representing a *view* of that document.

13. Acceptance Test Report

The acceptance test report captures and summarizes the test results in a form of a set of slides. The structure of the test report should be defined together with the test specification. Then, as the tests are carried out, the report is filled with information. Finally, when testing is over, the results are summarized.
Product Owner and customer are responsible for the set of slides. There can be a real document (for example spreadsheet) behind the scene used by the team (not mandatory), and the set of slides just representing a *view* of that document.



14. Final Project Report Document

The final project report document, limited to 20 pages (A4 single column format including tables, figures and literature) should summarize the experiences of the project, both in terms of the produced results (*product*) and of the project work (*process*). Identify as many parameters/metrics as possible that can be used to illustrate the project work. These metrics should be both on the level of the project, on the level of the two distributed sub teams and on the level of individual team members. Make an analysis of the results with respect to the original project plan, show in what aspects the original plan was followed and in what aspects the project deviated from the plan. Extensively use data gathered during sprints, compare performance among sprints, assess whether conclusions and measures agreed on sprint retrospectives lead to performance and/or quality improvements. Use suitable figures, diagrams and tables.
At least the following items should be addressed:

- Background information and references to other related documents.
- Project results.
  - Give an overview of the results produced in the project.
  - List the produced deliverables.
  - Refer to the Requirements definition and state how many of the requirements have been satisfied (fully or partially)
  - Comment on missing functionality and possibilities for improvements and extensions.

- Project work
  - Organization and routines (refer to project plan and comment if there have been any changes during the project).
  - Total project effort (e.g., in person-days) and how it was distributed over different activities (activity types, product backlog item types, sprints, weeks etc.)
  - Worked hours per group member (total, per sprint, per week, per product backlog item type, number of tasks done etc.)
  - Other suitable metrics.
  - Positive experiences (for example successful routines or techniques).
  - Improvement possibilities (for example, what would you do different if you were to do a similar project again?).

In case the project team is participating in the SCORE competition, the final report is the same document as the SCORE report.

Scrum Master is responsible for the document, but all team members must be involved in its creation.

Final Report Document Checklist

Responsible: Scrum Master, all team members should contribute

Revisions: no revisions, one document version finalized before the final course deadline