Preparation for Bachelor Project
E18 E6BAC-01

Designing Multichannel Audio- and Video-Playback System: Showman

Group members:

Name	Studentid.
Minik Nathanielsen Olsen	201600341

Contents

Chapter 1

Project Description

The following text is the project description pulled from the project catalog.

Development of multi-channel audio/video playback system for touring artists. This project will integrate both hardware- and software-design with emphasis on hardware design. Approx. 70-80 percent of modern Danish and international touring artists utilize audio backing tracks (prerecorded material played back on some chosen device) and video content projected on large screens as part of their performance.

However, there are a multitude of options ranging from iPod-playback to complex, customized systems operated by technical personnel.
Common for these systems is instability, making artists to purchase redundant backup-systems in order to deliver their audience a flawless and uninterrupted performance.

In a typical scenario, at least one laptop – often several laptops – with various audio- and/or videointerfaces, handles and runs the audio backing track and video-content in sync.

This project will integrate both audio- and video-handling in a single 19" rack-mounted device and provide the user with a user-friendly GUI (Graphical User Interface) while complying with international standards for audio and video and electronic devices.

For design and implementation of the device, a research is needed in:
- Circuit/hardware design
- D/A audio converters
- Standardized audio input/output formats
- Audio file formats
- Standardized video input/output formats

- Video file formats
- Time codes for synchronization
- User feedback for feasible GUI design

This project is already in the early stages of development as part of an apprenticeship in entrepreneurship (iværksætterpraktik) at Navitas Science and Innovation.
Current status on project:
- Requirements specification finished
- FURPS finished
- Market research ongoing

The project is requesting confidentiality as the device is in development intended for market release to prosumers in the music industry, thus an eventual project partner must sign a non-disclosure agreement.

Project partner profile: Electronic Engineering student or Information and Communication Technology student.

Chapter 2

Requirement Specification

Introduction

This chapter will describe the requirements needed for the multichannel audio/video playback system called Showman. This chapter contains an introductory overview of Showman in general that covers a System Description, System Overview and Usage Situation followed by Functional and Non-Functional Requirements. The chapter ends with a MoSCoW- and FURPS-analysis to organize a list of priorities needed to develop a functional prototype.

System Description

The purpose of Showman is to play back prerecorded audio- and video-material during a concert performance. Prerecorded audio- and video-playback material is colloquially called 'backing tracks'. The primary reason of backing track usage is to enhance a concert performance by play back additional material that artists on stage do not have the option or time to play themselves.

User (or artist) uploads synchronized audio files in 44.1 kHz 16-bit .WAV-format and video files in MPEG-4 format to Showman before using Showman during showtime. User creates, edits and organizes playlist with a software application Graphical User Interface (GUI) on laptop (PC or Mac). The content of the playlist is the backing track material that user produced beforehand. After playlist creation, user uploads the playlist via a USB-connection to Showman's internal memory.

Showman play back 8-tracks of audio along with video-content. The audio tracks can be connected directly into the XLR-inputs of a live sound mixer through Showman's XLR outputs, while the video-material can be connected to screens through Showman's HDMI-output. User starts each song in the playlist separately by pressing the 'Play'-button on Showman. The first song

on the playlist begins and playback stops automatically (auto-stop) after each ending of a song.

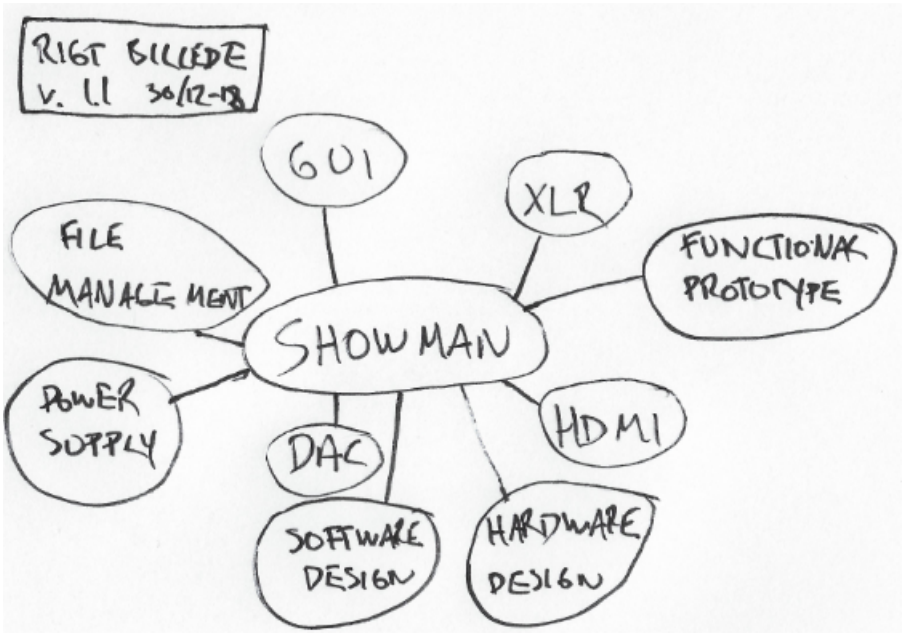

Figure 2.1 – Content of Showman as a rich picture.

Usage Situation

Showman's intended users are artists that utilize backing tracks at concert performances to enhance/augment their performance. Preconditions are essential to Showman, as user must have up to 4 stereo or 8 mono .WAV-files and a MPEG4-videofile available for each song that has backing tracks. The artist need to have produced the backing tracks beforehand.

As Showman's intended usage conditions are tough touring conditions and periodic hard handling, Showman's user interface is a 2U 19" rack mounted device with a LCD screen for user monitoring and panel buttons for navigation.

When the user have the necessary files available, the user assembles the playlist in the software application GUI and assigns the audio outputs. The GUI uploads the playlist project into specific folders in Showman's internal 500 GB flash memory.

When user need to start playback of a song, all user need to do is press 'Play' on the user panel. An auto-stop function after the end of each song in

the project is embedded in Showman.

Actor-Context Diagram

The figure below is an overview of actors that interact with Showman:

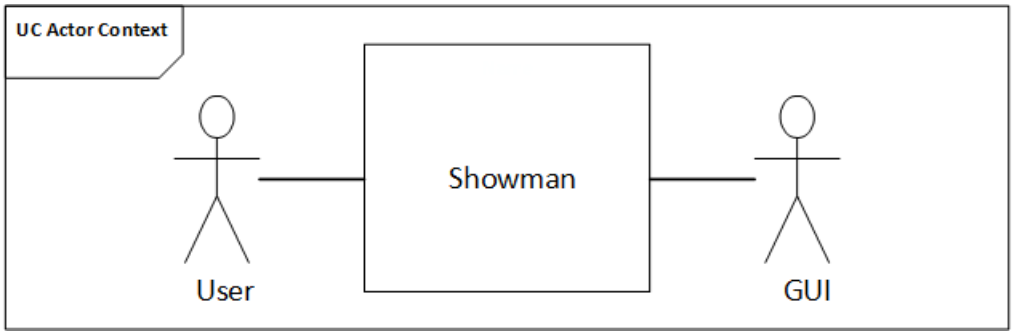

Figure 2.2 – Actor-Context diagram of Showman.

User

User is the primary actor. User creates the playlist and uploads the playlist to Showman. User operates Showman. User's interaction with Showman is outlined in detail in the specification for the individual Use Cases.

GUI

The Graphical User Interface (GUI) is the secondary actor. The GUI is the software application that handles the file management and playlist creation that is uploaded to Showman's interaction with Showman is outlined in detail in the specification for the individual Use Cases.

Functional Requirements

This section presents the functional requirements outlined for Showman. The figure below presents a Use-Case diagram that displays the actors' relations to the Use-Cases followed by Fully Dressed Use-Cases.

To keep this document brief and concise, only the first Use-Case is included. The additional Use-Cases will be explored further in the bachelor project.

Use-Case 1

Name:	Create Playlist
Scope	User wants to create a playlist
No. of concurrent events:	1
Primary Actor	User
Stakeholders	User wants to use backing tracks with Showman.
Preconditions	User has backing track files
Postconditions	User has uploaded files successfully
Main Success Scenario	1. User connects GUI to Showman via USB-connection. 2. User opens GUI. 3. User clicks on 'New Playlist'. 4. User sets desired playlist name. 5. GUI ask User to drag and drop files in Playlist Content panel. 6. GUI assist User in assigning output ports. 7. User saves playlist. 8. User clicks 'Showtime'. 9. GUI transfer playlist to Showman. 10. Showman is ready for use.
Extensions	[Extension 4a:] 1. GUI prompt displays 'Playlist already exist. Overwrite?'. 2. User prompts 'Cancel'. 3. GUI returns to Main Success Scenario step 4. [Extension 4b:] 1. GUI prompt displays 'Playlist already exist. Overwrite?'. 2. User prompts 'OK'. 3. User overwrites playlist. 4. GUI continues to Main Success Scenario 5. [Extension 5a:] 1. GUI displays 'INCORRECT FILE FORMAT'. 2. GUI returns to Main Success Scenario step 5. [Extension 5b:] 1. GUI displays 'The files exceeds memory limit. Please reduce file size.' 2. GUI returns to Main Success Scenario step 5. [Extension 6:] 1. GUI displays 'Too many files. Reduce number of files to accommodate number of available output ports. 2. GUI returns to Main Success Scenario 6.
Special Requirements	

Figure 2.3 – Use-Case diagram of Use-Case 1.

Non-Functional Requirements

Non-Functional Requirements (NFRs) are quality-demands, which are the qualities or constraints on the services of the functions offered by the system rather than a specific behaviour. Qualities are properties or characteristics of the system that its stakeholders care about and hence will affect their degree

of satisfaction with the system.
Quality demands/NFRs should satisfy two attributes:
- Must be verifiable - e.g. measurable metrics. - Should be objective.

The NFRs are drawn up using two techniques: FURPS and MoSCoW to determine system requirements and to prioritize and rank Non-Functional Requirements.

FURPS+

FURPS+ is an acronym that represents a model classify a product's properties: **F**unctionality, **U**sability, **R**eliability, **P**erformance, **S**upportability and + (Design and Physical constraints, Interfaces, Legal, Test, Reuse, Economic constraints, Aesthetics, Comprehensibility, Techology tradeoff, etc.)

A point-system from 1-5 is used to classify priorities, where 1 is the lowest and 5 the highest priority.

F - Functionality

The functionality describes the capability (size and generality of feature set), reusability (combability, interoperability, portability) and security (safety and exploitability) of a system.

Use-Case 1 - Create Playlist
UC1 has the highest priority as it is essential for Showman to create playlists with GUI in order to operate Showman.
Score: 5

U - Usability

Accessibility
Description of the ease of access to system
As the target group are touring artists and artist crew both with little to no time, the system need to be easy to use and operate with minimal learning curve.
Score: 4

User Interface
Description of the Graphical User Interface's functionality
The system need a GUI that need minimal learning curve, and therefore has to be similar to GUIs of software applications such as ProTools, Apple Logic

X, Ableton Live, Cymatic Audio uTool.
Score: 3

R - Reliability

Predictability
Description of the system's stability
This requirement is ranked highly in order for User to be reliant on that file-transfer does not malfunction nor malfunction does not occur during operation of the system.
Score: 4

Maintainability
Description of the system's maintainability
This requirement is also ranked highly in order for User to be able to acquire spare parts in case something breaks as the system will be in transit daily and is exposed to hard handling or manufacturer/producer updates firmware or software application.
Score: 4

P - Performance

GUI start-up time
Description of the software application GUI's accessibility when opening software application
Showman's GUI must be accessible max. 5 seconds after opening software application in order for User to create or edit a playlist and transfer playlist to Showman.
Score: 3

Showman start-up time
Description of Showman's User Interface accessibility when turning Showman on
Showman's UI must be available and accessible max. 5 seconds after turning the system on in order for User to play playlist.
Score: 3

Execution time - File transfer
Description of the fast the system can complete the given functionality described in the Use Cases
This property is not ranked highly, as long as the file transfer is finished within a few minutes. A research in file transfer speed is needed to determine file transfer speed.

Score: 2

Execution time - Play playlist

Description of how fast the system can complete the given functionality described in the Use Cases

This property is ranked in the middle. Showman must start playback within 5-10 microseconds of User pushing 'Play'-button. A research in push buttons is needed to implement and determine speed of play playlist execution time.

Score: 3

S - Supportability

Repairability - GUI

Description of GUI's level of repairability

The GUI's level of repairability is ranked low, as it is assumed that the User always have the latest software- and firmware-version installed on laptop and Showman.

Score: 1

Repairability - Showman

Description of Showman's level of repairability

Showman's level of repairability is ranked in the middle, as the goal is to create a system with as few as spare parts as possible except for a USB-cable for USB connection between GUI and Showman and a 230V 50-60Hz power cord for Showman. It is also assumed that Showman is racktmounted in a flight case designed for hard handled while in transit.

Score: 2

Compability - GUI

Description of the GUI's compability with other units

GUI's compability is ranked low as long as the User's laptop is compatible with GUI's technical requirements such as available hard disk space, RAM, CPU, etc. That is why the GUI is not required to be compatible with other units other than Showman.

Score: 2

Compability - Showman

Description of Showman's compaaility with other units

Showman's compability is ranked low as long as Showman is compatible with its corresponding GUI and has XLR-outputs for audio and HDMI-output for video. However, this project focuses on a **M**inimum **V**iable **P**roduct (MVP), that focuses solely on the necessary features for Showman to work. In a possible future scenario, a DANTE-protocol can be considered for implementation,

opting for digital audio connection via DANTE between Showman and audio mixing desk instead of using Showman's internal D/A converters and their analog XLR-outputs.
Score: 3

+ - Design constraints

The "+s" of the **FURPS+** acronym allows us to specify constraints, including design, implementation, interface, and physical constraints.

Legal

Data Protection
Showman's software application will be available for download from manufactorer's website when User has created a free account. User must fill out a form containing **name**, **username**, **contact info** (e-mail address, phone number, address, etc.) in order to sign up for an account. This user data will be regarded as confidential and must be treated and stored accordingly in compliance with the GDPR act introduced in May 2018.
Score: 4

Health and Safety

The final design and implementation of Showman must be compliant with EU rules on product safety, The General Product Safety Directive.
Score: 3

Test - Conditions

Stability is *essential* for Showman as that is how Showman differentiates itself from other key actors that produce and market similar products.
Stability for Showman includes a system that never stalls during playback and stable hardware that does not break down of wear and tear from hard handling.
Score: 5

Test - Environment

Showman will be exposed to hard handling during daily transit and changing weather conditions from humid air to dry air and temperatures below zero degrees Celsius to upwards of 55 degrees Celsius. It is therefore imperative that Showman is applicable during these hard conditions.
Score: 4

Reuse

It is the goal to produce Showman to have as few replaceable parts as possible because of the conditions and environment stated above. The only 2 replace-

able parts therefore need to be the USB-cable used for connection between GUI and Showman and a power cord for Showman. These two parts need to be widely available in most hardware- or electronics-retailers globally.
Score: 3

Enviromental - RoHS
RoHS stands for Restriction of Hazardous Substances. RoHS, also known as Directive 2002/95/EC, originated in the European Union and restricts the use of specific hazardous materials found in electrical and electronic products. All applicable products in the EU market after July 1, 2006 must pass RoHS compliance.
As Showman will be produced and marketed within the EU, Showman must be produced with respect to RoHS compliance. In practical terms, Showman need to be RoHS certified following a series of steps involved for RoHS certification:

Document Review: Review Bill of Materials, assembly drawings, Materials Declarations for each component and product, test reports and Conformance Certificates.

Audit: Inspect all manufacturing processes needed to meet RoHS compliance for the six restricted substances.

Testing: On-site portable XRF testing is done to determine values of the six restricted RoHS substances.

Certification: After successful audit, a RoHS certficate is issued.
Score: 4

Enviromental - WEEE
WEEE stands for **W**aste from **E**lectrical and **E**lectronic **E**quipment. WEEE Directive 2002/96/EC mandates the treatment, recovery and recycling of electric and electronic equipment (90 percent ends up in landfills). All applicable products in the EU market must pass WEEE compliance and carry the 'Wheelie Bin' sticker.
Score: 3

EMC
The Electromagnetic Compatibility (EMC) Directive 2014/30/EU ensures that electrical and electronic equipment does not generate, or is not affected by, electromagnetic disturbance.
The EMC Directive **limits electromagnetic emissions from equipment** in order to ensure that, when used as intended, such equipment does not disturb radio and telecommunication, as well as other equipment. The Directive **also**

governs the immunity of such equipment to interference and seeks to ensure that this equipment is not disturbed by radio emissions, when used as intended.
As Showman is exposed to electric devices or installations nearby such as GSM handsets, wireless communication devices such as wireless microphones, wireless In-Ear Monitoring (IEM) beltbacks, etc., it is a high priority to keep all those side effects under reasonable control.
Score: 4

MoSCoW

MoSCoW analysis is a prioritazion technique used in managemenst, business analysis, project management and software development to reach a common understanding with stakeholders on the importane they place on the delivery of each requirement.

M - Must have Requirements labeled as Must have are critical to the current delivery timebox in order for it to be a success. If even one Must have requirement is not included, the project delivery should be considered a failure.

S - Should have Requirements labeled as Should have are important but not necessary for delivery in the current delivery timebox. While Should have requirements can be as important as Must have, they are often not as time-critical or there may be another way to satisfy the requirement, so that it can be held back until a future delivery timebox.

C - Could have Requirements labeled as Could have are desirable but not necessary, and could improve user experience or customer satisfaction for little development cost. These will typically be included if time and resources permit.

W - Won't have/Would like Requirements labeled as Won't have have been agreed by stakeholders as the least-critical, lowest-payback items, or not appropriate at that time. As a result, Won't have requirements are not planned into the schedule for the next delivery timebox. Won't have requirements are either dropped or reconsidered for inclusion in a later timebox.

This section for requirement specifications is the *first draft*, so to keep this document concise, the **MoSCoW**-analysis is omitted as it needs a revision along with the rest of the requirement specification section.

Chapter 3

Project Plan

This section is an excerpt of the project plan. The project plan features descriptions of which experiments, technologies etc. to conduct, investigate and implement during the Bachelor's Project.

This section contains how the project work will be structured, which methods to be used, etc.

The Bachelor's Project

The Bachelor's Project (Thesis or Final Project) is an extensive project dealing with a realistic engineering assignment. The bachelor project should document the students' ability to work independently to apply engineering methods and theories in solving professional problems and development issues in a specific field. The project constitutes the conclusion of the programme and is placed in the last semester.

The Bachelor's Project workload is **20 ECTS points**. **1 ECTS** translates to **27 hours of workload**, totaling **540 hours** of workload for the Bachelor's Project. Assuming 18 weeks of Bachelor's Project work, total workload/week is 30 hours/week.

During the project period, 2 elective courses with 10 ECTS points in total to fulfill the goal of 30 ECTS workload each semester. Assuming 18 weeks of lectures and homework assignments and projects, total workload/week is 16-17 hours/week.

Total workload is therefore expected to be 45-50 hours/week during the semester. To keep track of the project, weekly SCRUM meetings with supervisor is essential along with project management tools:

- The ASE-model
- SCRUM
- Redmine

ASE-model

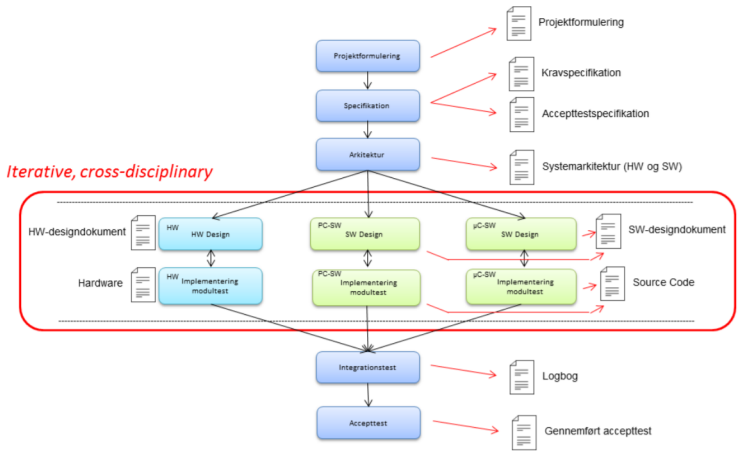

Figure 3.1 – The Aarhus School of Engineering model - The ASE-model.

The figure above is the ASE-model that engineering students at Aarhus School of Engineering (ASE) was introduced to during their 2nd semester. The ASE-model is a structured model for design and development of hardware- and software-systems.

A project of this magnitude contain and introduce several new technologies, with it many technical and project risks during the project work. Thus, an iterative project procedure is a helpful tool to ensure a clear procedure, as the beginning a new projects is usually difficult, necessitates management and overview to prevent wasting time on unnecessary activities.

The iterative, cross-disciplary model is useful when designing domain, technology, implementation possibilities, etc. Designing a system using the ASE-model reduces possible project risks and enables the option to specify requirements, formulate tests and design complete use-cases early in the project. Combining the ASE-model with an iterative process, the project process is robust from project risks and future modfications.

An iterative process produces functional product parts by completing tasks *piece by piece* (iterations) for each module of the project. Using this method, an early assessment of project success is possible along with an early risk pre-

vention assessment.

Every iteration contains elements from several processes outlined in the ASE-model. Hence, an iteration can contain both design, implementation and test for a minor part of the project.

Iterative development with the ASE-model

Even though a team chooses an iterative process, the project still has various phases as shown in Figure 3.1. The first iterations' focus is primarily a description of idea and establishing project requirements. The following iterations focuses on system development before moving on to implementation.
The difference between an iterative process and the ASE-model in Figure 3.1 is that we operate in *parallel* on several types of tasks within a given iteration. This is ongoing to better build, modify and refine the product on the basis of experiences and discoveries from earlier iterations.

For instance, the initial iterations outlines the first and the most important project requirements. When the reuirements are established, the problem domain is investigated: 'How can we implement GUI on a PC laptop?', 'How to measure Total Harmonic Distortion (THD) on audio output?, 'How do we establish USB-connection between GUI and Showman?', etc.
This iteration helps clarifying the thesis statement and allows the team the opportunities to conduct physical tests.

In each iteration, the team works on almost every project area. Documentation is produced during the process and needs to be regularly updated.

SCRUM

SCRUM is used to manage the iterative process. The reason is the welldocumented work procedures and team roles. Organizing the project by using SCRUM gives the project a clear overview and keeps the work flow going, even though external activities such as other courses can distract the process - In short, SCRUM makes the project manageable as the teams works toward small, but clear goals. Another reason is the widely accepted use of SCRUM in companies, giving the team necessary introduction and experience in using SCRUM.

The project team defines a framework with a set of **milestones** with mandatory assignments for each milestone, but gives the team the freedom

to reach these milestones.

Even though SCRUM is an integral part of project work at ASE, a detailed analysis and discussion of SCRUM is not within the scope of this document. The following is brief overview of selected elements of SCRUM.

Milestones

The table in Figure 3.2 below outlines the typical milestones for a project at ASE, weeknumbers relative to semesterstart:

Project Milestones	
Week 2	Group formation.
Week 3	Thesis statement hand-in to project supervisor.
Week 7	Review of requirement specifications, test specifications and system architecture (before beginning the construction sprints.).
Week 13	Hand-in of project artifacts.

Figure 3.2 – Example of SCRUM milestones.

Iterations

The ASE guide to development process for semesterproject 3 from Blackboard suggests following iterations for project completion. The first iterations primarily outlines the overview of the project. Later iterations outline hardware production and software-coding.

Iteration 1: Project Description (1 week)

The first project phase establishes clear overview of project idea, the most important requirements and the most significant technical risks:

Project Description: Project description outlines the project idea along with a rich picture. Goals corresponding requirements are set. MoSCoW-analysis is included to identify and prioritize system requirements.
Requirement Specification: Essential 'Must have'-requirements is selected for informal, brief use cases and their corresponding non-functional requirements.
System Architecture: Description of domain model based on Use Cases selected from the first iteration. A general risk assessment on system level is investigated. The assessment identifies technical risks critical for project: Unidentified interfaces, unidentified programming frameworks, unknown technology, etc.
Project Plan: A rough estimate of project plan, milestones and iterations.

Iteration 2: Identification and reduction of risks (2 weeks)

This iteration identifies risks and investigates the possibilites to reduce them by outlining the problem domain and which technologies and methods to include in the project work. Problem domain, architecture, code, hardware, algorithms, etc. are investigated to assess risks. This helps the next iteration of the requirement specification to be more precise and detailed.

System Architecture: This phase begins with an analysis of necessary hardware and software to reduce risks. The phase ends with a sketch of the system's general architecture such as block diagrams (BDDs), general internal block diagrams (IBDs) and identification of internal interfaces with sequence diagrams and system sequence diagrams. Initital application model for each CPU in the system is outlined based on the initial Use Cases. Application model in this iteration include a class diagram for each CPU for each CPU with boundary-, control- and domain classes.

Implementation and Test: Defining design and implementation of design with mock-ups, evaluation boards, virtual machines, emulators etc. to achieve the proof-of-concept and determine the chosen architecture's usefulness.

Requirement Specification: This phase finished by updating and clarification of requiremenents. Essential use cases are updated to fully dressed use cases and non-functional requirements i.e. external interfaces is clarified.

Project Plan: Is updated based on this phase.

Iteration 3: Early Implementation and Stabilizing Requirements (2 weeks)

This phase shift focus from establishing domains to design and implement a solution that meet the requirements based on use cases and non-functional requirements. The project is still in its early stage and many elements will most likely be modified during this iteration.

This edition of the system is the bare-bone version. This phase outlines a risk assessment, risk addressing and risk minimizing and stabilization of requirements.

Requirement Specification: This phase is conluded with a revision and clarification of initial and new requirements - They are apt to be relatively stable at this point.

System Architecture: This phase contains an analysis of the most necessary implementation(s). This results in an update of System Architecture with BDDs, IBDs, descriptions of internal interfaces and SW-architecture. Following this phase, the System Architecture is apt to be relatively stable.

HW/SW Design Specifications: The selected requirements are implemented in a design and undergoes a Module Test.

Project Plan: Is updated based on this phase.

Iteration 4-(N-1): Production of Project (2 weeks)

This phase focuses on production of the systen: Design, implementation and test to meet requirements. The requirements are selected and prioritized from theis business value.

This iteration lasts 2 weeks. First step is planning of assignments that are to be solved *completely* during the sprint: Task-specific system components is designed, implemented and module testes. This iteration integrates system components.

A clear-cut goal for this iteration is advantageous; it is a lot more satisfying and rewarding to have minimal, but functional GUI than to implement 10 percent of the complete, comprehensive system.

Requirement Specification: Is updated if necessary. They are usually minor updates at this stage of the project.

System Architecture: Is updated if HW/SW architecture and interfaces are modified. Architecture modifications and additions to the system components and their integrations are documented.

HW/SW Design Specification: Discoveries from previous helps modifying requirement design. Design modifications and additions to the system components and their integrations are documented.

Implementation and Test: The fabricated system components is implemented and tested - Both as isolated modules and system integrated modules.

Project Plan: No modifications of project plan unnecessary. The iterations are final; the *content* might be flexible, but *duration* is not modified.

Iteration 5: Project Hand-in (1-2 weeks)

This final iteration concludes implementation and project documentation. The product and documentation is now ready for hand in.

Requirement Specification: Is updated and completed.

System Architecture: Is updated and completed.

HW/SW Design Specification: Is updated and completed.

Implementation and Test: Implementation and test concludes. Project parts that did not meet deadline is removed from project.

Project- and Process Report: Is finalized.

SCRUM roles This project is a one-man project, thus an outline of SCRUM roles are redundant as they are all taken care of by one person.

Redmine

The online platform Redmine is typically used for SCRUM and project management at ASE projects to keep track of project events. Team member(s) can monitor tasks, where progress, 'burndown' for tasks is presented visually with Issues Burndown or Gantt diagrams.

As this project is a one-man team, the SCRUM roles are more fluid thus organized work flow and supervisor meetings are essential for the project to succeed.

Supervisor Meetings

Supervisor meetings are essential to this project as it is a one-man project. The supervisor will assist in keeping track of progress:
- Preliminary project plan.
- Gives feedback on detailed project plan.
- Organization and conduct ongoing supervision during the project to ensure project conclusion within given time frame.
- Gives continuous feedback on documentation content.

The supervisor is useful when conducting review- and retrospective meetings to track progress and reflections of sprints.

Timetable

The following Figure 3.3 outlines the time table for the project. They are subject to be modified during the first week of the process and after initial meeting with supervisor.

Sprint	Subject	Duration
1	Requirement Specification	2 weeks
2	Accept Test and System Architecture	3 weeks
3	Pre-Analysis	3 weeks
4	Design	2 weeks
5	Implementation	4 weeks
6	Module Test, Integration and Project Report	2 weeks

Figure 3.3 – A table of number of sprints and their duration.

Chapter 4

Project Research

Undersøgelse af tilsvarende projekter og relevant litteratur. This section investigates similar projects and products. Market analysis was conducted in the fall of 2018 to create an overview of the market. 70+ key figures from the Danish and international music business - musicians, recording producers, touring personnel, etc. - were interviewed for an investigation of their consumption patterns, gear run downs, their requests for useful features and the pitfalls of their gear.

The most experienced users are the so-called *backline technicians* (or backliners) that travel/tour with a touring artist or band to set up, tear down and maintain the artists' instruments and gear. In addition to setting up and tearing down the gear, the backline technician often operates electronics that run parts of the show such as backing tracks on some chosen device, MIDI changes, tune guitars/basses during the perfomance - In general to assist in making the artists' performance to run smoothly without unnecessay interruptions.

The most experienced backliners actively explore and research new technologies and products in between tours and/or during their spare time. Some even modify hardware and software to suit their clients' needs. And some backliners - usually on tours with larger production budgets - even design customized systems.

Similar Projects

Mircea Gabriel Eftemie, medialogy at Aalborg University.

Similar Products

Cymatic Audio LP-16

Ableton Live

QLab

MAINSTAGE

Logic Pro X

BlackBeagle

Arduino

Raspberry Pi

Linux Ubuntu Studio

Literature

Chapter 5

Project Expectations

After initial project proposal was accepted, an application for project working space was filed to secure an assigned working space at the university campus. As of this document's handin, no working space has been assigned yet.

Evt. aftale om forventet arbejdssted og tid.

Chapter 6

Conclusion

Konklusion på det indledende arbejde med forprojektet.

List of Figures