SOFTWARE ARCHITECTURE DOCUMENT

### **Version** 1.0

### PREPARED FOR

John Deere

### PREPARED BY

Apple Carplay Team

TU Kaiserslautern

[YOUR COMPANY’S LETTERHEAD]

EXECUTIVE SUMMARY

[150-600 word summary of the report that provides a high-level overview of the project]

|  |  |
| --- | --- |
| Signed as accepted by client: |  |

# 1. Introduction

[A detailed description of the project stating the aims, scope and intended operation]

#### 1.1. Requirements Overview

##### 1.1.1. The Goal

The goal of the Application is to help farm managers in making informed business decisions any time, from anywhere by providing users an ability for monitoring and documenting observations, and receiving work-related notifications at any time, outside the office. Connected to the car’s headset, this application is equipped with voice interactions to minimize the need for touch-based user’s intervention with the on-screen icons.

##### 1.1.2. Application context

Below the Applications possible usage scenarios are presented, linked by important features of it:

1. **Feature**: a map that displays the user’s location and marks the nearby fields with a coloring scheme.

**Usage**: the farm manager gets familiar with the Operation types happening in the fields. And on each field he/she can observe the icons that represent machine alert or another flag that occurred after their last visit.

1. **Feature**: a machine alert or flag in farm manager’s fields. Getting them by voice command and read them by Siri in an ordered way.

**Usage**: farm manager wants to be informed about problems happening with their technique. For that, they ask the Application for the machine alerts giving a simple voice command. Based on farm manager’s location, the Application will compose a list of alerts and flags from the nearest 3 fields, sort them in ascending order and read them. Here the alerts and flags read by Siri. Moreover all interactions between farm manager and the Application are carried out by Siri which represents the median between them.

1. **Feature**: interaction with the Application using shortcuts (shorter voice commands).

**Usage-1**: farm manager knows that the field to his right is his "Potato Field", and in order to make the conversation shorter they may even use a shorter voice command directly specifying the name of the field.

**Usage-2**: farm manager is driving near the "Sugar beet" and they want to hear about the progress of the seeding operation. So via a direct and simple voice interaction they can request those info.

1. **Feature**: taking (creating) general notes.

**Usage**: farm manager passes nearby their “Tomato” field, They notice a picture on the map, where brown spots on the leafs shown. Farm manager immediately calls the agronomist but then they would get the response from the answering machine telling them that the agronomist is on vacation and they would not be reachable before the day after tomorrow. So, farm manager tells the Application to create a general note “to call agronomist after 2 days”.

1. **Feature**: read notes saved by farm manager earlier.

**Usage**: After 2 days, while farm manager is on their way to visit fields again, they pass by his "Tomato" field, and notices an icon on the map that indicate that they have saved some reminder for that field. Farm manager asks the Application to tell the reminder, which would be about calling the agronomist that he has set 2 days before.

#### 1.2. Functional requirements

Below are listed functional requirements implemented in the application. For more information please refer to the Software Requirements Document.

Table 1. Functional requirements list

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Requirement** | **Rationale** | **Fit Criterion** |
| 1 | Application should be able to take input commands and give output using voice. | Driver should not be distracted by the use of application. | The devices in use should be compatible with Android/iOS voice recognition features. |
| 2 | Application should show user’s current location on a map | User should be able to see where he currently stands so he knows which direction he should head in | The application should support google maps/apple maps feature and shows the location zoomed in to the user |
| 3 | Application should update the user’s location as the car moves. | User should be able to see where he is heading, and where he is at that point in time | The application should support google maps/apple maps features and the map should update a location pointer as the car moves. |
|  | Application should show the area in the user vicinity on the map | User should be able to check which area he is in and plan as per the locations within vicinity | The application should support google maps/apple maps features and the map should be zoomed in such that the user’s location is centred and the area within the vicinity of XZY km should be visibly labelled. |
|  | Application should show custom field boundaries of the farms on the map | User should be able to instantly locate the fields over the basic map. | The application should support google maps/apple maps features and overlay feature libraries |
|  | Application should allow navigating to a custom field boundary | User should be equipped with the possibility of selecting fields he wants to drive to | The application should support google maps/apple maps features and overlay feature libraries. |
|  | Each field should have a proper name given to it that user can use to navigate to | It should be made easier for the user to point out the field he wants to drive to using verbal commands rather than selecting the area on map by touch | Proper naming conventions as per John Deere’s API should be followed |
|  | Application should allow the user to take notes verbally | User should not be distracted with typing notes while driving and hence minimize screen interaction by taking voice input | The car device in use should be compatible with Android/iOS voice recognition features. |
|  | Application should allow the user to save notes against a specific field | User may want to have the note for a specific farm area, so he can easily view his notes later categorized as per field |  |
|  | Application should allow the user to save general notes regardless of field. | User may want to have notes irrespective of field, like general reminders or points etc |  |
|  | Application should allow the user to discard the note | User may have created a note he doesn’t need or given wrong information that he does not want saved |  |
|  | Application should read out the notes to the user when asked for | User should not be distracted with reading the text off the screen while driving |  |
|  | Application should confirm if the note taken should be saved or not | User may want to discard a note in case its not correct, important or relevant |  |
|  | Application should ask which field note does the user want it to read | There could be multiple fields in the vicinity and user may want to read his notes for specific field only |  |
|  | Notes for a specific field should be saved with the field’s geo tag. | When user saves a notes for a particular field, the location tag is required to fetch field info |  |
|  | Application should read back the note to the user when taken | User should be allowed to ensure whether its correct or not |  |
|  | Application should only read back the note to the user before saving if its length is less than 140 characters | Reading long notes is not aesthetically pleasing, wastes times and delays the process of saving or discarding the note | Length of the notes should measure to less than 140 characters |
|  | Application should immediately save the notes locally on the device | Data loss should be avoided and notes should be readily available for later use | The devices in use has storage capacity. |
|  | Application should be able to detect if there are any fields within the user’s vicinity | User should be equipped with maximum relevant information regarding his surroundings, so he may schedule and plan things accordingly | The application should use John Deere’s API for the custom field maps and fetch data to see if there are any matching coordinates in the current vicinity |
|  | Application should be able to give information about the fields in the user’s vicinity | User does not need to go outside the car and inspect the field. | The John Deere’s API’s have all relevant data and is accessible by the application. |
|  | Information about the field and the machinary should be date specific | User should know how old the information is and has the situation changed over time or not |  |
|  | Application should provide the latest information about the field and machinery first | User would be more interest in knowing the latest updates rather older one |  |
|  | Application should be able to provide information about the machinery being used in the fields in the user’s vicinity | User does not need to go outside the car and inspect the machines.  User may want to schedule some tasks or take some notes as per the information he receives regarding the machines | The machine/Equipment information provided by the API in use is accurate. |
|  | Application should allow the user to ask if there is any important information regarding the field or the machinery | User should be allowed to check if there is any information rather than rely on guesses or assumptions |  |
|  | Application should provide the information field and/or the machinery only when asked for | User should not be flooded with information all the time, uninformed but should only have to know the information when desired |  |
|  | Application should colour the areas within custom field boundary as per priority of information about the field and the machinery within that field | User should be able to quickly pick the visual ques about which field needs more attention and hence prioritize the order of listening to information about the fields/machinery |  |
|  | Application should change the colour of the field area once the priority of information changes | The map should remain updated all the time so the user is aware of the current situation and remains updated at all times |  |
|  | Application should provide information about the nearby field/machinery through voice. | To be able to know about important information while driving the car without interacting with the mobile. | The application should be able to give information on the car device and read them out to the user |
|  | Application should allow dropping flags in custom field boundaries | User may want to drop a flag that’s related to a particular field |  |
|  | Application should show a flag in a field boundary every time someone creates a note |  |  |
|  | Users of the application can see flags pinned by other users. | User should be able to see if other users have found some information worthy of notifying |  |
|  | Application should show a tractor icon in the custom field boundary every time someone reports an issue with the machinery in that field. | User should easily be able to identify that  the nature of the problem pertains to machinery and may want to contact the operators in that field |  |
|  | | | |

#### 1.3. Stakeholders

Table 2. Stakeholders list

|  |  |  |
| --- | --- | --- |
| **No.** | **Role/Name** | **Expectations** |
| 1 | Requirements Engineer | Review the requirements overview part |
| 2 | Software Architect | Develop the software architecture and document it |
| 3 | Lead Developer | Prepare skeleton of the Application and distribute tasks. Consult Software Architects about technologies. Review the architecture document. |
| 4 | Software Testers | Prepare test-cases for white-box testing |
| 5 | Project Manager | Review the document and use for project planning accordingly |
| 6 | Client | Get familiar with the Application’s architecture and be able to enhance software if required |

#### 

# 2. Architecture Drivers

**Statuses used:** Open, To Do, Done, Rejected**.**

**Priorities:** High, Medium, Low.

Further the “Notes” notion is used for machine alerts, flags and general notes.

#### 2.1. Key functional requirements

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Voice interaction with the Application | |
| **ID** | AD.01.FUNCTIONAL COMPLETENESS | |
| **Status** | Done | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. Application is running in normal mode and connected to the car’s head-set. Internet is available | Previous logins >= 1;  Mode = Normal |
| **Stimulus** | User gives voice command to the Application |  |
| **Response** | Application responses with voice accordingly | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Showing map on the car’s head-set | |
| **ID** | AD.02.FUNCTIONAL COMPLETENESS | |
| **Status** | Done | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. Application is running in normal mode and connected to the car’s head-set. Internet is available | Previous logins >= 1;  Mode = Normal |
| **Stimulus** | User starts the Application and connects to the car’s head-set. |  |
| **Response** | Application shows map on the car’s head-set | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Mark fields on the map and show on the car’s head-set | |
| **ID** | AD.03.FUNCTIONAL COMPLETENESS | |
| **Status** | Done | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. Application is running in normal mode and connected to the car’s head-set. Internet is available | Previous logins >= 1;  Mode = Normal |
| **Stimulus** | User starts the Application and connects to the car’s head-set. |  |
| **Response** | Application shows map on the car’s head-set and marks fields with boundaries and colors them accordingly | Response time ≤ 1 sec |

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Internet Connection availability when login | |
| **ID** | AD.04.RELIABILITY | |
| **Status** | Done | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. Application is running in normal mode. Internet is available | Previous logins >= 0;  Mode = Normal |
| **Stimulus** | User logs in | Data size ≤ 1 KB |
| **Response** | User is transferred to the home screen | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Internet Connection availability when creating note | |
| **ID** | AD.05.RELIABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. Application is connected to Carplay. Internet connection is available | Previous starts >= 0;  Mode = Normal |
| **Stimulus** | User tells Application to create note | Data size ≤ 1 KB |
| **Response** | The data is locally saved and delivered to JD database | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Internet Connection is NOT available when creating note | |
| **ID** | AD.06.RELIABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. Application is connected to Carplay. Internet connection is NOT available | Previous starts >= 0;  Mode = Normal |
| **Stimulus** | User tells Application to create a note | Data size ≤ 1 KB |
| **Response** | The data is locally saved and delivered to JD database whenever Internet connection becomes available | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Internet Connection availability when receiving notes | |
| **ID** | AD.07.RELIABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed and connected to Carplay. Internet connection is available | Previous starts >= 0;  Mode = Normal |
| **Stimulus** | User tells Application to receive notes |  |
| **Response** | Application receives notifications and read by Siri for the User | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Data Correctness when creating note and Internet is available | |
| **ID** | AD.08.FUNCTIONAL SUITABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | User is logged in. Application is running normally. Internet connection is available | Previous starts >= 0;  Internet Availability = TRUE |
| **Stimulus** | User tells Application to create a note |  |
| **Response** | Data exchange happened using JD API. As a result data is correctly saved in JD DB |  |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Data Correctness when receiving notes and Internet is available | |
| **ID** | AD.09.FUNCTIONAL SUITABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | User is logged in. Application is running normally. Internet connection is available | Previous starts >= 0;  Internet Availability = TRUE |
| **Stimulus** | User tells Application to receive notes |  |
| **Response** | Data exchange using JD API happened. As a result User received latest data from JD DB |  |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Data Correctness when creating note and Internet is NOT available | |
| **ID** | AD.10.FUNCTIONAL SUITABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | User is logged in. Application is running normally. Internet connection is NOT available | Previous starts >= 0;  Internet Availability = TRUE |
| **Stimulus** | User tells Application to create a note |  |
| **Response** | Data is saved locally and marked “to be synced” as soon as Internet connection is there |  |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Geolocation Data is Complete | |
| **ID** | AD.11.FUNCTIONAL SUITABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | User is logged in and has enabled geolocation capabilities. Application is running normally | Previous starts >= 0;  Geolocation = TRUE |
| **Stimulus** | User tells Application to create a note |  |
| **Response** | The geolocation is tagged to the note successfully and normal execution resumes. | Coordinates accuracy < 5m |

##### 

#### 2.2. Quality Requirements

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Data Resource Utilization for upload | |
| **ID** | AD.12.PERFORMANCE EFFICIENCY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | User is logged in and has internet access. System is running normally | Previous starts >= 0;  Internet Availability = TRUE  **144Kb/s ≤** Internet connection bandwidth **≤ 1Mb/s** |
| **Stimulus** | User tells Application to create a note | Data size ≤ 1 KB |
| **Response** | All information is transferred successfully to the JD Database using JD API | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Data Resource Utilization for download | |
| **ID** | AD.13.PERFORMANCE EFFICIENCY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | User is logged in and has internet access. System is running normally | Previous starts >= 0;  Internet Availability = TRUE  **144Kb/s ≤** Internet connection bandwidth **≤ 1Mb/s** |
| **Stimulus** | User tells Application to receive notes | Data size ≤ 10 KB |
| **Response** | All information is received successfully from the JD Database | Response time ≤ 1 sec |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Product maintainability | |
| **ID** | AD.14.MAINTAINABILITY | |
| **Status** | Open | |
| **Priority** | Low | |
|  | **Description** | **Quantification** |
| **Environment** | Development Environment (XCode) |  |
| **Stimulus** | Development Activity |  |
| **Response** | Complete Architecture and Features should be developed in such a way to satisfy the following criteria:   * Modularity | For each User screen should have 1 View, 1 Delegate, 1 Controller classes |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Product maintainability | |
| **ID** | AD.15.MAINTAINABILITY | |
| **Status** | Open | |
| **Priority** | Low | |
|  | **Description** | **Quantification** |
| **Environment** | Development Environment (XCode) |  |
| **Stimulus** | Development Activity |  |
| **Response** | Complete Architecture and Features should be developed in such a way to satisfy the following criteria:   * Reusability | Generalization of classes should be used actively. For each action seperate method should be developed, so that it could be reused later. |

##### 

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Product maintainability | |
| **ID** | AD.16.MAINTAINABILITY | |
| **Status** | Open | |
| **Priority** | Low | |
|  | **Description** | **Quantification** |
| **Environment** | Development Environment (XCode) |  |
| **Stimulus** | Development Activity |  |
| **Response** | Complete Architecture and Features should be developed in such a way to satisfy the following criteria:   * Analysability | Each software component should be documented properly |

#### 2.3. Other constraints

# 3. Architecture Scope

#### 3.1. Business context

[Here put Level 1 System context diagram]

#### 3.2. Technical context

[Here put Level 1 Technical context, interaction with other systems]

# 4. Solution Strategy

[Here put Architecture decisions and Solutions]

# 5. Detailed Solutions

5.1. Component diagram (Level 2)

5.2. Class diagram (Level 3)

5.3. Deployment view

# 6. Risks and Technical Debts

[List of risks and/or technical debts, probably including suggested measures to minimize, mitigate or avoid risks or reduce technical debts.]

# 7. Glossary

[A list of software technologies that will be used in the development of the proposed software]