SOFTWARE ARCHITECTURE DOCUMENT

### **Version** 1.0

### PREPARED FOR

John Deer

### 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

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.

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 |  |
|  | | | |

#### 1.2. Architecture Drivers

##### 1.2.1. Key functional requirements

|  |  |  |
| --- | --- | --- |
| **Driver Name** | Internet Connection availability when login | |
| **ID** | AD.01.RELIABILITY | |
| **Status** | Open | |
| **Priority** | High | |
|  | **Description** | **Quantification** |
| **Environment** | Application is installed. System 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.02.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 Siri 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.03.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 Siri 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.04.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 Siri 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.05.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 Siri 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.06.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 Siri 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.07.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 Siri to create a note |  |
| **Response** | Data is saved locally and marked “to be synced” as soon as Internet connection is there |  |

##### 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Notes added shall be synchronized with JD Database. | The user can access or refer the notes in a later point of time. | The notes added by the user is stored in JD database for longer time periods. | Medium |
|  | When the data connectivity is limited, store the data in local device and upload when back online. | The user can use the product in locations with poor internet connection or of no connectivity. | The devices in use has storage capacity. | High |
|  | App should provide synchronization between Local Database and API | Changes made in Local Database should be reflected in API and vice versa | APIs and Database should be synced so that no information is lost or is redundant | High |

1. the first feature of our JD Driver: a map that displays the user location and marks the nearby fields with a coloring scheme that Arian is already familiar with (Reflecting the Operation type in the field). And on each field there will be icons that represent (machine alert, flag) that occurred after Arian's last visit.
2. the second feature of JD Driver, Arian sees a machine alert in his '' Field , and he wants to know what is it about , he asks JD Driver for the machine alerts giving a simple voice command. Here it is worth mentioning that this interaction between Arian and JD Driver is carried out by Siri which represents the median between him and JD Driver. Based on Arian's location JD driver will compose the list of the nearest 3 fields, sort them in an ascending order,
3. Arian can do the same to get the Flags on the nearby field, and since he is a skilled Manager, he knows that the field to his right is his "Potato Field", and he likes to make the conversation shorter so he may even use a shorter voice command directly specifying the name of the field6- Now Arian is driving nearby the "Sugar beet" and he wants to hear about the progress of the seeding operation. So via a direct and simple voice interaction he can request those info.
4. He passes nearby his/her “Tomato” field , He notices brown spots on the leafs!! He immediately calls the agronomist but then he would get the voice of the answering machine telling him that the agronomist is on a vacation and he wouldn't be reachable before the day after tomorrow.

feature of JD Driver , "Taking the notes"

• of creating a general note

• last feature in our app, After 2 days, While Arian is on his way to visit his fields again, he passes by his "Potato" field, he notices an icon on the map that indicate that he has some reminder for that field and he asks JD Driver to tell him the reminder. "which would be about calling the agronomist that he has set 2 days before".

• In the same way Arian can also ask JD Driver to tell him the general notes.

# 2. Obstacles

[A description of the possible risks involved with the project and how you will manage them]

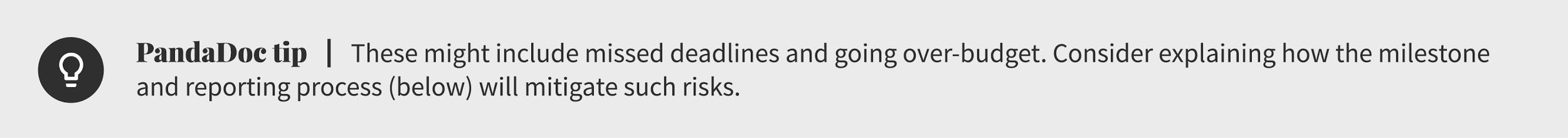
# 3. Technical Obstacles

[Any technical obstacles like integration between different systems, as well as mitigation strategies]

# 4. Industry and Market Risks

[Any industry or market-related risks]

# 5. Budgetary Risks

[Budgetary risks]

# 6. Hardware

[The hardware that the proposed software will be compatible with]

# 7. Software

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

# 8. Milestones and Reporting

### Total estimation of man hours: 226

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Milestone** | **Tasks** | **Reporting** | **Hrs** | **Date** |
| **1 - Analysis** | | | | |
| 1.1 | Analysis and design stage, gather data and create system mockup | None | 20 | 20/01/15 |
| 1.2 | Architecture design | None | 4 | 01/02/15 |
| 1.3 | Design work plan (distribution of tasks to development teams) | Client meeting to review work plan | 10 | 07/02/15 |
| **2 - Development** | | | | |
| 2.1 | Create database | None | 5 | 14/02/15 |
| 2.2 | Import existing client data | None | 5 | 21/02/15 |
| 2.3 | Clean data | None | 5 | 28/02/15 |
| 2.4 | Create GUI | Client meeting to review GUI | 30 | 01/04/15 |
| 2.5 | Integration with PaperlessOffice.net | None | 10 | 14/04/15 |
| 2.6 | Integration with smartphone network | Email report | 10 | 21/04/15 |
| **3 - Testing** | | | | |
| 3.1 | Alpha testing desktop application (Closed) | Email report | 25 | 07/05/15 |
| 3.2 | Alpha testing smartphone application (Closed) | None | 25 | 14/05/15 |
| 3.3 | Open Beta (volunteer employees) | Client meeting | 22 | 21/05/15 |
| 3.4 | Finalise documentation | None | 20 | 28/05/15 |
| **4 - Deployment** | | | | |
| 4.1 | Deployment to desktops | None | 5 | 01/06/15 |
| 4.2 | Deployment to smartphones | None | 10 | 07/06/15 |
| **5 - Training** | | | | |
| 5.1 | Inhouse training | Client meeting | 16 | 14/06/15 |
| 5.2 | AdHoc training | None | 4 | 30/06/15 |