decidim.org

(Version 0.9- March 1, 2018)

Master Test Plan

(Test Plan Identifier: DEC_V:0.4)

Xavier Arque Q.A.

Abstract

This document provides an overview of the project and the product test strategy, a list of testing deliverables and plan for development

Table of Contents

1. Introduction	4
1.1 Purpose	5
1.2 Test Objectives	5
1.3 Scope	5
2. System Overview	10
3. Test Requirements	12
4. Test Strategy	13
4.1 Test Levels	14
5 Estimates of Test Effort	15
6. Test Schedules	16
7. Entrance and Exit Criteria	17
8. Assumptions and Constraints	17
9. Risk assessments and contingencies	17
10. Test and Defects Management	18
11. Test Metrics	20
12. Responsibilities and Approvals	20
13. Supporting documents	21

1. Introduction

The product under test is decidim website. Decidim is a participatory democracy framework, written in Ruby on Rails, originally developed for the Barcelona City government online and offline participation website.

> "Decidim [https://decidim.org] is a digital infrastructure for participatory democracy written on free software (Ruby on Rails), which enables the creation of citizen participation portals. Users of the platform (participants) interact through participatory mechanisms known as components which provide specific functions for the various participatory spaces. In other words, participatory spaces such as Initiatives, Assemblies, Processes and Consultations have components at their disposal which work together as participatory mechanisms. The more notable components include in-person meetings, surveys, proposals, votes, results monitoring and comments. For example, the various stages of a specific participatory process, such as the preparation of participatory budgets, can incorporate various components to: convene and capture the results of in-person meetings with citizens, conduct a survey, make proposals and, during the final stages of the process, hold a vote to decide on the projects with a budget expenditure system (always after a period for commentary and debate), conduct an assessment survey and, finally, monitor the implementation of the projects selected." (Barandiaran, X.E. & Romero, C. (2017))[1]

> "Portals created with Decidim allow any organization (local city council, association, university, NGO, neighborhood or cooperative) to create mass processes for strategic planning, participatory budgeting, collaborative regulatory design, urban spaces and elections. It also enables the organization of in-person meetings, meeting invites, registrations, the publication of minutes, the structuring of government bodies or assemblies, the convening of consultations, referendums or channelling citizen or member initiatives to impact different decision making processes." (Cabot, J and Cànovas J.(2017))[2] .

1.1 Purpose

This document describes the master test plan and strategy for testing of the decidim.org platform. It will be used to document all aspects of the testing process and to obtain official input and approvals from the different people and teams involved in the project. Once approved, the document will be used for verifying that all aspects of testing have been completed and the platform decidim.org v.1.0 can be fully deployed (or at the extent that can be deployed functionally)

Some parts of this document -like the Estimates of Test Effort or the Schedules-, will be introduced

1 Barandiaran, X.E. & Romero, C. (2017) Funcionalidades y características de Decidim. v.1.0.
2 Cabot, J and Cànovas J.(2017) Page 11. Code Governance in Decidim December 22, 2017.

as a template or mere references to be fulfilled later on as the QA team grows and gets integrated into the development.

1.2 Test Objectives

Test Objective is the overall goal and achievement of the test execution. We can define the Test Objective of the project Decidim.org as following

- Check that whether website decidim.org **functionality** is working as expected without any error or bugs in real user environment finding as many software defects as possible and ensure that the software under test is **bug free** before release.
- Check that the external interface of the website such as **UI** is working as expected and meet the user needs.
- Verify the **usability** of the website. Are those functionalities convenient for user or not?

Nowadays the objectives of the decidim tests[3] are to verify that the product decidim.org meets the explicit and implicit specifications whether functional or non-functional detailed in the document "Decidim's functionalities and features. Roadmap 2017-2018 v1.0 04/09/2017" available at https://decidim.org/pdf/features-roadmap-en.pdf[4]

1.3 Scope

It is assumed that proper unit and integration testing have been conducted by the development team prior to delivering the code and that the unit testing will include 100% code coverage (lines of code and branch) and reasonable data coverage[5]. All critical defects must be fixed prior to code delivery and a list of any other existing and/or known defects should be send to the test team[6].

No DDBB testing will be specified. It is assumed that all Data Mapping has been checked. Like whether the fields in the UI/front-end forms are mapped consistently with the corresponding fields in the DB table as well as whenever certain actions are performed in the front end of an application, and corresponding CRUD (Create, Retrieve, Update and Delete) action gets invoked at the back end. There will be no ACID properties validation neither.

---

3 Specifically for these release, and due to a tight timeline and no QA team the objective of the next test will be to set up the Test Plan and Test Suits an a Smoke Testing.

4 Although there is a new release on its way. (https://decidim.org/blog/en/2018-02-06-release-0-9-0/) So far we will use this one until the next one is officially published

5 Page 30. - According to Code Governance in Decidim December 22, 2017. Jordi Cabot / Javier Canovas: CODE QUALITY The platform employs a number of code quality tools, namely: CIRCLE CI24 , as continuous integration and delivery platform; CODE CLIMATE25 , to perform code review for test coverage, complexity, duplication, security, and style; CODE COV26 for code coverage; CROWDIN27 , a localization project management platfom and translation tool; INCH CI28 , to assess the documentation quality; and ROCKET VALIDATOR29 , to validate the websites included in the platform.

6 Currently there is a list in https://github.com/decidim/decidim/issues

With the current QA team its impossible to do mobile tests before decidim.org V.1 release. But if the team increases or the release is delayed we will try to perform some manual testing and to set up a testing environment with Appium and an account in the Saucelabs Web for automated testing.

Testing levels to be completed should include:

Functional Test: Define and execute test cases to verify all functional requirements of the platform. There are three main users' profiles that have to be tested.

- Visitor. Somebody that just stops by the website and wants to get some information

- User. (registered members[7]) Somebody that has been registered and uses the Decidim platform

- Admin. (verified members[8]) Somebody that has to administer the Decidim platform. "The platform has been designed in such a way that processes, assemblies and mechanisms can be set up easily and deployed from an administration panel, where no knowledge of programming is required to install, configure and activate it." (Roadmap 2017-2018 v1.0 04/09/2017 page 8) therefore all admin test will be executed based on this premise.

The three profiles have been ordered according to its importance relative to decidim.org release 1.0 which is visitor, user, admin. That is to say, because of the public release v 1.0 we have to guarantee first that the platform can be explored and the visitor obtains information, second, that the platform can be used, third that the platform can be managed. Then, according to the document Decidim's functionalities and features. Roadmap 2017-2018 v1.0 04/09/2017 points 3 Components and 4 Participants (index page) the most critical functionalities for each profile have been listed in order to be tested.

Any user must be able to access the platform with the navigator and OS specified in (XXXXX.txt) and with the three default languages (Catalan, Spanish and English).

Visitor- Build 1.0

Once in the site he has to be able to change language, obtain information about decidim.org visit the main participatory spaces, activate the join the community option and access the sign-up page.

Therefore the most critical performance/areas/actions/measures to test for the visitor profile are:

1. Access to the platform
2. Access to the four/three main participatory spaces[9]

7 p.11- According to Code Governance in Decidim December 22, 2017. Jordi Cabot / Javier and Decidim's functionalities and features. Roadmap 2017-2018 v1.0 04/09/2017 page 8

8 p.11- According to Code Governance in Decidim December 22, 2017. Jordi Cabot / Javier and Decidim's functionalities and features. Roadmap 2017-2018 v1.0 04/09/2017 page 8

9 "Participatory Spaces: A space defines a participatory channel or media through which citizens or members of an organisation can process requests or coordinate proposals and make decisions. In Decidim we find four types of

3. Use language pop-down menu
4. Access the more Information menu
5. Home menu and return to Home Page
6. Sign-up option/menu
7. Join the community button

User (Registered members) - Build 1.0

A user must have access to the platform, be able to login, access personal account and use the interactive options in the participatory spaces

Therefore the most critical performance measures to test for the user profile are:

1. Access to the platform
2. Sign-in
3. Assert language selected is active
4. Use language pop-down menu
5. Access to the four/three main participatory spaces
6. Participatory options in the participatory spaces
7. account pop-down menu

Admin (Verified members) - Build 1.0

An admin must have access to the platform, be able to login, access personal account and use the interactive options in the participatory spaces

Therefore the most critical performance measures to test for the user profile are:

1. Access to the platform
2. Sign-in
3. Access to the four/three main participatory spaces as admin
4. Participatory options in the participatory spaces

spaces, namely: initiatives (e.g., a citizen initiative), processes (e.g., a participatory budget), assemblies (e.g., a workshop to discuss a proposal) and consultations (e.g., a call for voting for/against a proposal).” Page 11 Decidim's functionalities and features. Roadmap 2017-2018 v1.0 04/09/2017 pag 8

5. Account pop-down menu
6. Access to the Dashboard Menu
7. Dashboard menus

Although some Test Cases will be the same for each different profiles, each one may have different priorities. Decidim's functionalities and features. Roadmap 2017-2018 v.1.0 04/09/2017. Page 8 contains more detailed information about those profiles and the functionalities of each one.

Proposals - Build 1.0

Being proposals Decidim's most important component[10], all its functionalities will be tested in the Smoke Testing (see 4.Test Strategy)[11].

Caveats

The Test Team is very small so we will begin with the functional tests, rate the importance of each feature/functionality (Levels: Critical = 1, High = 2, Medium = 3, Low = 4) and begin a Smoke Testing with the critical ones for each profile.

Once the functional test work-frame for the three profiles and the proposals screen is already setup, then we will move to the next functional tests and later on, the other tests:

User Interface Test: Verify the usability of menus and ease of navigation between screens. Verify that usability standards for GUI product are met. Verify ease of navigation conforms to decidim.org standards.[12]

System Test: Define and execute a set of typical sequence of functions (at least 5 scenarios to include all the main functionality of the system) that end user would execute on daily basis. Verify response time of login and menu navigation. Also execute the customer scenarios to be used in the user acceptance test.

User Acceptance Test: Performed by 3-5 end users using a set of agreed on scenarios.

---

10 A component defines a mechanism that enables a specific interaction between users and spaces. In Decidim we currently find sixteen types of components, namely: meetings , conferences , conferences , incubators , proposals , participatory texts , surveys , discussions and debates ,results , monitoring , votes , pages , blogs , observations , newsletters and search engines .

11 More information about its keys features in page 12-13 Decidim's functionalities and features. Roadmap 2017-2018 v.1.0 04/09/2017.

12 For instance: "The System shall be easy-to-use and shall be appropriate for the target of computer-literate citizens."- "The desktop user-interface shall be Windows XX, OS XX Linux XX and Android XX compliant."- "The user interface of the Registration System and the participatory components shall be designed for ease-of-use and shall be appropriate for a computer-literate user community with no additional training on the System."- "Each feature of the platform shall have built-in online help for the user. Online Help shall include step by step instructions on using it. Online Help shall include definitions for terms and acronyms."

Performance, Load Testing and Stress Testing: They strongly rely on client provider so it doesn't make sense to verify or test them.

All Security related tests and issues will not be specify in this test plan. (All this information has to be specify in a document Security&Storage-DecidimSpect.doc to be created)

Details on each test will be determined later as Test Suites and Test Cases are identified and Test Procedures developed. All Test Cases and Test Suites will be detailed in Test-Link

2. System Overview

The decidim platform is a GUI based website with menu driven interfaces[13]. A visitor may navigate through out most of the web site but in order to use the platform end user first needs to register (login credentials and related security issues will be covered under separate document – Security&Storage-DecidimSpect.doc).

Home page

The home screen is the first screen of the platform and it provides information and access to the main menus, languages and register/sign-in menu but a user will not be able to do some things without first registering and login into the system using an e-mail and password[14].

The user may login as an administrator and get access to the Dashboard and the “Welcome to the Decidim Admin Panel” to manage the website.

Participants functionalities[15]

Registration and verification [some functions expected for 2018Q1-3, AjB-Lote1].

Public registration of administrative activity

User profile and personal settings [some functions expected for 2018Q1-3, AjB-Lote1].

Monitoring and notifications

Communication among participants [some functions expected for 2018Q1-3, AjB-Lote1].

Engagement [some functions expected for 2018Q1-3, AjB-Lote1].

User groups [some functions expected for 2018Q1-3, AjB-Lote1].

13 The technical specifications which outline decidim's new functions can be found and downloaded from the following link: https://contractaciopublica.gencat.cat/ecofin_pscp/AppJava/ca_ES/notice.pscp?idDoc=24253271&reqCode=viewCn

14 The system should have to issue error message if no credentials are entered or incorrect ones are used, although so far it doesn't do it.

15 For a more detailed information about functionalities and expected functions refer to page 18 Decidim's functionalities and features. Roadmap 2017-2018 v.1.0 04/09/2017.

Participatory spaces

- Processes is a space that permits a user to create, activate/deactivate and manage various participatory processes, linking them together at different stages in which all of the components can be incorporated.

- Assemblies is a space which offers a user the possibility of defining bodies or groups that meet up periodically, detailing their composition, listing their meetings with geo-location and taking part in them (attending if the seating capacity and nature of the assembly so permits, adding items to the agenda or commenting on the proposals and decisions taken by that body).

- Initiatives is a space that allows a user to collaboratively create initiatives, define their trajectory and aims, gather support , discuss, debate and disseminate initiatives and define meeting points where signatures can be collected from attendees.

- Consultations is a space that makes it possible to coordinate referendums , trigger discussions and debates, and is connected to a secure e-voting system where results can be published. Function expected for 2018Q2, AjB]

Components.

As stated before, the proposal- incubator component is the only one that will be included in the Smoke Testing. The proposals component allows a user to create an official proposal with geo-location tracking, upload documents and browse and filter proposals. Not all the Propossal functions are expected to be functional prior to release 1, for instance Proposal incubator function is expected for 2018Q1. Those functions will not be included in the functional tests r.1[16]

The other main components[17] are:

- Participatory texts [function expected for 2018Q2, AjB-Lote3Mod4]:
- Results
- Monitoring results
- Votes and/or endorsements
- Comments
- Informative pages
- Discussions and debates
- Surveys

16 For a more detailed information about functionalities and expected functions refer to page 12 Point 3-Components. Decidim's functionalities and features. Roadmap 2017-2018 v.1.0 04/09/2017.

17 The components (participatory components) that are integrated are independent from each other and can be developed, activated and deactivated independently.

- In-person meetings
- Conferences [function expected for 2018Q2, AjB-Lote2Mod5]
- Blogs
- General and targeted newsletter
- Search engine

3. Test Requirements (soft&hard)

Credential interface (mocked or actual system) for login support.

iMac with Mac OS El Capitan v. 10.11.6

Web browsers: FireFox Quantum 57.0 and Google Chrome Versió 61.0.3163.100 (although so far no versions have been defined)

Test-link as web-based test management system for Test Cases, Test Suites, Test Plans, test projects and user management, as well as various reports and statistics.

Automated Testing: Java Runtime Environment (JRE) assuming Java implementation. Software IntelliJ Idea + Maven + TestNG or Junit

GIThub o Bitbucket as a web-based hosting service for version control using git

The meta-decidim platform will be used for bug reporting but if it is not available then Mantis Bug Tracker, a free and open source web-based bug tracking system, will be used.

Once the testing in the IOS environment is done, a virtual machine with Windows 7 will be set up for Windows OS testing purposes and later on for mobil:

- Android emulator, OS 5.1, Appium 1.6.4 → Google Chrome
- iPhone6 , OS iOS, 10.0, Appium 1.7.1 → Safari

Other Dependencies

None

4. Test Strategy

Due to the lack of a QA team and the closer deadline, we will perform an urgent Smoke Testing (“Build Verification Testing”). This is a type of software testing that comprises of a non-exhaustive set of tests that aim at ensuring that the most important functions work. The results of this testing

will be used to decide if a build is stable enough to proceed with further testing. Once we complete smoke testing then only we will start a deeper functional testing.

4.1 Test Levels

Functional Test (FT):

This type of testing is based upon black box techniques, that is, verifying the application by interacting with the application via the GUI and analyzing the output (results).

Test objective: There will be a test case defined for each menu and requirements to include valid and invalid inputs and data rated Critical. Verify expected behavior for valid data and appropriate error message on invalid data. Make sure system does not crash or have any other unexpected behavior. Some special Components will be checked as well.

Deliverables: FT will document all test cases (both manual and automated), document execution findings will open defects, and test fixes.

Milestones: The following milestones will be tracked:

•Test cases defined – all test cases are defined, reviewed, and approved, ready to be executed

• Start execution – test system is ready and software is installed ready for testing (to include test credentials)

• 100% test cases executed – all defined test cases have been executed

• Regression test complete – all L1 and L2 defects have been fixed and regression test have been run

• Exit FT – All function test tasks are complete

User Interface Test (UI):

User Interface testing verifies a user's interaction with the software. The goal of UI Testing is to ensure that the User Interface provides the user with the appropriate access and navigation through the functions of the applications. In addition, UI Testing ensures that the objects within the UI function as expected and conform to corporate or industry standards. ROCKET- VALIDATOR[18], will be used to validate the website and A11y[19].

Test objective: Verify navigation between screens meets requirements and standards Verify the

18 https://rocketvalidator.com/s/2c6ce461-b007-4071-85d7-75e49bc768e8/v

19 "A11y" is a numeronym presenting "accessibility" as "a" followed by 11 more letters, followed by "y".

size, colors, and position of the elements meets the design and requirements.

Deliverables: Document test cases and findings; open and test defects

Milestones: The following milestones will be tracked:

• Test cases defined – all test cases are defined, reviewed, and approved, ready to be executed

• Start execution – test system is ready and software is installed ready for testing (to include test credentials)

• 100% test cases executed – all defined test cases have been executed

• Exit – All test tasks are complete and L1 and L2 test cases are fixed and closed

System Test (ST):

System testing is the type of testing to check the behavior of a complete and fully integrated software product based on the software requirements specification (SRS) document. The main focus of this testing is to evaluate End-user requirements.

Test Objectives: Verify real-life scenarios (sequence of events) that typical users will use the application for and make sure that both valid and invalid inputs work as expected. Verify response time of login and menu navigation meet the standards. Verify the customer scenarios to be used in the user acceptance test are working as expected.

Deliverables: Document test scenarios and document findings; open and test defects

Milestones: The following milestones will be tracked:

• Test scenarios defined – all test scenarios are defined, reviewed, and approved, ready to be executed

• Start execution – test system is ready and software is installed ready for testing (to include test credentials)

• 100% test cases executed – all defined test scenarios have been executed

• 100% acceptance scenarios executed

• Exit – All function test tasks are complete and L1 and L2 test cases are fixed and closed

Test deliverables are provided **after** the testing cycles is over.

- **Test Results/reports**
- Defect Report
- Installation/ Test procedures guidelines

- **Release notes**

5 Estimates of Test Effort

Test Level	Task	Estimate	Comments
Function test	Define and automate tests	30PD[20]	There will be a mixture of manual and automated tests
	Execute tests	10D	
	Execute regression	2D	Run a subset of automated tests as a regression to make sure defect fixes did not break anything
Usability test	Define tests	10D	All screens, menu, selections, and components have to be tested
	Execute tests	5D	
System Test	Define scenarios	15D	Define and get approval from user to make sure they reflect real-life operations
	Execute tests	5D	
All levels	Admin tasks	40D	Drafting documents, meetings, test system setup and maintenance, approvals, defects management, etc.
Total		**117D**	

6. Test Schedules

Its impossible to set up a schedule because so far all tests will be done when the QA have some spare time from their other jobs. The table it's only for a reference[21].

20 PD stands for Person Day

21 Notes:
1. Function test can start defining tests before unit test is complete
2. Acceptance scenarios need to be defined before system test starts executing but they need to be verified and tweaked as needed once system test is completed and critical and severe defects are fixed.
3. After final test is executed, there is a buffer to complete processing the rest of the defects.

Level	Task	Start	End	Comments
Unit Test	**Execute and fix defects**			**L1 and L2 defects fixed**
Test Start				**Start defining tests/scenarios**
Function test	Define and automate tests			Testers at 50%
	Execute tests			Testers run tests and open defects
	Execute regression			tester run tests and open defects
Development	**Design User Interface**			**Pre-req for Usability Test**
Usability test	Define tests			tester
	Execute tests			tester run tests and open defects (multiple testers running same scenarios)
System Test	Define scenarios			2 testers
	Execute tests			tester run tests and open defects
Test Exit				**All defects fixed**

7. Entrance and Exit Criteria

Function Test:

Entry: Function test execution will start, for each system functionality, once unit test is 100% complete and all L1 and L2 defects (critical and severe) are fixed.

Exit: All planned test cases have been executed. All defects have been assigned to development. L1 and L2 defects have been fixed. L3 and L4 defects have agreed on course of action.

User Interface Test:

Entry: Test execution will not start until all_interfaces for the product are complete and unit tested. All L1 and L2 defects must be fixed.

Exit: Each screen navigation and behavior works as expected and meets GUI and decidim.org standards. All L1 and L2 defects are fixed.

System Test:

Entry: Test execution can be staged per testing scenario. All functionality for the scenario must be 100% function test complete with L1 and L2 defects fixed and 90% of L3 and L4 defects fixed.

Exit: All defined scenarios have been executed and L1-L3 defects are fixed.

8. Assumptions and Constraints

8.1 Assumptions

There is an Internet connection and a website available at http://staging.decidim.codegram.com/

8.2 Constraints

Some of the testing maybe will be done with mocked up code, specifically for credentials and final user interfaces until the actual code is available. Testing will only be done on Windows and MAC systems.

9. Risk assessments and contingencies

Item	Impact	Risk	Contingency Plan
Development Delivery Late	Medium	High	Track progress closely and plan to test in phases as needed
Function Test End Late	High	High	Track progress closely and prioritize tests completion; Start ST in phases as needed
High FT Defect rate	High	Medium	2 additional developers can be dedicated full time to

			fix as needed
User Interface severe defects	High	Low	Customer to review and approve design; FT to open defects when see any issues; 2 developers can be dedicated full time to fix as needed
System Test End Late	Medium	Medium	Track progress and add tester from FT team as needed to complete on time
High ST Defect rate	High	Medium	2 additional developers can be dedicated full time to fix as needed
User Acceptance severe defects	High	Low	Tests will be defined early and approved by user; System test will test first; 2 developers can be dedicated full time to fix as needed
System environment issue	High	Low	Dedicated tester setting up; Can add resource as needed
Login and credentials not complete on time	High	Medium	Test with mocked up code and then run regression once real code available

Legend: FT: Function Test – ST: System Test – UIT: User Interface Test

10. Test and Defects Management

Currently there is already a protocol implemented in-between Github and the Meta-decidim platform to deal with this issues.

Information about bug fixing in Github can be found here:

https://github.com/decidim/decidim/issues

Information about bug fixing in Metadecidim can be found here:

More information in:

https://meta.decidim.barcelona/processes/bug-report?locale=en

To report bugs:

https://meta.decidim.barcelona/processes/bug-report/f/210/

The latest release changelog with a list of new features:

https://decidim.org/blog/en/2018-02-06-release-0-9-0/

If finally there is a QA team. Defects will be opened by individual testers and assigned to the development lead. Development lead will review and either return (e.g. ask for more information) or assign to a developer. Developer will work with tester to debug the issue and will fix the issue. Once code is integrated into current driver, the defect will be marked as fixed and assigned to the tester. Tester will rerun the test that failed and either close the defect or reopen it to be assigned to development (e.g. fix did not work or there is a regression).

The actual terminologies, and their meaning, can vary depending on people or projects but the following is a normally accepted classification.

Defects will be classified as follows:

Severity	Classification Criteria Based on Impact to the User or System
1 (Critical)	Issue can cause system crash, loss of data, hang, or data corruption. Example: Unsuccessful installation, complete failure of a feature.
2 (Severe/Major)	Issue causes incorrect or unexpected behavior for a critical functionality. The defect affects major functionality or major data. A feature is not functional from one module but the task is doable if X complicated indirect steps are followed in another module/s.
3 (Medium/Minor)	Issue causes incorrect behavior or non-critical data but there is a workaround. Example: A minor feature that is not functional in one module but the same task is easily doable from another module.
4 (Low/Trivial)	There is an issue with a message or other issue with small impact on overall functionality. It does not impact productivity or efficiency. It is merely an inconvenience. Example: Petty layout discrepancies, spelling/grammatical errors.

11. Test Metrics

Status and charts

If we can set up an automated way of doing it, -probably with Test-link- it will be convenient to provide weekly reports and graphs that include:

- Total number of opened defects with rate of creation

- Number of currently opened defects grouped by severity
- Number of blocking defects (and their defect number)
- Number of fixed defects
- Number of defects currently assigned to development
- Number of defects currently assigned to test

The following data and metrics should be calculated periodically and presented at status meetings:

- Number of defined test cases per test level
- Number of defects found per test level
- Total number of defects found
- Number of defects found per functionality (Function test)
- Number of defects per severity (Function test)
- Total number of defects per severity
- Rate of defect arrival and rate of defect removal
- Percentage of executed test cases per test level
- Percentage of total executed test cases
- Number of blocking defects and how many tests they are blocking

12. Responsibilities & Approvals

Role	Name	Responsibility	Approvals
Project Manager		Overall responsibility for quality and due dates	Approve Master Test Plan
Product Owner		Overall responsibility for design, dev, and test execution	Approve Master Test Plan
Dev Lead		Dev schedule and	Approve Master Test Plan

	delivery	
Test Manager	Overall test effort and dates	Approve Master Test Plan
Test Lead	Test quality and execution	Approve Master Test Plan
Customer tester		Approve dates related to acceptance test and final delivery dates

13. References and supporting documents

- Barandiaran, X.E. and Romero, C. (2017) Decidim's functionalities and features. Roadmap 2017-2018 v.1.0 04/09/2017.

- Cabot,Jordi and Canovas, Javier (22-XII-2017) Code Governance in Decidim

- PLEC DE PRESCRIPCIONS TÈCNIQUES PER A LA CONTRACTACIÓ DELS SERVEIS DE CONSTRUCCIÓ I IMPLANTACIÓ DE LA PLATAFORMA DECIDIM.BARCELONA AMB INCORPORACIÓ D'OBJECTIUS D'EFICIÈNCIA SOCIAL . Institut Municipal d'Informàtica Direcció d'Estratègia i Nous Projectes