# Project CASA: Project Plan

## Project information

Project CASA is part of a larger ongoing academic project, information on it can be found here:  
[**Quality Assurance and testing methodology for IoT**](https://www.researchgate.net/project/Quality-Assurance-and-testing-methodology-for-IoT?_esc=profileProjectCards&_sg=GNWc9Zh9M-1M952KIYE0MAN8P-qDywz96oYzkqZ4L8cFxtHJBASyTZFBFp2F04yCBz_uj45oHhr5IqmsBZkEdw)

Since the main deliverable of the project is having working code at the end, we have opted for agile project management, which means less strict rotation of project roles with more iterations between shorter development phases.

The main use case is to recreate the original CASA program 1 to 1 in relation to structure. Some parts of the program, namely the “sat” and “minisat” packages might be online in Java already, which will save time.

## Requirements

### Initial Requirements

1. Write new program in Java with same functionality as CASA.
   1. Create covering array with test cases created by pairwise testing method.
   2. Using constraints that affects calculation of covering array.
   3. Set the seed value for the random number generator.
   4. Set the initial number of iterations allowed at each array size.
   5. Set the number of retries allowed at the same array size.
   6. Set the weight of the upper bound in the binary search partition.
   7. Set the initial temperature.
   8. Set the temperature multiplier applied each iteration.
   9. Let the covering array be no smaller than the given size.
   10. Let the covering array be no larger than the given size.
   11. Lock the covering array at the given size.
2. Keep the CASA solving speed.

### Revised Requirement

1. Write new program in Java with same functionality as CASA.
   1. Create covering array with test cases created by pairwise testing method.
   2. Using constraints that affects calculation of covering array.

### Operational project requirements

1. Find SAT and minisat module in Java.
2. Find a way to use code in C language from Java.
3. Compare Choco solver with SAT solver in Java.

## Use case

### Generate test cases

The use case of the program is to generate covering array of test cases. There are few parameters.

* (Required) Model file with
  + strength of testing
  + number of options
  + number of values in each option
* (Optional) Constraint file with
  + Number of disjunctive clauses
  + Number of terms in each disjunctive clause
  + Terms itself from each clause
* (Optional) Name of the Output file

## Task overview

### Analysis

#### Running C code in Java

### Business requirements specification

#### Testing strategy

### Coding

#### Search

#### Space

#### State

#### Utility

#### posix

#### Cost

#### Filter

#### Heuristic

#### Report

#### Goal

#### IO

#### Bookkeeping

#### Events

## Task specification

TBD

## Overall project schedule



## Development plan

