Software Requirements Specification

for

Improved Garbanzo

Version 1.0 approved

Prepared by Norbert Szulc

Gdańsk University of Technology

2018-11-29

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Norbert Szulc	2018-11-29	Created	1.0

1. Introduction

1.1 Purpose

Document defines basic requirements for final project. It describes all project features and outlines implementation schedule for final evaluation.

1.2 Document Conventions

Feature priority is based tag:
REQ: required to progess.
OPT: optional, shouldn't have negative influence on project success if not completed.

1.3 Product Scope

Improved Garbanzo is toy project testing viability of Julia Programming Language for solving N-body problems using numerical integration.

1.4 References

- N-body simulations (gravitational) Michele Trenti and Piet Hut (2008), Scholarpedia, 3(5):3930.
- Moving Stars Around by Piet Hut and Jun Makino – http://www.artcompsci.org/
- pynbody – Python package 2013ascl.soft05002P https://github.com/pynbody/pynbody
- Julia Programming Language – https://julialang.org/

2. Overall Description

2.1 Product Perspective

Improved Garbanzo is final project for programming course at GUT. It's creation is motivated by learning Julia Programming Language and numerical methods for simulating gravitational influences in star clusters and smaller systems.

2.2 Product Functions

Main function of Improved Garbanzo is integration over time of forces acting in n-body systems. All minor functionalities like data IO and visualization will be provided by a well defined API.

2.3 User Classes and Characteristics

Course instructors – will use this documentation and final product for grading.
Me – playing around.

2.4 Operating Environment

Software will be support Linux Mint 19 and Julia 1.0.2 with personal grade hardware. No further support is planned.

2.5 Design and Implementation Constraints

Improved Garbanzo development will be limited in time by programming course, giving around 3 months to create final product. Another limiting factor is small ecosystem around Julia Programming Language. Lack of experience in both Julia and numerical integration will add necessary research time for project owner.

2.6 User Documentation

- Documentation in code.
- Example Jupyter Notebooks as tutorials
- Generated reference. (not required, options will be tested)

2.7 Assumptions and Dependencies

Dependent on Julia Programming Language and community created modules:
- Plots.jl – Visualisation
- Ijulia.jl – Jupyter Notebook kernel
- TBD

3. External Interface Requirements

3.1 User Interfaces

Improved Garbanzo won’t provide any GUI. There is possibility for CLI (TBD).
User should be able to visualize basic simulation with provided API.
User can improve experience by using Jupyter Notebooks.

3.2 Hardware Interfaces

Doesn’t apply.

3.3 Software Interfaces

Improved Garbanzo should leverage existing Julia type system. It’s simplest way to provide API working with other Julia modules without significant data transformations.
Simulation engine will consume vectors of position and speed of particles. During simulation it should emit current particle states as vectors.
Programming API TBD.

3.4 Communications Interfaces

Doesn’t apply.

4. System Features

4.1 Basic Simulation

4.1.1 Description and Priority

High priority. Program should run short simulation for specified number of particles with random states. After execution it should output numerical data.

4.1.2 Functional Requirements

REQ-1:	Simulate system by one step with specified time
REQ-2:	Simulate in a loop
REQ-3:	Simulate with specified integration method

4.2 Euler method

4.2.1 Description and Priority

High priority. The most basic integration. Simple but error accumulation needs to be accounted for. $O(N^2)$

4.2.2 Functional Requirements

REQ-4: Integrate system by specified time using Euler method

4.3 Runge-Kutta method

4.3.1 Description and Priority

Medium priority. Exact method TBD. $O(N^2)$

4.3.2 Functional Requirements

REQ-5: Integrate system by specified time using Runge-Kutta method

4.4 Tree code method

4.4.1 Description and Priority

Low priority (optional). Fast method for collisionless systems. O(Nlog(N))

4.4.2 Functional Requirements

OPT-1: Integrate system by specified time using Tree code method

4.5 Visualization

4.5.1 Description and Priority

Medium priority. Basic visualisation and animation utilities.

4.5.2 Functional Requirements

REQ-7: Plot distribution of particles in a system.
REQ-8: Plot trajectories of particles in a system during simulation.
OPT-2: Animate trajectories over time. TBD

4.6 Command Line Interface

4.6.1 Description and Priority

Low priority. Set of CLI utilites to make work with program easier

4.6.2 Functional Requitrements

REQ-9 Run simulation with data supplied as command arguments
OPT-3 Animate trajectories with data supplied as command arguments

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Performance of basic simulation can be 2x-5x slower than a state-of-the-art program. Bigger performance are expected as Julia Interpreter needs some optimizations and will be resolved on a case basis.

5.2 Safety Requirements

Doesn’t apply.

5.3 Security Requirements

Doesn’t apply.

5.4 Software Quality Attributes

Test suite should be provided for at least integration logic. Modularity of integration methods should allow for interchangability of given methods without modification of simulation logic.

5.5 Business Rules

Doesn’t apply

6. Other Requirements

Appendix A: Glossary

Program – software package specified by this documentation
User – person using software
Particle – point mass, smallest object of simulation
System – ensemble of particles
Integration – process of integration of diffrential formulas over time
Simulation – continous process of integration on a system
TBD – To Be Defined
CLI – Command Line Interface
MVP – Minimal Viable Product

Appendix B: Schedule

2018-12-2 – setting up environment
2018-12-9 – research, testing
2018-12-16 – MVP
2018-12-22 – implement REQ-1, REQ-2, REQ-4
2018-12-29 – implement REQ-3 REQ-5, REQ-7
2019-01-4 – implement REQ-8, stabilize features
2019-01-11 – implement OPT’s, documentation
2019-01-18 – implement OPT’s, testing
2019-01-25 – testing, documentation

Appendix C: To Be Determined List

1. CLI interface – precise user facing commands
2. Animation – precise behavior and package
3. Runge-Kutta – precise methos