Alpha Delivery: RUP Artifacts

Packer for Control Systems

Github repository Link

Prepared by: Anton Buguev, Kirill Glinskiy, Mirna Alnoukari, Timur Galiev

05.09.2021

Version History

Editor's Name	**Date** (DD/MM/YYYY)	**Reason for Changes/Sections Updated**	**Version**
Anton Buguev+Timur Galeev+Mirna Alnoukari	25/08/2021	Initial Information for planning	1.0
Kirill Glinskiy	04/09/2021	The stakeholders list have changed	1.01
Kirill Glinskiy	05/09/2021	Non-Functional Requirements added Glossary (~40%) added	1.02
Anton Buguev+Kirill Glinskiy	05/09/2021	Non-Functional Requirements ways of implementation added	1.03

Table of Contents

1. Business Goals and Objectives

The main goal for this project is to generate multiple different libraries that transfer any mechanical system into a control system that is used by the control and robotics engineers, without anyone having to write his own library manually every time.

1.1. The goal is to create a packer that turns physical equations into a control system in the form of a usable library or a packet for some hardware.
1.2. It has to unify a solution for a lot of control problems, including simulation, control of robots and creation of interactive systems.
1.3. The packer should give the opportunity to easy access to the control system from any language or hardware.
1.4. The system should coexist with existing libraries.
1.5. The packer should have high performance.
1.6. The result libraries should be applicable for different projects.

2. Roles and responsibilities

Stakeholder's Name	Roles	Responsibilities
Simeon Nedulchev	Project owner	Provide us with information about the system that needs to be controlled
Anton Buguev	Developer, tester	Can run builds, view logs ● Make calculations to transform physical equations into the system
Timur Galiev	Developer, tester	Can run builds, view logs ● Addition of integration of the control system. ● Create C headers.
Mirna Alnoukari	Product manager, technical documentation writer	Manage the process between team members. Write documentation
Kirill Glinskiy	Developer	Assist in the development process
name1	End users	check if this software works, and use it if it does
name2	Admins	add

Note: The previous names of team members will be written in short taking the first letters of the first and last name of each person.

3. Requirement Analysis and Specifications

3.1. Features

ID #	User Story Title	Priority	Any Other Label
1	Creation of the system from physical equations.	Must	
2.1	C++ library creation	should have	
2.2	Python library creation	should have	

3.2. User Stories

User Type	User Story Title	User stories
Control system user	Creation of the system from physical equations.	● As a developer I want my physical equations transformed into a control system so that I can use them in my code ● As a robotics developer, I want to get a control system out of my robot's physical equations so that I can import them into my robot for control ● As a developer I want the system to be parameterized so that I can use it with different parameters
	Python library creation	As a developer, I want the system to be a Python library so that I can easily import it
	C++ library creation	As a developer, I want to be able to use the system as a C++ library so that I can easily import it

4. Software Development plan

Inception Phase				
#Iteration	**Timeline**	**Stakeholders**	**Activities**	**Artifacts**
#1	24/08/2021-25/08/2021	KG, SN	Determine goals and objectives with valid justification	Deliver the documentation of achieved milestones
#2	25/08/2021-26/08/2021	TG	Requirement engineering(20% user stories) Identify Risks	Update the documentation of achieved milestones with User stories and Risk Lists
#3	24/08/2021 - 25/08/2021	MN	Identify the stakeholders Establish roles and responsibilities	Update the documentation with plans and responsibilities.

Elaboration Phase				
#Iteration	**Timeline**	**Stakeholders**	**Activities**	**Artifacts**
#1	26/08/2021-05/09/2021	TG, MN, AB	Revise User Stories (100%)	Document 100% user stories
#2	25/08/2021-06/09/2021	KG MN	Software development planning	Iteration Plan
#3	30/08/2021 - 10/09/2021	AB	Software Architecture Test Plan	Software architecture document Test Plan Document

Construction Phase				
#Iteration	**Timeline**	**Stakeholders**	**Activities**	**Artifacts**
#1	01/09/2021-05/09/2021	AB,KG,MN	Implement Feature 1/ user story 1 Unit test cases for feature 1	Deliver the documentation of achieved milestones
#2		TBD	Implement Feature 2 Unit test cases for feature 2	Working feature 1 branch Unit test results

Transition Phase				
#Iteration	**Timeline**	**Stakeholders**	**Activities**	**Artifacts**
#1	24/08/2021-25/08/2021	MN	Integration, End to end testing Training for Users and Developers	Github repository Merged branches Integration and ended to end test results README for developers and Users
#2		TBD	Final product release	Working Product

5. Non-Functional Requirements

ID #	Parameter	Priority	Requirement	How we will reach the requirements
1.	Scalability.	Medium	The system should allow the addition of libraries for new languages without or with no less than 10% damage to performance.	Using as minimum RAM as possible and low-level libraries it will be possible to add generation of libraries for other languages without significant loss in performance.
2.	Portability.	High	The system must be installable on major operating systems (WinOS / Linux-based systems).	C headers, .py files, .cpp files are available for use on both platforms. We must avoid OS-dependent code (e.g. multiprocessing)
3.	Compatibility.	Medium	The system should be able to coexist with any other subroutine libraries.	The code should take as minimum RAM as possible (not to hurt other libraries performance) and not use third-party modules.
4.	Localization.	High	The system should have the ability to be integrated into the existing projects of Innopolis in the field of control theory.	The system will generate libraries for different languages so it will be useful for many projects.
5.	Performance	High	The system should be executed in less than 1 minute.	Using low-level libraries we can reach high performance.

6. Glossary

Packer - a program that allows you to turn input (energy equations) into compact libraries for various programming languages.

Control system - a system, which provides the desired response by controlling the output.

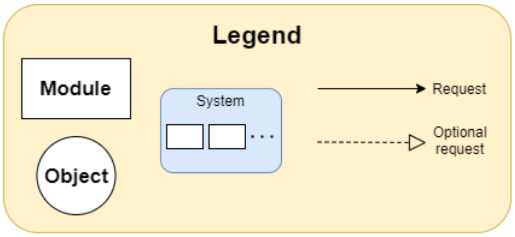
Header - a file containing C language declarations and macro definitions to be shared between several source files.

Library - a collection of non-volatile resources used by computer programs for software development.

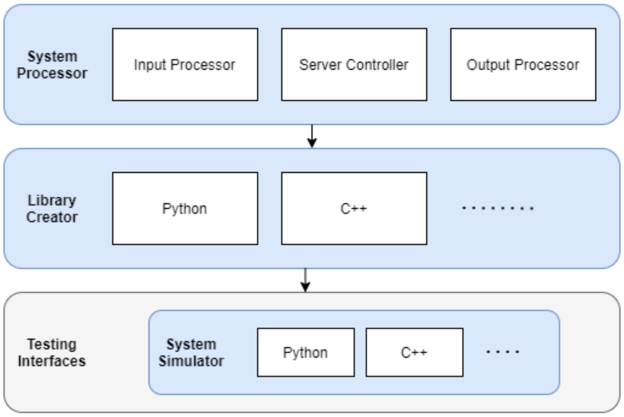
Energy equations - potential and kinetic energies equations in symbolic format.

Method of Lagrange multipliers - strategy for finding the local maxima and minima of a function subject to equality constraints.

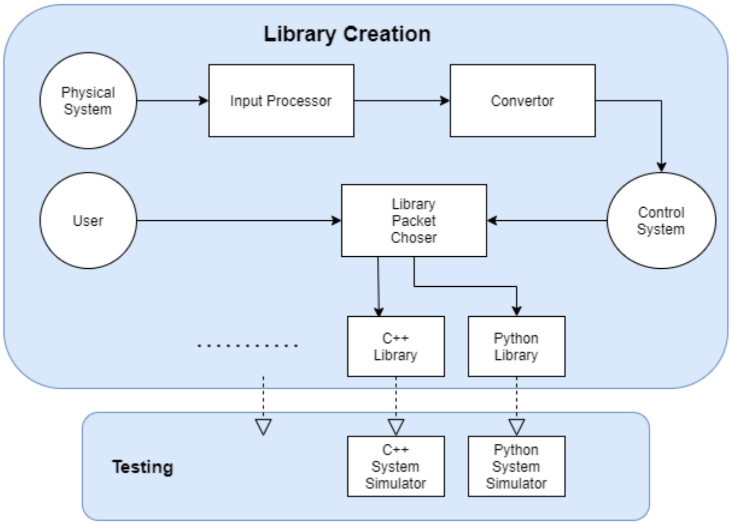
Perspectives



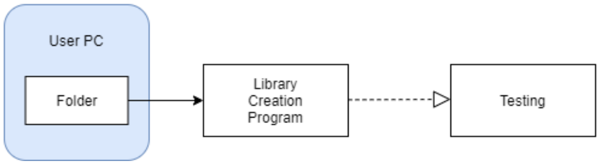
Static View



Dynamic View



Allocation View


User PC
Folder
Library Creation Program
Testing