

Software Requirements Specification for 4TB6 - Mechatronics Capstone: Smart Bike Lock

Abi Nevo
Elsa Bassi
Steffi Ralph
Abdul Iqbal
Stephen De Jong
Anthony Shenouda

October 4, 2022

Contents

1	Reference Material	iv
1.1	Table of Units	iv
1.2	Table of Symbols	iv
1.3	Abbreviations and Acronyms	1
2	Introduction	1
2.1	Purpose of Document	1
2.2	Scope of Requirements	1
2.3	Characteristics of Intended Reader	2
2.4	Organization of Document	2
3	General System Description	2
3.1	System Context	3
3.2	User Characteristics	4
3.3	System Constraints	4
4	Specific System Description	4
4.1	Problem Description	4
4.1.1	Terminology and Definitions	4
4.1.2	Physical System Description	5
4.1.3	Goal Statements	5
4.2	Solution Characteristics Specification	5
4.2.1	Assumptions	7
4.2.2	Theoretical Models	7
4.2.3	General Definitions	9
4.2.4	Data Definitions	10
4.2.5	Data Types	11
4.2.6	Instance Models	12
4.2.7	Input Data Constraints	13
4.2.8	Properties of a Correct Solution	13
5	Requirements	14
5.1	Functional Requirements	15
5.1.1	User Input Related	15
5.1.2	Bike Input Related	15
5.1.3	Output Related	15
5.2	Nonfunctional Requirements	15
5.2.1	Smart Phone	15
5.2.2	Physical Design	15
5.2.3	Accuracy	15

5.2.4	Usability	15
5.2.5	Maintainability	15
5.2.6	Portability	15
5.2.7	Security	15
6	Likely Changes	15
7	Unlikely Changes	16
8	Traceability Matrices and Graphs	16
9	Development Plan	16
10	Values of Auxiliary Constants	19

Revision History

Date	Version	Name	Notes
02-10-22	1.0	Elsa	Drafted first draft
03-10-22	1.1	Stephen	Drafted 1 & 5, and formating updates

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d’Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
m	length	metre
kg	mass	kilogram
s	time	second
J	energy	joule
P or W	power	watt ($W = J s^{-1}$)
A or I	current	ampere
Ω or R	resistance	ohm
V	voltage	volt
N	force	newton
NM	torque	newton meter

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	unit	description
B_L	hour	battery life in hours
B_C	mAH	battery capacity in amp hours
L_C	A/actuation	current drawn per motor actuation
A_{num}	actuation/charge	number of actuations per charge

1.3 Abbreviations and Acronyms

symbol	description
SRS	Software Requirements Specification

2 Introduction

The purpose of the Smart Lock project is to design and build a product that will provide bicycle users with a safer, easier, and more accessible way to secure their bike through their smartphone. Additionally, it will provide users with a GPS feature to locate the lock in case of bike theft or misplacement. It will consist of a physical lock that mounts to a bike and a smartphone application that will function as the user interface through which the lock can be engaged and disengaged wirelessly, as well as located. The project will provide an engineering solution using wireless communication, mechanical design, and smartphone application development. More broadly, it seeks to encourage members of society to pursue biking, in both a transportation and recreational capacity, improving the health of society's citizens and its environment.

2.1 Purpose of Document

The Requirements Documentation seeks to provide a complete overview of the requirements of the Smart Lock system so as to define the project. The assumptions, inputs and outputs will also be defined in order to outline the problem. Several models will be developed and lastly, likely and unlikely changes will be described. This will communicate a unified and documented plan for the project that can be used after the design stage to verify its functionality and provide an important benchmark.

2.2 Scope of Requirements

The Smart Lock project will build off existing designs to tie together principles of wireless communication, GPS, a mechanical locking mechanism, electrical actuation, and smartphone application development. The system will provide all functions necessary for effective and secure locking and location.

The Smart Lock project will be designed to perform the following functions for the user.

1. Wireless communication from the smartphone to the lock and vice versa.
2. Display of lock and battery status information on the app.
3. Display of GPS location on the app.
4. Store and use a replaceable battery.

5. House a mechanical lock frame.
6. Perform electrical engagement/disengagement of a locking mechanism.
7. Be waterproof.

Not included in the project scope will be Bluetooth wireless connective capabilities, a fully autonomous locking mechanism (one that will both open/close and engage/disengage the lock wirelessly) and a rechargeable battery. We will also be ignoring extreme weather and temperature conditions and their impact on the material properties.

2.3 Characteristics of Intended Reader

The reader will have basic knowledge of how bikes are secured in public settings, namely using mechanical locks and keys or combination locks. They will understand why bike locking is necessary and what it means for a bike to be vulnerable to theft, i.e. that all main parts of the bike must be secured including the frame and wheels. They will also have basic knowledge of GPS and wireless communications that can be accessed through smartphone applications. Lastly, the reader will have a high-school level understanding of kinematic and electronics-related physics and math such that they can grasp the mechanical and electrical functionality of the system.

2.4 Organization of Document

The document will follow the base template provided by the Capstone 4TB6 course GitLab. It will be organized with an increasing order of specificity, beginning with the General System Description that will include an overview of the Smart Lock project. The Specific System Description will make up the bulk of the document, describing the project in greater detail with its assumptions, inputs and outputs, goals and models. Finally, since the context of the project is previously stated, the Functional and Non-Functional Requirements will be outlined, as well as any foreseen Likely and Unlikely Changes to the project. Lastly, the Traceability Matrix is included to define the relationships between various relevant formulas, sections and information in the document and how they be used to build the project.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints. [This text can likely be borrowed verbatim. —TPLT]

[The purpose of this section is to provide general information about the system so the specific requirements in the next section will be easier to understand. The general system description section is designed to be changeable independent of changes to the functional requirements documented in the specific system description. The general system description

provides a context for a family of related models. The general description can stay the same, while specific details are changed between family members. —TPLT]

3.1 System Context

[Your system context will include a figure that shows the abstract view of the software. Often in a scientific context, the program can be viewed abstractly following the design pattern of Inputs \rightarrow Calculations \rightarrow Outputs. The system context will therefore often follow this pattern. The user provides inputs, the system does the calculations, and then provides the outputs to the user. The figure should not show all of the inputs, just an abstract view of the main categories of inputs (like material properties, geometry, etc.). Likewise, the outputs should be presented from an abstract point of view. In some cases the diagram will show other external entities, besides the user. For instance, when the software product is a library, the user will be another software program, not an actual end user. If there are system constraints that the software must work with external libraries, these libraries can also be shown on the System Context diagram. They should only be named with a specific library name if this is required by the system constraint. —TPLT]

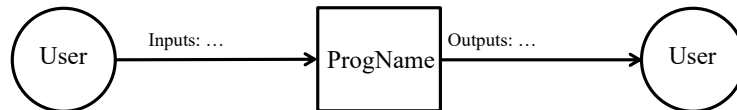


Figure 1: System Context

[For each of the entities in the system context diagram its responsibilities should be listed. Whenever possible the system should check for data quality, but for some cases the user will need to assume that responsibility. The list of responsibilities should be about the inputs and outputs only, and they should be abstract. Details should not be presented here. However, the information should not be so abstract as to just say “inputs” and “outputs”. A summarizing phrase can be used to characterize the inputs. For instance, saying “material properties” provides some information, but it stays away from the detail of listing every required properties. —TPLT]

- User Responsibilities:

-

- 4TB6 - Mechatronics Capstone Responsibilities:

- Detect data type mismatch, such as a string of characters instead of a floating point number

-

3.2 User Characteristics

[This section summarizes the knowledge/skills expected of the user. Measuring usability, which is often a required non-function requirement, requires knowledge of a typical user. As mentioned above, the user is a different role from the “intended reader,” as given in Section 2.3. As in Section 2.3, the user characteristics should be specific and unambiguous. For instance, “The end user of 4TB6 - Mechatronics Capstone should have an understanding of undergraduate Level 1 Calculus and Physics.” —TPLT]

3.3 System Constraints

[System constraints differ from other type of requirements because they limit the developers’ options in the system design and they identify how the eventual system must fit into the world. This is the only place in the SRS where design decisions can be specified. That is, the quality requirement for abstraction is relaxed here. However, system constraints should only be included if they are truly required. —TPLT]

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models. [Add any project specific details that are relevant for the section overview. —TPLT]

4.1 Problem Description

4TB6 - Mechatronics Capstone is intended to solve ... [What problem does your program solve? The description here should be in the problem space, not the solution space. —TPLT]

4.1.1 Terminology and Definitions

[This section is expressed in words, not with equations. It provide the meaning of the different words and phrases used in the domain of the problem. The terminology is used to introduce concepts from the world outside of the mathematical model The terminology provides a real world connection to give the mathematical model meaning. —TPLT]

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

-

4.1.2 Physical System Description

[The purpose of this section is to clearly and unambiguously state the physical system that is to be modelled. Effective problem solving requires a logical and organized approach. The statements on the physical system to be studied should cover enough information to solve the problem. The physical description involves element identification, where elements are defined as independent and separable items of the physical system. Some example elements include acceleration due to gravity, the mass of an object, and the size and shape of an object. Each element should be identified and labelled, with their interesting properties specified clearly. The physical description can also include interactions of the elements, such as the following: i) the interactions between the elements and their physical environment; ii) the interactions between elements; and, iii) the initial or boundary conditions. —TPLT]

The physical system of 4TB6 - Mechatronics Capstone, as shown in Figure ?, includes the following elements:

PS1:

PS2: ...

[A figure here makes sense for most SRS documents —TPLT]

4.1.3 Goal Statements

[The goal statements refine the “Problem Description” (Section 4.1). A goal is a functional objective the system under consideration should achieve. Goals provide criteria for sufficient completeness of a requirements specification and for requirements pertinence. Goals will be refined in Section “Instantiated Models” (Section 4.2.6). Large and complex goals should be decomposed into smaller sub-goals. The goals are written abstractly, with a minimal amount of technical language. They should be understandable by non-domain experts. —TPLT]

Given the [inputs —TPLT], the goal statements are:

GS1: [One sentence description of the goal. There may be more than one. Each Goal should have a meaningful label. —TPLT]

4.2 Solution Characteristics Specification

[This section specifies the information in the solution domain of the system to be developed. This section is intended to express what is required in such a way that analysts and stakeholders get a clear picture, and the latter will accept it. The purpose of this section is to reduce the problem into one expressed in mathematical terms. Mathematical expertise is used to extract the essentials from the underlying physical description of the problem, and to collect and substantiate all physical data pertinent to the problem. —TPLT]

[This section presents the solution characteristics by successively refining models. It starts with the abstract/general Theoretical Models (TMs) and refines them to the concrete/specific

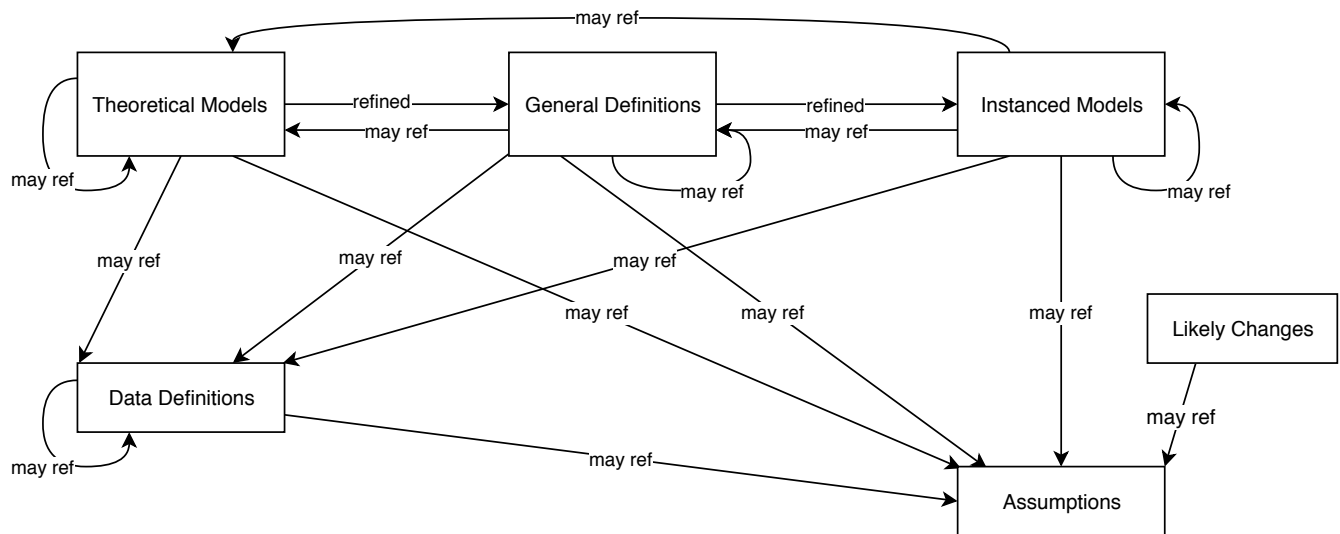
Instance Models (IMs). If necessary there are intermediate refinements to General Definitions (GDs). All of these refinements can potentially use Assumptions (A) and Data Definitions (DD). TMs are refined to create new models, that are called GMs or IMs. DDs are not refined; they are just used. GDs and IMs are derived, or refined, from other models. DDs are not derived; they are just given. TMs are also just given, but they are refined, not used. If a potential DD includes a derivation, then that means it is refining other models, which would make it a GD or an IM. —TPLT]

[The above makes a distinction between “refined” and “used.” A model is refined to another model if it is changed by the refinement. When we change a general 3D equation to a 2D equation, we are making a refinement, by applying the assumption that the third dimension does not matter. If we use a definition, like the definition of density, we aren’t refining, or changing that definition, we are just using it. —TPLT]

[The same information can be a TM in one problem and a DD in another. It is about how the information is used. In one problem the definition of acceleration can be a TM, in another it would be a DD. —TPLT]

[There is repetition between the information given in the different chunks (TM, GDs etc) with other information in the document. For instance, the meaning of the symbols, the units etc are repeated. This is so that the chunks can stand on their own when being read by a reviewer/user. It also facilitates reuse of the models in a different context. —TPLT]

[The relationships between the parts of the document are show in the following figure. In this diagram “may ref” has the same role as “uses” above. The figure adds “Likely Changes,” which are able to reference (use) Assumptions. —TPLT]



The instance models that govern 4TB6 - Mechatronics Capstone are presented in Subsection 4.2.6. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

4.2.1 Assumptions

[The assumptions are a refinement of the scope. The scope is general, where the assumptions are specific. All assumptions should be listed, even those that domain experts know so well that they are rarely (if ever) written down. —TPLT] [The document should not take for granted that the reader knows which assumptions have been made. In the case of unusual assumptions, it is recommended that the documentation either include, or point to, an explanation and justification for the assumption. —TPLT]

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [T], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

- A1: [Short description of each assumption. Each assumption should have a meaningful label. Use cross-references to identify the appropriate traceability to T, GD, DD etc., using commands like dref, ddref etc. Each assumption should be atomic - that is, there should not be an explicit (or implicit) “and” in the text of an assumption. —TPLT]

4.2.2 Theoretical Models

[Theoretical models are sets of abstract mathematical equations or axioms for solving the problem described in Section “Physical System Description” (Section 4.1.2). Examples of theoretical models are physical laws, constitutive equations, relevant conversion factors, etc. —TPLT]

This section focuses on the general equations and laws that 4TB6 - Mechatronics Capstone is based on. [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

4.2.2.1 RefName:**T:COE****4.2.2.2 Label:**Conservation of thermal energy

4.2.2.3 Equation:

$$-\nabla \cdot \mathbf{q} + g = \rho C \frac{\partial T}{\partial t}$$

4.2.2.4 Description:

The above equation gives the conservation of energy for transient heat transfer in a material of specific heat capacity C ($\text{J kg}^{-1} \text{ } ^\circ\text{C}^{-1}$) and density ρ (kg m^{-3}), where \mathbf{q} is the thermal flux vector (W m^{-2}), g is the volumetric heat generation (W m^{-3}), T is the temperature ($^\circ\text{C}$), t is time (s), and ∇ is the gradient operator. For this equation to apply, other forms of energy, such as mechanical energy, are assumed to be negligible in the system (A??). In general, the material properties (ρ and C) depend on temperature.

4.2.2.5 Notes:

None.

4.2.2.6 Source:http://www.efunda.com/formulae/heat_transfer/conduction/overview_cond.cfm**4.2.2.7 Ref. By:**

GD??

4.2.2.8 Preconditions for [T:COE](#):

None

4.2.2.9 Derivation for T:COE:

Not Applicable

[“Ref. By” is used repeatedly with the different types of information. This stands for Referenced By. It means that the models, definitions and assumptions listed reference the current model, definition or assumption. This information is given for traceability. Ref. By provides a pointer in the opposite direction to what we commonly do. You still need to have a reference in the other direction pointing to the current model, definition or assumption. As an example, if T1 is referenced by G2, that means that G2 will explicitly include a reference to T1. —TPLT]

4.2.3 General Definitions

[General Definitions (GDs) are a refinement of one or more TMs, and/or of other GDs. The GDs are less abstract than the TMs. Generally the reduction in abstraction is possible through invoking (using/referencing) Assumptions. For instance, the TM could be Newton’s Law of Cooling stated abstracting. The GD could take the general law and apply it to get a 1D equation. —TPLT]

This section collects the laws and equations that will be used in building the instance models.

[Some projects may not have any content for this section, but the section heading should be kept. —TPLT] [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	GD1
Label	Newton's law of cooling
SI Units	W m^{-2}
Equation	$q(t) = h\Delta T(t)$
Description	<p>Newton's law of cooling describes convective cooling from a surface. The law is stated as: the rate of heat loss from a body is proportional to the difference in temperatures between the body and its surroundings.</p> <p>$q(t)$ is the thermal flux (W m^{-2}).</p> <p>h is the heat transfer coefficient, assumed independent of T (A??) ($\text{W m}^{-2} \text{ }^{\circ}\text{C}^{-1}$).</p> <p>$\Delta T(t) = T(t) - T_{\text{env}}(t)$ is the time-dependent thermal gradient between the environment and the object ($^{\circ}\text{C}$).</p>
Source	Citation here
Ref. By	DD1, DD??

Detailed derivation of simplified rate of change of temperature

[This may be necessary when the necessary information does not fit in the description field. —TPLT] [Derivations are important for justifying a given GD. You want it to be clear where the equation came from. —TPLT]

4.2.4 Data Definitions

[The Data Definitions are definitions of symbols and equations that are given for the problem. They are not derived; they are simply used by other models. For instance, if a problem depends on density, there may be a data definition for the equation defining density. The DDs are given information that you can use in your other modules. —TPLT]

[All Data Definitions should be used (referenced) by at least one other model. —TPLT]

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Number	DD1
Label	Heat flux out of coil
Symbol	q_C
SI Units	W m^{-2}
Equation	$q_C(t) = h_C(T_C - T_W(t))$, over area A_C
Description	T_C is the temperature of the coil ($^{\circ}\text{C}$). T_W is the temperature of the water ($^{\circ}\text{C}$). The heat flux out of the coil, q_C (W m^{-2}), is found by assuming that Newton's Law of Cooling applies (A??). This law (GD1) is used on the surface of the coil, which has area A_C (m^2) and heat transfer coefficient h_C ($\text{W m}^{-2} ^{\circ}\text{C}^{-1}$). This equation assumes that the temperature of the coil is constant over time (A??) and that it does not vary along the length of the coil (A??).
Sources	Citation here
Ref. By	IM1

4.2.5 Data Types

[This section is optional. In many scientific computing programs it isn't necessary, since the inputs and output are straightforward types, like reals, integers, and sequences of reals and integers. However, for some problems it is very helpful to capture the type information. —TPLT]

[The data types are not derived; they are simply stated and used by other models. —TPLT]

[All data types must be used by at least one of the models. —TPLT]

[For the mathematical notation for expressing types, the recommendation is to use the notation of Hoffman and Strooper (1995). —TPLT]

This section collects and defines all the data types needed to document the models. [Modify the examples below for your problem, and add additional definitions as appropriate. —TPLT]

Type Name	Name for Type
Type Def	mathematical definition of the type
Description	description here
Sources	Citation here, if the type is borrowed from another source

4.2.6 Instance Models

[The motivation for this section is to reduce the problem defined in “Physical System Description” (Section 4.1.2) to one expressed in mathematical terms. The IMs are built by refining the TMs and/or GDs. This section should remain abstract. The SRS should specify the requirements without considering the implementation. —TPLT]

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.2.4 to replace the abstract symbols in the models identified in Sections 4.2.2 and 4.2.3.

The goals [reference your goals —TPLT] are solved by [reference your instance models —TPLT]. [other details, with cross-references where appropriate. —TPLT] [Modify the examples below for your problem, and add additional models as appropriate. —TPLT]

Number	IM1
Label	Energy balance on water to find T_W
Input	$m_W, C_W, h_C, A_C, h_P, A_P, t_{\text{final}}, T_C, T_{\text{init}}, T_P(t)$ from IM?? The input is constrained so that $T_{\text{init}} \leq T_C$ (A??)
Output	$T_W(t)$, $0 \leq t \leq t_{\text{final}}$, such that $\frac{dT_W}{dt} = \frac{1}{\tau_W}[(T_C - T_W(t)) + \eta(T_P(t) - T_W(t))]$, $T_W(0) = T_P(0) = T_{\text{init}}$ (A??) and $T_P(t)$ from IM??
Description	T_W is the water temperature ($^{\circ}\text{C}$). T_P is the PCM temperature ($^{\circ}\text{C}$). T_C is the coil temperature ($^{\circ}\text{C}$). $\tau_W = \frac{m_W C_W}{h_C A_C}$ is a constant (s). $\eta = \frac{h_P A_P}{h_C A_C}$ is a constant (dimensionless). The above equation applies as long as the water is in liquid form, $0 < T_W < 100^{\circ}\text{C}$, where 0°C and 100°C are the melting and boiling points of water, respectively (A??, A??).
Sources	Citation here
Ref. By	IM??

Derivation of ...

[The derivation shows how the IM is derived from the TMs/GDs. In cases where the derivation cannot be described under the Description field, it will be necessary to include this subsection. —TPLT]

4.2.7 Input Data Constraints

Table 1 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

The specification parameters in Table 1 are listed in Table 2.

Table 1: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
L	$L > 0$	$L_{\min} \leq L \leq L_{\max}$	1.5 m	10%

(*) [you might need to add some notes or clarifications —TPLT]

Table 2: Specification Parameter Values

Var	Value
L_{\min}	0.1 m

4.2.8 Properties of a Correct Solution

A correct solution must exhibit [fill in the details —TPLT]. [These properties are in addition to the stated requirements. There is no need to repeat the requirements here. These additional properties may not exist for every problem. Examples include conservation laws

Table 3: Output Variables

Var	Physical Constraints
T_W	$T_{\text{init}} \leq T_W \leq T_C$ (by A??)

(like conservation of energy or mass) and known constraints on outputs, which are usually summarized in tabular form. A sample table is shown in Table 3 —TPLT]

[This section is not for test cases or techniques for verification and validation. Those topics will be addressed in the Verification and Validation plan. —TPLT]

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

5.1.1 User Input Related

- 5.1.1.1 LockButton input must engage the lock on the bike.
- 5.1.1.2 UnlockButton input must disengage the lock on the bike
- 5.1.1.3 When lock is engaged, change lock status to locked
- 5.1.1.4 When lock is unengaged, change lock status to unlocked
- 5.1.1.5 Moving locking bar closed must move OpenClosedstatus to Closed
- 5.1.1.6 Moving locking bar open must move OpenClosedstatus to open
- 5.1.1.7 Location (coordinates) of user's phone must be shown on app as User-Position

5.1.2 Bike Input Related

- 5.1.2.1 The lock can be mounted to the bikes frame

5.1.3 Output Related

- 5.1.3.1 Battery percentage must be shown on phone app
- 5.1.3.2 Location (coordinates) of bike must be shown on app as BikePosition
- 5.1.3.3 Battery must output enough power to engage/disengage the lock

5.2 Nonfunctional Requirements

5.2.1 Smart Phone

- 5.2.1.1 can be used by people speaking different languages
- 5.2.1.2 Can reasonably be used without requiring an instruction manual

5.2.2 Physical Design

- 5.2.2.1 The design must be visual appealing
- 5.2.2.2 The design must not impede normal bike functions
- 5.2.2.3 The lock must be waterproofed to withstand normal rainfall
- 5.2.2.4 The lock must be waterproofed to withstand normal splashing while riding

5.2.3 Accuracy

- 5.2.3.1 Accuracy of bike lock status must be above 90%
- 5.2.3.2 The accuracy of gps positioning must be accurate within 10M
- 5.2.3.3 Battery percentage must be calculated accurately within 10%

5.2.4 Usability

- 5.2.4.1 iPhone app locking must be quicker to use than a typical keyed/combo bike lock

LC1: [Give the likely changes, with a reference to the related assumption (aref), as appropriate. —TPLT]

7 Unlikely Changes

LC2: [Give the unlikely changes. The design can assume that the changes listed will not occur. —TPLT]

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an “X” may have to be modified as well. Table 4 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 5 shows the dependencies of instance models, requirements, and data constraints on each other. Table 6 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1’s derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is “used by” GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure ?? shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure ?? shows the dependencies of instance models, requirements, and data constraints on each other.

9 Development Plan

[This section is optional. It is used to explain the plan for developing the software. In particular, this section gives a list of the order in which the requirements will be implemented. In the context of a course this is where you can indicate which requirements will be implemented as part of the course, and which will be “faked” as future work. This section can be

	T??	T??	T??	GD1	GD??	DD1	DD??	DD??	DD??	IM1	IM??	IM??	IM??
T??													
T??			X										
T??													
GD1													
GD??	X												
DD1				X									
DD??				X									
DD??													
DD??								X					
IM1					X	X	X				X		
IM??					X		X		X	X			X
IM??		X											
IM??		X	X				X	X	X		X		

Table 4: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM1	IM??	IM??	IM??	4.2.7	R??	R??
IM1		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R??	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R??			X	X			
R??		X					
R??		X					

Table 5: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
T??	X																		
T??																			
T??																			
GD1		X																	
GD??			X	X	X	X													
DD1							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM1											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 6: Traceability Matrix Showing the Connections Between Assumptions and Other Items

organized as a prioritized list of requirements, or it could should the requirements that will be implemented for “phase 1”, “phase 2”, etc. —TPLT]

10 Values of Auxiliary Constants

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

References

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL <http://citeseer.ist.psu.edu/428727.html>.

[The following is not part of the template, just some things to consider when filing in the template. —TPLT]

[Grammar, flow and L^AT_EX advice:

- For Mac users *.DS_Store should be in .gitignore
- L^AT_EX and formatting rules
 - Variables are italic, everything else not, includes subscripts ([link to document](#))
 - * [Conventions](#)
 - * Watch out for implied multiplication
 - Use BibTeX
 - Use cross-referencing
- Grammar and writing rules
 - Acronyms expanded on first usage (not just in table of acronyms)
 - “In order to” should be “to”

—TPLT]

[Advice on using the template:

- Difference between physical and software constraints
- Properties of a correct solution means *additional* properties, not a restating of the requirements (may be “not applicable” for your problem). If you have a table of output constraints, then these are properties of a correct solution.
- Assumptions have to be invoked somewhere
- “Referenced by” implies that there is an explicit reference
- Think of traceability matrix, list of assumption invocations and list of reference by fields as automatically generatable
- If you say the format of the output (plot, table etc), then your requirement could be more abstract

—TPLT]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.
2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?