Griffin Robotics Engineering Process

Introduction

The Griffin Robotics Engineering process is a process (or set of generalized procedures) that the Nex+Gen Griffin Robotics Teams use to create robots for First Technology Challenge competitions. An advantage to using flexible development procedures, such as these, is to introduce consistency and predictability into the development process. Students understand more clearly what needs to be done and what their contribution to the creation of a robot should be. The process lends structure to the activities, which can reduce confusion and lend confidence to the developers.

The Process

The engineering process is the same regardless of the branch of engineering used. The process is iterative in nature, where each cycle is one of gradual refinement (see Fig. 1). The terminology may be a little different from one engineering branch to the next, but the general idea is the same.

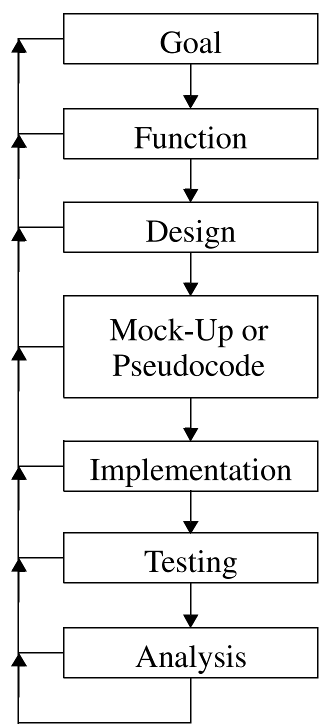

Figure 1 - Engineering Process

The idea is that each stage provide the foundation for the next stage in the process. For example, it only makes sense to form a strategy to accomplish a goal once you have a clear idea of a goal in mind. However, that does not mean that a stage needs to be completed in its entirety before beginning the next stage. For example, a person might begin analyzing the data before the last test is completed, or start writing Java code before the last line of pseudocode is written. In fact, some processes in use today *encourage* the next stage to begin before the previous stage is completed.

Goal

This is the big picture goal for the project. It is also sometimes presented as a mission statement. It identifies at the highest level what the project is about. A good goal is simple, clear and measurable. For example, a good goal might be something like: “Build a robot to successfully compete in the 2018 FTC Competition.” It imparts a clear idea of what you want to accomplish at the end of the project.

A better goal might be: “Build a robot to win the Design Award and score in the top 5 robots in the 2018 FTC Competition.” Notice how there are two clear, measurable items in this statement. The FTC criteria are clear and specific about what you must do for each of the items. The items are measurable as well. At the end of the competition, either you will have won the Design Award or you will not. Either you will score in the top 5 or you will not. There is clarity in the what you are trying to achieve and the outcome is measurable.

A *less* useful goal, because of its ambiguity, might be: “Learn cool stuff about robots.” What is cool? What stuff do you want to learn? How will you know when you have learned it?

There is a single shared goal statement for the project. There is no need for separate goal statements for each engineering discipline.

Function

The functional specification is where you begin to develop your strategy for achieving your goal. Once you have a good goal, it needs to be broken down into the features and functions your robot will provide. What strategy will you use to achieve your goal? What does your robot need to do to achieve your strategy? For example, if the FTC Competition includes loading foam blocks into a box, how will your robot get them there? Will you use a ramp, a claw, or some other means of transporting them? What means of locomotion will you use to move your robot around the playing field? Holonomic or tank? And if it’s holonomic, will you use mecanum wheels or omni wheels?

The design process is flexible here. Whether to use a tank or holonomic drive for locomotion is clearly part of the function (or strategy) because many subsystems are affected and the robot behaves differently based on this choice. But some of the details, such as whether to use mecanum or omni wheels, could be considered as part of the functional specification or part of the robot design. Whether you include some detail in the functional specification or the design depends on how far reaching its impact could be. Tank drive requires a completely different mechanical, electrical and programming approach than a holonomic drive system. Choosing between omni and mecanum wheels has much less impact on how the robot *functions*, so that choice should probably be included as part of the design rather than part of its function.

A functional description should also include a description of any requirements the functionality must meet. For example, if the robot must operate within a 12’ x 12’ play field, or fit within an 18” x 18” x 18” cube-shaped box. External requirements such as these may be spelled out in the document, or they may be included as references to other documents, such as the FTC competition rules.

There is one functional description for a project, not one for each engineering discipline.

Design

The next level of organization is the robot component design. How is each function in the functional specification going to work? The functional specification describes *what* the robot does, e.g., move forward, backward, left and right without turning, while the design specification describes *how* each subsystem works. For example, the robot is able to move left and right without turning by using a holonomic drive system with mecanum wheels.

Of course a complete design must describe the drive system in *much* more detail than that, including how a holonomic drive system works, how the power to the drive motors will be determined, etc. The design must be complete and have enough detail that a person who is reasonably profficient in the field could build the robot from this document. When appropriate, the document should include any mathematical reasoning used to determine how a component is implemented. For example, one component might translate joystick positions to power settings on the drive motors as part of its design. Be sure to show both the math that transforms the joystick to the drive power and how that transformation was derived.

This is also a good place to write down any expectations or assumptions inherent to the design. For example, will the design make use of a linear or iterative op-mode (programming)? Will the balancing tables be especially slick (mechanical)? What are the advantages and disadvantages of the possible choices and why was *this* selection made?

There is one design specification for each engineering discipline used, e.g., one specification for programming, one for electrical, one for mechanical, etc.

Mock-Up, Prototype Or Pseudocode

The "mock-up" carries the design to the next level. The "big animal" features are integrated together in whatever manner is appropriate to the discipline. For example, in the mechanical domain you may want to put together a prototype to test whether components fit together, claws grasp blocks with sufficient force, etc. In the electrical domain you may want to put together a prototype to ensure ports map appropriately to devices, wires reach, and voltages match. In the programming domain you may want to write out pseudocode to ensure that all input cases and conditions are adequately covered, control flow is clear, and data structures maintain the needed state.

Once created, this prototype can be morphed into the final product, or it can be used as the basis for further prototyping. In the case of programming, once the designer is satisfied with the pseudocode it can be used as the template for creating the "final[1]" program.

Implementation

This is the deliverable product. For a mechanical or electrical engineer, it is a physical robot with working motors, claws, etc. For a programmer it is the Java program to read the sensors and drive the motors to accomplish the goals of the project.

1. "Final" in this case means "sufficiently complete to move on to the next stage." The program need not be "final" in the sense of the final deliverable product.

What is important to note, though, is this process may produce many implementations, where each one is (hopefully) closer to satisfying your goals. While it may be the deliverable product, the first time through your product may not be in a deliverable state. Once you have a product it must be passed forward to the next stages for testing and analysis.

Testing

With a working product in place, the next stage is to test whether it accomplishes the goals of the project. *Ideally* the testing criteria have been thought out in advance and tests have been created that adequately cover the goals. While that may be ideal, in reality much of the design is an exercise in learning about the project, whether it is learning about the new FTC Challenge rules, the new techniques (e.g., sliders and holonomic drives), or new devices (e.g., the new REV Hub), etc. Tests can *and should* be developed along with every phase of the project. As more is learned, new tests can be added to the pool.

Analysis

Analysis takes the results of the tests and compares their results against the desired goals. This can be used provide feedback to *any* of the previous stages of the process. You should always be on the lookout for gaps in the testing, as well as anything the tests might say about gaps in your implementation. In fact, it is often worthwhile to use your analysis to reexamine your goals, strategies and tactics, as well. Did you set the right goals? Should your goals be more specific, or less? Do you have the right strategy for achieving your goals? Did you leave anything out of your strategy or are there gaps? With the tests you have completed, is there new information you now realize that you need to be successful?

Your analysis can and should be used to reexamine all levels of your design, from goals to strategies to tactics to implementation, and even your tests and analysis. It is the perfect opportunity to look once again at the project as a whole, to see whether everything still fits together as well as you once thought, or whether something can be productively adjusted.