A Systems Engineering Approach to the Development of an Autonomous Sailing Vessel

B. E. Bishop, J. Bradshaw, C. Keef, and N. Taschner

Abstract Over the past four years, undergraduate students in the Systems Engineering degree program at the United States Naval Academy (USNA) have pursued autonomous sailboat systems development as part of their required capstone project in conjunction with students from the Naval Architecture program that design and fabricate the vessels. In this paper, we discuss the pedagogy of robotic sailing as a capstone for Systems Engineering students and discuss our approach to interdisciplinary tasks such as this. We also outline the design philosophy associated with the systems side of the equation, focusing on the on-board instrumentation and control hardware.

Bradley E, Bishop
United States Naval Academy, Annapolis, MD, USA, email: bishop@usna.edu

Joseph Bradshaw
United States Naval Academy, Annapolis, MD, USA, email: bradshaw@usna.edu

Cody Keef
United States Naval Academy, Annapolis, MD, USA, email:
cody.keef36@gmail.com

Nicholas Taschner
United States Naval Academy, Annapolis, MD, USA, email: m116552@usna.edu

1 The Pedagogy of Robotic Sailing Competitions

The United States Naval Academy (USNA) is a baccalaureate institution that provides to every student both a degree and a commission in the United States military, primarily the Navy or Marine Corps. Students in the Weapons and Systems Engineering department are required to complete a capstone project as part of their ABET-accredited degree in Systems Engineering. The students take a semester course on project management in the Fall semester of their 1/C (senior) year. In this course, students propose a project and are assigned a faculty advisor. In the following (Spring) semester, students execute their plan and produce a fully-integrated capstone project with a formal performance and design review.

Systems Engineering at USNA focuses on mechatronics, robotics and feedback control. As such, our projects tend to be very hardware-intensive and must include some form of autonomy or automation. Due to the marine-centric nature of our service, many students opt to work on projects that involve autonomous marine vehicles, both surface and underwater. Many students choose autonomous sailboats for their capstone projects due to their familiarity with sailing or from a desire to apply advanced control and automation in a competitive framework.

1.1 Robotic Sailing as a Capstone Project

While autonomous sailing vessels have been developed at USNA prior to the effort discussed in this paper, the origin of this work was initially the SailBot competition [2, 3], and later the World Robotic Sailing Championship (WRSC) [4]. It has been the point of view the USNA Systems Engineering department for many years that student participation in a competition can be both a blessing and a curse. While the students are highly motivated to win, it is easy for them to lose sight of good design principles and the requirements of a capstone project in the face of the immediate demands of the competition. It is our belief that winning the competition should never be the primary goal, but rather should be a consequence of a proper design experience that begins with an appropriate needs analysis.

While it seems logical that the winning entry in a competition will of needs be the best designed of the field, it is not necessarily the case. Stop-gap measures, kludges, and inelegant coding, wiring and design are the hallmarks of projects whose only metrics are victory at any cost. While one of these vessels may win a competition that spans a few days, they will often have deep design flaws that may not be obvious from their performance. These flaws in design are the hallmarks of a failed capstone project: one in which the students did not learn the appropriate lessons of design and built an overly-inflated view of kit bashing and caffeine-fueled desperation that will not stand the test of a real-world engineering problem.

It is therefore essential that students engaged in a capstone design whose output is to be used in a competition be guided appropriately through the proper engineering design, and be evaluated on the same. This is a problem of perception on the part of the students and expectation on the part of the advisor(s) and the host

department, both of which need to be managed carefully from the outset of the project.

1.2 Project Execution Methodology: SailBot

SailBot is a student-driven competition involving automated sailboats that compete in racing, navigation and station-keeping contests. SailBot class vessels are 2 m or less in length, while Open Class vessels can be up to 4 m in length. Full details of vessel requirements can be found in [2].

It is important to note that the SailBot project at USNA is an interdisciplinary effort between the Systems Engineering department and students pursuing a Naval Architecture degree. The Naval Architecture students design and build the vessel in a two-course sequence (Fall-Spring), and the Systems Engineers provide instrumentation, actuation and automation as a follow-on effort the next academic year. This process has yielded quality results for many years now, and there are important lessons that have been learned. In this section, we will discuss the formal pedagogy of SailBot as a capstone design project for the Systems Engineering students, starting with a completed but strictly R/C vessel generated by the Naval Architecture students.

Students in Systems Engineering at USNA are expected to formally define a full set of objectives, metrics, functions and demonstration plans for their design projects [1]. Through the remainder of this section, we will discuss an appropriate scope for the SailBot project and lay out the objectives and performance metrics. For purposes of this discussion, we will assume that the problem and needs statements for the design are completely defined by the fixed vessel (provided by the Naval Architecture students) and the SailBot competition rules [2, 3]. Further, there are constraints on weight and on size/volume of each component that come from the fixed vessel design and must be discussed with the Naval Architecture students. It is impossible, for example, to place a standard laptop PC into the SailBot designs that have thus far been generated at USNA due to the access hatch size and hull form, so selecting such a system is *a priori* prohibited.

1.2.1 Objectives and Functions

The first step in a proper system design, after a clear needs analysis, is to lay out the objectives of the design. An *objective* for a design is one of a set of characteristics that differentiate a *good* design from a bad one [1]. Objectives describe what the system *is*, not what it *does*, and a fully defined objectives tree is a crucial step toward developing a good project that meets the needs of the problem.

Students typically struggle to generate the appropriate depth and breadth of objectives for a complex project. Wanting to focus on what the vessel does (functions) and how it will accomplish those functions (means), students skip over the vital step of defining design objectives. The task presented to the students is to define the objectives and then push toward the functions that support those objec-

tives. The following is a discussion of the primary objectives that are crucial to a good SailBot system design (in bold), with the high-level functions that are needed to support them. There are some additional design objectives and further refinements that are not listed here due to space restrictions.

The system must *be*:

1. **Competitive**: this objective focuses on the tasks that are required for victory in the SailBot competition. The vessel and its controller must be:

   a. **Fast**: the system must achieve its navigation and sailing tasks quickly in order to compete.
   b. **Accurate**: the system must minimize error in navigation and in executing selected sailing commands
   c. **Energy-efficient**: the system must not waste battery power, as some of the tasks are long-duration

2. **Reliable**: the systems must be designed to reduce maintenance and update costs (in time and money):

   a. **Robust**: the system must be capable of carrying out its mission with minimal sustained damage over many cycles. Must be concerned with temporary power loss, water intrusion, damage from slamming (waves), and possible collisions
   b. **User-Friendly**: the system must admit quick setup, configuration, power recharge, component replacement and in-situ reconfiguration

Note that the full set of functions is not defined here, but there are several functions described above that would not be clear from the basic SailBot task. Students must fully enumerate the set of functions while avoiding solution bias once the full set of objectives is defined.

The set of functions (things that the system must *do*) is given by the implicit functions above as well as the following:

1. **Allow configuration of task**: the software or hardware must allow a user to select from a suite of tasks (station keeping, match racing, etc.) *and* configure the parameters of the task (maximum allowable deviation, GPS coordinates, etc.)

2. **Compute sailing commands**: the system must be able to determine appropriate sailing commands to control the available surfaces (typically, for our vessels, the rudder and the main sail).

3. **Carry out sailing commands**: the system must execute sailing commands, meaning that it must actuate the surfaces at the appropriate time and to the appropriate position.

4. **Measure appropriate vessel states**: the system must measure data as required by the sailing command generation subsystem. The exact nature of the data required should not be specified at this point.

5. **Avoid obstacles**: the system must recognize and avoid navigation obstacles, including other vessels and structures that might damage the vessel.

6. **Provide remote override**: the system must allow a human to remotely take control of all sailing functions in order to maintain safety.

7. **Transmit data**: the system must provide telemetry to the shore, indicating all data gathered and computed sailing commands.

8. **Provide power**: the system must provide power sufficient for all subsystems for the specified operational period.

1.2.2 Demonstration Plan and Metrics

The set of design objectives and functions above are the hallmarks of a good design, but they are vague and open to interpretation. Students must be provided with a method by which they can test and validate their design in a rigorous and meaningful way. It is here than many competition-based projects run into trouble. Specifically, it is inappropriate to design to a metric that cannot be evaluated during the design process. If victory at the competition is the only performance goal, it will be impossible for the students to carry out the iterative design of the individual components and subsystems that is required by a quality design process. As such, students pursuing SailBot at USNA must define a set of metrics by which their system can be assessed *before* the competition. This set of metrics must be driven by the tasks of the competition, and realistic performance levels must be enumerated (we use a 0 – 4 system, with 4 being the best result and 0 being unacceptable).

The following describe a full set of metrics for the primary objectives of the SailBot system, with the point values for each level of performance:

1. **Fast**: given a change in environment state or task configuration, the system with compute and reach a new course in:

4 points: less than 5 seconds
3 points: between 5 and 7 seconds
2 points: between 7 and 9 seconds
1 point : between 9 and 10 seconds
0 points: greater than 10 seconds

2. **Accurate**: passes within a specified distance, on a specified heading, of a target GPS point (corresponds to scoring buoys in races). The best score is achieved when passing inside the start marks, which are 3m apart. There are further marks an additional 3m beyond those start points:

   4 points: passes within 1m of target
   3 points: passes within 1-1.5m of target
   2 points: passes within 1.5-3m of target
   1 point : passes within 3-4.5m of target
   0 points: greater than 4.5m from target

3. **Energy-efficient**: the system must provide energy for endurance trials. The scoring rubric is battery life (in hours) divided by 2, with a max score of 4 (8 hours of battery is desirable).

4. **Robust**: this metric scores based on 1-point binary decisions for a total score of 0 - 4:
   +1: system automatically reboots and restarts configured task when power is interrupted
   +1: system components are watertight to 6”
   +1: system components are secured against shock (mean time between failures in normal operation of greater than one day)
   +1: system can function in any navigable weather condition

5. **User-Friendly**: again, this metric scores on a sequence of one-point binary decisions for a total of 0 – 4 points:

   +1: allows wireless programming and reconfiguration
   +1: allows for battery replacement underway
   +1: allows for one-point disconnect of any device or subsystem
   +1: allows for one-point access to any subsystem or component

It is well and good to define a set of metrics, but it is necessary that the system be designed to meet those metrics, and that there be a plan for testing the performance. We utilize the pairwise comparison chart (PCC), morphological chart and decision matrices to select components to achieve the functions and hopefully

meet the metrics [1]. Briefly, the PCC provides a weighting for each objective, relative to each other objective. The morphological chart enumerates possible methods by which each function can be achieved (e.g., sensing could be a GPS and an IMU or it could be a weather station; there might be sensors for sail pressure or for relative wind speed; etc.). A decision matrix is generated for each subsystem by taking each potential implementation for that subsystem and scoring it based on the appropriate metrics. An overall score for each possible solution is generated based on the scores and on the appropriate objective weightings. The potential solution with the highest composite score for each subsystem is the preliminary design. Details of this process are beyond the scope of this work, but can be found in [1].

The advisor plays a crucial role in the development of the preliminary design. Students will tend to gravitate toward well-understood or proven components to fill out the morphological chart, and will often resort to nonsensical entries to fill out a row. For example, when selecting components for the power system, a typical morphological chart will show: batteries (unspecified) and solar, with the occasional internal combustion or nuclear power option. Students must be guided to look into battery technologies and study their characteristics to select an appropriate suite of choices using the metrics and constraints.

Once a realistic set of options have been enumerated, the students must rely on the available data and their engineering insight to select the preliminary design using a decision matrix. Here, the advisor must assist with realistic scores for metrics that may be difficult to predict. Whenever possible, direct experience with the systems under consideration is appropriate. In 2011, the SailBot systems team developed, constructed and tested two completely different and independent sensing systems in order to evaluate the efficacy of each.

Having selected a preliminary design for each subsystem, a compatibility check is run. If two subsystems are incompatible, a decision must be made as to which to disallow. Assuming that such a decision can be made, a replacement for one or more of the incompatible systems is selected, starting with compatibility with the remaining system as a prerequisite. The process is carried out iteratively until a complete and compatible preliminary solution is obtained.

Finally, students must generate a demonstration plan by which they will test each component of the system and measure the actual performance. Again, guidance by the advisor is crucial during this step, as students will tend toward the "build the system and test the whole thing" approach. Students who work on SailBot are required to carry out the following tests on every subsystem:

1. **Component bench test**. Using a stable (wall) power supply and the least-complex interface available (manufacturer-provided testing software, simple RS232, etc.), demonstrate that the performance of the component meets its specs. This will often involve generation of test inputs and parameterized test runs, and will often result in extensive troubleshooting and (sometimes) a return to the decision matrix for an alternative solution.

E.g.: when testing a wind sensor, students provided power from a regulated wall source and mounted the sensor on a gimbal used for testing IMUs in our autonomous vehicles class. Students provided a reasonable airflow using a shop fan and looked at the system output on a dummy terminal using HyperTerminal as the attitude of the sensor was adjusted on the gimbal. The results of the test showed that the system could reliably measure airflow with a relative angle of up to 30 degrees. The test did NOT, however, show that the measurements were accurate. A good bench test would have a means of verifying the measured data.

2. **Interoperability test**. Here, each functional component or subsystem that interacts with any additional component(s) is tested with each. This is done first with each communicating component individually and then by adding components/subsystems individually (as is reasonable and achievable… extremely complex systems have a combinatorial explosion in this test and will require additional care in selecting subsystems to test). When and if failures occur, individual components can be isolated for further analysis and troubleshooting. This set of tests is only complete when the entire, fully-connected system is functional with stable wall power. It is crucial to point out that the system does not need to be fully programmed to test the interoperability, but that any and all communication modalities and control signals must be exercised to show their operation. It is typical to start with the processor in this test, adding peripheral components one at a time.

3. **Powered tests.** Using the designed power supply, each component is again tested to show full performance is met. This step is optional and may only be required if the multi-system powered test fails (see below).

4. **Multi-system powered test.** Additional loading to the power supply is provided by bringing up each subsystem that has passed the powered test, one at a time, and verifying that each maintains its performance and interoperability.

5. **In-situ test**. The components are installed and again tested.

6. **Full system trials.** Here, each capability of the system is tested as well as possible with variables removed. Navigation tests are conducted on land, moving the vessel on a cart. This is followed by on-water tests of basic GPS-based navigation, then maneuvering (tacking, etc.), and so forth, building up sophistication.

When students follow the design process, they will inevitably run across a subsystem that does not meet specs. This is where the iterative nature of the design becomes apparent, and where all of the preliminary work pays off. Students have fall-back options in place for any subsystem that fails, and can refine decisions

and metrics based on experience in the testing phase. Without this careful and systematic approach, students are often left accepting less than desirable performance because they simply cannot determine where things have gone awry and, in most cases, are not prepared to generate an alternative solution even if they did understand the primary failure mechanism.

1.3 Pedagogy of Robotic Sailing

The means by which students carry out their project are an end of sorts of their own, providing students with experience and insight into development of a complex system. It is unlikely that any student will be building robot sailboats for a living at any point after graduation, so the main outcome of the effort must of needs be a deeper understanding of the engineering process in general. It is true, however, that robotic sailing is a highly specialized task that involves all of the disparate aspects of systems engineering, from power through expert control. As students progress through the tasks associated with this project, there are several key points at which domain-specific lessons can be driven home, tying the design experience more closely to the students' core discipline and coursework.

Functional Blocks: Because the task of generating a complete sailing robotic system in just one semester is quite daunting, the planning phase becomes of vital importance. Students cannot afford the luxury of working together on every aspect of the task, and so must divide the work. Because of the tightly integrated nature of the systems, each sub-team must understand and work to a specific set of interface and power requirements. As such, every subsystem is managed as a 'functional block' in the overall system. Each sub-team leader is responsible for maintaining an accurate list of input requirements, output format, power requirements, and total volume and weight.

Budget: Because the vessels we use are optimized for sail racing, the budget aspect of the systems design goes well beyond the understood monetary level to include weight, power and volume. Because total battery capacity, weight, and volume are often related, tradeoffs must be made to select appropriate batteries. These decisions can impact available sensing and actuation technologies. Design to a four-part budget is a great exercise in optimization and planning.

Communication: Because this is a multi-disciplinary team, comprising systems engineers as well as naval architects, communication is especially challenging. The two disparate engineering disciplines do not necessarily speak the same language nor mutually understand the requirements of their distinct tasks. As such, a continual dialog is necessary. To accomplish this, there are advisors from both disciplines on the project, and representatives from the Systems team are required to attend meetings of the Naval Architects. Further, during a Fall semester course

offered to the Naval Architects, the Systems advisor gives an overview of the basics of the automation and control of the previous year's vessel and explains some of the common issues related to the systems and naval architecture interaction.

Planning: To execute a project of this nature in only one semester takes a commitment to the task and a solid plan. Students can be taught the importance of planning as this project progresses, especially if the deliverables include a full set of test results as discussed previously. When a week-by-week plan of action and milestones (POA&M) is completed as part of the preparatory course, students do not appreciate how much effort each of their tasks will take. We require our students to update the POA&M every week, and provide discussion on their progress. This allows the advisor to show the students how to generate optimality in the task order and how to better estimate actual required time. This is a skill that we find cannot be taught in a lecture, but which students rapidly appreciate when the realities of a complex task loom large and their grade is in the balance. To assist us at USNA, there are actually formal marking periods at the 6 and 12 week points in the semester, offering good targets for intermediate deliverables and maintaining a steady pressure on the students across the time of the project.

2 System Architecture of USNA SailBots

As mentioned, all of the vessels used by the USNA team for SailBot and WRSC are designed and constructed by students in the Naval Architecture department. Systems Engineering students primarily focus on selection and design of the power, sensing, control and communication subsystems and to some lesser extent the actuation. Over the last four years, the teams have developed a systems architecture that has proven robust and easily customizable, suited to wide variety of tasks. In this section, we will outline the basic components of the system and discuss the unique aspects of the system design.

2.1 *Basic System Architecture*

As discussed previously, the systems used for robotic sailing at USNA are developed with a 'functional block' approach. The primary subsystems and associated requirements are as follows, where each subsystem is (again) managed as a functional block that is under the purview of one subsystem manager who must communicate with the rest of the team to guarantee proper performance.

Processing: This subsystem is the heart of the robot. Here are carried out all of the computations required for planning and executing the assigned tasks. SailBot requires that all processing be on-board, and the form factor of the vessels de-

signed precludes the use of a standard laptop or netbook. Power issues and heat dissipation must also be considered in the selection process.

Actuation: Because the system is designed to allow a human to take over from shore or a chase boat at any time, the actuators are selected to be R/C friendly. As of now, only the main sail and the rudder are actuated, although actuators have been selected for independent control of the jib as well. The key concerns when choosing the actuators are power consumption, accuracy and speed. It is desirable to have actuators that have no required holding torque (using a mechanical brake or a non-backdrivable system such as a worm gear).

Communication: The system must communicate its state to a base station for monitoring and data collection, and must allow remote reconfiguration.

Control: The system must be able to take outputs from the processor and use them to manage the actuation. Further, there must be a remote control capability that is the system default (that is, when the processor fails the system automatically defaults to remote control).

Sensing: GPS, wind data, obstacle locations, etc. must be provided to the processor as needed for the appropriate algorithms.

Power: An integrated power system must be generated that will provide appropriate power to all components regardless of load. This often includes multiple supplies as well as regulation to achieve optimal balance of weight, volume and capacity.

Containment/Mounting: All system components must be securely mounted and protected from water incursions, which are common in small amounts. The components must be easily accessible and must fit through the access panels on the vessel.

Most of the components used in USNA SailBot are commercial off the shelf (COTS) products, from the R/C sail winch to the AirMar weather station used for wind measurement and GPS data. The full list of components for SailBot changes from year to year and is beyond the scope of this paper. There is one component, however, that is unique to USNA SailBot. In the following section, we will discuss that component and its implications for autonomous system control.

2.2 *The NavBoard3*

As mentioned, the form factor of most SailBot systems precludes the use of a laptop or netbook as the primary processor. While we have used a small form computer (Pico system) in the past, the Technical Support Department (TSD) within the Systems Engineering department at USNA has developed a custom computation and sensing board that has a small form factor, low overhead, moderate cost and high capability. The power and flexibility of this system has been proven in its use across a wide array of projects, but none so thoroughly as the autonomous robotic sailboat. In this section, we will discuss the key features of this crucial component and provide the overall design concept behind its development so that the interested reader can adopt and adapt the design principles to suit their needs and the wider needs of the autonomous vehicle community.

The Rabbit Navigation Board version 3.0 (NavBoard3) is a programmable single board computer that functions as an embedded controller for a variety of unmanned systems requiring navigation-based sensor feedback and output control signals. The system was designed to facilitate the ever-increasing number of unmanned system related projects hosted by the Systems Engineering department, from underwater autonomous vehicles, surface vessels and ground vehicles to a variety of aerial systems including helicopters, planes, and experimental craft.

The NavBoard3 uses a Rabbit 3000 processor core module programmed in a C programming design environment called Dynamic C. The core module can be programmed with a special cable using a PC RS232 serial communication port or over a wireless interface if a special boot-loader is used (as is done with SailBot to provide in-situ reprogramming).

The Rabbit 3000 processor, although relatively slow when compared to many modern processors, offers the advantages of a highly dependable In-Circuit Debugger for development, many built-in software libraries and functions, preemptive and cooperative software multitasking capability, large on-board memory capacity of 512K bytes of Flash program memory and 512K bytes SRAM, a Real Time Clock, and a multitude of additional input and output capability. A special NavBoard3 software library gives users access to the onboard hardware via standard C function calls, allowing the user to focus on the higher level overall system implementation and control algorithms as opposed to the low level data acquisition and digital communication interfaces.

This control board encompasses a variety of sensors including a 3-axis accelerometer and 3 angular rate gyros, a 3-axis magnetic field sensing module for deriving heading, an on-board Xbee wireless communication transceiver, four servo control ports with programmable pulse width modulation, and a GPS receiver. These sensors provide the ability to collect navigation information and sense external body rates and forces for robotic platform state estimation. There are many undedicated I/O ports to which the user can interface external sensors, both analog and digital. The NavBoard3 also has a variety of external communication bus interfaces such as I2C, SPI and serial UARTs, making it easy to interface with almost any COTS product. The overall layout of the custom board is shown in Fig. 1.

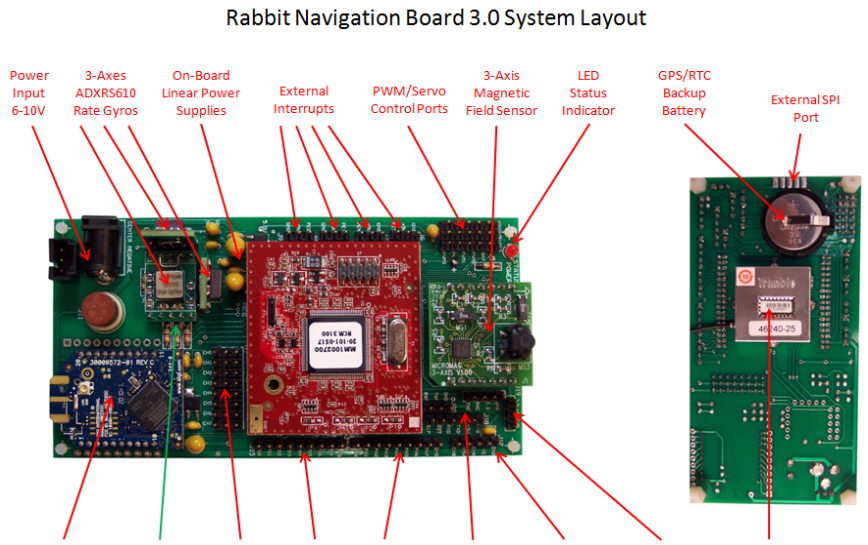

Fig. 1: The NavBoard3 layout. The board main face is shown on the left, with the view of the underside on the right. Green items indicate a component that cannot be seen in the given view (occluded by other components).

The NavBoard3 is modular and easy to maintain, with most components being easily removable for replacement. The components are all themselves COTS, and the PCB is manufactured in bulk. This device uses the processor that is the core programming and control learning platform for the department, reducing the learning curve for students adopting it as their processor. The integrated IMU capabilities reduce overall system complexity for autonomous vehicles, and allow for higher-level work on the part of the students. At the same time, any and all capabilities embedded in the board can be supplanted by external devices as needed.

3 Conclusions

Robotic sailing is a complex, interdisciplinary task that requires significant effort in real design. While competition-driven, it is possible to achieve high-quality pedagogical results using a systematic design methodology and appropriately scoped deliverables and metrics. In this paper, we have discussed an appropriate set of design tools and tactics for system design. We have given details of our interdisciplinary interaction model, which has been developed over four years of running this project. The best recommendation that we can provide is to generate each vessel as a two-stage process, with one year of effort on naval architecture followed by a year of system design. While this is challenging, the rewards are real and substantial, and actually improve the quality of the design process.

References

1. C. L. Dym, P. Little, E. J. Orwin, and R. E. Spjut. *Engineering Design: A Project-Based Introduction (3rd edition).* New York : John Wiley & Sons, Inc., (2009).
2. SailBot Class Rules. *SailBot: 5th International Robotic Sailing Competition.* [Online] December 23, 2010. [Cited: May 31, 2011.] http://www.usna.edu/Users/naome/phmiller/SailBot/SailBot.htm.
3. SailBot: Sailing Instructions. *SailBot: 5th International Robotic Sailing Competition.* [Online] December 23, 2010. [Cited: May 31, 2011.] http://www.usna.edu/Users/naome/phmiller/SailBot/SailBot.htm.
4. World Robotic Sailing Championship. [Online] INNOC - Österreichische Gesellschaft für innovative Computerwissenschaften , 2010. [Cited: May 31, 2011.] http://www.roboticsailing.org/.