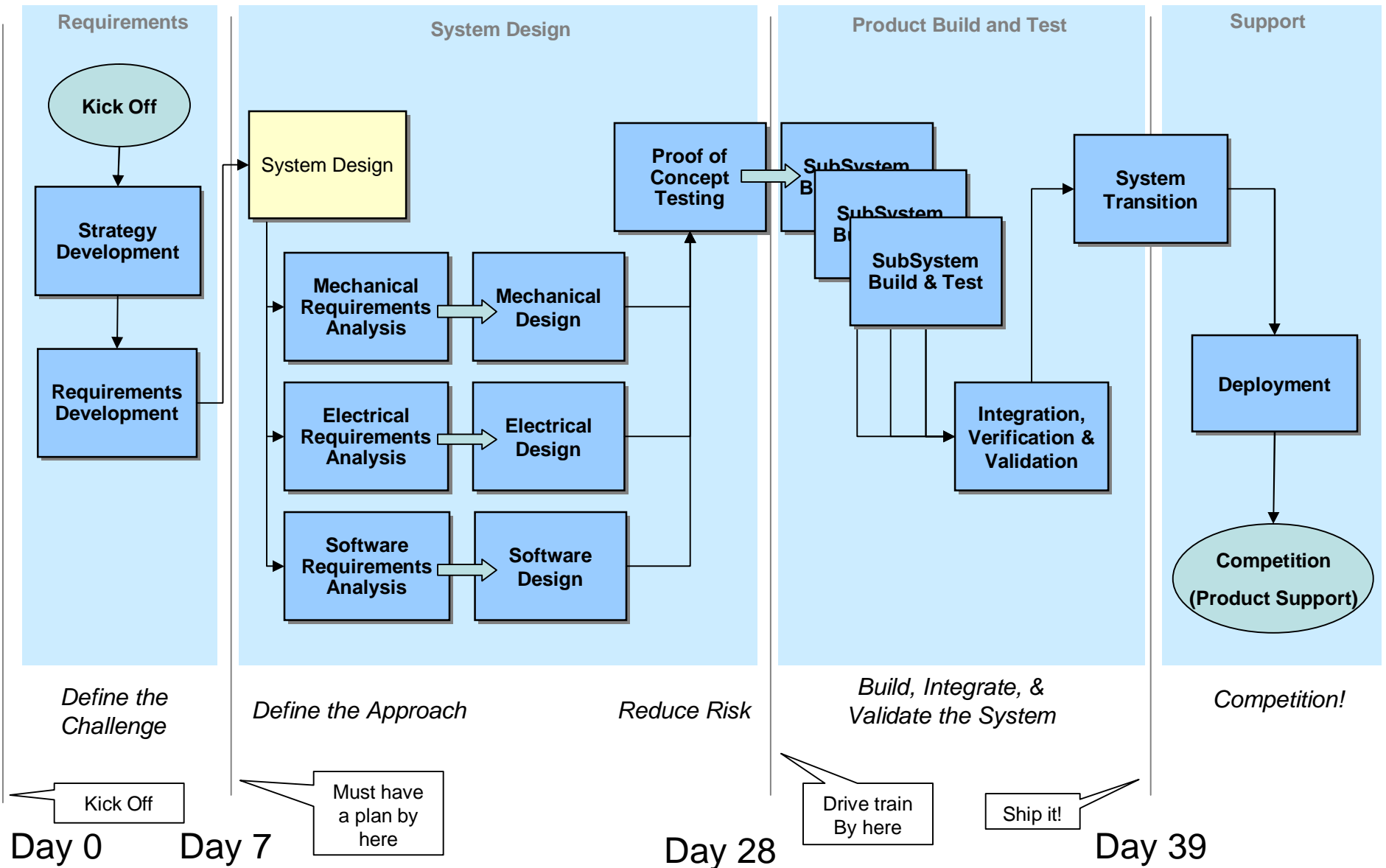


FIRST Robotics
A view from the
Systems Engineering Perspective

Chris Mikus
January 2, 2006
Rev 0.2

FIRST Robotics - Process Context

Robot Engineering Process Flow



Introduction

- The development of a FIRST Robot from the view of a Systems Engineer
 - Process oriented
 - Requirements driven
 - Product development cycle
 - Plan the task – set hard dates
 - Study the effects
- The process flow for a FIRST robot is more of a prototype build and test rather than a complete product development cycle, i.e., the system you are building is the system you are delivering.
- The **achievability** of the designs is crucial – an exotic design that cannot be completed in time is useless.
 - There will be less iteration in this short-term project than would be performed with more time available.
 - Participants must look at the phase of the project they are in – as described in these slides – and also look ahead to identify and address possible problems.
 - A design can fail just as surely because of missing data and analysis as because of missing components.

What does a Systems Engineer focus on?

- Overall design of the product
- Understanding the required operation
- Translating the requirements to feasible design
- Planning the individual technical tasks and interrelationships
- Making a workable schedule and meeting it
- Performance of the delivered product
- Reliability of the delivered product
- Robustness of the delivered product

The Challenge drives the design

- Example Elements from Previous Games
 - Human and robot team members required to shoot balls through hoop to score points
 - Human and robot must work together to satisfy requirements
 - Robot can shoot balls for more points, human feeds balls into robot
 - Human can shoot balls through hoop, robot collects balls
 - Human can shoot balls through hoop, robot can block other robot
 - Robots operated autonomously for first 15 seconds of each 2-minute round, then under operator control.
 - Number of robots in test area simultaneously
 - One only
 - Two competing
 - Three competing as individuals
 - Two teams of two competing

Strategy Development

- Concept of Operations Development
 - Understand the game requirements – robot and human
 - What activities accumulate points?
 - Agree on a strategy to collect points – as an individual and as a partner
 - Brainstorm the maneuvers that your teams needs to accomplish in order to satisfy your strategy
- Example questions to explore
 - What will the robot do – autonomously and controlled?
 - What will the human player do?
 - What are your top 3 offensive and defensive maneuvers?
 - What is the maximum score your team can expect to achieve on its own? With a partner?
- Exit Criteria:
 - A clear description of how your team is going to play the game.

Requirements

Requirements Development

- Requirements Development
 - Brainstorm the attributes that your team will build into your robot in order to satisfy your Concept of Operations.
- Example questions to explore
 - How will you retrieve and place game objects?
 - What type of chassis will you need (wheels, tracks, 4WD, etc)?
 - Will you need a lift? An arm? A scoop? Other?
 - How long will it take for your robot to perform its tasks?
- Exit Criteria:
 - A clear definition of **what** your robot will do and **how** it will do it.
 - Produce a prioritized list that the designers can work from

Requirements

Requirements Example

1. Our robot must start low and double its height to reach the goal in order to be offensive.
2. Our robot must climb the stairs in order to be defensive.
3. Our robot shall carry both large and small scoring objects.
4. Our robot may score during the autonomous period.
5. Our human player will load our robot.

Advice

Mechanical Requirements Analysis

- Mechanical Requirements Analysis
 - What motions and features are needed to perform your strategy?
 - *We need a lift platform to raise up our capture / dispenser subsystem.*
- Types of questions to explore
 - What capabilities must your robot have?
 - Maneuverability – 2 WD, 4 WD, tracks?
 - Speed – Does your strategy require a certain pace?
 - Controllability – Can it be driven?
 - Can we use a scissor lift, Archimedes screw or pulley lift?
- Exit Criteria:
 - Analysis or model results that indicate design will satisfy requirements

Electrical Requirements Analysis

- Electrical Requirements Analysis
 - What controls and feedback are needed to perform your strategy?
- Types of questions to explore
 - What controls must your robot have?
 - Controllability – Single or dual stick drive?
 - Sensors – What can switches and sensors provide?
 - Can the required controlled and autonomous behavior be achieved?
 - Can available components provide the required mobility and motion? How? With what margin?
 - Etc
- Exit Criteria:
 - Analysis or model results that indicate design will satisfy requirements

System Design

Software Requirements Analysis

- Software Requirements Analysis
 - What capabilities can software provide to improve your strategy?
- Types of questions to explore
 - Will the default software work as is?
 - Can you add efficiency with additional software?
 - Can you increase reliability with additional software?
- Exit Criteria:
 - Analysis or model results that indicate design will satisfy requirements

System Design

- System Design
 - What components can be used to satisfy the individual requirements?
 - What is the best way to assemble the pieces required?
 - How can these components be assembled into a working achievable robot?
- Types of questions to explorer
 - What stock systems can you use? What custom systems do you need to build?
 - Consider your capability to build the selected design, not just the capabilities of the design itself
 - If your design calls for a lift, what type should you build
 - A scissors lift
 - A elevator and cable lift
 - Do you have the capability, skills, experience to build such a device?
 - Can your design met the schedule and budget?
- Exit Criteria:
 - A clear definition of
 - the subsystems your robot will be made of
 - how they will interoperate
 - how they will be obtained

System Design

How is System Design done?

- System design is accomplished by examining the needs of the product
 - Understand the problem from many angles
- Educating the designers
 - Making sure the people performing this work have the information they need to make solid decisions
- Communicating among your team
 - Let others know what your intentions are
- Drawing in the strengths of your team
 - Leverage the skills and knowledge you all ready have
- Filling any voids
 - Seek the skills and knowledge you need – within your time constraints
- Driving the assembly
 - What is the best way to assemble the pieces required?

System Design

Proof of Concept Testing

- Proof of Concept Testing
 - Small-scale table top experiments to reduce risk. Hands on.
- Types of questions to explore
 - Does the theory match reality?
 - Will that motor / gear ratio real lift that load?
 - Will the chassis go as fast as expected? Carry the load?
- Exit Criteria:
 - Design changes

Build and Subsystem Test

- Build and CI Test
 - Final Fabrication
 - Subsystem Assembly

Integration, Validation & Verification

- Integrations, Validation & Verification
 - System assembly
 - Does it work as expected?
 - Will this be adequate?

System Transition

- System Transition
 - Out of the workshop and onto the field
 - From engineers to users (the drivers and coaches)
 - Pre-event Unveiling Event
 - Shipping
 - Hand off to the drive and pit team

Product Build and Test → Support

Deployment

- Deployment
 - How does it function on the real field, with other robots
 - Creating replacement parts

Support

Product Support

- Product Support
 - Further refinement
 - Fix breaks
 - Pit support

Support

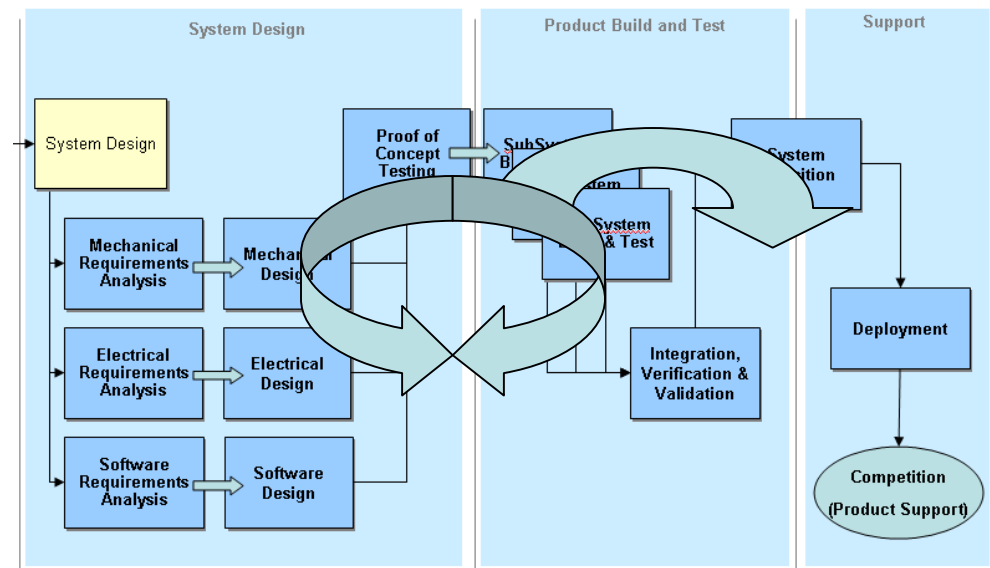
Feedback

- Post Mortem Collection
 - What worked well
 - What didn't work so well
 - Did you meet the schedule
 - Did you meet the budget
 - Feed into next year

Support

Getting stuck

- Avoid getting stuck on the loop of getting stuck in one place and not making progress.
- Too much planning
- Lack of design
- Poor decision making



Advice