Countdown to Launch

Managing New Product Development

Prepared by: Dan McCarthy

15.875 Applications of System Dynamics

Agenda

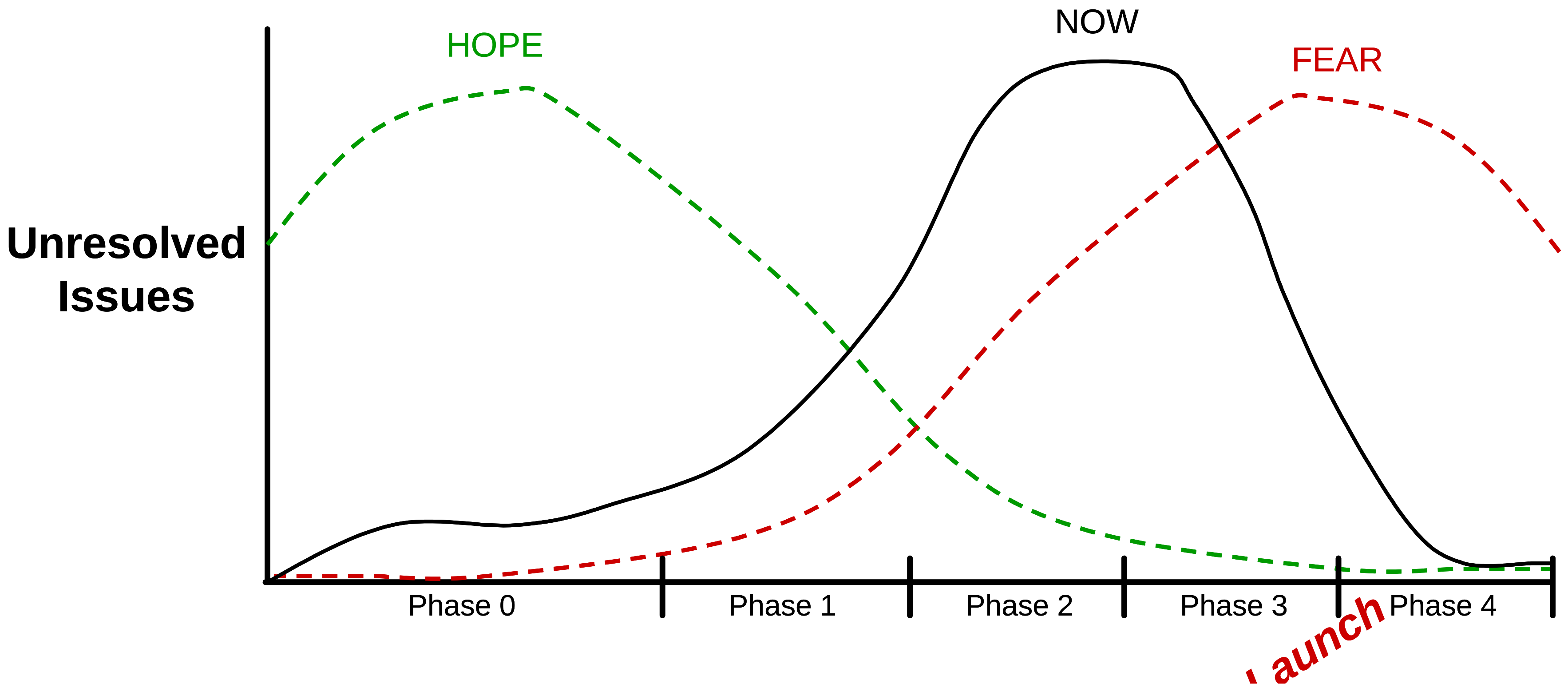
- Company Background
- The Problem
- The Process
- The Insights
  - Insight #1: ***You don't know what you don't know***
  - Insight #2: ***If you build it, they will come…***
  - Insight #3: ***…but at what cost?***
  - Insight #4: ***Parts ain't parts***
- Lessons Learned (Mine)
- Future Work
- Questions

Company Background

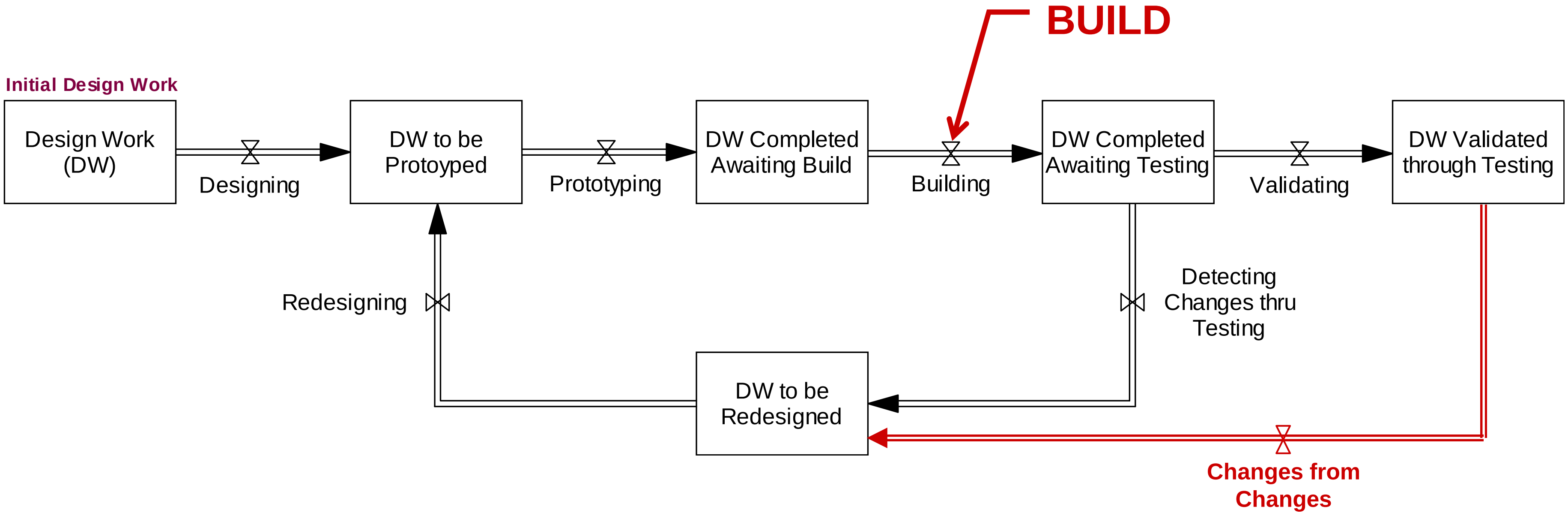
- The Early Years
- The 70’s
  - Takeover & Rapid Growth
  - Foreign Competition & Poor Quality
  - Low Market Share & Near Bankruptcy
- The 80’s
  - Buyout from within & 40% Reduction
  - Back to the Basics
  - TQM & JIT
- The 90’s
  - Focus on Product Development
  - Concurrent Engineering
- Today
  - Growth in Overall Demand
  - Demand for Innovation

Problem Statement

We have been experiencing difficulty managing our New Product Development process resulting in a large number of unresolved issues and design changes very late in the product development life-cycle. We want to develop policies that will turn this situation around.



The Product Development Process



- There is a large variation in iteration / cycle time among components between builds
- Testing takes time
- Prototyping can be the long pole in the tent (key opportunity)
- Simultaneous launch of products (projects)

Insight #1: *You don’t know what you don’t know*

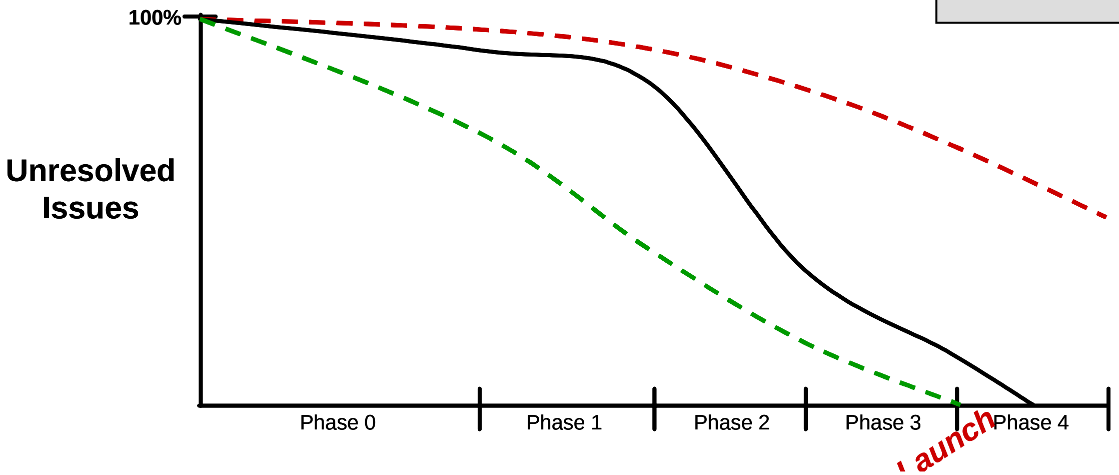
- **It can be hard to define the problem (symptoms of the problem) in terms that are quantifiable**
  - Firefighting
  - Quality
  - Unresolved issues
- **The client doesn't always know what they don't know:**
  - Unresolved issues **vs** Unresolved AND UNDISCOVERED issues
  - Manufacturing **vs** DESIGN

Two Problems:

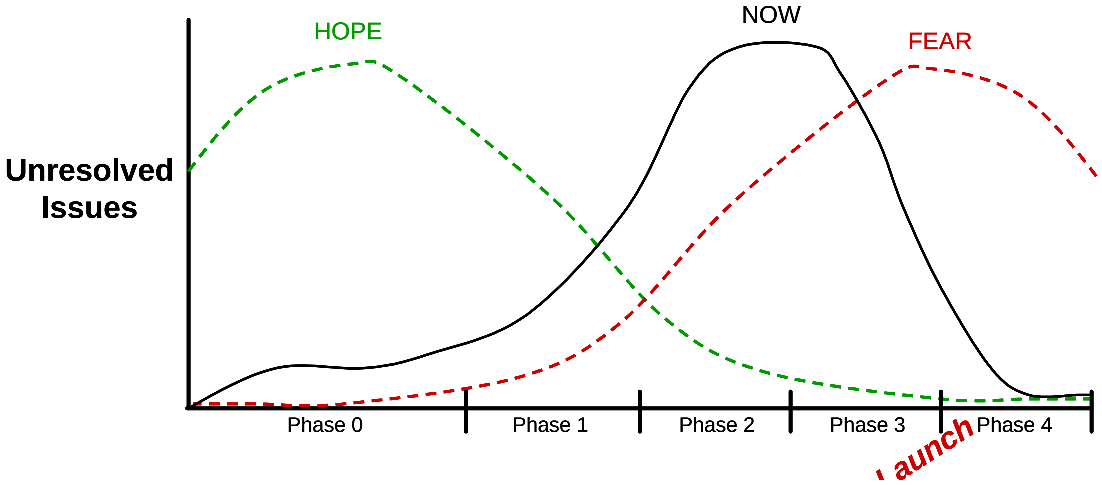
Discovery of Issues

Resolution of Issues

NOW **HOPE** **FEAR**



Original



Revised

Insight #1: *You don't know what you don't know*

- **It can be hard to define the problem (symptoms of the problem) in terms that are quantifiable**
  - Firefighting
  - Quality
  - Unresolved issues
- **The client doesn't always know what they don't know:**
  - Unresolved issues **vs** Unresolved AND UNDISCOVERED issues
  - Manufacturing **vs** DESIGN
- **Implications on modeling**
  - Traditional Project Model (Drains a stock of work)
    - Presumes a finite and known set of tasks (issues)
    - Undiscovered Rework (not undiscovered to the modeler)
  - Issues can be a complex notion
    - Fit **vs** Function
    - Component **vs** Integration
    - How and when they are discovered / resolved

Insight #2: *If you build it, they will come…*

- Reviews: Project reviews, End of Phase Reviews, Mock-up Reviews, ***BUILDS***
- Reviews are only as useful as:
  - The amount of ***active participation*** in the event
  - The amount of ***realism*** brought to the event
  - The amount of ***perceived commitment*** achieved
- Reviews are important when the ***boss*** shows up
- ***BUILDS*** require all of the above
- Builds (and subsequent testing) are when most issues are ***discovered***
- Meaningful ***integration*** rarely happens outside of build events
- Conclusion: ***HAVE MORE BUILDS!***

Insight #3: *…but at what cost?*

- “***More Builds***” has many implications:
  - Longer test queues → less ***value-added engineering***
  - More parts to procure/track → less value-added engineering
  - Higher cost
- ***Less time between builds*** involves a tradeoff:
  - Quicker iterations help to discover problems sooner
  - Less time to solve problems / ***less integration*** of designs
  - ***Learning leapfrogs*** the builds when not all components are “ready” for the build
  - Resolve issues sooner but with more overall work (cost)
- ***Building more bikes*** at a build has additional implications:
  - Even longer test queues
  - Longer prototype lead time
  - More parts to procure/track
  - Higher cost
- ***What is the right number / time between builds?***

Insight #4: *Parts ain't parts*

- How do you measure the success of a project? (***How do you know a project is in trouble?***)
- Assessing Success (Risk) for a New Product Design project is not easy:
  - **During the Project**: Tracking Parts (# Authorized)
    - ***Parts ain't parts***: Washer vs Fuel Tank
      - Require different amounts of effort (variability/uncertainty of effort)
      - Have different interaction effects on other parts (propagating changes)
    - Level of aggregation: part, component, sub-system, motorcycle model, platform
  - **Prior to the project**: Assessing Risk / Complexity
    - # new parts / # new suppliers
    - new technology (product / process)
    - # interacting components
    - volume of product(s) affected
    - # and scope of competing projects

Insight #4: *Parts ain't parts (cont)*

- Level of Aggregation / Unit of analysis is important in shaping the modeling effort:

  Do we need/want to track parts?

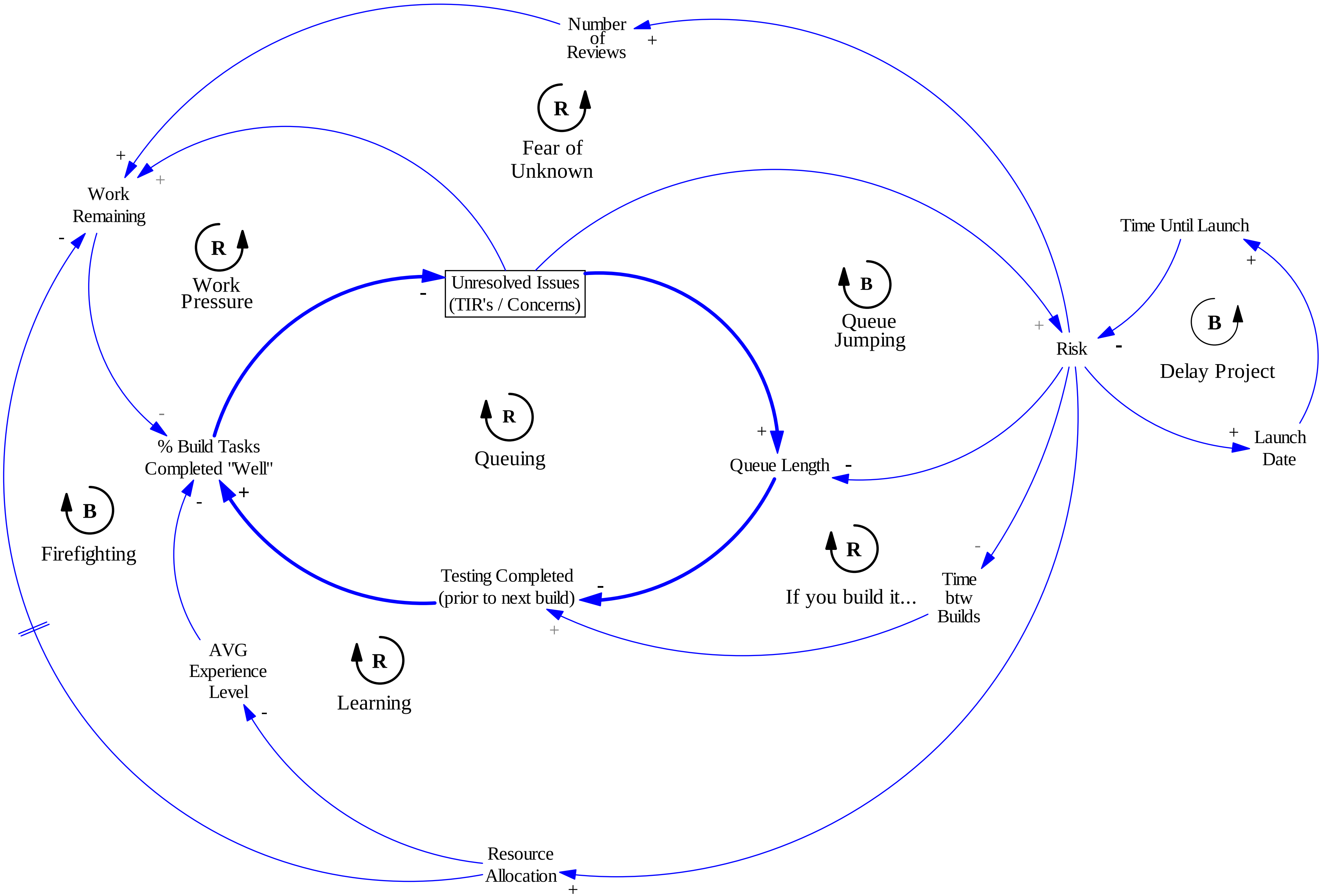
  – To understand the dynamic behavior of the system?

  – To develop a useful training tool for PM's?

  – To develop a decision support tool for management?

Lessons Learned (Mine)

- All clients are not created equal
  - Having an SD-informed client can be a double-edged sword
  - Be weary of the client who wants an “outside” opinion
- Central loop doesn’t mean the simple loop when modeling


Number of Reviews
+
+
+
R
Fear of Unknown
Work Remaining
-
R
Work Pressure
Unresolved Issues (TIR's / Concerns)
-
Time Until Launch
+
B
Queue Jumping
+
Risk
-
B
Delay Project
-
% Build Tasks Completed "Well"
R
Queuing
+
Queue Length
-
+
Launch Date
+
-
B
Firefighting
Testing Completed (prior to next build)
-
R
If you build it...
-
Time btw Builds
+
AVG Experience Level
-
R
Learning
Resource Allocation
+

Lessons Learned (Mine)

- All clients are not created equal
  - Having an SD-informed client can be a double-edged sword
  - Be weary of the client who wants an “outside” opinion
- Central loop doesn’t mean the simple loop when modeling
- Central loop can mean a lot of learning about the problem / difficulties of modeling
- There’s more than one way to skin a cat (model a problem) and trying a few (many) different ways doesn’t guarantee that you’ll get the fur
- Having it all done “right” is not a requirement for getting useful feedback
  - Designer to Customer (VP Rides)
  - Consultant to client
- Is SD right for this problem?

Golden Insight

The company is convinced that the ***structure*** of their New Product Development process, **not the project managers**, ***is responsible for the problems*** they have been experiencing

Future Work

- Expanded Model of Single Project
- Flight Simulator based on the Single Project Model
  - *Training tool for Project Managers*
- Multi-Project Model
  - *Decision Support Tool (DST) for HD management*
- Flight Simulator based on the Multi-Project Model
  - *Training tool for Project Managers and HD Management*
- Dan’s dissertation

Questions