



# Bridging the Gap at Raytheon

What is lacking when a college grad walks thru the door and what we do about it.

Janet Bratton  
Kathy Noguees  
April 25, 2008

# Author Contact Info

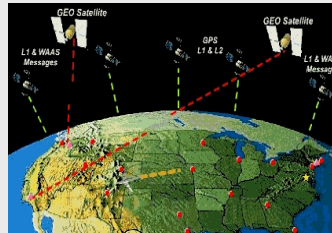
- Janet Bratton
- Network Centric Systems Western  
Regional Raytheon Six Sigma Lead
- Raytheon Six Sigma Expert –  
currently pursuing certification
- Sr. Manager Software Engineering
- Network Centric Systems
- Raytheon Company
- Fullerton, CA
- [JABratton@Raytheon.com](mailto:JABratton@Raytheon.com)
- 714-446-4502
- Kathy Nogues
- Engineering Training Manager,  
2003 – Present
- Raytheon Six Sigma Specialist
- Principal Software Engineer
- Network Centric Systems
- Raytheon Company
- Fullerton, CA
- [KLNogues@Raytheon.com](mailto:KLNogues@Raytheon.com)
- 714-446-4219

# Raytheon - Who we are

## Major Programs

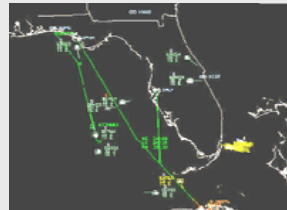
### Raytheon Systems

- **Tactical Communications Systems**  
EPLRS - Enhanced Position Location Reporting System. High bandwidth tactical data link. A critical component of the Tactical Internet.
- **Airspace Management & Homeland Security**  
Satellite Navigation Systems,  
Navigation Landing Systems,
- **HTMS - Highway Traffic Management System**  
Automated highway toll collection



### Thales Raytheon Systems

- **Air Defense Radars**  
Most modern 3D, long range, fixed/mobile multi mode, solid state radar on the market
- **Battlefield Radars**  
Battlefield surveillance and projectile tracking radars
- **Tactical and Operational level C2 system**  
Air defense and Joint operations systems



## Fullerton, California



**73,000 Employees World Wide**

**1100 engineers in Fullerton –  
450 SW, 350 SE, 200 HW**

**Projects are typically very large -  
Typical project size 50 to 220 software  
engineers.**

**Decade of focus on process  
improvement.**

**June 2007 - SEI CMMI Level 5 across 5  
Raytheon sites - SW, SE & HW**

# Bridging the Gap at Raytheon

- Gaps
  - From Managers' view
  - From College Grads' view
- What we do about it

# Managers' View

- They need to develop an awareness of focusing on customer needs (via communication, documentation of rqmts, demos, etc).
- Software development processes –
  - Understanding Software Requirements
  - How To Test Your Code, the Art of Unit Testing and Integration
  - How do you go about debugging your software?
- New grads seem uncomfortable with Peer Reviews - feeling like they are being graded - versus feeling like they are learning and sharing with their peer
- They may know the syntax of a software language, but have a difficult time developing/changing software when suddenly confronted with 10's of thousands of lines of code.
- How to program complex algorithms?

# College Grads' View

- CMMI- a general CMMI overview should be explained so that young engineers know how important it is in real practice
- Technical Writing was not emphasized in college, when in reality it is important to have well written, clear, testable requirements
- SW code maintenance - practice adding to or modifying code that already exists
- Relate the technical curriculum to real-world applications, especially in the defense industry

# What We Do To Bridge the Gap



- High School Intern Program –
  - Troy and Sunny Hills
- College Intern Program –
  - CSUF, UCI, Cal Poly
- New Hire Orientation
- PERCH –
  - Program for
  - Engineering
  - Rotations of
  - College
  - Hires
- Mentoring
- Formal Training
- OJT - On the Job Training
- IDP - Individual Development Plan

# Training We Provide



- Domain Specific
  - Ada83/C++/JAVA Swing/Assembly Language
  - Air Defense/Radio/Command and Control/Navigational Landing
- Processes
  - Peer Review
  - R6Sigma
  - SEI CMMI
- SW Lifecycle
  - SW Architecture
  - Requirements analysis
  - Code and Unit Test
  - IV&V
  - Maintenance

# More Training We Provide



- Principles of Systems Engineering
- Technical Writing
- Proposal Writing
- Front Line Leadership
- SW Program Management
- Working in Teams
- Communication skills
- Presentation skills
- Generational awareness
- Diversity

# Summary

- We need Team Players that are Problem Solvers with good Communication skills
- Colleges provide the building blocks
- Raytheon provides domain specific, process training
- Raytheon partners with the Universities to make sure our needs are met

# Bridging the Gap at Raytheon

Questions?



# Back up slides

- These slides are the responses by the SW managers and college grads to a survey on what the gaps are at Raytheon in Fullerton.

# Managers' View

- - The actual mechanics of working through our software development processes is overwhelming. School only teaches you about the high-level concepts of the software development cycle, but not necessarily the actual step-by-step process needed to develop, test and release software.
- - During school, students only develop a handful of small programs. They may know the syntax of a software language, but have a difficult time developing/changing software when suddenly confronted with 10's of thousands of lines of code.
- - How do you go about debugging your software? Unfortunately, a lot of it is knowledge gained through experience, but there are basic techniques and guidance that could be taught through a class/learning environment. <SW manager 1>
- The ones I have interviewed seem to know about the Software Development Cycle (Code, Unit Test and Integration). Our grads want to jump right into coding and slide to Integration. They need to learn the importance of all three (Code, Unit Test and Integration). Training on "Understanding Software Requirements" and "How To Test Your Code, the Art of Unit Testing and Integration" is where I think more training would be helpful. <SW manager 2>
- I think the concept of peer reviews would be one that I would like colleges to have their students try out. Our new grads seem uncomfortable with peer reviews - feeling like they are being graded - versus feeling like they are learning and sharing with their peer. <SW manager 3>
- The problem is that they're coming in the door hoping to be great software developers and that does not always align with solving the customer's problem. What's missing is the practical application of software development methods. That is something that could be taught.

# College Grads' View

- There is only one "Software Engineering" class required at CSUF for undergrads. One item that is skipped over is CMMI. I think that the gates & a general CMMI overview should be explained a little bit so that young engineers know how important it is in real practice.

Also, there's an elective track a student can choose that consists of the following classes

CPSC 462 (Software Design) / CPSC 463 (Software Testing) / CPSC 464 (Software Architecture) / CPSC 466 (Software Process)

These classes are also electives for the Master's Programs, so the majority of the spots go to Master's students. The majority of the classes are only offered once a year which makes this elective track not look too appealing. Personally for me, I waited on these classes because I want to take them as a masters student. <SW Engineer>

- Technical Writing was not emphasized in college, when in reality it is important to have well written, clear, testable requirements <SE Engineer>
- I think schools are not doing enough to relate the technical curriculum to real-world applications, especially in the defense industry. I have not run across any new grad that know anything about an Air Defense or Command Control (C2) Systems, including myself when I first got hired. Over the years I have seen many young software engineers coding different aspects of an C2/C4I system, but not knowing how it will be used in the operational environment. Lunch time series like APL, ACCS, or EPLRS 101 helps the the new hires and I think its should be part of their required trainings. We can also develop more general trainings like Defense System 101 or Crash Course in C2/C4I Systems and extend them to college students. <SE Engineer>

# College Grads' View

- I don't know how practical this is for school, but the students could practice adding to or modifying code that already exists. For example, the professor could have a three part project.
- Part 1: Each student writes a program that satisfies requirements list A.
- Part 2: The students get randomly assigned someone else's code from Part 1, then they must either add to or modify this code to satisfy requirements from requirements list B.
- Part 3: The students get randomly assigned someone else's code from Part 2, then they must either add to or modify this code to satisfy requirements from requirements list C.
- When you join an existing project that is already up and running, you may go through a process similar to the one above. From my experience in school, I usually wrote code from scratch. Rarely would I be assigned to modify/add to existing code. This exercise would probably just force the students to adapt to styles of coding other than their own. <SW Engineer>