Real World Software Engineering

• January 17, 2017
• Heather Ellsworth
• Test Engineer

Customer Requirements

• What techniques does your organization use to understand, specify, and communicate what the customer wants?
  - Understand: Customer input comes from multiple sources
    - Customer Visits
    - Understand both functional and operational needs
    - Input from Sales Department
    - Product Owner (Agile definition) Input
    - State of the art: tech press articles, RFC’s, conferences
    - Industry trends: NFV virtualization, Big Data
    - Support Calls
    - Specify & Communicate
    - Rally User Stories
  - In hockey terms: Understanding “where the puck is going” as opposed to “where the puck is now.” Listening to customers can be dangerously close to the latter.
Problem Solving

• B. What techniques do teams and individuals use to understand and solve problems? We are especially interested in techniques that go beyond algorithm development.

• Developing New Functionality

  • Problem solving requires the ability to do abstraction. Taking details from the 100 ft. level and elevating the discussion to the bigger picture at the 10,000 ft. level. What solutions would be scalable, resilient, elegant, performant, etc.
  • Planning, design review, rinse, repeat.
  • Integration not only with Secure64, but with the rest of the world (BGP/Cisco)

Problem Solving

• B. What techniques do teams and individuals use to understand and solve problems? We are especially interested in techniques that go beyond algorithm development.

• Investigating Customer Issues

  • What should we do vs what are we doing – full understanding
  • Duplication of the ENTIRE customer environment as closely as possible
  • Approach to solving the problem is very situationally dependent.
  • Understand the issue as completely as possible, before proposing a fix. Can the customer get involved in verification process?
Teamwork

C. What teamwork skills do you expect? What does ‘teamwork’ mean in your company?

• Teamwork is mission critical - If you can’t work in a team, you can’t work in our company

• Teamwork takes many forms:
  - Pairing new engineers with experienced engineers – they each learn from the other
  - Peer code reviews
  - Design reviews
  - Task swarming (Agile term)
  - Cross department help (engineering backs up sales, sales engineers back up support, etc…)
  - One big team

Processes & Tools

D. What collaborative tools does your organization use?

• Light Agile Process
  - Standup Meeting
  - Small Vertical Slices
  - Iteration Demos to rest of the company
  - Flexibility
  - Test Coverage Analysis
  - Coverity
  - Source Control (SVN and GIT)
  - Rally (Agile Project Management Tool)
  - Defect Tracking System
  - Iteration, Release & Requirements Management
Communication

- E. What kind of communication skills do you expect? E.g. for written communications—short messages/documents/slides, for verbal communications—presentations, face-to-face conferences, distributed conferencing, etc.
  - Written communication
    - Email: Concise, articulate, clear
    - Slack: Quick response time, broader audience
    - Dev documentation: wiki pages, test plans
  - Verbal communication
    - Approachable, amicable
    - Speak up and ask questions
  - We do have a documentation person skilled at translating from “engineer” to “customer”

Engineering and Teamwork

Software Engineering at HP Inc.
Nathan Nuber / Jan 17, 2017
Understanding the Needs of the Customer

The feedback cycle

• **Customers** tell the **Product Owner** what they want...
  - “I want to do X. Add Y to the program so I can.”

• **Product Owner** adds user stories to the backlog...
  - “As a user, I want Y so that I can X.”
  - **What** and **Why**

• **Developers** release new features to satisfy user stories.

• Rinse and repeat.
Anatomy of a User Story

• A one-sentence “story”
  “As a user, I want the program to remember my payment information, so that I can save time at checkout.”

• Acceptance Criteria
  • Store data so that entering payment info can be skipped by the user
  • Stored data must be encrypted
  • Allow the user to delete old payment information through settings

Developers and Product Owner revisit the backlog of user stories for clarity.

What Makes a good developer?
What Makes a Good Developer?

“I can write it all myself.”

The Hacker-Ninja

• Code of Silence: Doesn’t ask questions, questions are a sign of weakness.

• Works in the Shadows: Accepts a task and disappears until it’s done.

• One Man Army: Works alone. Doesn’t need help.
What Makes a Good Developer?

“I can write it all myself.”

The Hacker Ninja

• Code of Silence: Doesn’t ask questions, questions are a sign of weakness.
  • Interprets instructions incorrectly, works on the wrong things.
• Vanishing Act: Accepts a task and disappears until it’s done.
  • Unpredictable—When will the work be done?
• One Man Army: Works alone. Doesn’t need help.
  • Doesn’t offer help.
  • Gets stuck and doesn’t ask questions.

What Makes a Good Developer?

“Hidey-Ho Neighbor!”

The Ned Flanders of Software

• Friendly: Approachable and Forgiving
• Helpful: Offers and accepts constructive criticism
• Communicative: Asks questions, gives answers, responds quickly

• Even if Ned isn’t the best programmer, Ned is the best engineer.
What is good communication?

- Concise
- Specific
- Constructive
- Timely
Collaboration Tools

**Code Inspection**

- Face to face meetings are best for communicating design, provides context
- “Code Collaborator” for code review

**Code Repository**

- SVN or GIT
- Source Control used to maintain a central repository.
- Central repo is the “true state” of the code base
Collaboration Tools

Issue Management

- CA Agile Central or Jira
- Geared towards Agile/SCRUM process
- Same Basic Capabilities
  - Define issues (User Stories, Bugs, misc. Tasks)
  - Predict how long issue will take
  - Put issues into “Sprints” (work iterations)
  - Track To-Do/in-progress/complete

- All about communication and predictability

Thank you

Nathan Nuber
HP Inc.
CS314 Panel Discussion
Ron Vaughn, January 17, 2017

Collaborative tools

Mastering tools is very important

- Source control: internal git, perforce
- Code review: gerrit
- Bug tracking: internal system (think bugzilla)
- Project management: Jira
- Development tools: gcc, python interpreter, java interpreter
- Info, docs: Wiki pages, Confluence, cloud storage, PowerPoint, Word
- Build farm: internal system
- Automated test farm: internal system

All of these systems are tied together
All of these systems are way more complicated than new-hires are used to
You have to master the tools to be an effective engineer
Collaborative tools
Using git in college is **not** like using git at Nvidia

- A single source code tree:
  - 700 individual git repositories
  - Each at the correct location, each at the correct branch
  - 50G of source (before you compile)

On a scale of 1 to 10, you use git at a ‘4’. We go to 11.

---

Software Development Processes
500 engineers submitting code to the same project...

- The development process is required because of the scale of the projects
  - Hundreds of developers submitting code
  - Multiple customers or partners
  - Multiple releases
  - Multiple hardware platforms
- The development process can be dictated by the customer or product category (e.g., ASIL-D for automotive)
- Nvidia is gravitating towards Agile. It varies by team.
- “Software engineering” (scoping, documentation, meetings, code reviews etc.) can easily be 50% of an engineer’s job
Teamwork skills

- Become an Nvidian, be a team player
- Help teammates in need -- code reviews, code is broken, brainstorming, explain
- Don’t work in a vacuum
  - Understand your project’s bigger picture
  - Understand Nvidia’s bigger picture

Communication skills at Nvidia

- Email
- Email
- Email
- Email
- Email
Communication skills
OK... There is more than email

- Concise emails are invaluable
  - Communicate at the right level for your audience
- Maintaining wiki pages
- Running a meeting, with concise meeting notes
- Concise PowerPoint presentations
  - Project overview, schedule, software processes
- Software architecture, requirements, design documents

Good communication skills increase your impact as an engineer
Good communication skills are a key step to management