

My Experience with an Agile Software Development Process

John Bartholomew
Nethra Imaging, Inc.
Beaverton OR

10/28/09

2009 Pacific NW Software Quality Conference



Background

- **Software professional with 20+ years experience in EDA and semiconductor tools development**
- **Seen many development processes/methodologies in numerous companies, projects, teams.**
- ***Disclaimer:* I'm in no way affiliated with any agile development trainer, author, etc. Nor am I certified as a Scrum Master.**
- **My goal is to relate my direct experience as an example of how an agile development process (Scrum) can help a team deliver software releases with higher reliability and quality.**

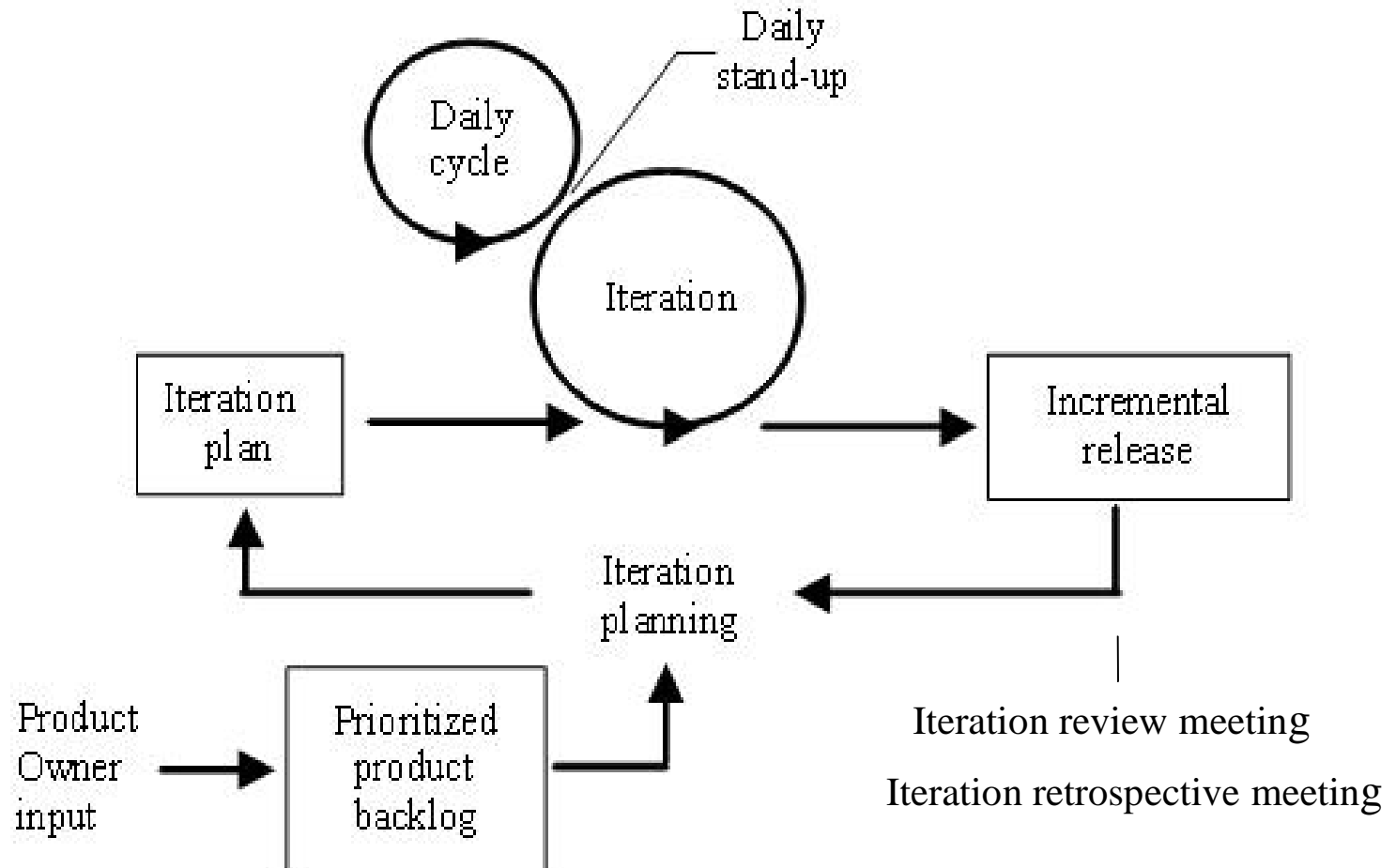
Overview

- **Overview of Scrum process**
- **Product/Team Overview**
- **Key Findings across Releases**
 - **Alpha1**
 - **Alpha2**
 - **Beta / FCS 1.0**
 - **1.1**
 - **1.2**
 - **1.3**
- **Summary/Conclusions**

Scrum Overview

- Why consider an agile process?
 - Scrum recognizes that project requirements often change, and allows for such changes to be more easily accommodated.
 - This flexibility was attractive for a startup company developing a new software product of significant size and complexity.

Scrum Overview



Scrum Overview

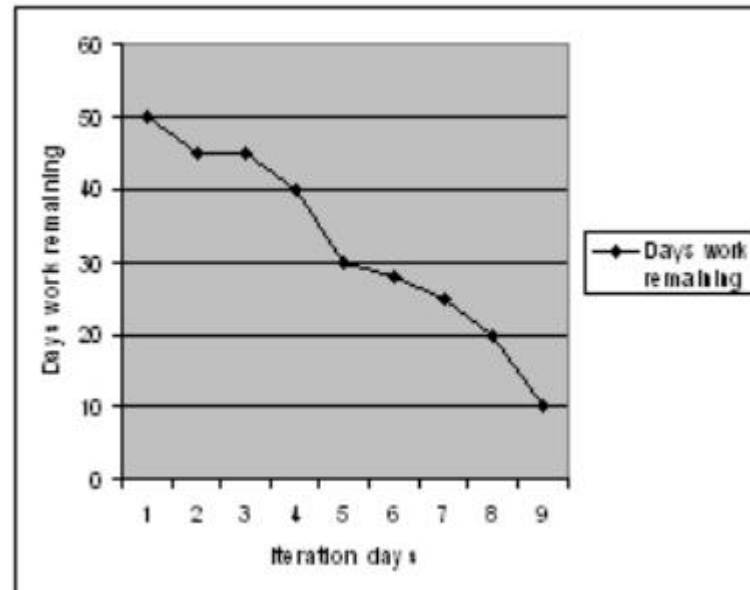
Term definitions: [Schwaber]

- **Backlog (iteration and release)**
 - **Prioritized list of requirements with estimated times to turn them into completed product functionality**
- **Product Owner**
 - **The person responsible for managing the Product Backlog in order to maximize the value of the project**
- **Scrum Master**
 - **The person responsible for the Scrum process, its correct implementation and maximization of its benefits**

Scrum Overview

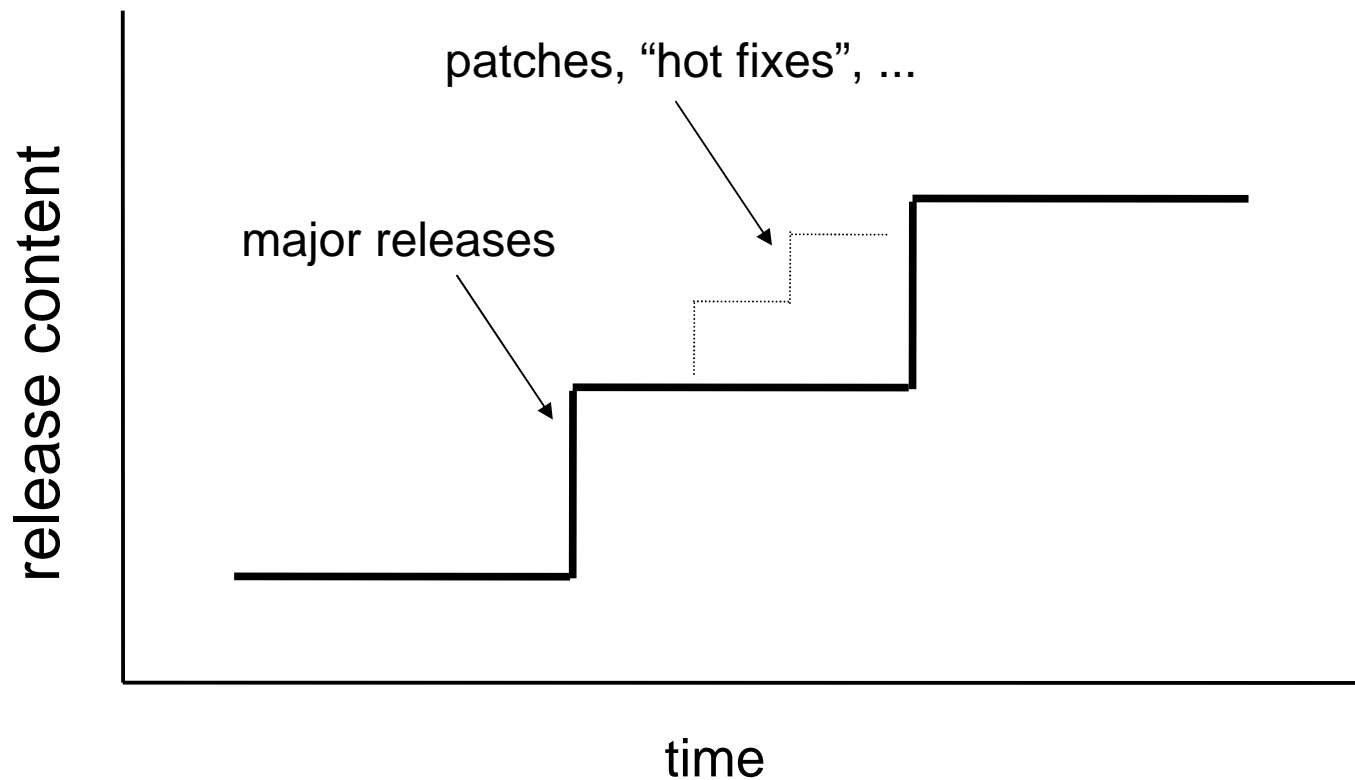
Term definitions: [Schwaber]

- **Burndown Chart**
 - Diagram showing the amount of work vs time remaining in an iteration, release or product



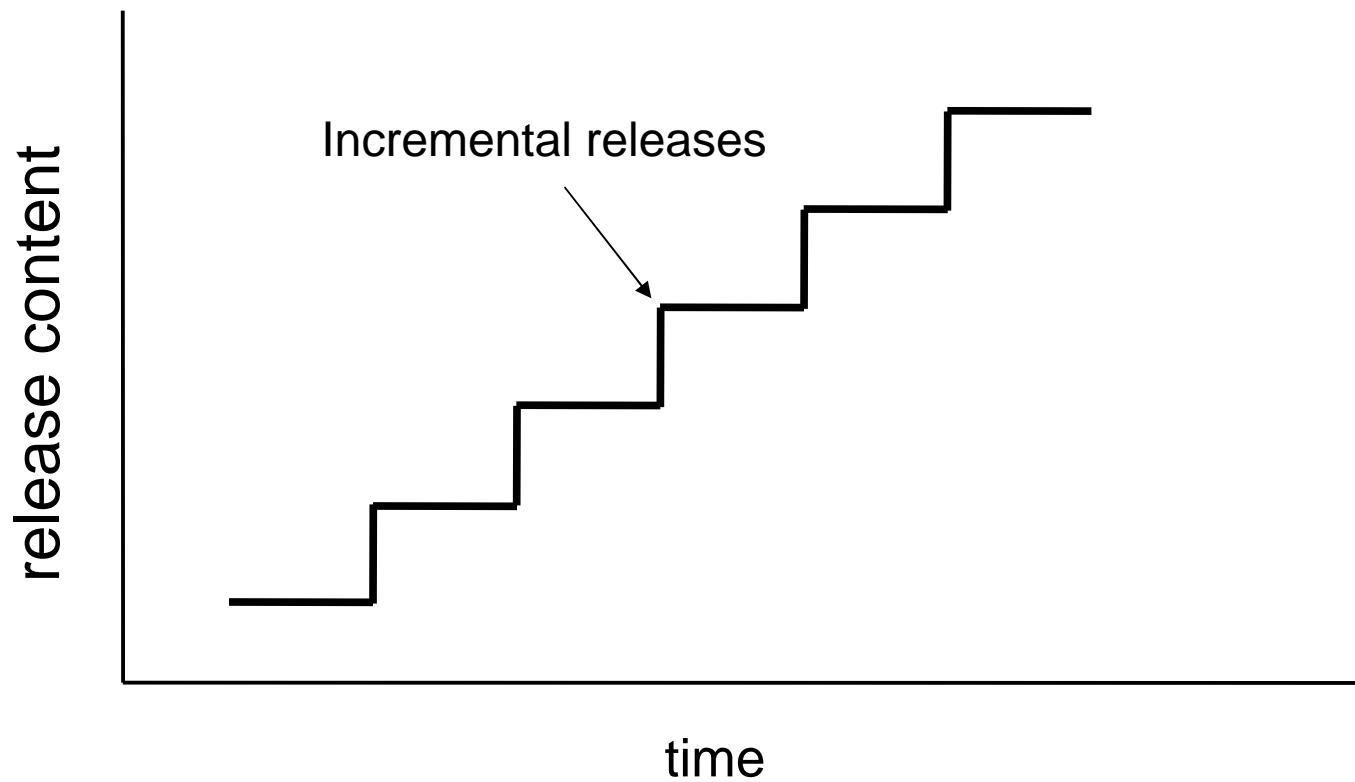
Scrum Overview

Standard release profile



Scrum Overview

Agile release profile



Scrum Overview

Our additions to/variations from the Scrum process:

- Automated nightly regression test pass
 - All tests, every night
 - Automated email and report generation
- Every development team (3-4 engineers) had a dedicated QA engineer
- Release cycle consisted of several iterations
- Several “bug hunts” performed during an overall product release cycle
- Product Owner and Scrum Master were the same person for the first few releases
- Daily stand-up meetings done via sub-teams



Product/Team Overview

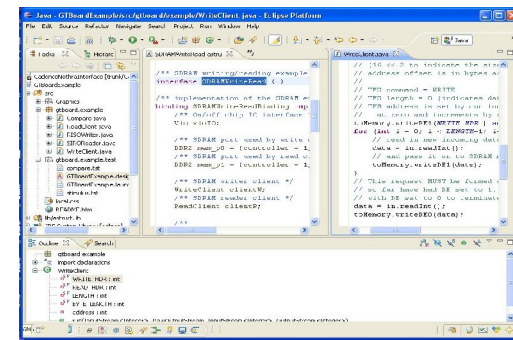
Ambric Technology and Team

- **Ambric MPPA (Massively Parallel Processor Array) chip – 344 processors, 6 Mb RAM, 2 DDR ports, various IO channels**



- **Delivered in several form factors including single and dual chip PCIe cards**

- **Software includes compiler, assembler, debugger, configuration and performance analysis components**



Ambric Technology and Team

- **Ambric Inc. founded in 2003, funded in 2004.**
- **Grew 2005-2008, reaching 70 employees.**
- **Company closed in Nov 2008 when it failed to receive third round of funding in the financial abyss of Q4 2008.**
- **IP purchased by Nethra Imaging of Santa Clara CA, a supplier of video imaging hw/sw solutions in April 2009.**

- **Ambric chip and toolset are well-suited for applications like video encode/decode, medical imaging, encryption/decryption, and image and digital signal processing.**

Ambric Technology and Team

- **Software team : 12 engineers, 4 QA engineers**
 - The focus of this presentation
 - One product under development
- **Applications team: 6 engineers, 4 QA/integration engineers**
 - Also adopted Scrum process for their development
- **Hardware team(12), marketing/sales, field engineering, executive/admin staff.**

Ambric Technology and Team

- **Unique business model:**
 - **SW provided early tools to HW team for more thorough chip design testing**
 - **Apps team developed first large designs, pushing bounds of early tools releases**
- **This supported our agile development process by strongly driving the product backlog (i.e., requirements) very early on, even in the absence of external customers. ***



Releases

Releases

Release Alpha1: late 2006-early 2007

Release Alpha2

Release Beta/FCS 1.0

Release 1.1

Release 1.2

Release 1.3

Release: Alpha 1

- Main theme: How to be Not Very Agile At All...
- Classic scramble to get a long list of “critical functionality” into a release, to demonstrate to prospective customers.
- Result: moving release deadline + feature creep
- Quality was not well understood. Final release testing turned up many problems that then took additional time to fix.
- VP of SW Engineering had brought in a Scrum tool to help us with planning and tracking, but it was lacking in depth, ease of use, etc...
- Team morale was down by the end of this death march

Release: Alpha 1

Result:

- Release was completed, but it was significantly delayed and team was tired.
- We needed a better process – our “Scrum-lite” attempt didn't work.

Actions:

- Brought in a full Scrum methodology and better toolset to support it for the next release cycle. *
- Observation: this was a new product going through major initial development, supporting new technology using a new programming paradigm.
- Also: no installed customer base yet, which gave us some release delivery freedom...?

Releases

Release Alpha1: late 2006-mid 2007

Release Alpha2: Nov 2007

Release Beta/FCS 1.0

Release 1.1

Release 1.2

Release 1.3

Release: Alpha 2

- **Main theme: Learning the New Process**

- **Defined our sprint structure (“iteration”)**

	Monday	Tuesday	Wednesday	Thursday	Friday
Week One	Feature design; coordinate team interdependencies	Develop/test	Develop/test	Develop/test	Develop/test
Week Two	Develop/test	Develop/test	Develop/test	Qualified release build/test	Fix outstanding issues, if any

- **Chose two week sprints**
- **Point of ongoing discussion – maybe should have chosen 3-4 weeks instead to provide higher development to overhead ratio ***

Release: Alpha 2

- **Release structure**

Iteration 1	Feature planning, design, initial documentation; development start
Iteration 2	Development/test
Iteration 3	Development /test
Iteration 4	Development /test
Iteration 5	Development /test
Iteration 6 [three weeks]	Release testing, final bug fixing (no new feature code), release packaging

- **Duration chosen to allow 4 releases per year**
- **Final iteration longer to allow for more thorough integration testing**

Release: Alpha 2

- **At the close of each iteration, a Qualified Build was built and tested.**
 - **Snapshot release, used by Application Team as latest stable build.**
 - **Could be used as a field demo, or customer patch release.**
 - **Good team confidence builder : focused on stable good build, not constantly chasing the next feature/bug. ***
- **Quality:**
 - **More automated nightly build/regression test process put in place.**
 - **Defined weekly open bug review with tech leads to prioritize new incoming issues and review status of critical must-fix ones**

Release: Alpha 2

- **“Bug hunts” introduced**
- **As a team, we clearly defined our development work flow: repository branching/merging/tagging**
 - **Side effect of agile framework: we'd organized our efforts at the macro-level, so we spent time cleaning up our act at the subteam/developer level too.**
 - **Benefits of Scrum's lighter-weight framework/practices than XP encouraged team members to rise to the challenge and suggest process improvements rather than being bogged down in controversial process details.**

Release: Alpha 2

- **Results:**
 - **Clearly-defined process, agreed upon by all developers and QA engineers.**
 - **Example: all significant feature development performed on a branch, shared between team members as needed. QA testing used this branch prior to code merge to repository main line.**
 - **Better planning and communication when focusing on smaller iterations. ***
 - **Release delivered a few weeks later than promised, but far better than our prior release!**

Release: Alpha 2

- **BUT: Team really struggled to get our plan done by the end of the release time frame!**
 - **We overestimated our ability to deliver new product features within this new process ***
 - **Task estimation no easier in Scrum – especially backlog item estimation**
 - **Large bug-fixing effort near the end of the release**
 - **We largely ignored a key aspect of Scrum: prioritize new features, and move lower priority ones out of release as deadline approaches.**

Releases

Release Alpha1: late 2006-mid 2007

Release Alpha2: Nov 2007

Release Beta/FCS 1.0: Feb 2008

Release 1.1

Release 1.2

Release 1.3

Release: Beta / FCS

- **Main theme: Grappling with quality**
- **We carried a large open bug count into this release**
 - **To address this, we allocated tasks for unspecified bug fix time tasks in second half of release**
- **For business reasons, this release was changed from a Beta release to our first official customer shipment (“1.0”)**

Release: Beta / FCS

Results

- **Completed release within 2 weeks of planned date**
- **BUT: still struggled with quality issues:**
 - **Closed lots of bugs in this release, but still had 250 open issues at release completion.**
 - **Being “our own worst critics” on quality**
 - **Make sure fixed bugs really are! * [1 in 15 weren't]**
- **Actions:**
 - **Increased effort on adding new regression tests**
 - **Increased effort at more thoroughly verifying bug fixes and new features earlier in the release cycle to avoid surprises at the end**

Releases

Release Alpha1: late 2006-mid 2007

Release Alpha2: Nov 2007

Release Beta/FCS 1.0: Feb 2008

Release 1.1: May 2008

Release 1.2

Release 1.3

Release: 1.1

- **Main theme: Priority One: Prioritization**
- **Began to include specific larger known bug fix tasks as scheduled development tasks.**
- **Scheduled the release “critical features” list for the first 3 iterations (of 5) only. ***
 - **Gave development and QA a needed buffer**
 - **Any other new features were deemed “on the fence” and not guaranteed to make the release build. Agreed to with the Product Owner.**
 - **Continued to raise release reliability**

Release: 1.1

- **Results:**
 - **Delivered release within one week of planned date over a 3 month timeframe**
 - **Quality: open bug count reduced from 250 to 160**
- **Team morale vastly improved from Alpha2 release**
- **Sales/field force confident in software team's ability to deliver new features on time with good quality**
- **Summary: Scrum process is proving itself!**

Releases

Release Alpha1: late 2006-mid 2007

Release Alpha2: Nov 2007

Release Beta/FCS 1.0: Feb 2008

Release 1.1: May 2008

Release 1.2: August 2008

Release 1.3

Release: 1.2

- Main theme: Are We There Yet...?
- **At half way through release (iteration 3), most critical features had been completed**
 - **Possible candidates for “on the fence” features to be pushed out of the release had been identified.**
- **Bug hunts changed to include several smaller ones throughout release cycle to target new features and big bug fixes, plus one final full release one**

Release: 1.2

- Results:
 - Delivered on time (OK, one weekend late...) over a three month time frame
 - Quality: open bug count down from 160 to 115
- The fact that I don't have much to say about this release is a very good thing...! Our team was maturing in its use of Scrum.

Releases

Release Alpha1: late 2006-mid 2007

Release Alpha2: Nov 2007

Release Beta/FCS 1.0: Feb 2008

Release 1.1: May 2008

Release 1.2: August 2008

Release 1.3: December 2008

Release: 1.3

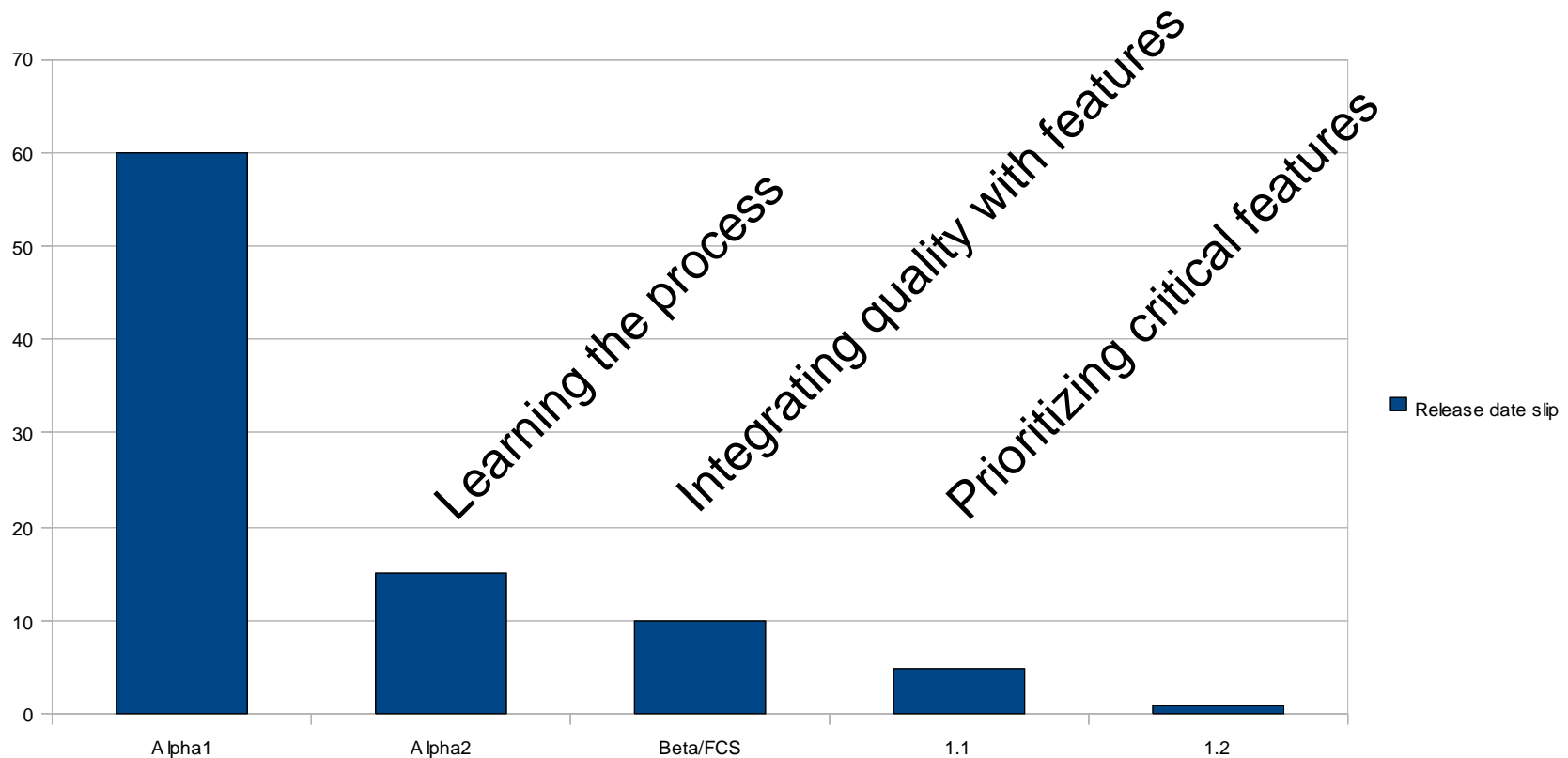
- Main theme: Disaster strikes...
- Release was entering fourth iteration in mid November when company closed due to lack of third round of investment funding.
- However, we were on track to deliver release in December.



Conclusions

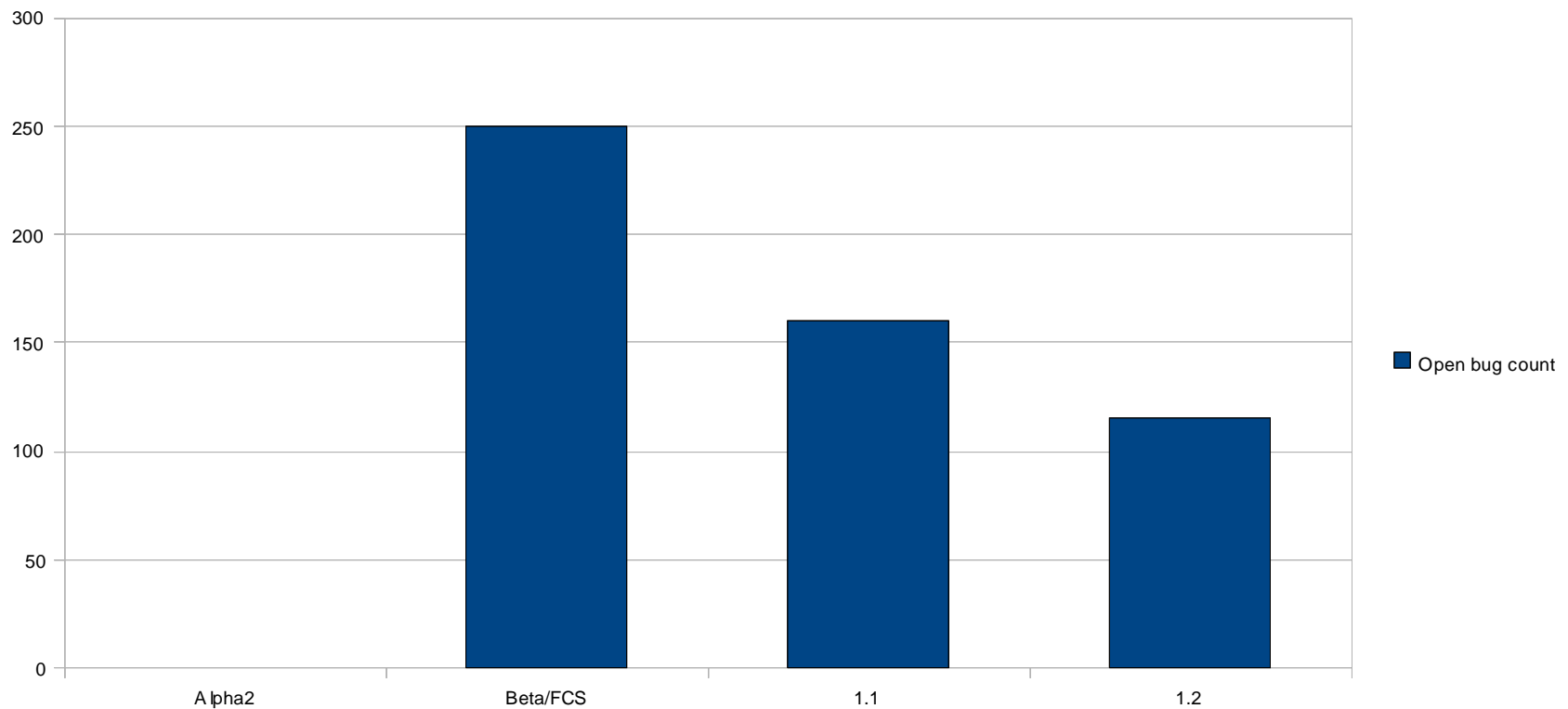
Key Findings

- Refine, refine, refine - **it takes time to instill good agile practices**



Key Findings

- Managing release quality takes time and you must budget for it.



Key Findings

- **Stack the feature deck for success** – prioritize critical feature addition across the first half of the release cycle
- **Verify quality early and often** – incremental feature development demands incremental testing! (bug hunts)
- **Agile process requires good feature backlog to produce a worthwhile product** – do everything you can to derive quality customer requirements to build this list
- **The team benefits as well as the product** – team morale is raised when you focus on what you've completed rather than what's still missing (burndown chart, iteration review)

Conclusions

- Many smaller iterations helped with product planning and delivery compared to longer, ill-defined release
 - Qualified release build per iteration provided structure
- Release date delivery became far more predictable
 - Sales and execs trusted engineering promises
- Release quality improved significantly over time
 - Very, very few customer issues ever reported
 - Customer quote: tool doesn't crash; for a first release, this is better than many mature FPGA/DSP tool sets

Summary

- An agile development process (Scrum) helped our team achieve better quality and far more reliable delivery dates
 - BUT it takes time to learn to use the process well
 - AND requires constant attention to both quality goals and critical feature status



Thanks!