Using Great Product Development Process to Achieve Great Results

Eric Krock
Director of Product Management for Broadband Content Services,
VeriSign Inc. (formerly of Kontiki Inc.)

February 7, 2007

Where it all comes together.™

My Background

- 15 years experience in software product management, sales engineering, technology evangelism, training, support
- Director of Product Management, Broadband Content Services at VeriSign, Inc.
- Previously:
  - Director of Product Management (Kontiki) for 4.0x, 4.1, 4.2, 5.0 ...
  - Senior Sales Engineer (Kontiki) for E&Y, Verizon
  - Group Product Manager (Kontiki) for 1.0, 1.1, 1.2, 2.0
  - Group Product Manager (Netscape) for Gecko Browser Engine
  - Senior Technology Evangelist (Netscape) for JavaScript
  - Technology Evangelist (Interleaf Japan)
  - B.S. in Computer Science & B.A. in Japanese from Stanford

About VeriSign and Kontiki

+ Kontiki Inc.
  - Founded November 2000
  - Developed secure commercial peer-to-peer (P2P) content delivery solution used by both enterprises for internal communication and media companies for B2C content delivery
  - Acquired March 2006 by VeriSign for $63M
+ VeriSign Inc.
  - Mission: to enable and protect all the world's network interactions
  - Supports secure content distribution to both mobile and broadband consumers
  - Launched VeriSign Intelligent CDN December 2006 based on Kontiki technology & VeriSign Constellation infrastructure
    - Next-generation content delivery network (CDN) that enables delivery of higher quality rich media at lower cost by use of VeriSign Kontiki P2P technology

What We're Going to Cover

- Common problems in software development
- How process can improve product definition and development and company performance
- A product definition/development, feature drop, and change control process that worked for Kontiki Inc. for two years and 13 releases
- Results of following those processes

You Know Your Process Has Problems When … (all real problems seen various places)

+ No one in eng believes schedule can be met.
  - Company commits to schedule anyway.
+ "Wait, shouldn't the schedule have accounted for QAing the product on Windows?"
+ Major destabilizing "skunk works" architectural changes introduced well into development without adjusting schedule.
  - ANY new features are introduced after PRD (Product Requirements Document) complete, design complete, and/or schedule commit without adjusting the schedule.
+ Company has no definition of "PRD complete," "design complete," "feature complete," and "code complete."

You Know Your Process Has Problems When … (all real problems seen various places)

- 6 weeks before scheduled Release To Manufacturing (RTM), 250 bugs & enhancements are untriaged
- Company has committed to ship a product with a fatal flaw that will cause it to fail customer security reviews
  - "It would have taken too long to fix it"
- Call from legal counsel: "Hey, you'd better file for export approval … the last PM never did …"
- Customer beta feedback: "Hey, why'd you take that feature out … we depend on it!" (Sales Engineer was sure no one used it … no one checked with the customer …)
- No one can figure out what certain lines in the PRD were supposed to mean

You Know Your Process Has Problems When … (all real problems seen various places)

- There's no specification for the product user interface
- "Can anyone tell me how this feature is supposed to work?"
- Company has deemed unimportant known issues that key customers consider unacceptable.
- "We've never seen an email like that from product management before." (sales rep, upon receiving a request for input)
- "Wait, customer X HAS to have feature Y … why isn't that in the plan?"

You Know Your Process Has Problems When … (all real problems seen various places)

- Customer: “We feel like you stopped talking to us.”
- Sales: “We decided we weren’t even going to *mention* that to [customer name] for now …”
- You have impressively detailed 24 month product roadmaps that have no relation to reality.
- Numerous stop-ship bugs are found just before RTM
- One day before the beta date, an executive review determines the beta will actually have to slip by six months
- You slip 6 month server/client dev plans by 4 and 6 months, 2 months before planned RTM
- You never hit your release dates

Myth

- "Engineers will take as much time to ship products as you give them."
  - Consequences:
    1. Company imposes unrealistic/unbuffered schedules thinking this will speed delivery.
    2. Eng/QA becomes frustrated and demoralized.
    3. Eng won't make maximum efforts to hit a schedule they never believed in.
    4. Frustrated engineers leave company.
    5. Releases miss announced dates.
    6. Return to step 1 and repeat until fired or bankrupt …
- Antidote:
  - Make detailed, rigorous, complete PRDs reviewed by sales, executives, and engineering.
  - Have engineering do thorough bottoms-up level of effort estimates on all features and "major bugs."
  - Iteratively negotiate work/schedule tradeoff until optimal tradeoff is identified.

“Roadmap” -- the root of all confusion?

+ Executives, Customers: “I want to see the roadmap.”
+ Sales: “But I saw it on the roadmap … it was committed!”
+ PM: “That was only a roadmap.”

+ The problem:
  - Some consider “roadmap” an illustration of possibilities, subject to change (e.g. marketing & PM, selling the dream)
  - Others consider “roadmap” hard and fast commitments that can be promised to customers (e.g. sales)

+ A solution:
  - Have a clear, written Plan of Record that shows only firm commitments
  - Rather than a “roadmap,” show “what is committed” and “what is under consideration”

The Iron Triangle of Sales Needs

+ Sales primarily wants three things from product management:
  - Deliver the features and benefits customers/prospects are asking for within the timeframes they want it.
  - Never miss a commitment to anyone.
  - Flexibility to add new deliverables and/or change the plan at any time.
+ These goals are each good, legitimate, and understandable, but are fundamentally in conflict and irreconcilable.
+ Each account team sells to different customers, leading to additional conflict among requests.

The Iron Triangle of Sales Needs: The Solution

+ Define a solid product definition and delivery process.
+ Educate company about process and why it's important.
+ Get process approved and made official.
+ Follow the process.
+ Use the process to develop the best possible plan with the time, resources, and information available.
  - Spend more time on due diligence. If you don't get every decision right (enough), you can't defend plan later.
  - Incorporate adequate buffer to cover the unexpected & unknown.
+ Get everyone's buy-in to the plan before company commits.
+ Communicate the outcome and commitment to all concerned.
+ Fight like hell to stick to plan once committed.
+ Insist on schedule adjustments where needed if plan must be changed.

Problems That Good Process Can Help Prevent

+ Committing way too much work in a given time period and planning to cut features later “if we run out of time.”
+ Opportunity of the Day Syndrome
+ Shiny Object Syndrome
+ Pursuing off-strategy opportunities
+ Sales or executives going directly to engineering
  - Process gives eng a way to say no, force an exec meeting
+ Engineering building what it wants instead of what the customers want
+ Product management guessing what market needs
+ Company building what it THINKS customers want instead of what they DO want

Commitment Techniques in Product Management

+ Lessons from social psychology research studies:
  - People are happier and more likely to comply if they feel VALIDATED.
    - You listened.
    - You understood.
    - You repeated what you heard with empathy to prove it.
  - People are more likely to comply if they say they will.
    - … especially if they say so in front of other people.

Key Processes Used by Kontiki Inc.

+ Feature Drop Process
+ Product Definition & Development Process
+ Change Control Process

A Feature Drop Process that Worked for Kontiki

- Confirm with core product team (PM, eng, QA) you really want to drop feature
- Document in writing everything you are proposing to drop, including:
  - Product UI (fields, controls)
  - API calls
  - Preferences or configuration settings
  - Data (schema fields, etc.)
  - Capabilities & benefits (e.g. ability to centrally manage X, support for platform Y, etc.)
- Update PRD and (if needed) road map to note that the proposed change is pending
- Email proposal to sales, support, PM, eng
  - Specify deadline for objections
- Restate proposal at next sales meeting
- Identify any customers currently using feature

A Feature Drop Process that Worked for Kontiki (cont'd)

+ Work with account teams to communicate change to relevant customers
+ At some point, email proposal to customers
+ Include proposal in overall "road map update" presentation emailed and/or presented to each customer
  - Archive presentations you give/send to each customer!
+ Get confirmation from responsible party at each affected customer that they accept feature drop proposal
  - Permanently document confirmer name & date received
+ After verifying change won't hurt existing customers or block sales at prospects, do change control process to ratify decision at executive staff

A Feature Drop Process that Worked for Kontiki (cont'd)

+ Notify sales, support, PM, eng of ratification; list customer approvals
+ ONLY THEN: file bug reports to start removing the feature!

Results of Following Feature Drop Process

+ 63 features (including support for obsolete browser, mail client, OS, media player versions) successfully permanently dropped over two years
+ 10 more features removed at client architectural transition with possibility of future restoration
+ Zero customer escalations
+ Zero emergency restorations or respins
+ Major savings of engineering & QA effort
+ Only two customer representatives later expressed surprise
  - One had not been informed by co-worker who approved removal
  - One had forgotten they approved feature drop
  - Both accepted change when shown presentation we'd given and that their company had accepted

A Product Definition and Development Process that Worked for Kontiki for Two Years & 13 Releases

+ Release Definition / Requirements Gathering
  - Define high-level objectives, target audience/customer(s), and scope of release (major, minor, or maintenance) [PM]
  - Define codename [PM]
  - Create bug database keywords for nomination, approval, and rejection of changes for the release [QA]
  - Solicit input from sales, eng, QA, support, operations of proposed work [PM]
  - Inform customers "release planning window" has opened [PM]
  - Meet with each key customer and solicit input [PM]

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Product Work Definition:
  - Create first draft PRD(s) for release. [PM] Cover:
    - Target market
    - Features & benefits
    - Quality target
    - Security
    - Performance targets including quantitative metrics
    - Language support
    - Supported OS & 3rd party component versions including browsers, players, helper apps, virus checkers, etc. New versions? Add/drop?

A Product Definition and Development Process that Worked for Kontiki (cont'd)

- Create first draft PRD(s) (cont'd)
  - Necessary and desirable timeframe for release
  - Quantifiable success criteria for release including per-customer revenue enablement goals (if any)
  - Assumptions
  - Open issues and any risk areas

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Product Work Definition:
  - Define bug triage criteria [PM]
  - Review bugs & enhancements: [PM]
    - Definite or possible sales blockers for prospects
    - Known enhancement requests from customers
    - All bugs doable for this kind of release (maintenance, minor, or major)
    - Any possible security issues
    - Bugs rejected for previous release & not already assigned to maintenance, minor, major, future buckets
    - Nominate/approve bugs/enhancements

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Product Work Definition
  - Kick off Feature Drop Process where necessary [PM]
  - Kick off eng/QA Level of Effort estimates: major features, then minor features, then "big bugs"
  - Write presentation describing proposed release, with tentative dates, for customer review meetings [PM]
  - Review presentation with each customer [PM]
    - Archive what you showed each customer!
  - Iteratively review PRDs with sales / eng / qa / ops [PM]
  - Hold "cross-PRD customer requirements review meeting" [PM]
  - Finalize PRDs = "PRD Complete" milestone
  - Official hand-off of PRD

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Product Work Definition
  - Kick off engineering design effort
  - Do pre-design sanity check security review discussion [eng, PM]
  - Ensure adequate spec of UI and functionality requirements exists [PM, eng]
  - As appropriate, review design with SEs, client services, customers [PM, eng]
  - As time/resources allow, do usability testing [PM]
  - Complete Level of Effort estimates for all items [eng, QA, PM]
  - Do bottoms-up schedule, timeline for all items [eng]
  - Determine amount of buffer for release date [eng, PM, QA]

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Product Work Definition
  - Review eng spec to confirm it satisfies PRD requirements [PM, eng]
  - Conduct security review [PM, eng, QA, SE]
  - Get executive staff approval of final schedule Communicate approved final release plan & external schedule to customers [PM]
  - Archive the presentation!
  - Document QA test plan [QA]
  - Review QA test plan with sales to ensure understand of what is/isn't tested and how deeply [PM, QA]
  - Gather documentation requirements [doc lead]
  - Define proposed doc plan for release [doc lead]
  - Iteratively review doc plan with customers, sales, eng, QA ops until finalized [doc lead]

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Legal Compliance
  - If creating new product with strong cryptography or increasing strength of existing cryptography, work with legal to determine if export certification is needed. If needed, start early: 30-45 day lead time!
  - If more open source software is to be used with product, update list of o.s. software and work with legal to verify license acceptability and compliance
    - Ditto for any o.s. software distribution
  - Evaluate any changes to legal privacy requirements
  - If new product/product area, do general review with legal (End User Licensing Agreement, etc.)

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Development Process Management
  - Schedule & attend team meetings as needed [PM]
  - Keep "for PM review" bugs list at zero [PM]
  - Triage new bugs and keep nominees list at zero [program manager, QA, with PM help as needed]
  - Flag bugs for release notes as needed [all]

+ Beta program
  - Define goals [PM]
  - Identify target customer(s) [PM]
  - Get agreement to participate from customer(s) [PM]
  - Ensue doc created to support beta [PM]
  - Provide customers beta software, doc [support, ops]
  - Gather feedback from customers & support [PM]

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Shipping Process
  - Identify issues that will affect deployment process (e.g. migration) & communicate to support, SEs, client services, as appropriate [PM]
  - Keep "for PM review" bug tally at zero! [PM]
  - Ensure all needed release notes get written [PM, support]
  - Copy draft release notes from bug reports into actual release notes document [support]
  - Review WONTFIX bugs and mark CLOSED to approve [PM]
+ Launch
  - Finalize name/number of release [PM]
  - Work with outbound marketing as needed to support product launch [PM]
+ Deployment
  - Support sales & support in customer communication [PM]

A Product Definition and Development Process that Worked for Kontiki (cont'd)

+ Post-Release
  - Schedule and hold post-mortem review
    - Identify action items for future process improvement.
  - Permanently archive all release-related documents

Change Control Process

- When used:
  - New feature request that isn't included in the current product plan as specified by the written Plan of Record (POR) and PRDs
  - Any feature or implementation change that impacts schedule on POR
  - When action will take more than an hour of time from a developer, QA tester, or UI designer
- Steps:
  - Define work
  - Estimate level of effort, get PM/Eng/QA approval
  - Review at executive staff meeting, get approval
- Benefits:
  - Gives eng, QA a way to say "no" to execs, sales, PMs, others
  - Provides a structured way to change plans when needed

Iterative Refinement: A Hybrid Between Waterfall and Extreme Programming

- Define the big “must do, will do, universally agreed” chunks first and get engineering to start work on them right away
- Continue evaluating “may do” features and iteratively refining the list based on customer/prospect/sales/support/executive input and engineering/QA level of effort estimates
- Postpone finalizing decisions where you need to and can afford to
- Iteratively develop and communicate tentative top-down, tentative bottoms-up, and final bottoms-up schedules
- Kontiki CEO on largest release company ever did: “I’ve never seen a release definition finalized as late in the process as we did …” yet the release shipped 6 weeks ahead of schedule after an 18 month development cycle

Results of Following This Process*

- 13 consecutive releases shipped on or ahead of schedule (on a resource-constant basis)
  - In every case that release was necessary to get customer business, release met deadline, passed customer acceptance, and deal closed
- Record number of new customers for company on both enterprise and media sides
- Profitability achieved for first time
- Company valuation greatly increased over 16 months after third round of financing
- $63 million acquisition by VeriSign 16 months after third round of financing
- Company products and technology used as a foundation for VeriSign Intelligent CDN, major new managed serving offering launched 9 months after acquisition closed

* and an improving market, new company leadership, etc. …

Results: Annual Employee Survey Results (Dec ’04 – Dec ’05)

Biggest deviations from last employee survey:

Teamwork

+ After the debate, we move forward and support decisions made. (95%)
+ Everyone has the opportunity to express their opinions (95%)
+ We challenge each other's thinking to get the best idea/solution (90%)
+ We are driven by facts, not opinions. (74%)
+ We foster open debate (84%)

Management

+ We focus on the critical few vs. the trivial many (69%)
+ We get results by focusing motivated capable people on a shared objective (95%)

Communication

+ We focus on listening and understanding. (90%)
+ We foster open and direct communication. (95%)
+ Overall, I have a good understanding of the company's focus and efforts (95%)
+ We communicate early and often. (90%)

Best Practice: Record Everything On the Intranet

- What if you or a co-worker leave or go on vacation?
- What if someone needs information you know and doesn't know you know it?
- What if a new employee needs to get up to speed and review an account's history?
- What if you forget the status of something?

Best Practice: Use a Wiki for Your Intranet

- "I can never find anything on our intranet and it's all out of date anyway." (Sound familiar?)
- If you want people to keep intranet pages up to date, how do you get them to do that? MAKE IT EASY!
- If you want people to collaborate, how do you get them to do that? MAKE IT EASY!
- If you want a culture in which everyone feels free to contribute, how do you achieve that? MAKE IT EASY!
- Wikis address all of these needs.
  - To update a page, you just click Edit, type, and Save. It's too easy NOT to correct something that's wrong or out of date!
  - To add a new page, you click Edit, type something like [Page Pretty Name|Internal Name], hit Save, click the new link, type, and Save
  - Don't need to know HTML. ANYONE CAN DO THIS!

Great Uses for Wiki Pages

- "To Do" lists
- "Open Issues" lists
- Individual status page for every open issue
  - Known problems
  - New features
  - Page for each release
- Maintaining FAQs
- Lists of links to other resources (documents, videos, FAQs, repositories)
- List of contacts by function for a project
- Training steps for new hires
- History of decisions on project
- Archiving all "FYI" emails sent out by product management, especially to sales force
- Archiving customer meeting notes
- Individual status page for every customer
  - Summarize deployment, use cases, known issues, platforms, application, requirements, account history, etc.

Disclaimers: Your Mileage May Vary

- For this to work:
  - Company culture must follow Jim Barksdale's rule: "The product manager is the CEO of the product."
  - Company executives must agree to follow the process.
  - As a product manager you must execute the process faithfully, thoroughly, and successfully or management will not let you continue to follow it.
- We were working with finite number of large accounts
- Much easier to do at a small company than at a large one.
- Employees I was working with during this better were way better than average quality.
  - Most team members had been working together for 2-3 years, knew each other's strengths and weaknesses

Summary

+ Good process:
  - Saves more effort than it consumes
  - Prevents more conflict than it creates
  - Surfaces issues sooner, enabling easier/better resolution
  - Enables increased sales
  - Improves customer satisfaction
  - Improves employee satisfaction & retention
  - Increases odds of company success (and survival!)
  - Is totally, totally worth it

Q & A

Deadly Sins of Product Management

- Sloth: Failing to do sufficient due diligence
- Vagueness: writing doc that could be misunderstood
- Undercommunication: could anyone not know?
- Pride:
  - thinking you know the answer instead of asking
  - ignoring eng, QA, sales, customer, exec input
- Wishful thinking
- “Inside the Beltway Thinking:” adhering to beliefs inside the building that don’t apply outside the building
  - “This issue isn’t a big deal because it doesn’t happen often”

Principles to Live By

+ Humility: “I am a fool” -- Socrates
+ Listen: to customers, sales, engineering
+ Write it down: take and archive notes
+ Tell the truth: to yourself, customers, executives, sales, engineering
  - If you don’t know, say so!
  - Bad news doesn’t improve with age
+ Get buy-in: from executives, eng, sales, customers
+ Communicate in writing: especially to sales and customers
+ Document customer agreement
+ Archive what you communicated
+ Expect the unexpected
  - Practice realistic scheduling
  - Include adequate schedule buffer: 30%