Experiences in Software Engineering: Product Incubation Lessons

Derick Campbell
Director of Engineering, Microsoft Research Connections

May 24, 2012

This presentation is available online: http://docs.com/@derickc
Agenda

Intro
Me, You, Our Business

Lessons
1. Express Capacity Meticulously
2. Throw Away Good Projects
3. Add Drama
4. Mandate Business Reviews
5. Open Source Prematurely
6. Partner Fanatically
7. Be the Edge
8. Capitalize on Risk

Demos
Who Is This Guy?

Places of Work
- Microsoft
- Lucent Technologies
- AT&T
- Toronto Police Services
- Crown Life Insurance
- AdScan
- The Unicorn Pub
- The Bombshelter
- Chuck E. Cheese

Variety of Roles
- Software Engineering Management
- Program Management
- Product Marketing
- Consulting Management, Consulting
- IT Management
- Software Development, IT Architecture
- Helpdesk Technician
- Data Entry Clerk
- Bouncer
- Pizza Chef, Pizza Delivery Guy
- Software Consulting
- Cartoon Rat

Father.
Canadian.
Geek.
Who This Talk Is For

Teams
Product Incubations
Software Startups
Software Labs
Prototyping Teams
Open Source Projects

Team Qualities
Usually small
Future investments uncertain
Big breakthroughs expected
Projects may run in parallel
Competition for resources

Software Engineers
Software Managers
Students and Academics
Startup Businesses
Our Engineering Business

Deep software experience

Empowered teams of peers

Agile process, some rules

Enable the pursuit of scientific breakthroughs

Accelerate scientific exploration with computing

Inspire emerging computer and research scientists

Community Releases

Forum-Based Community Support

Disclaimer: Microsoft Research is different...
Free Software Shipped: Highlights

- Millions of unique visitors
- Hundreds of thousands of downloads
- Dozens of schools and classes

http://research.microsoft.com/Accelerators
Layerscape demo
Incubation Lessons Learned

Express Capacity Meticulously
Throw Away Good Projects
Add Drama
Mandate Business Reviews

Open Source Prematurely
Partner Fanatically
Be the Edge
Capitalize on Risk
Express Capacity Meticulously
Be cautious about over-committing

How

Ideal scope.
Ideal duration.
Budget parameters.

Why

Expectations, risks.
Leave room to aspire and amaze.
Focus!

Keep in Mind

Pressure to do more.
“Nobody ships early”.

Forecast Schedule
(No. of days)

Actual Schedule
(No. of days)
Throw Away Good Projects
Keep the Great Ones

Selection
Ideas must be great!
(Of the best of available options)
A high bar sets the right tone.
Caution: high bars create anxiety and tension too!
Trust people to identify better projects over time.

Sacrifice
Be prepared to de-invest in legacy projects that are holding the team back.

Polish
Make something in each project shine.
Translator Hub

demo
Add Drama
Drive Action with a Compelling Story

“This software does 4,000 amazing things!”

That people will never remember, and distract from the core message.

The Story Drives Action
Try the software.
Approve the project.
Provide funding!

Simplify The Story
Focus on primary audiences.

Make It Compelling
Audience validation.

Engineers On The Project
Are NOT the audience.
Mandate Business Reviews
More Value Than You First Expect

Benefits

Team morale and alignment
Forward progress and evolution
Commitment, recommitment
Visibility and understanding
Better team preparation
Expectations management
Annual performance reviews
Getting help!

Caution

Remember who the decision makers are
Schedule early to avoid conflicts
Too many reviews can impede progress
Open Source Prematurely
Planning for an OSS Release is Critical

Why
Shifting to OSS late introduces risk and offers diminishing returns.
Shifting off OSS is far easier.

When
OSS is “Mission aligned”.

Consider
Governance and increased costs.

Licensing
Remember: Developers are not lawyers!

IP Management

Governance and increased costs.
F# is ideal for data-rich, concurrent and algorithmic development: “simple code to solve complex problems”. F# is a simple and pragmatic programming language combining functional, object-oriented and scripting programming, and supports cross-platform environments including PC, Mac, and Linux.

We'll provide the tutorials, resources and tools you'll need to begin working with F# right away.

Try F# demo
Our engineering team is eight “full-timers”. Actual project teams are larger – thanks to partnering. The potential benefits are greater than you expect.
Challenge Boundaries
Don’t just live on the edge: define part of it

Advance Your Art
Incubations are about learning and risk taking

(Poor) Reasons To Avoid New Technology:
It’s too hard!
Tools aren’t ready yet
Spec is changing
Missing skills

Never underestimate personal passion!
Program Management
Overall risk management.
Risk-driven scheduling.

Culture
Embrace risk awareness!
Pursue riskier opportunities over time.

Development
Classify team by competencies.
Be strategic about who does what.

Test
Business prioritized testing.
Platform multiplier strategies.
ChronoZoom demo
Throw Away Good Projects
Mandate Business Reviews
Open Source Prematurely
Partner Fanatically
Add Drama
Be the Edge
Express Capacity Meticulously
Capitalize on Risk