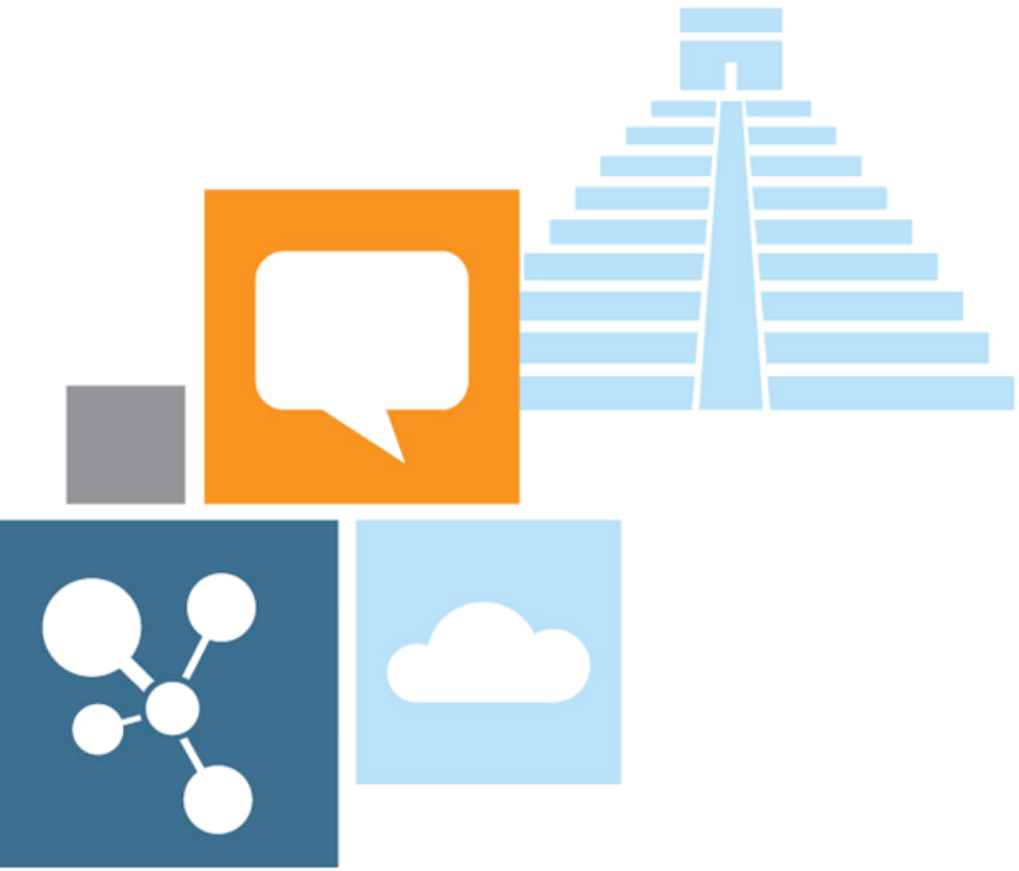
Microsoft



Microsoft® Research

Faculty Summit 2012

Riviera Maya, Mexico | May 23-25 | In partnership with CONACYT

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

Christophe

Danny

Dean

Derick

Kal

Lori

Mike

Roman

Agenda

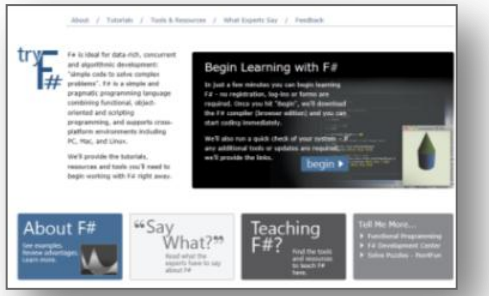
Intro

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

Software Engineers
Software Managers
Students and Academics
Startup Businesses

Teams

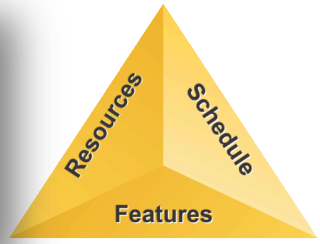
Product Incubations
Software Startups
Software Labs
Prototyping Teams
Open Source Projects

Team Qualities

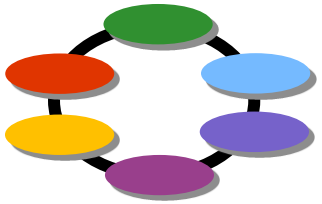
Usually small
Future investments uncertain
Big breakthroughs expected
Projects may run in parallel
Competition for resources

Our Engineering Business

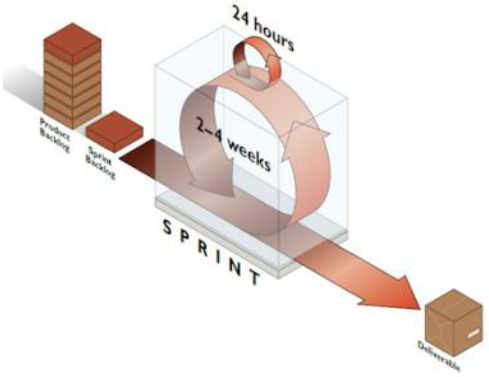
Deep software experience

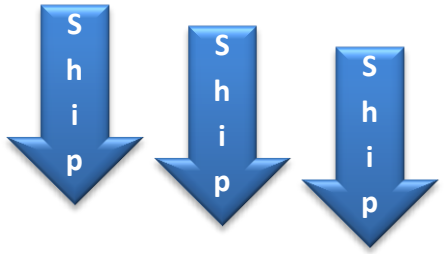


Empowered teams of peers

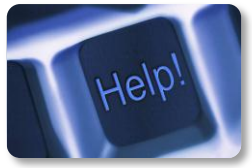


Agile process, some rules

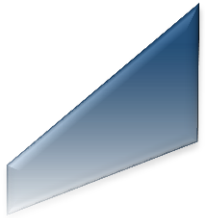




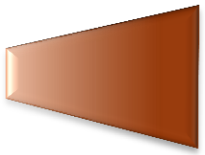
Community Releases



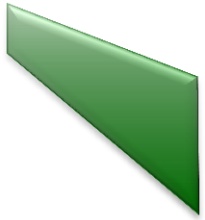
Forum-Based Community Support



Enable the pursuit of scientific breakthroughs



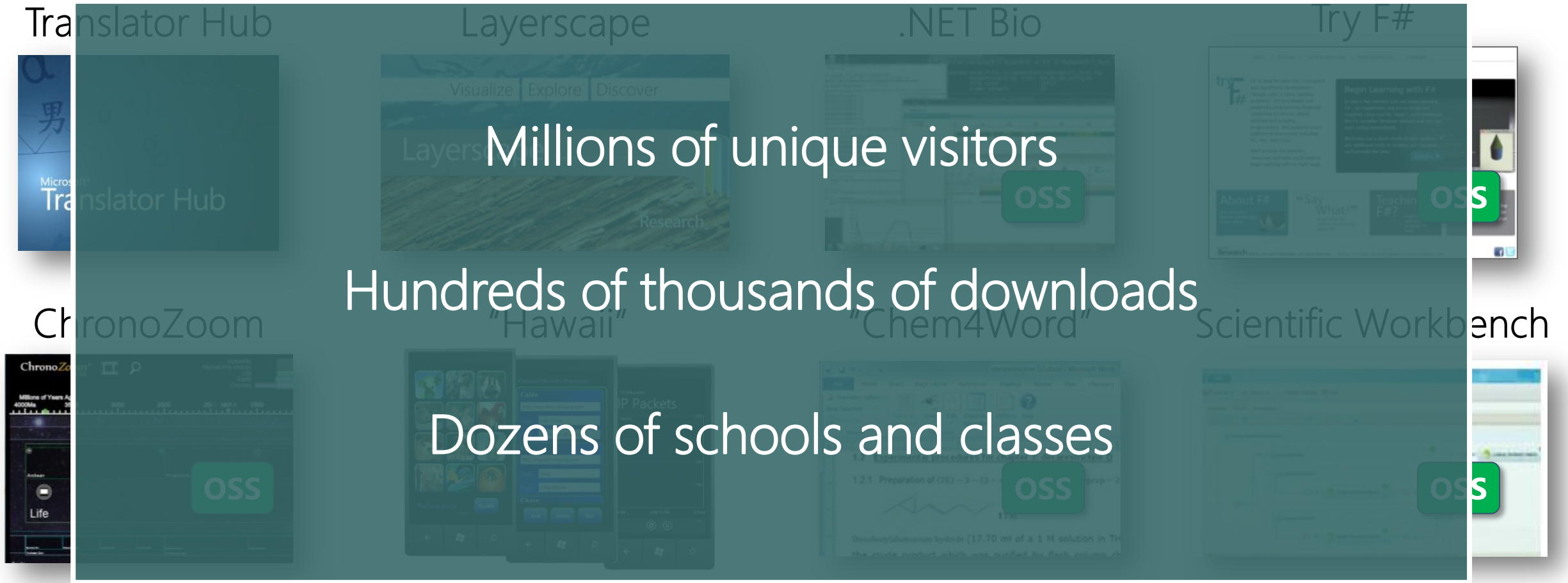
Accelerate scientific exploration with computing



Inspire emerging computer and research scientists

***Disclaimer**: Microsoft Research is different...*

Free Software Shipped: Highlights



Layerscape demo

Incubation Lessons Learned

Express Capacity Meticulously

Open Source Prematurely

Throw Away Good Projects

Partner Fanatically

Add Drama

Be the Edge

Mandate Business Reviews

Capitalize on Risk

Express Capacity Meticulously

Be cautious about over-committing

How

Ideal scope.

Ideal duration.

Budget parameters.

Why

Expectations, risks.

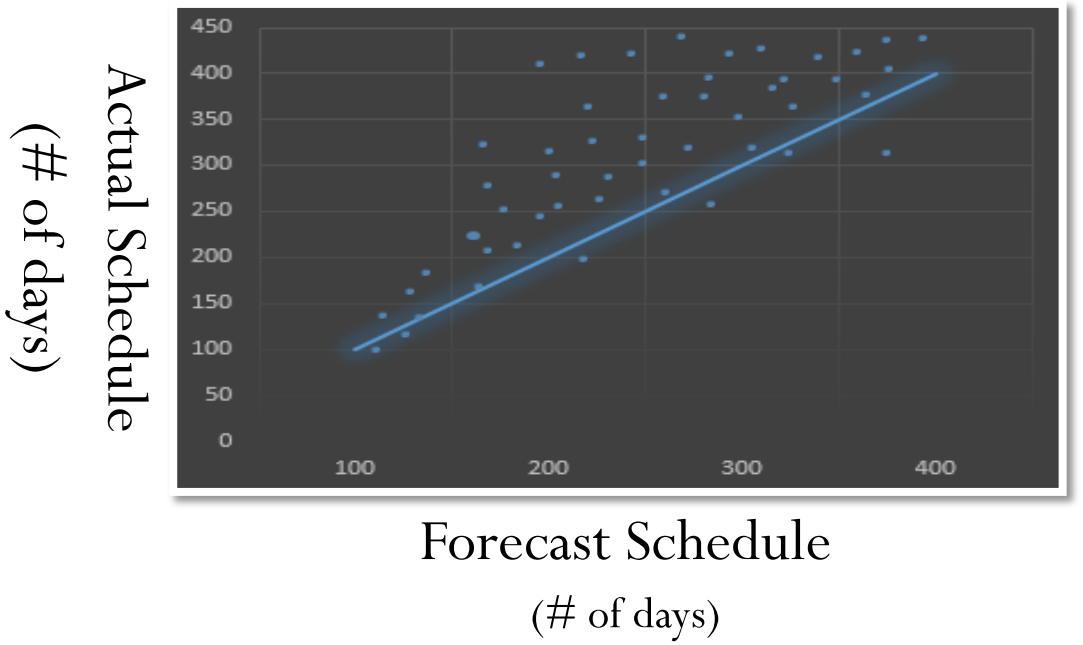
Leave room to aspire and amaze.

Focus!

Keep in Mind

Pressure to do more.

"Nobody ships early".



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

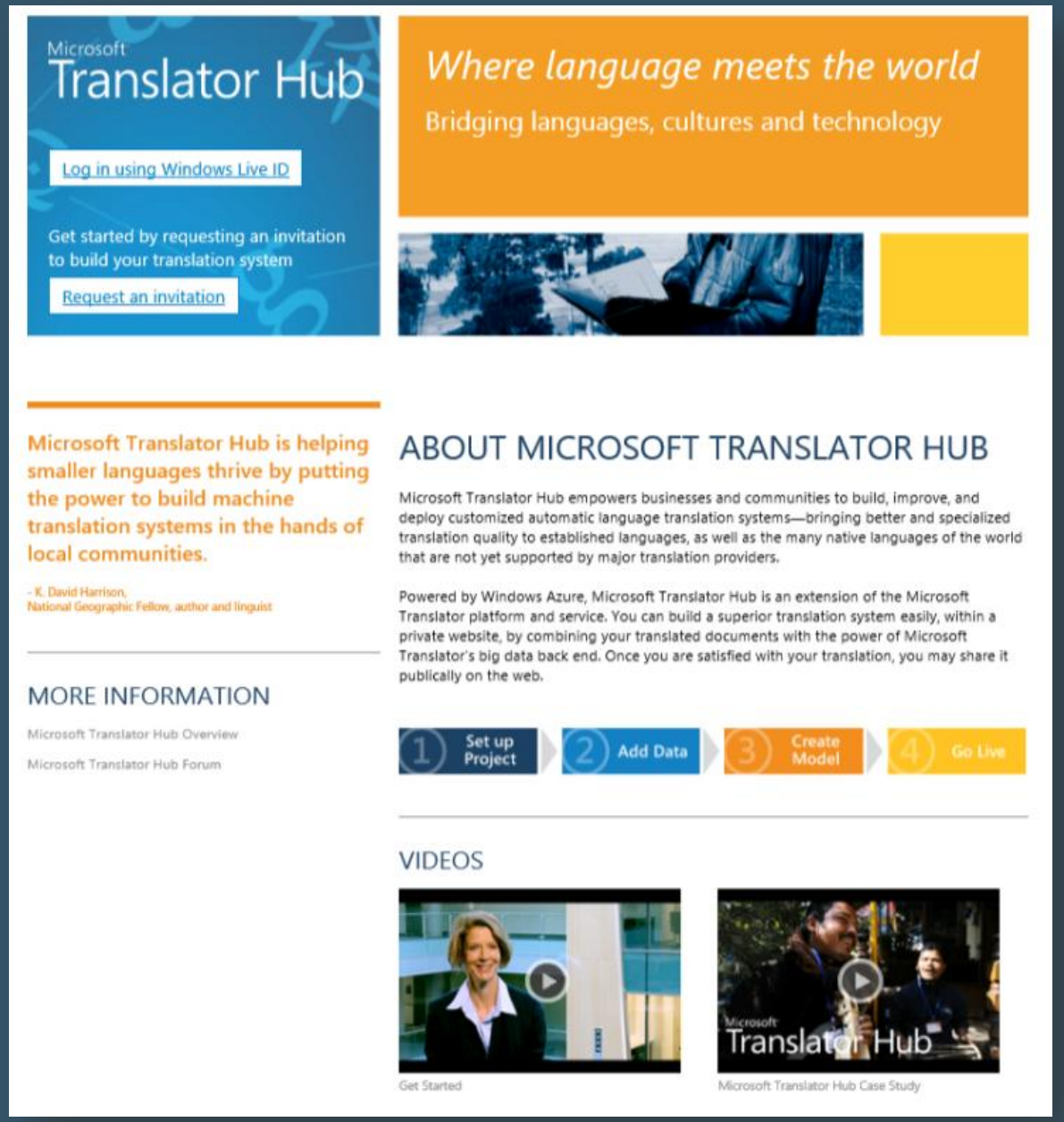
Be prepared to de-invest in legacy projects that are holding the team back.

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

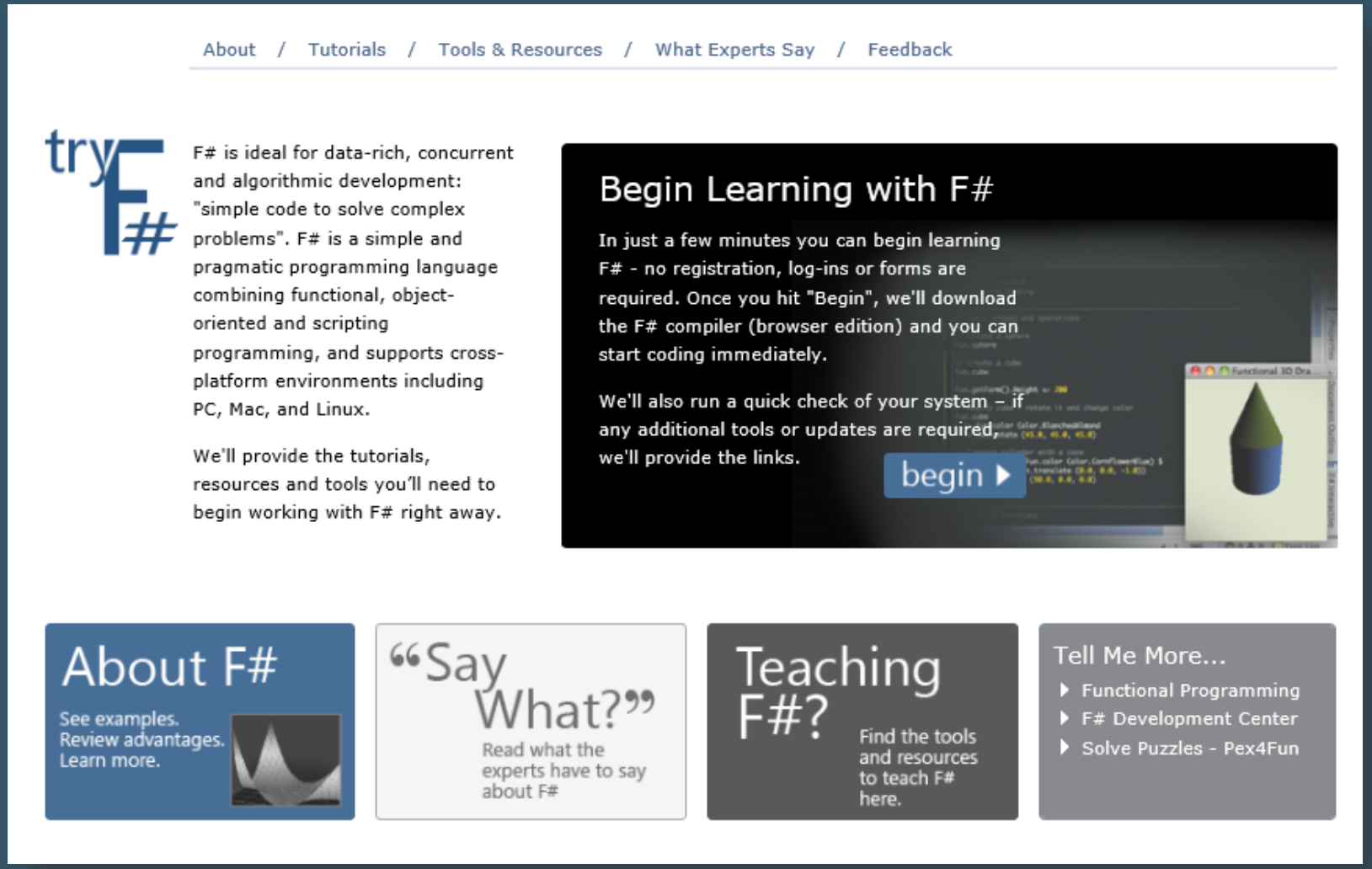
Remember: Developers are not lawyers!

IP Management

OUTERCURVE FOUNDATION

Try F#

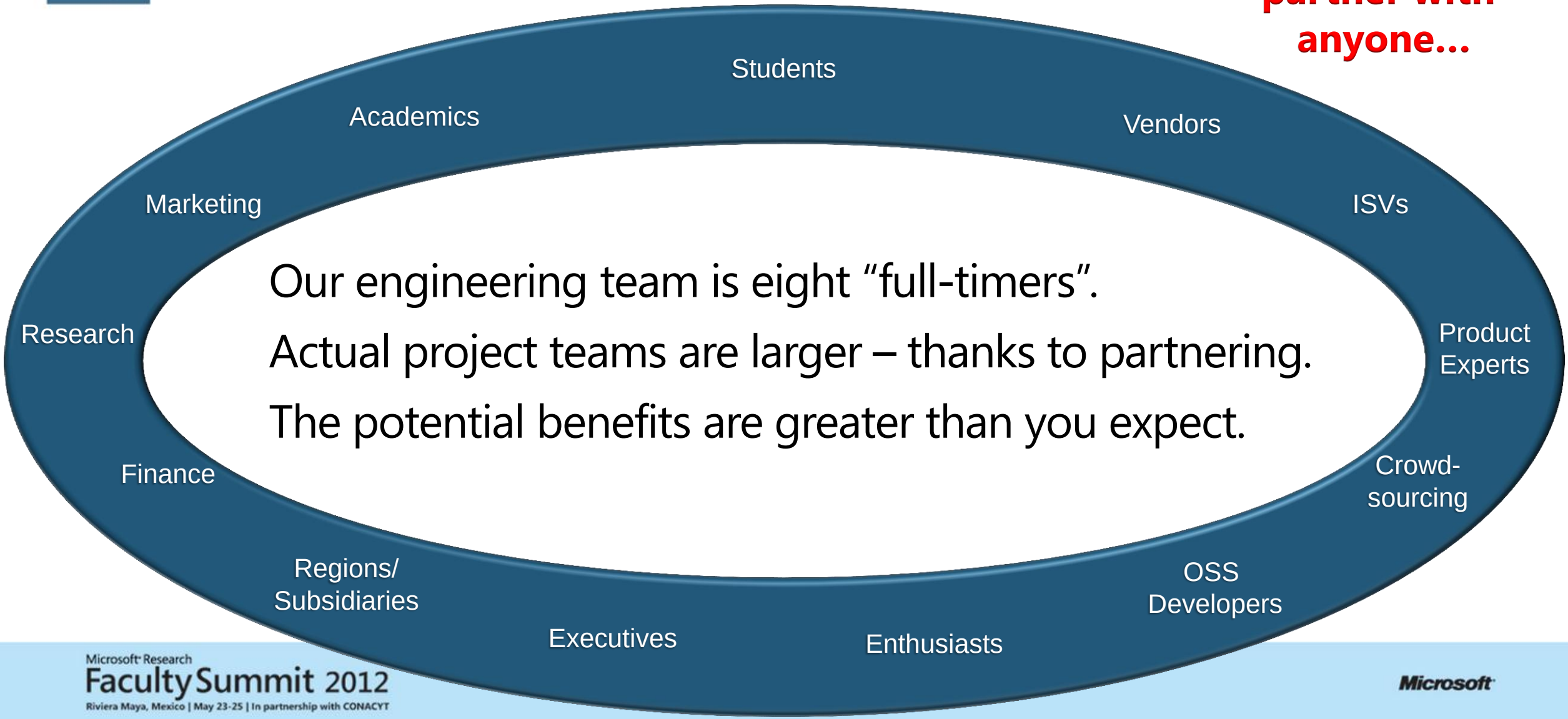
demo


About / Tutorials / Tools & Resources / What Experts Say / Feedback
try F#
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.
Begin Learning with F#
In just a few minutes you can begin learning F# - no registration, log-ins or forms are required. Once you hit "Begin", we'll download the F# compiler (browser edition) and you can start coding immediately.
We'll also run a quick check of your system – if any additional tools or updates are required, we'll provide the links.
begin
About F#
See examples. Review advantages. Learn more.
"Say What?"
Read what the experts have to say about F#
Teaching F#?
Find the tools and resources to teach F# here.
Tell Me More...
Functional Programming
F# Development Center
Solve Puzzles - Pex4Fun

Partner Fanatically

1 + 1 > 2

This does not imply you partner with anyone…



Our engineering team is eight "full-timers".

Actual project teams are larger – thanks to partnering.

The potential benefits are greater than you expect.

Be the Edge

Embrace and Push New Technologies

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!

Capitalize on Risk

Manage It, Pursue It

Program Management

Overall risk management.

Risk-driven scheduling.

Development

Classify team by competencies.

Be strategic about who does what.

Test

Business prioritized testing.

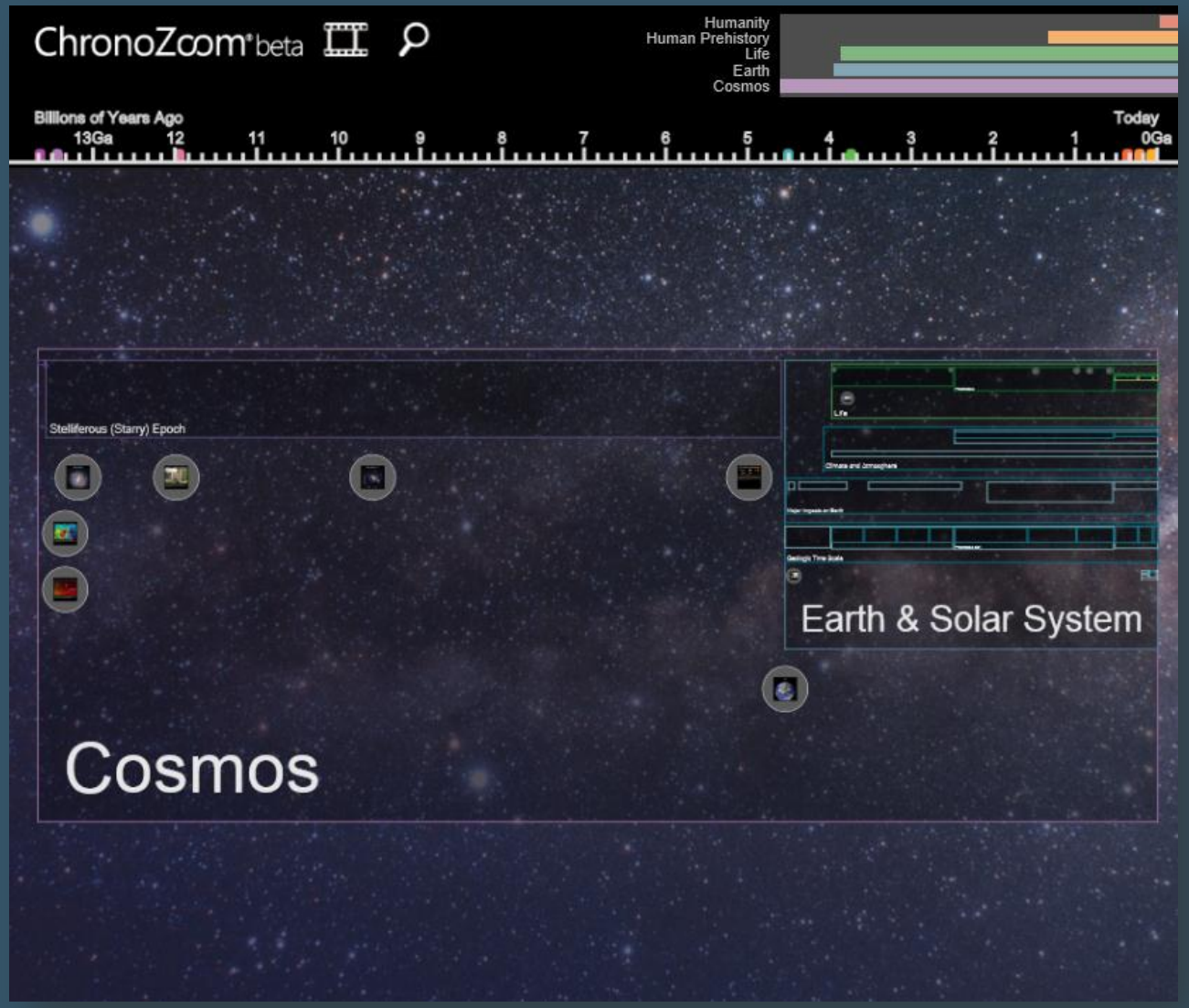
Platform multiplier strategies.

Culture

Embrace risk awareness!

Pursue riskier opportunities over time.

ChronoZoom demo

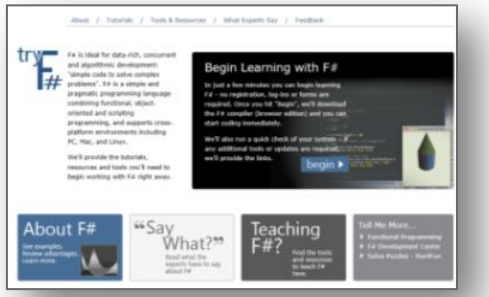


Recap

Incubation Lessons Learned

Express Capacity Meticulously

Open Source Prematurely



Throw Away Good Projects

Partner Fanatically

Add Drama

Be the Edge

Mandate Business Reviews

Capitalize on Risk

Microsoft®
Research