Lesson 8:
Why Your Open Source Start-Up Will Fail

Stephen R. Walli

Objectives

- Understand the basic ideas behind a software business model
- Understand the three sorts of software you create as a business
- Understand the difference between projects and products

Notes

https://github.com/jhu-ospo-courses/JHU-EN.601.210/tree/main/lessons/8

As a Company
You are either CONSUMING or PRODUCING
open source licensed projects

If you are CONSUMING OSI-licensed project components, then

- Orders of Magnitude of Value Capture
- It’s all about Engineering Economics
- Software Process is needed to protect customer stability
- Build vs Buy vs Borrow + Share

Congratulations … you're Red Hat

"We're not an open-source company. We're an enterprise software company with an open-source development model." – Paul Cormier, Red Hat President

If you are PRODUCING project components under OSI-licenses, then the software is either

- Core Value Proposition to Customers
- Complement Value Add to Core Value Proposition
- Context (Software Exhaust)

Congratulations … you're still a Software Business

The Business Model Canvas

Designed for:

Designed by:

On:

Iteration:

Key Partners

Who are our Key Partners?
Who are our key suppliers?
Which Key Resources are we acquiring from partners?
Which Key Activities do partners perform?

Key Activities

What Key Activities do our Value Propositions require?
Our Distribution Channels?
Customer Relationships?
Revenue Streams?

Key Resources

What Key Resources do our Value Propositions require?
Our Distribution Channels? Customer Relationships?
Revenue Streams?

Value Propositions

What value do we deliver to the customer?
Which one of our customer's problems are we helping to solve?
What bundles of products and services are we offering to each Customer Segment?
Which customer needs are we satisfying?

Collaborate in real time

Customer Relationships

What type of relationship does each of our Customer Segments expect us to establish and maintain with them?
Which ones have we established?
How are they integrated with the rest of our business model?
How costly are they?

Channels

Through which Channels do our Customer Segments want to be reached?
How are we reaching them now?
How are our Channels integrated?
Which ones work best?
Which ones are most cost-efficient?
How are we integrating them with customer routines?

Customer Segments

For whom are we creating value?
Who are our most important customers?

Cost Structure

What are the most important costs inherent in our business model?
Which Key Resources are most expensive?
Which Key Activities are most expensive?

Revenue Streams

For what value are our customers really willing to pay?
For what do they currently pay?
How are they currently paying?
How would they prefer to pay?
How much does each Revenue Stream contribute to overall revenues?

www.businessmodelgeneration.com

Three Types of Software You Can Build

- Core Value Proposition: The solution for the customer for money
- Complement Value Add: Additional components of the solution that makes the solution to the customer more complete
- Context: Tooling and process that you build for yourself to aid developing customer solutions

Building a project community in context spaces:

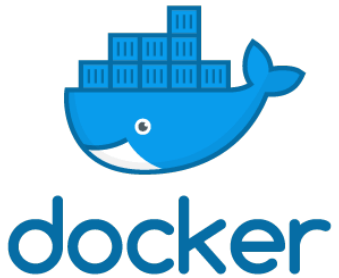
- Validates the approach to a problem
- Demonstrates expertise that can be used in recruitment
- Improves the quality of recruitment candidates
- Demonstrates committed values to collaboration amongst developers that further recruitment goals
- Captures innovation from outside sources

Building a project community in complement value-add spaces:

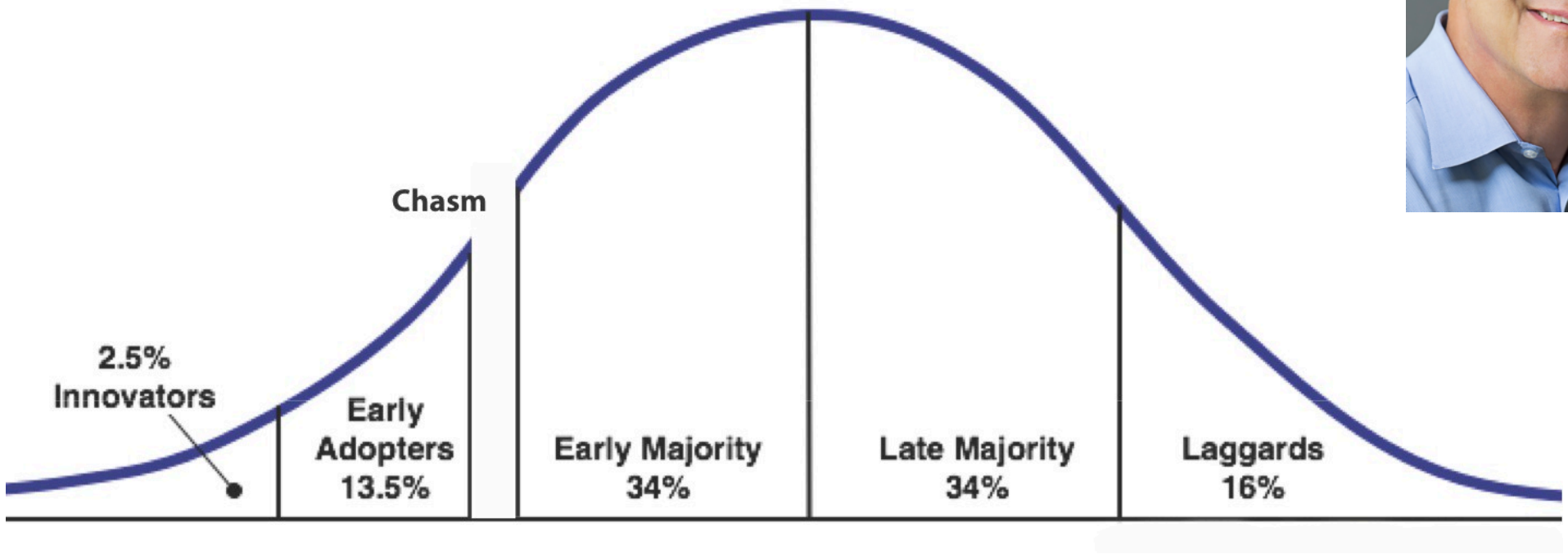
- Creates stickiness/inertia for the core value.
- Creates experts, advocates, and evangelists around the technology
- Hardens the complements with new configurations and contributions
- Captures direct value to the complements (indirectly to the core)
- Is possibly disruptive to competitors

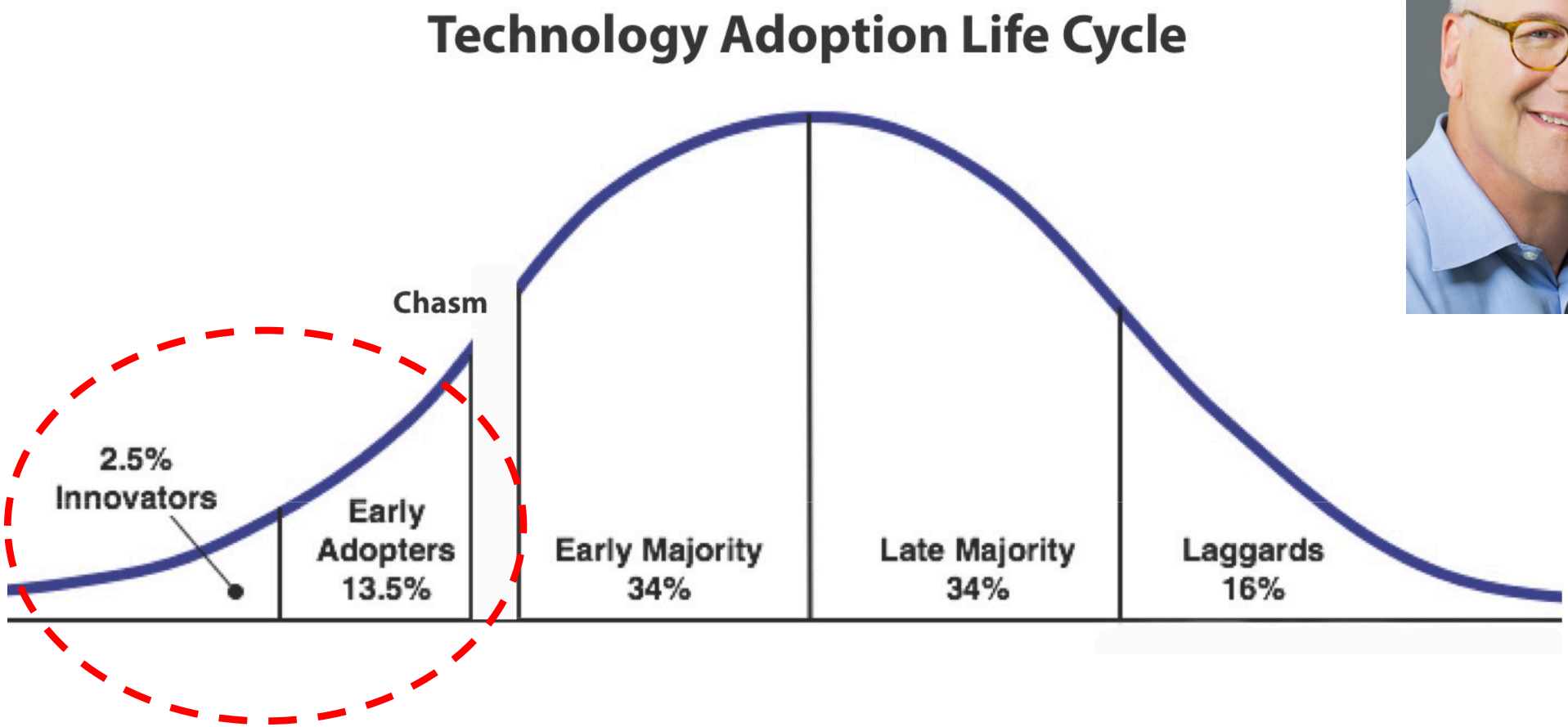
But –
publishing your core value proposition under an open source license:

- Makes potential partners into competitors
- Allows savvy IT consumers to avoid becoming customers
- Creates confusion for the sales and marketing team



Technology Adoption Life Cycle



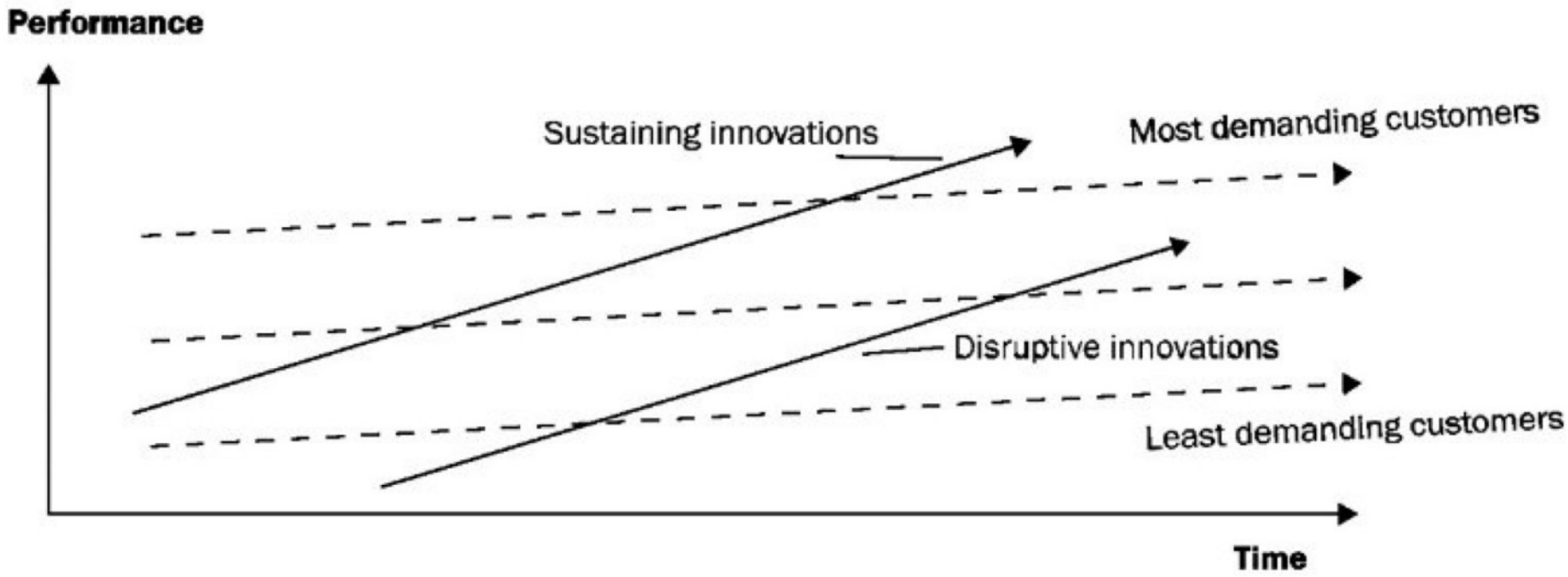


Crossing the Chasm, 3rd Edition: Marketing and Selling Disruptive Products to Mainstream Customers
Geoffrey Moore

If you invest in building community around the project that sits on your core value proposition:

- You create tension when competitors and partners and advanced IT users contribute value they want in your core value proposition.
- The power of innovation capture in community around a complement becomes the problem of innovation dilution in your core value proposition.
- You accelerate the creation of a community of early adopting users that aren't interested in paying for your software, instead of creating early adopting customers that understood your product solution sufficiently to give you money.

EXHIBIT 1
The Theory Of Disruptive Innovation



SOURCE: C.M. Christensen, *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail* (Boston: Harvard Business School Press, 1997).

Christensen's Tests

New market Disruption

- *Is there a large group of people who historically have not had the money, equipment or skill to do this thing for themselves?*

– OR –

- *To use the product or service, do customers need to go to an inconvenient, centralized location?*

Low-end market Disruption

- *Are there customers who would be happy to purchase a product with less (but good enough) performance?*

– AND –

- *Can we create a business model that enables us to earn attractive profits at the discounted prices?*

Then

- *Is the business innovation disruptive to ALL of the significant incumbent firms in the industry?*
- *If it appears to be a sustaining innovation to one or more significant players in an industry, then the odds are stacked in that firm's favour, and the new entrant unlikely to win.*

Projects are not Products

Projects are interesting buckets of technology developed collaboratively by like-minded engineers

Projects are interesting buckets of technology developed collaboratively by like-minded engineers

---

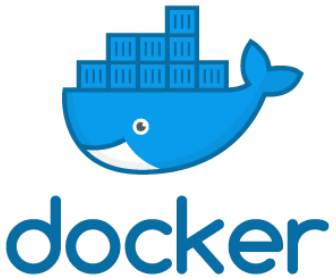
Products solve customer problems and money is exchanged for perceived value

Projects have communities
Communities have time and no money

Projects have communities
Communities have time and no money

Products have customers
Customers have money and no time

Parking your identity brand on any open source project you own, instead of the product/solution your customers buy, creates confusion for your messaging to customers.



Open Source Software is about Engineering Economics

Collaboratively-Developed Liberally-Licensed Software is about Engineering Economics

Open Source Software is about developers collaborating
Go build great software businesses