

Product Strategies

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

COSS C04

Licensed under CC BY 4.0 International

Agenda

1. Product management in context
2. Software features
3. Feature differentiation
4. The open core model
5. IP rights management
6. Cloud computing challenges
7. Commercial open source life-cycle

Structure product and services so that you

- 1. Maximize conversion to paying customer**
- 2. While benefiting from user community**
- 3. And keeping the competition at bay**

1. Product Management in Context

Product Management in Context (Recap)

| Management | | | | | |
|---------------------|----------------------|---|-------------|---------|--------------------------------------|
| Sales and Marketing | | Product Management (Inbound Marketing) | Engineering | | Other (HR, Office, Finances, ...) |
| Sales | (Outbound) Marketing | | Development | Support | |
| | | | | | |
| | | | | | |
| | | | | | |

The Main Questions for Product Management to Answer

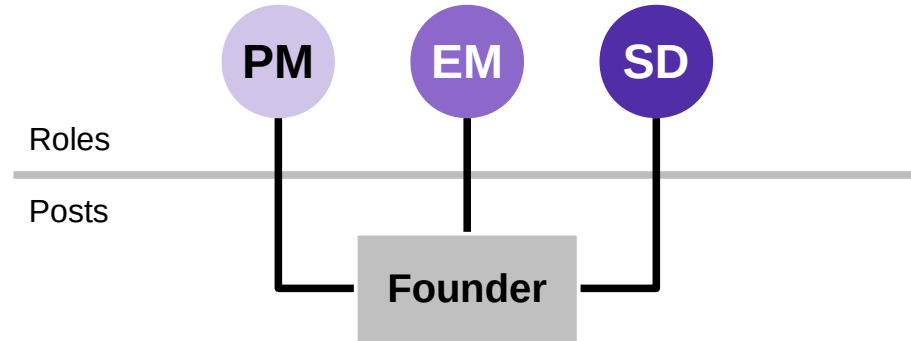
- **What / why / when should be doing it?**
 - For the company (strategic product management)
 - For the product (technical product management)
- Strategic product management
 - Performs market research
 - Defines the opportunity
- Technical product management
 - Performs market research
 - Specifies the product

The Role of the Product Manager

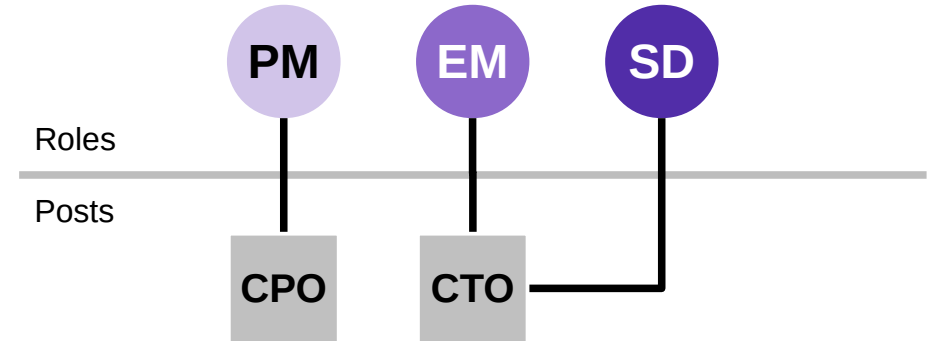
- CEO of the product
 - Holds overall responsibility for the product (including revenue responsibility)
- ~~Voice of the customer~~
 - ~~Channels market requirements towards development~~

Who is a Product Manager?

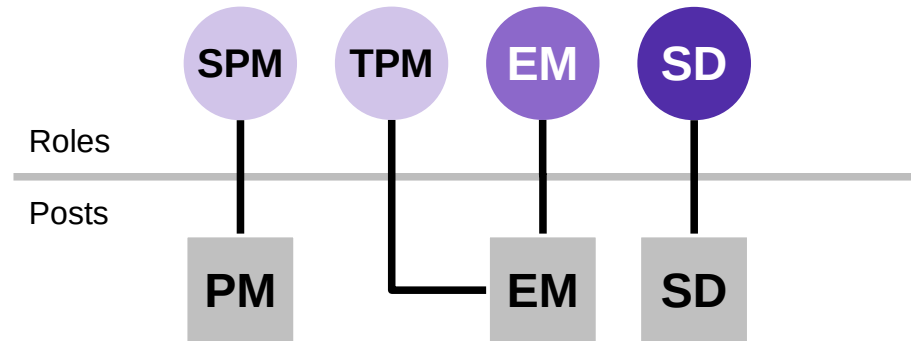
Solo entrepreneur



Team-based startup



Mature vendor



PM = Product manager (strategic + technical)
SPM = Strategic product manager
TPM = Technical product manager
CPO = Chief Product Officer
EM = Engineering manager
VPE = VP of engineering
CTO = Chief Technology Officer
SD = Software developer

Project vs. Product Manager

- A project manager
 - Manages projects (including people)
- A product manager
 - Defines products, may lead projects

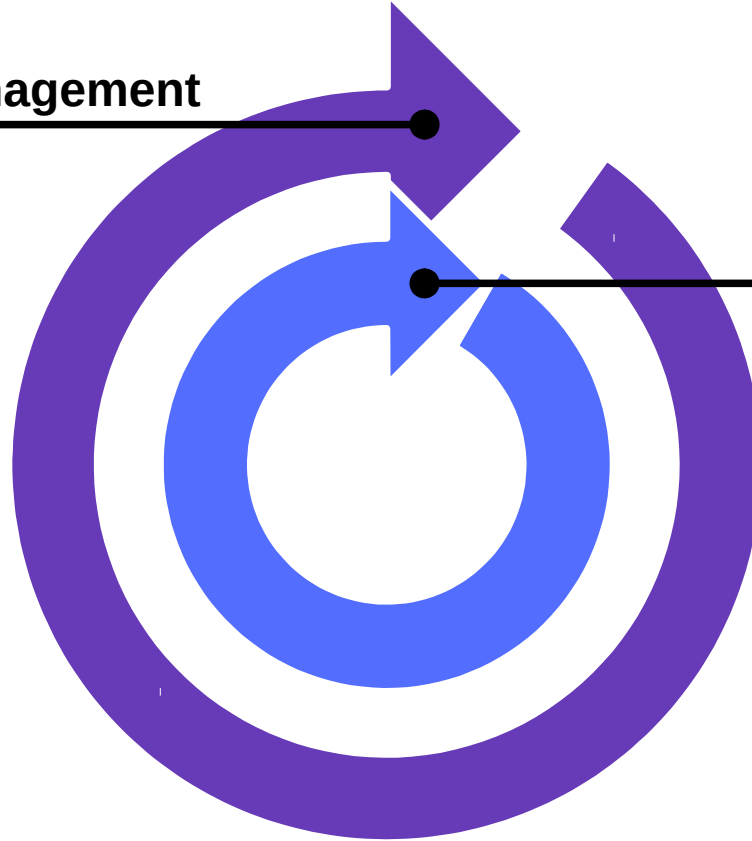
Product Management is an Ongoing Process

Strategic product management

- Opportunity
- Segmentation
- Partnerships
- Architecture
- Roadmap
- ...

Technical product management

- Releases
- Requirements
- User experience
- Compliance
- ...



2. Software Features

Tools of the Trade

- Vision
- Glossary
- Features

Why Written Communication?

- Writing helps you ...
 - Discipline your thinking
 - Uncover errors
 - Identify omissions
 - Avoid repeating yourself
 - Communicate asynchronously
 - Multiply your powers

Product Vision

- A product vision ...
 - Captures the essence of the product and the reasons for its existence
 - As the business value it provides to users
 - Names users and customers
 - Embodies the core structure of an underlying business model
 - Is inspiring (inspires users / customers)
 - Is timeless (not bound to a schedule)
 - Is focused (reduces to the essentials)
 - Is a decision aid (the ultimate arbiter)

Wahlzeit

[[show](#) | [tell](#)] — [[signup](#) | [login](#) | [configure](#)]

Last Viewed

Praise: 9.67



Photo by [testuser](#)

Photo Filter

[Click to toggle filter!](#)

Community

[It is Wahlzeit!](#)



Who/what/where is that?

[Click to show/hide description!](#)

Photo by [testuser](#)

Praise it!

☐ 10
☐ 9
☐ 8
☐ 7
☐ 6
☐ 5
☐ 4
☐ 3
☐ 2
☐ 1
Or [skip](#) it.

a friend about this photo: <http://localhost:8585/x1ac1.html> — Send to the owner of this photo!

Please help keep this community site clean! photo as inappropriate if necessary.

This website is to show the best in photos!

[[blog](#)] — [[about](#) | [contact](#) / [imprint](#) | [terms](#)] — [language: en | [de](#)] — [photo size: [XS](#) | [S](#) | M | [L](#) | [XL](#)] — [debug: [reset](#)]

[processing time: 0.004 seconds]

Example Product Vision

- Flowers Product Vision
 - The Flowers social network helps flower enthusiasts worldwide to connect with each other and enjoy following their favorite hobby online. Centered on showing and rating favorite flower photos, it inspires growing and presenting ever more beautiful flowers. With a highly engaged user community, Flowers is the best place for producers and sellers of gardening supply to reach out to customers and engage with them. Such engagement involves understanding flower enthusiasts' needs around gardening supplies and selling to them.

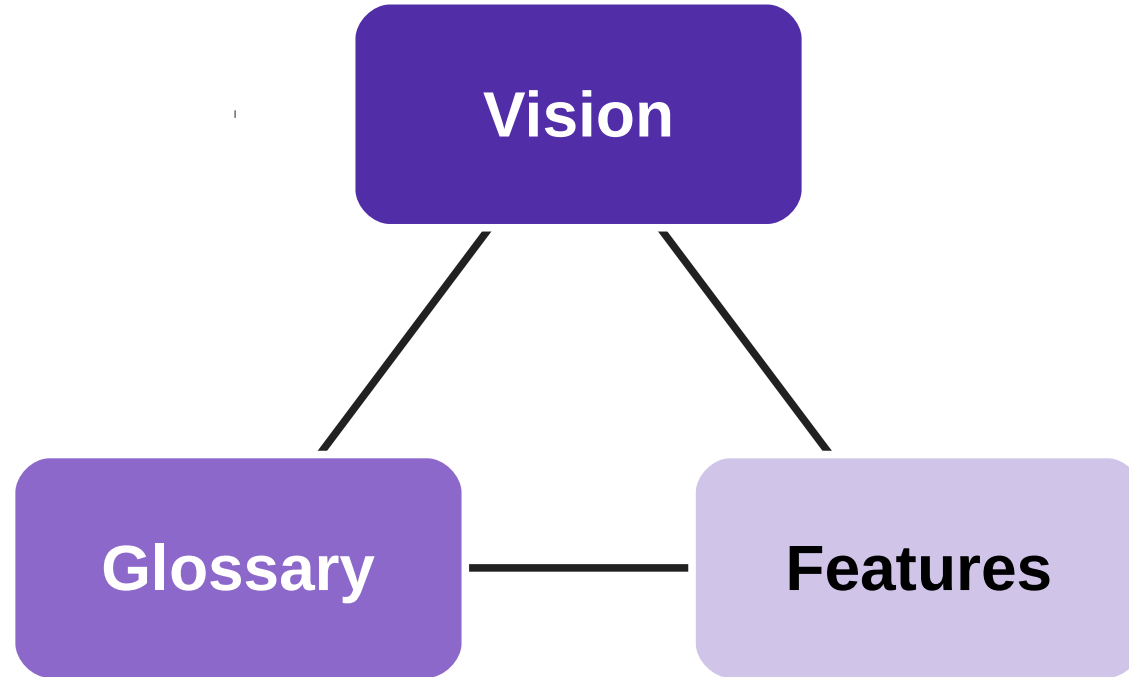
Product (Domain) Glossary

- A domain glossary ...
 - Is a list of concepts and their definition from the product being developed
 - Is the missing link between the vision and the requirements
 - Is a “poor man’s approach” to a domain model
- A glossary term ...
 - Is a domain concept with a clearly defined name
 - Has a crisp “is a” definition, followed by secondary properties
 - Is defined using a single sentence, at max. a paragraph

Example Domain Glossary

| Term | Definition |
|-------------------------|---|
| Photo | A photo is a digital image provided in either JPEG or PNG format. It must be at least of size 512x512 pixel. |
| Flower photo | A flower photo is a photo of one or two flowers. If more flowers are visible and in-focus, the photo is a flower bed photo. |
| Flower bed photo | A flower bed photo is a photo of three or more flowers in-focus. If less than three flowers are visible, it is a flower photo. |
| Photo Rating | A photo rating is a numerical value in the range of 1..10 that expresses the appreciation of a photo numerically. 1 is lowest, 10 is highest. |
| Individual photo rating | An individual photo rating is a photo rating that expresses a particular individual's appreciation of a photo at a particular point in time. |

Connecting Vision with Features Through Glossary




Product Feature

- A (software) feature is ...
 - A distinguishing characteristic of a software item (for example, performance, portability, or functionality) [IEEE 829]
- Also, in Scrum
 - An epic is ...
 - A large feature awaiting break-down into smaller features; it acts as a placeholder for these smaller features
 - A (user) story is ...
 - A feature presented using a the user-story-pattern that is small enough to be implemented in a sprint
- Refactoring
- Bug report



[[show](#) | [tell](#)] — [[signup](#) | [login](#) | [configure](#)]

Tell a friend!

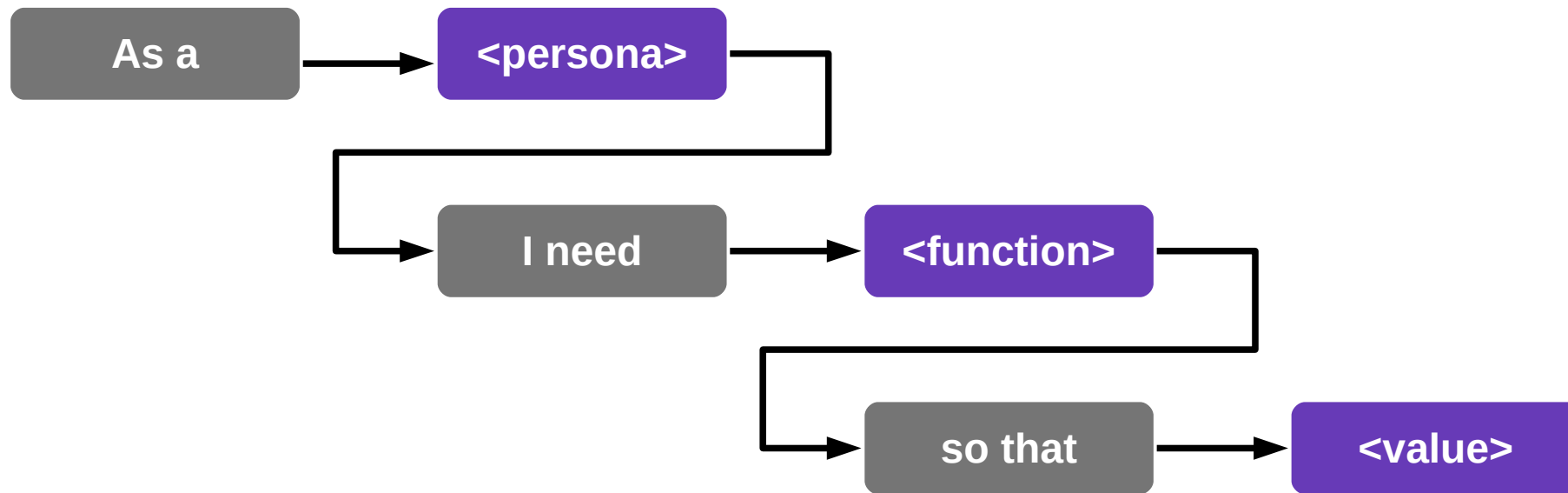
| | | |
|---|---------------------|--|
|  | From Email Address: | <input type="text" value="dirk@riehle.org"/> |
| | To Email Address: | <input type="text"/> |
| | Email Subject: | <input type="text"/> |
| | Message: | <div><div>http://localhost:8585/ http://localhost:8585/x1ac5.html</div><div>Tell!</div></div> |

This website is to show the best in photos!

[[blog](#)] — [[about](#) | [contact](#) / [imprint](#) | [terms](#)] — [language: en | [de](#)] — [photo size: [XS](#) | [S](#) | M | [L](#) | [XL](#)] — [debug: [reset](#)]

[processing time: 0.012 seconds]

User Story (Sentence Template)



As a visitor, I can tell a friend about Flowers to share my enthusiasm for the service

3. Feature Differentiation

Structure product and services so that you

- 1. Maximize conversion to paying customer**
- 2. While benefiting from user community**
- 3. And keeping the competition at bay**

Which Feature Goes Where? (Recap)

- **Community edition**

- **Core product**

- **Core software**
 - **Provided under an open source license**
 - Some complementary artifacts
 - Self-help services

- **Commercial edition**

- **Core product**

- **Core software**
 - **Provided under a commercial license**
 - **Additional functionality**
 - Complementary artifacts
 - Self-help services

- **Basic product = core product +**

- Fitness for use / certification
 - Indemnification
 - Support services

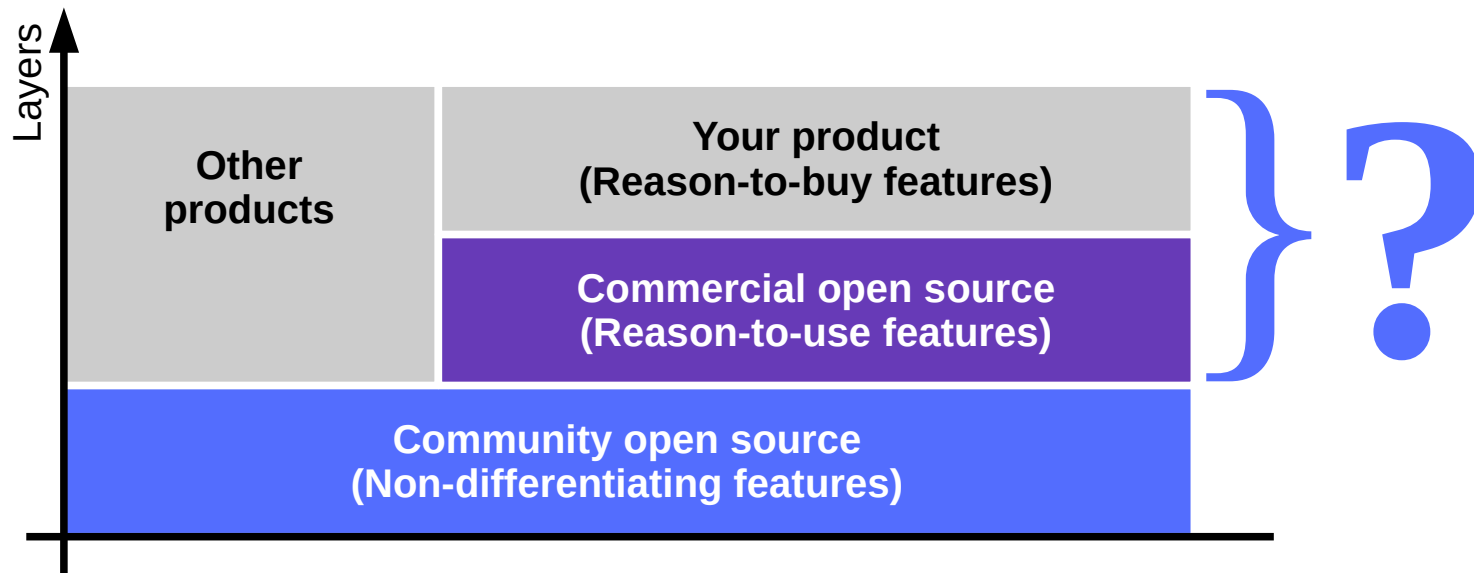
- **Whole product = basic product +**

- Training
 - Consulting
 - Operations

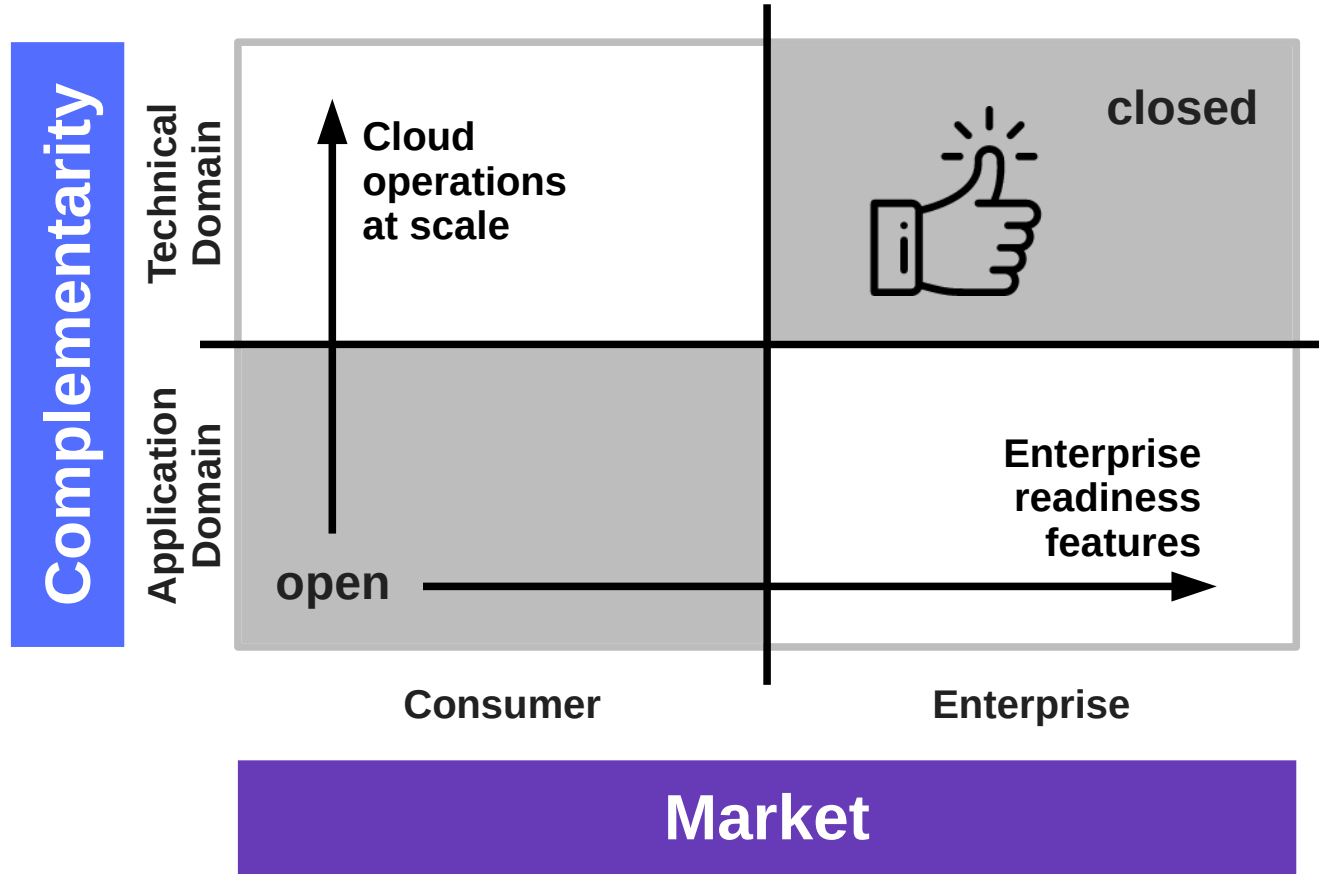
A Commercial Open Source Feature Classification

- Non-differentiating
 - The feature is competitively not differentiating and readily available in community open source
- Reason-to-use
 - Users come to your software, because the feature is not ubiquitous
- Reason-to-buy
 - Users upgrade to paying customers to receive this feature

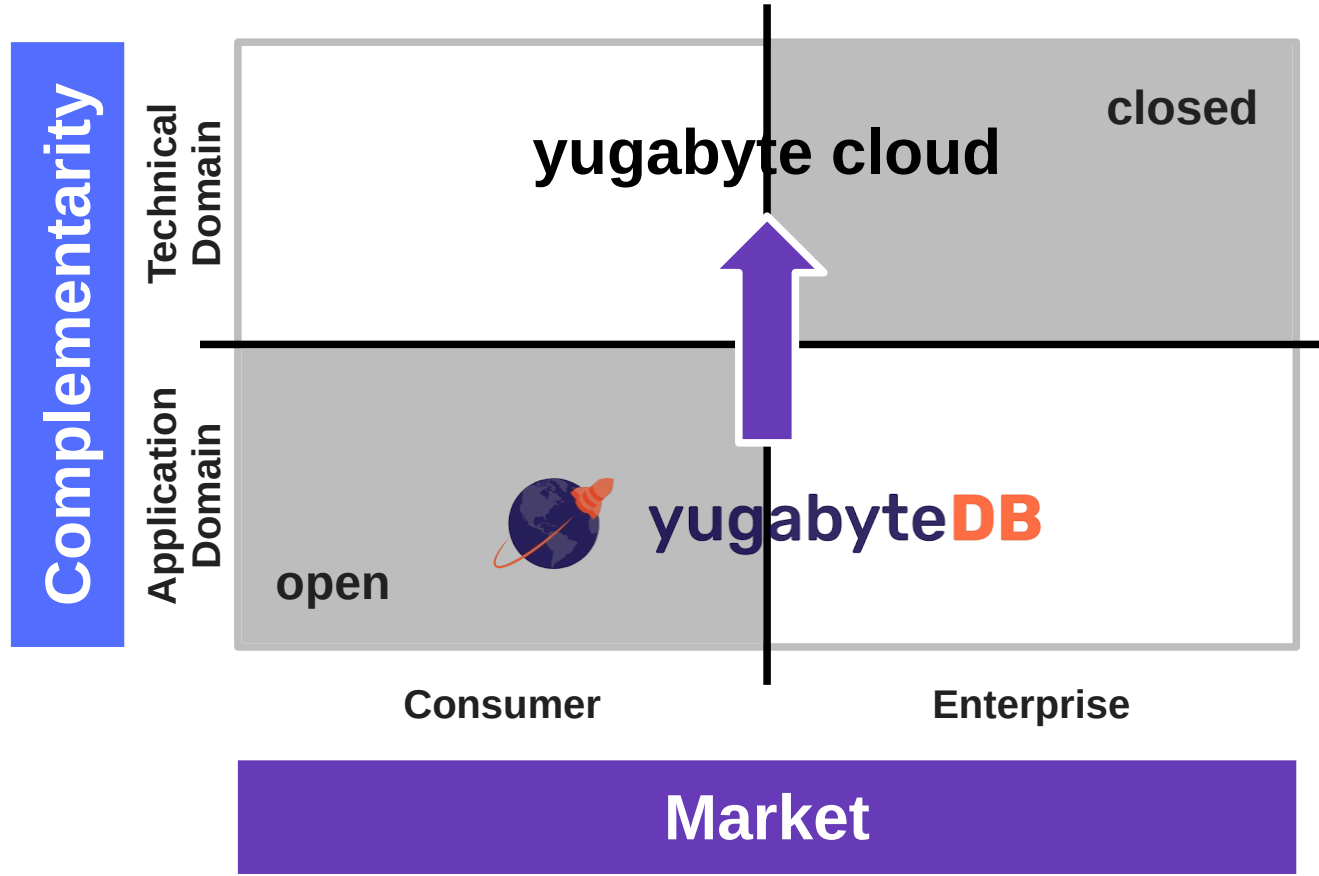
Open / Closed Software Feature Differentiation



How to Think About Feature Differentiation



Yugabyte 2020 (Example Feature Differentiation)



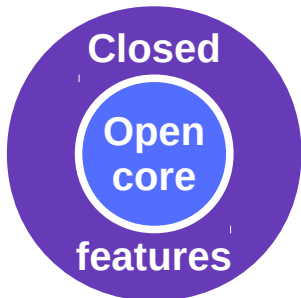
4. The Open Core Model

Intellectual Property Modularity

- Intellectual property (IP) modularity
 - The practice of splitting software into modules of different licenses



- Open core model
 - A particular form of IP modularity where there is
 - An “open” core available under an open source license and
 - Functional extensions of the core available only under a commercial license



What do Open Core Users Worry About?

- That the vendor
 - Withholds critical features
 - Stops updating old features
 - Delays delivery of new features
 - Lacks backwards compatibility

The Commercial Open Source Pledge

- Design choices for a pledge to create trust
 - Always open source or not
 - Permissive or copyleft
 - Allow competition or not
 - Copyright transfer or not
 - Support community or not
 - Influence roadmap or not

[1] See <https://dirkriehle.com/2019/06/11/the-commercial-open-source-pledge/>

5. IP Rights Management

Structure product and services so that you

1. Maximize conversion to paying customer
2. While benefiting from user community
3. And keeping the competition at bay

Commercial Forks of Commercial Open Source Software

Compiere®



Nagios®



IP Rights Management (Recap)

- Intellectual property rights imperative (of single-vendor open source)
 - “Always act in such a way that you, and only you, possess the right to provide the open source project under a license of your choice.” [1]
- Use contributor agreement to maintain ownership
 - Almost all single-vendor open source firms require copyright transfer for any contributions to maintain full IP ownership [2]

[1] Riehle, D. (2009). [The intellectual property rights imperative](#). Web-published.

[2] All you really need is a relicensing right though

Dual / Multi-Licensing

- Dual licensing / multi-licensing
 - The practice of licensing a piece of software under two or more licenses
- Commercial open source licensing
 - At least one open source and one commercial license
- Just which open source license(s)?
 - Basic idea: Use copyleft to keep competitors away

Product Types

- Application
 - A piece of software that can be used as is for a business purpose
 - Example applications
 - A financial accounting system
 - A compiler
 - Applications dominated the second wave of commercial open source
- Component
 - A piece of software that is integrated with other components to become an application
 - Example components
 - Functional library
 - Database system
 - Components dominate the current third wave of commercial open source

[1] Riehle, D. (2009). The intellectual property rights imperative. Available at <http://wp.me/pe4V6-io>

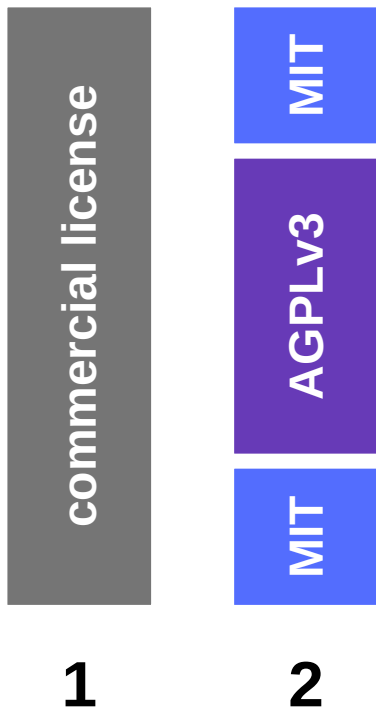
[2] All you really need is a relicensing right though

Choice of Open Source License in Commercial Open Source

| Deployment | Product Type | |
|--------------|-------------------|-------------|
| | Component | Application |
| In-the-cloud | AGPLv3 + shims | N/A |
| On-premise | GPLv2 | AGPLv3 |

Resulting Licensing Structures (Until Recently)

Component



Application



6. Cloud Computing Challenges

The Move Into the Cloud

- A tectonic shift
 - (Almost) everything is moving into the cloud
- Open source
 - Becomes an on-ramp to the cloud
- Conversion is
 - From self-hosted to vendor-hosted
- The hyperscalers
 - Are possibly the new competition






A Resulting License Change

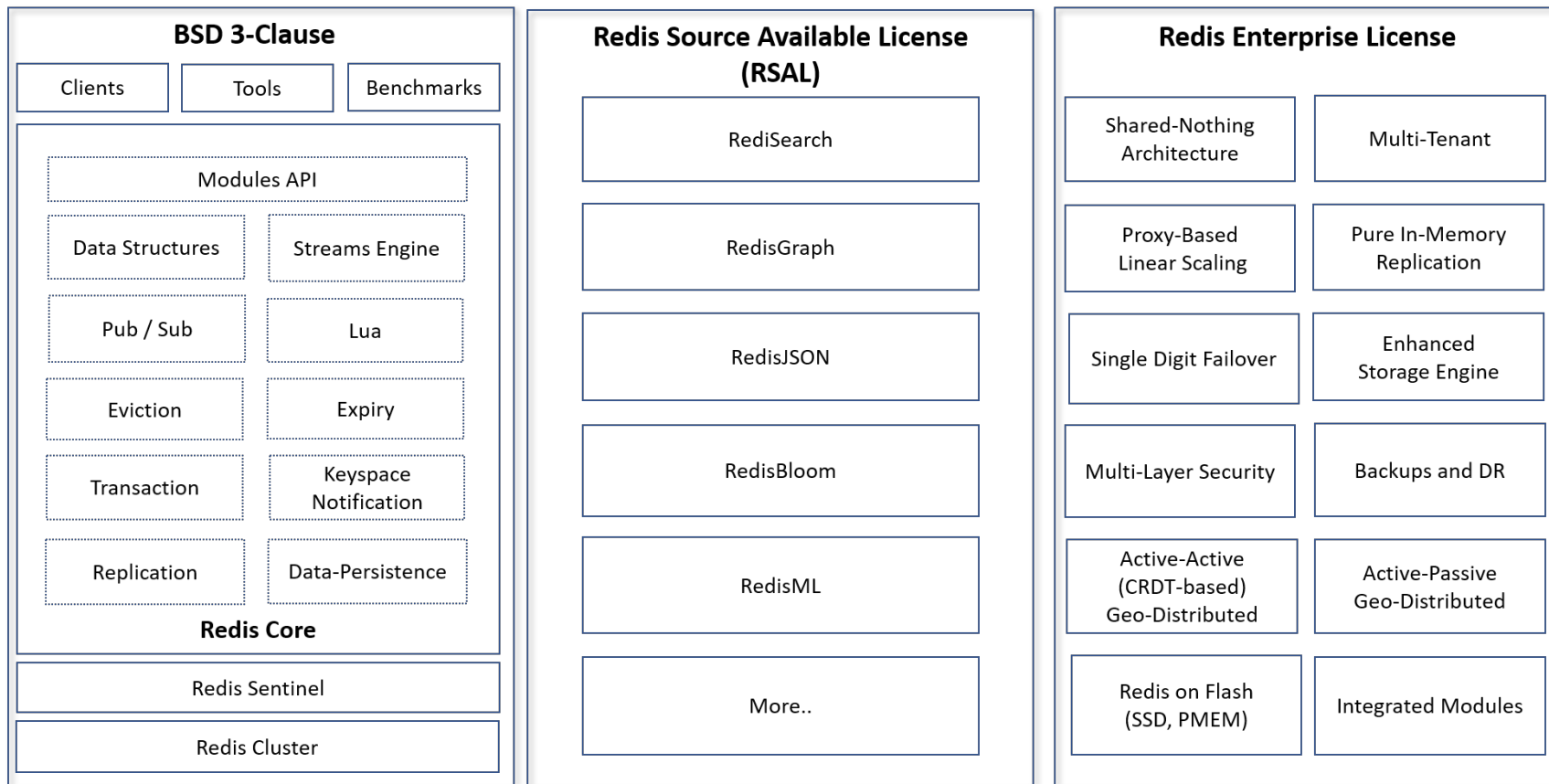


| Component | From-License | To-License |
|------------------------|--------------------------------|--------------------------------|
| Community server | AGPLv3 (and commercial) | SSPL (and commercial) |
| Connectors and drivers | Apache 2.0 (and commercial) | Apache 2.0 (and commercial) |
| Cloud management | Commercial (only) | Commercial (only) |

More Recent Licensing Changes

| Who? | What? | When? | From License | To License |
|---|----------------------------|-------|--------------|------------|
|  | MaxScale (Proxy Server) | 2016 | GPLv2 | BSL |
|  | Extensions | 2019 | Apache 2.0 | CCL |
|  | Extensions | 2019 | AGPLv3 | RSAL |

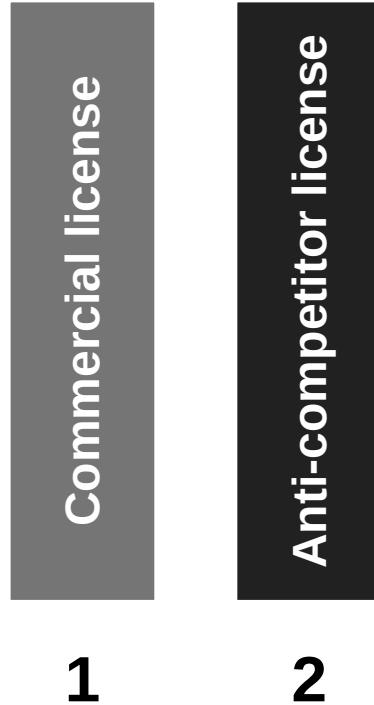
Redis After Licensing Change (AGPLv3 to RSAL) [1]



[1] <https://redislabs.com/blog/redis-labs-modules-license-changes/>

Triple-Licensing Components to Keep Competitors at Bay

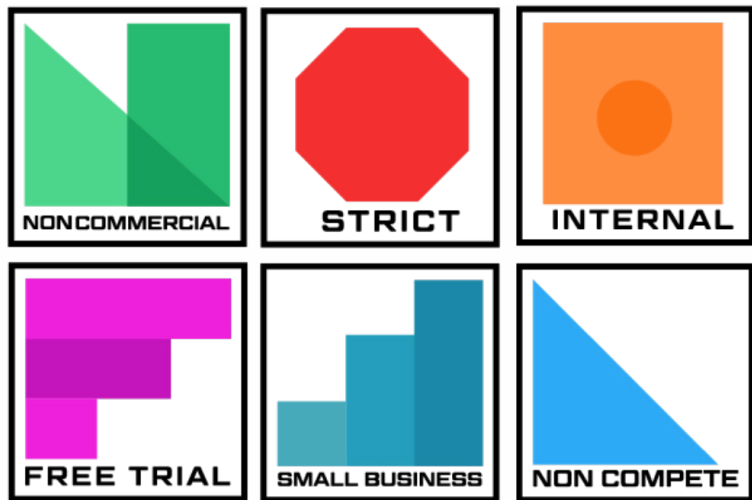
Dual-Licensing



Triple-Licensing



The Polyform Project [1]



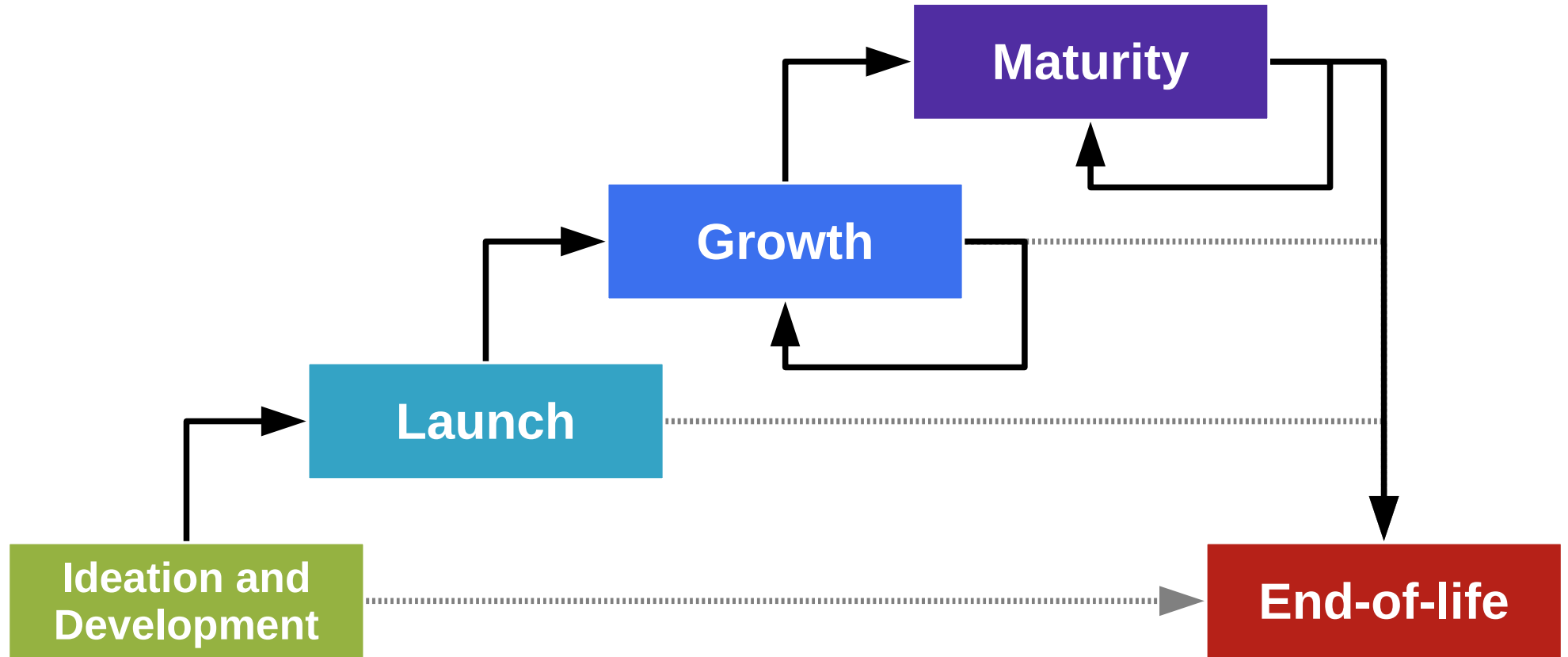
What is PolyForm?

PolyForm is a project to draft and make freely available plain-language source code licenses with limited rights.

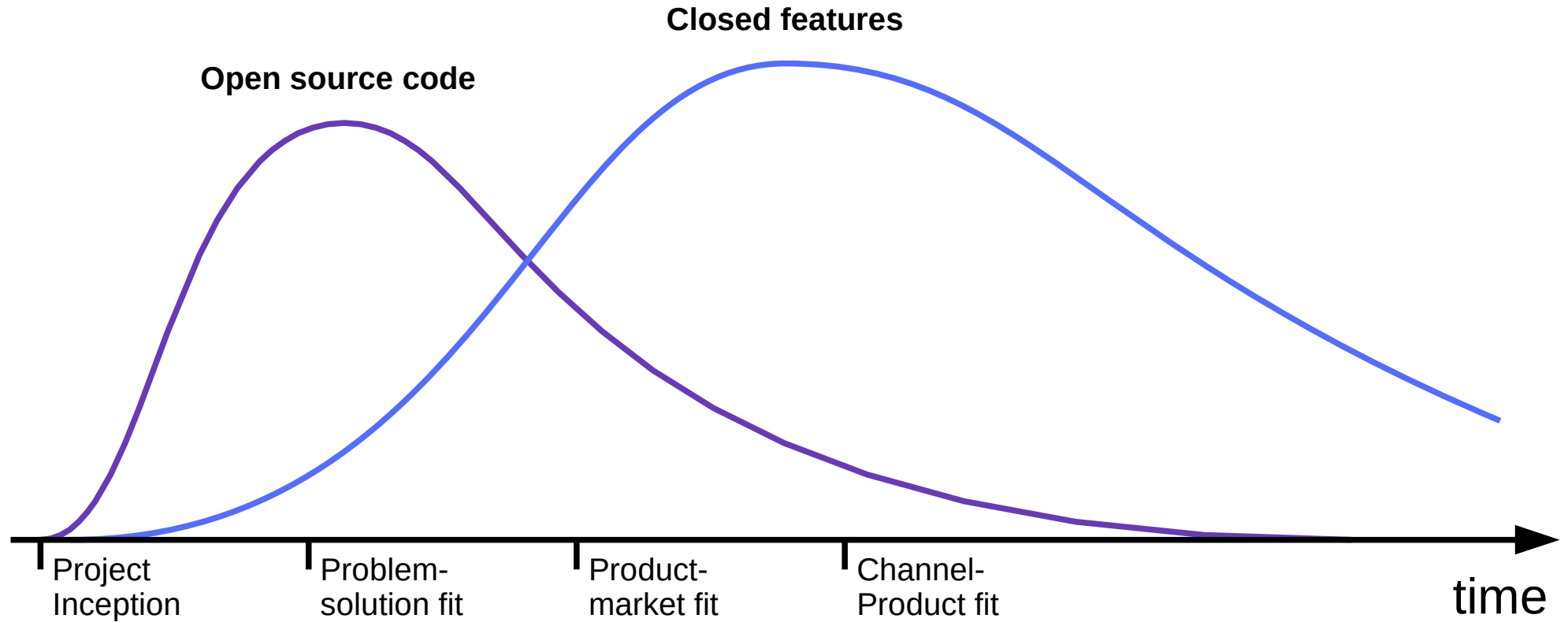
[1] See <https://polyformproject.org/>

7. The Commercial Open Source Life-cycle

Product Life-cycle



Where New Features (Innovation) Goes



Life-cycle of Single-Vendor Firms

- Early years
 - Full fair open source play
- Growth years
 - Full fair open source play
- Maturity
 - Increased closing of product

Summary

1. Product management in context
2. Software features
3. Feature differentiation
4. The open core model
5. IP rights management
6. Cloud computing challenges
7. Commercial open source life-cycle

Thank you! Questions?

dirk.riehle@fau.de – <http://osr.cs.fau.de>

dirk@riehle.org – <http://dirkriehle.com> – [@dirkriehle](#)

Credits and License

- Original version
 - © 2020 Dirk Riehle, some rights reserved
 - Licensed under [Creative Commons Attribution 4.0 International License](#)
- Contributions
 - None yet