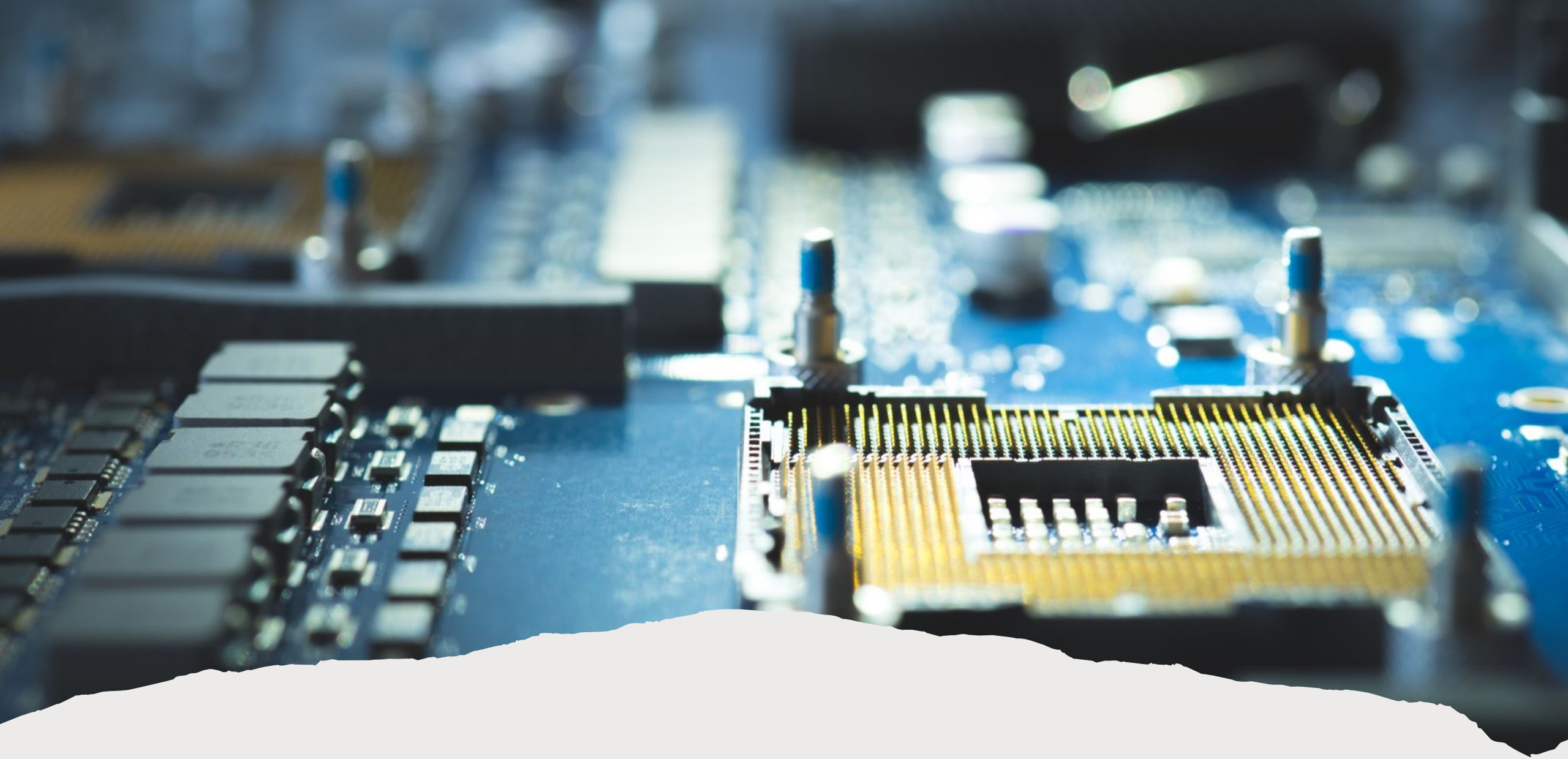
PRODUCT ENGINEERING MINDSET - WORKSHOP

Balaji Thiruvengadam

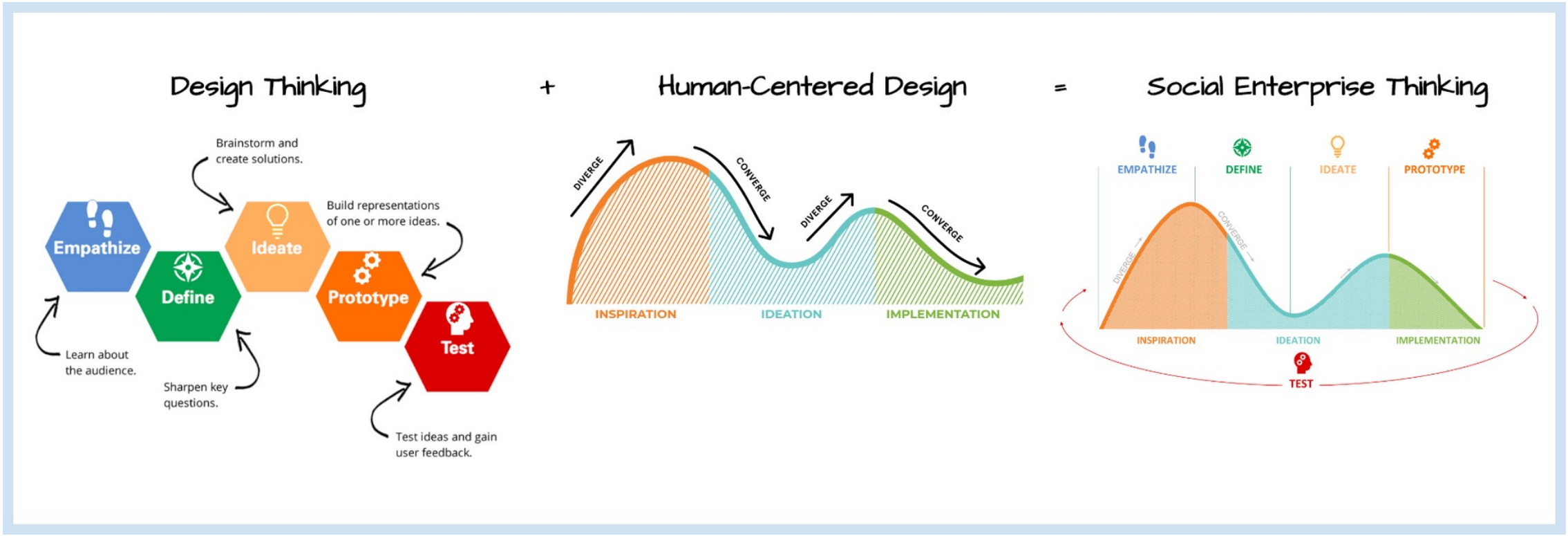
DAY 3 - AGENDA

- Recap of day2
- Swiggy problem Exercise: Convergent & Divergent thinking
- Swiggy problem Exercise: Prioritise and build MVP
- Does product version matters to customer?
- Why versioning and its strategy is important for enterprise product?
- Branching Strategy
- How product and features to be designed for customers having different versions?

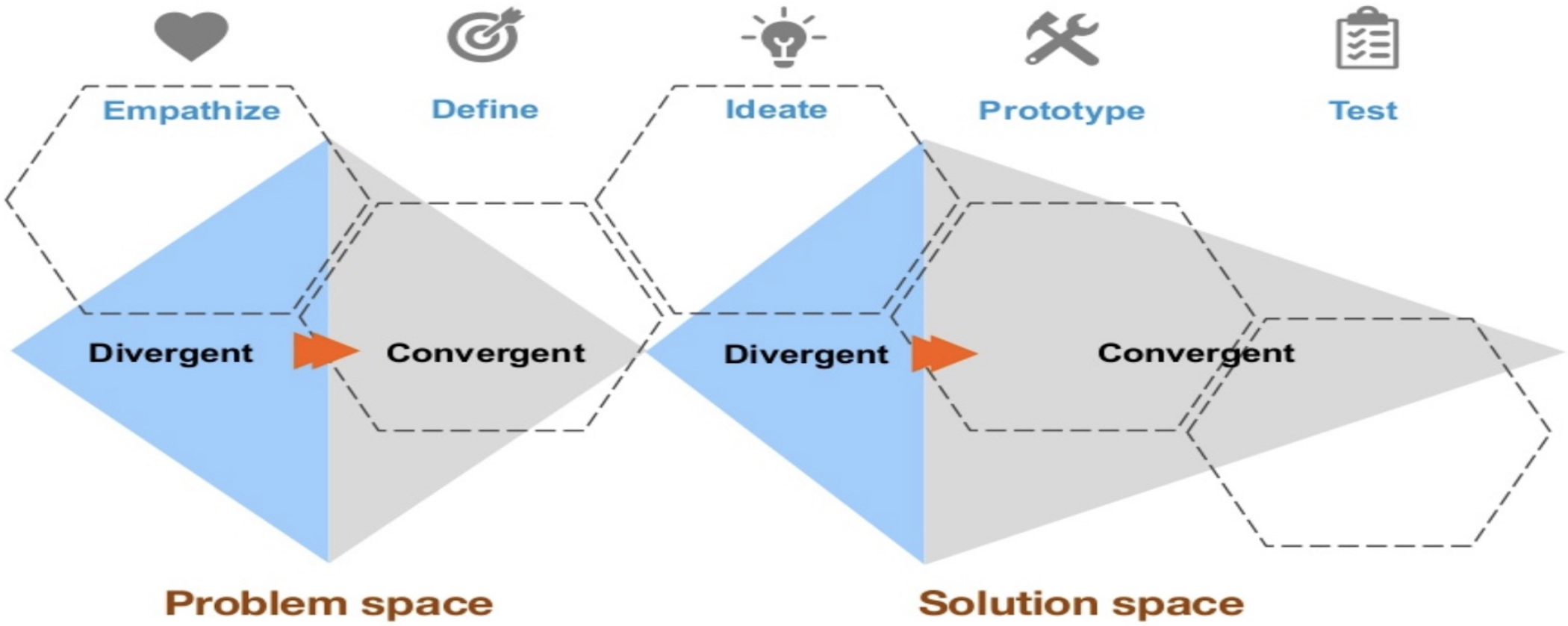


PRODUCT ENGINEERING – RECAP DAY2

SOCIAL ENTERPRISE THINKING

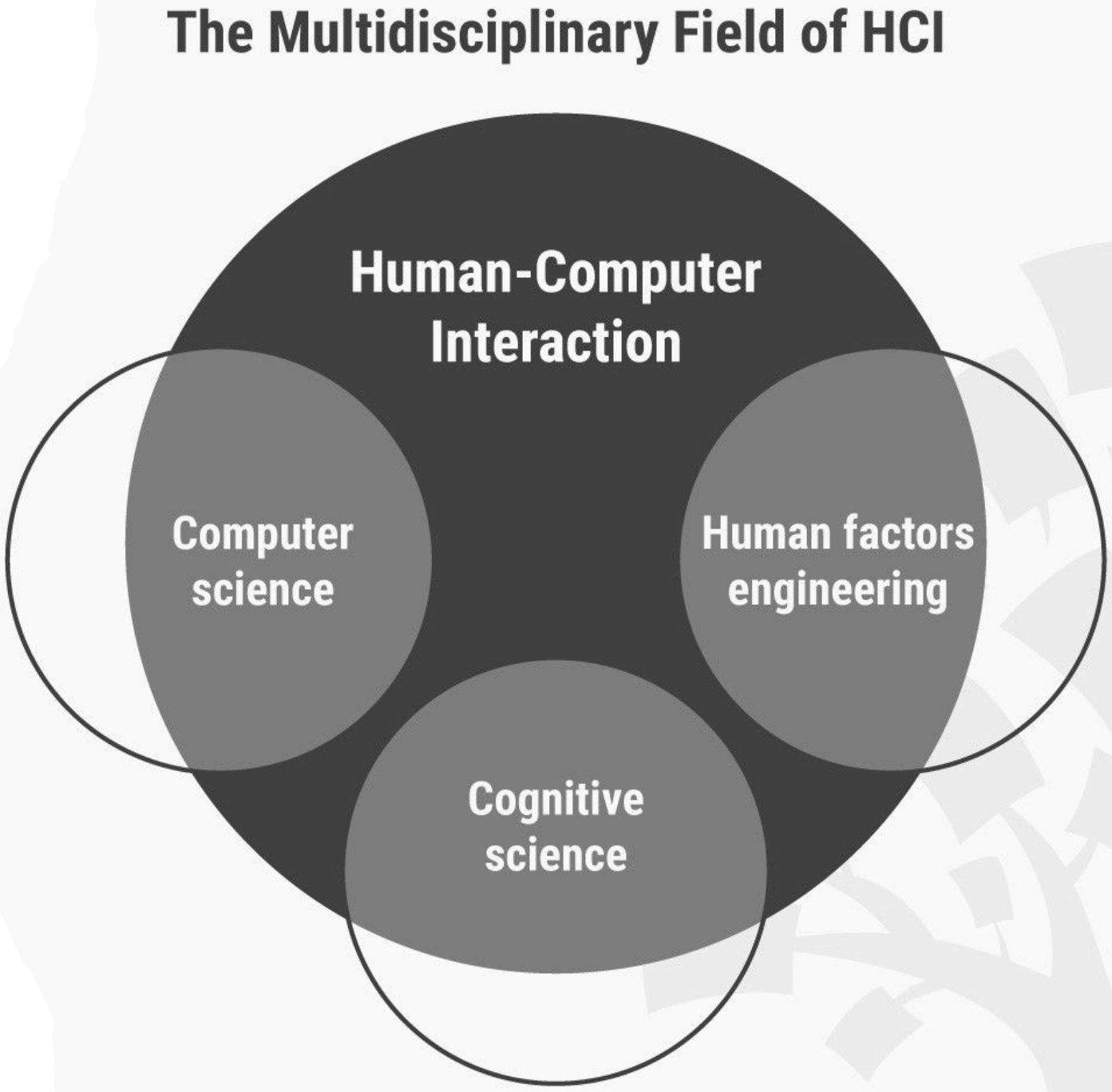


DIVERGENT AND CONVERGENT THINKING



HUMAN COMPUTER INTERACTION

Human-computer interaction (HCI) is a multidisciplinary field of study focusing on the design of computer technology and, in particular, the interaction between humans (the users) and computers. While initially concerned with computers, HCI has since expanded to cover almost all forms of information technology design.



VALUE PROPOSITION CANVAS

Customer Profile

•***Gains*** – the benefits which the customer expects and needs, what would delight customers and the things which may increase likelihood of adopting a value proposition.

•***Pains*** – the negative experiences, emotions and risks that the customer experiences in the process of getting the job done.

•***Customer jobs*** – the functional, social and emotional tasks customers are trying to perform, problems they are trying to solve and needs they wish to satisfy.

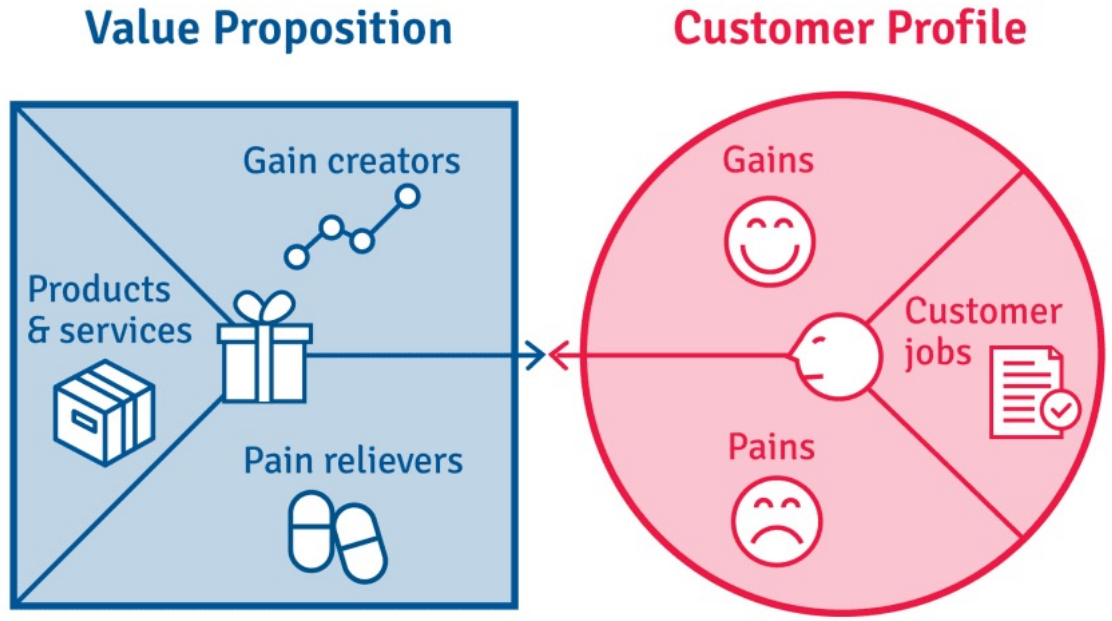
A customer profile should be created for each customer segment, as each segment has distinct gains, pains and jobs.

Value Map

•***Gain creators*** – how the product or service creates customer gains and how it offers added value to the customer.

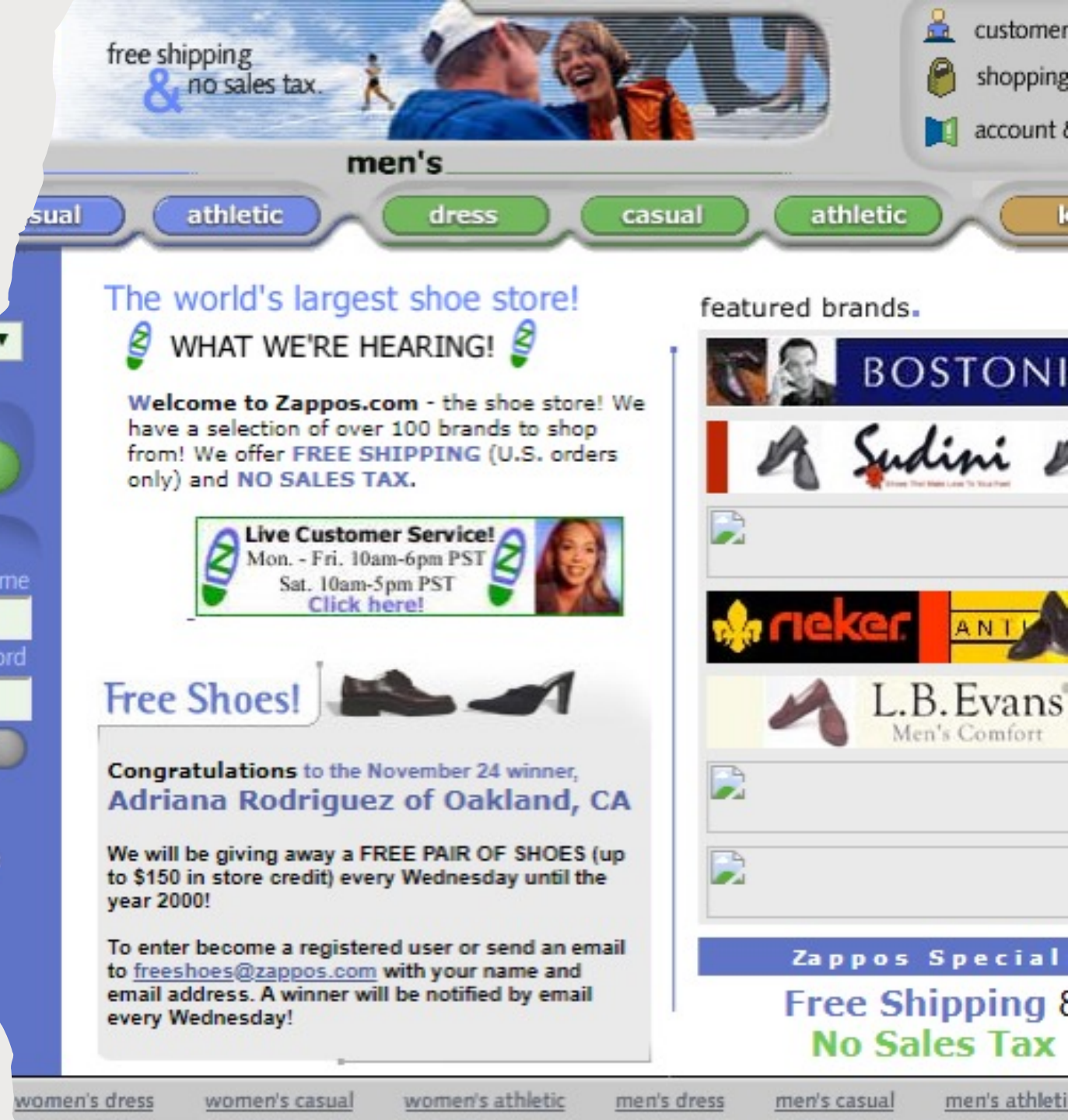
•***Pain relievers*** – a description of exactly how the product or service alleviates customer pains.

•***Products and services*** – the products and services which create gain and relieve pain, and which underpin the creation of value for the customer.



MINIMUM VIABLE PRODUCT

- The year was 1999. There was a young guy who wished to own a particular pair of shoes. He went to a mall close to his place but, unfortunately, he was unable to find the pair.
- Frustrated, he came up with an idea to sell shoes online and that's where it all started.
- Minimum Viable Product (MVP) was born.
- Rather than conducting extensive and expensive market research, he built a basic website. Then, he approached a shoe store, clicked pictures of shoes, and placed them on his site. Upon receiving the order, he purchased the shoes from the store and shipped them out.
- Although he lost money on every sale, it was an incredible way to test a business idea. Once he inferred that customers are willing to purchase shoes online, he started to turn his idea into a fully functioning business.
- This is how Nick Swinmurn built the company Zappos, which was later acquired by Amazon for $1.2 billion USD.
- The approach that Nick followed is what is termed MVP Development in today's time.



WHAT IS MVP?

An MVP (Minimum viable product) is a basic, launchable version of the product that supports minimal yet must-have features (which define its value proposition). An MVP is created with an intent to enable faster time to market, attract early adopters, and achieve product-market fit from early on.

Once the MVP is launched, initial feedback is awaited. Based on this feedback, the company will reiterate to fix the bugs and introduce new features that those early adopters suggest.

The MVP approach allows for:

- Making an early market entry which leads to a competitive advantage
- Enabling early testing of the idea with actual users to check whether the product is able to solve their problems efficiently
- Working effectively towards developing a fully-fledged product that integrates user feedback and suggestions

MINIMUM VIABLE PRODUCT

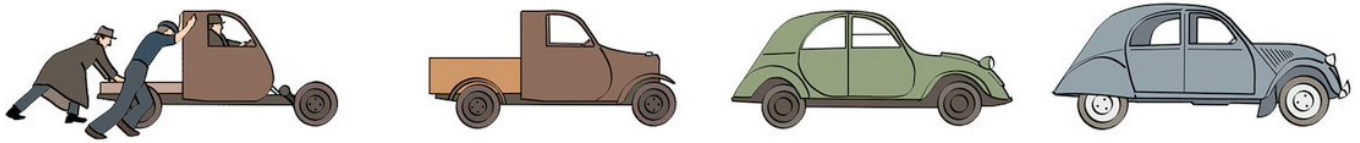
HOW NOT TO BUILD A MINIMUM VIABLE PRODUCT

1 2 3 4

ALSO HOW NOT TO BUILD A MINIMUM VIABLE PRODUCT

1 2 3 4

HOW TO BUILD A MINIMUM VIABLE PRODUCT



1 2 3 4

VERSIONING

Goal Driven Versioning: A Better Schema for Product Releases

Goal Driven Versioning (GDV) is a better take on MVP, and the following wins immediately become available with it:

- Better alignment around the level of maturity a product is and should be at
- A consistent progression of goals set for every product you ship
- A better understanding of the expected cost of each level of maturity
- An easy way to strategize and communicate investment in each of your products over time

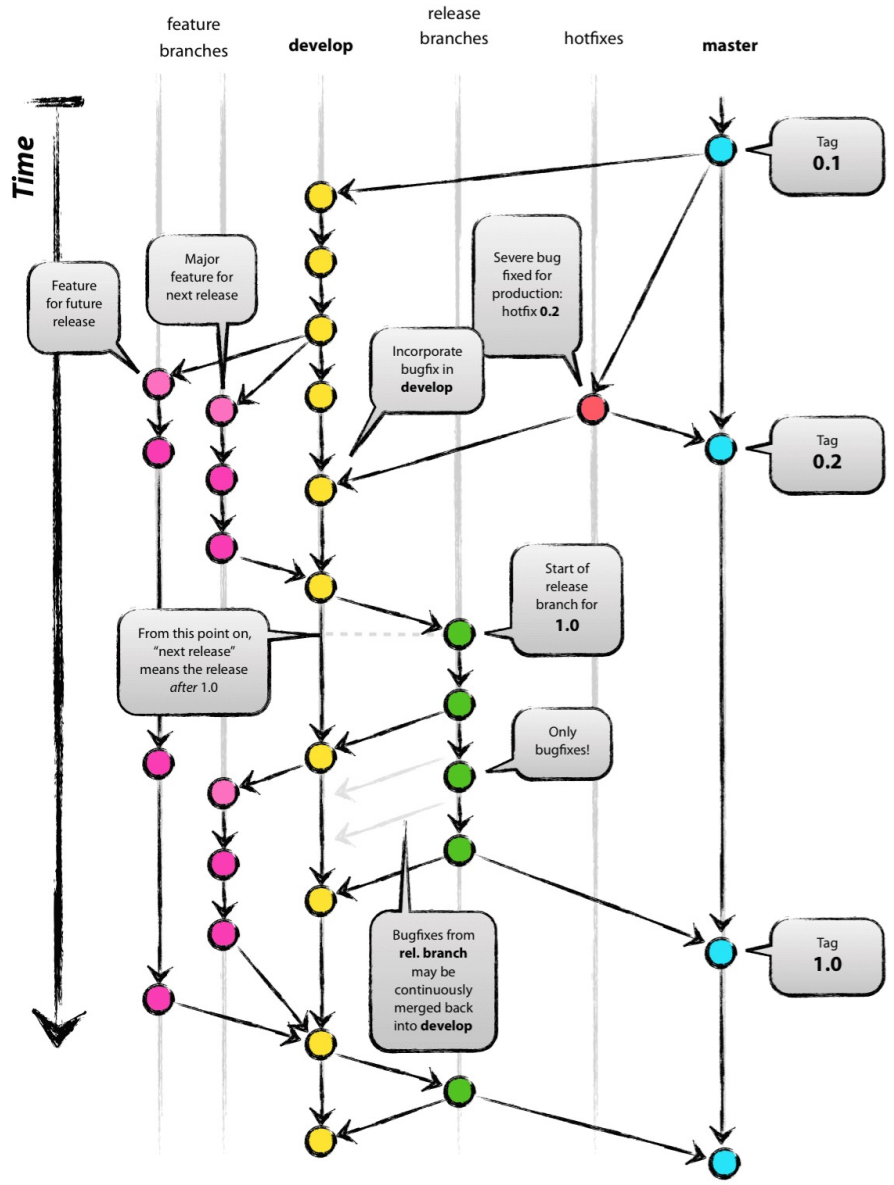
GOAL DRIVEN VERSIONING

You version your products internally using semantic versioning (e.g., v3.1.9), and attach specific major versions to a few milestones you deem essential for each product you reach moving forward. To pursue those milestones, you release incremental versions along the way, better communicating where your product is and where you want it to be!

Here's an example recipe I like to use for my GDV:

Version	Goal	Measure
v0.1.0	Desirable	The product demo can convince new prospects to sign up
v1.0.0	Valuable	The product can retain current customers
v10.0.0	Delightful	Customers are happy and recommend the product

BRANCHING STRATEGY



MAIN BRANCHES

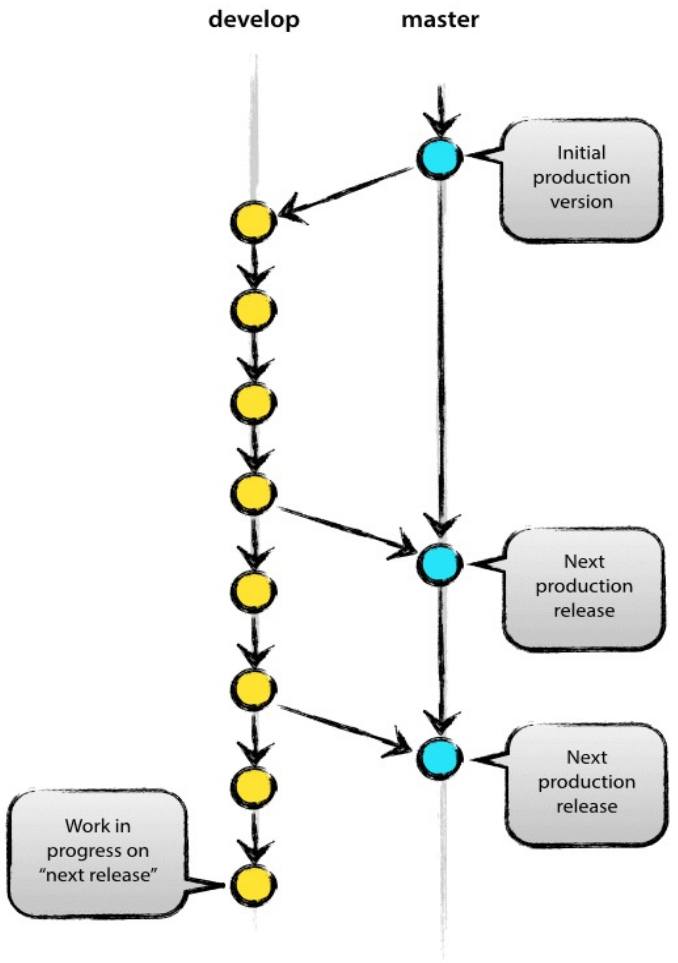
At the core, the development model is greatly inspired by existing models out there. The central repo holds two main branches with an infinite lifetime:

- ***master***
- ***develop***

The master branch at origin should be familiar to every Git user. Parallel to the master branch, another branch exists called develop.

We consider origin/master to be the main branch where the source code of HEAD always reflects a *production-ready* state.

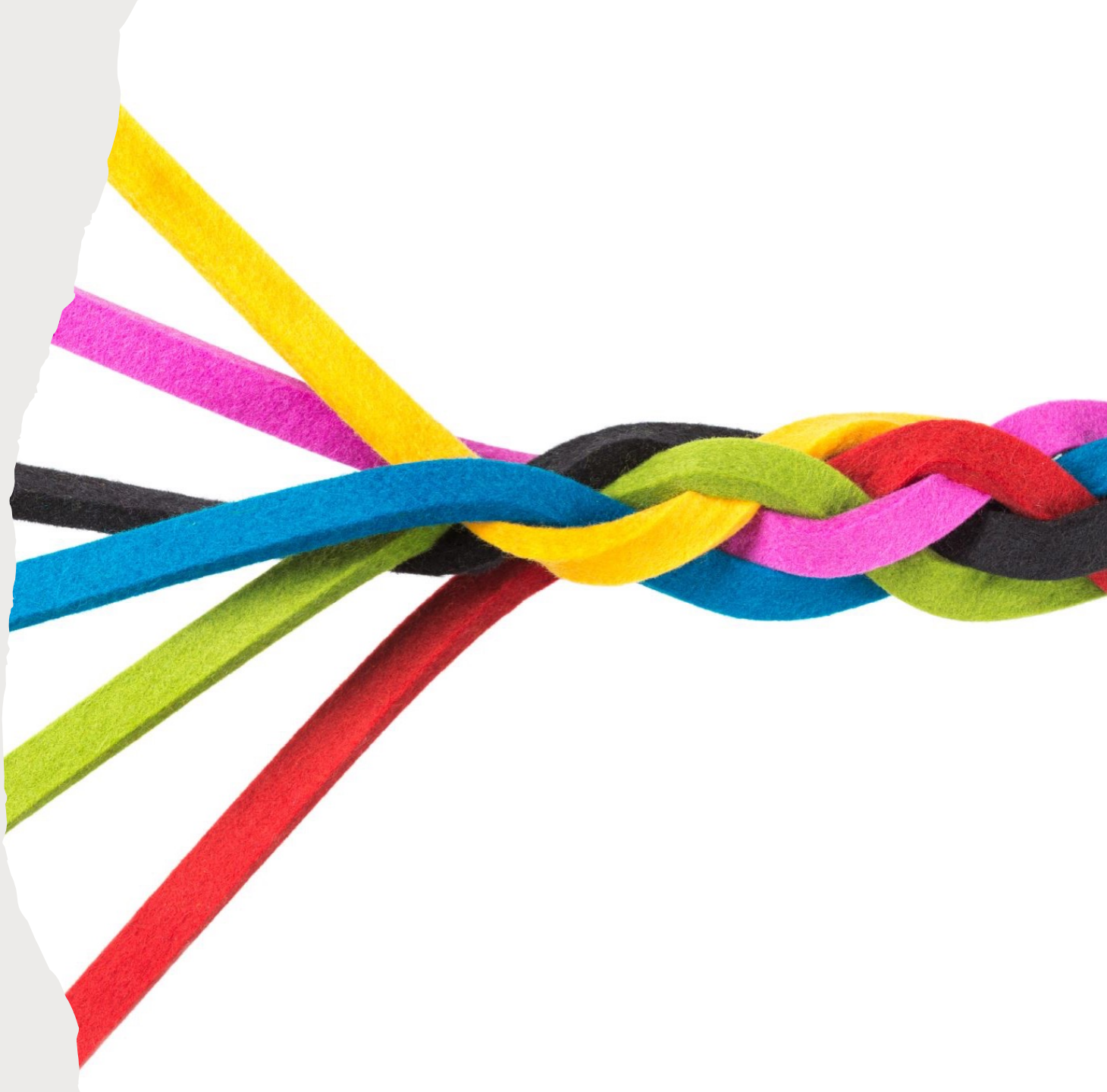
We consider origin/develop to be the main branch where the source code of HEAD always reflects a state with the latest delivered development changes for the next release.



SUPPORTING BRANCHES

The different types of supporting branches

- Feature branches
- Release branches
- Hotfix branches



HOW PRODUCT AND FEATURES TO BE DESIGNED FOR CUSTOMERS HAVING DIFFERENT VERSIONS?

- Key challenges
  - Customer can move between plans
  - Product upgrades to customers having different versions

- Configurability
  - Individual feature level enablement
  - Subscription based
  - Packaged model

	Free	Standard	Premium	Enterprise
	$0 Always free for 10 users Get started	**$7.50** per user (average) $75 a month Start trial	**$14.50** per user (average) $145 a month Start trial	Billed annually. Switch the Billing cycle to Annual to view Enterprise pricing. Contact us
	For small teams to plan and track work more efficiently	For growing teams focused on building more together	For organizations that need to scale how they collaborate and track work	For enterprises with global scale, security, and governance needs
Features				
User limit (per site)	10 users	20,000 users	20,000 users	20,000 users
Site limit	One	One	One	Unlimited
Scrum and Kanban boards	✓	✓	✓	✓
Backlog	✓	✓	✓	✓
Agile reporting	✓	✓	✓	✓
Customizable workflows	✓	✓	✓	✓
Apps and integrations ↗	✓	✓	✓	✓
Automation	Single project	Single project	Global and multi-project	Global and multi-project
Roadmaps	Basic	Basic	Advanced	Advanced
Dependency management	Basic	Basic	Advanced	Advanced
Capacity planning	-	-	✓	✓
Project archiving	-	-	✓	✓
Admin Controls				
Domain verification & account capture	✓	✓	✓	✓
Session duration management (desktop)	✓	✓	✓	✓
Project roles	-	✓	✓	✓
Advanced permissions	-	✓	✓	✓
Admin insights	-	-	✓	✓
Sandbox	-	-	✓	✓
Release tracks	-	-	✓	✓