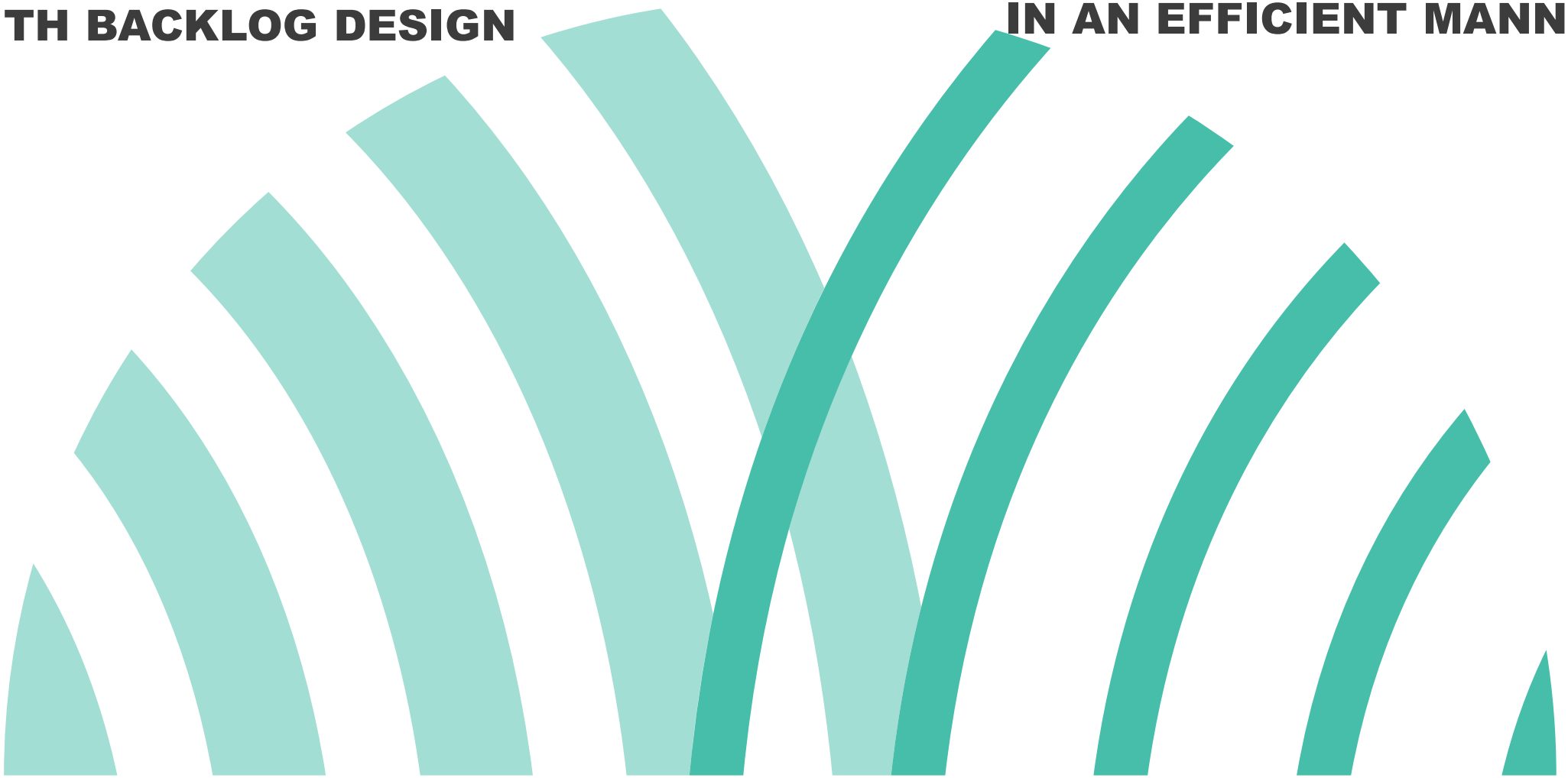
DRIVING FOCUSED AGILITY WITH BACKLOG DESIGN

HOW TO SPLIT YOUR BACKLOG IN AN EFFICIENT MANNER



Digital Payments
for a Trusted World

WORLDLINE

24/02/2022

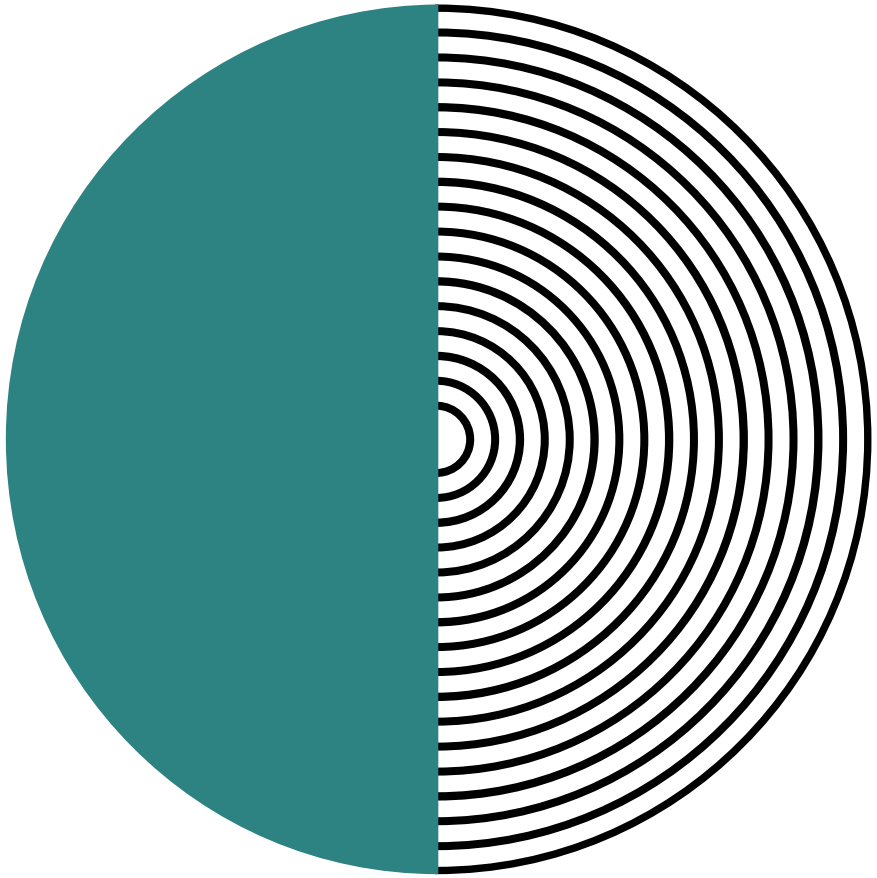
Agenda

PITCH

PEOPLE & BACKLOG

HOW TO SPLIT

EXAMPLES



ABOUT ME

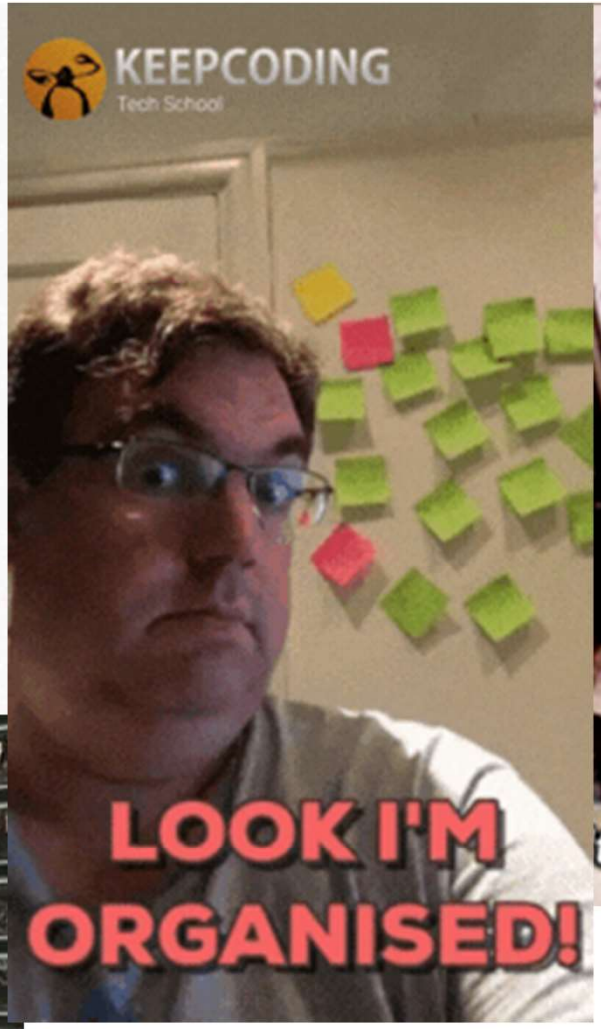
- Hybrid IT/Finance profile, discovering Scrum in 2008 as Scrum Master
- Joined successively 2 big IT Fintech companies with strong envy to deliver with Scrum
- Often facing medium/big projects with numerous stakeholders but also deep new IT paradigms
- Recently implemented DevOps inside our organization (200+ devs / 15 DevOps squads)
- Coaching Product Owners inside 1 Agile DevOps Tribe
- Convinced that people managers can also act and coach to deliver more efficiently

ABOUT WORDLINE

- **Ogone** : small start-up set-up in 90 on emerging market (2 to 4 Scrum squads)
- **Ingenico** : joining a multi-national group and converging to Agile 2.0 model (Spotify inspired : 5 Tribes for 15 dev squads)
- **Worldline** : Integrated into TOP-4 FINTECH actor but
    - 95 % of Dev squads use Scrum with all roles (rest is Kanban)
    - No PjM anymore but strong Product Owners, comfy with IT
    - DevOps model implemented with strong CI/CD tooling

OUR PITCH

You wanted to be an efficient Product Owner ?



WORLDLINE

But then reality strikes again

Product Owner backlog design can sometimes be challenging and require some anticipation with regards of classical pitfalls :

- **Big projects** sometimes – often – do have a **long timeline**, being interrupted/freezed or cancelled because of changing priorities
- Working with Scrum is ok for most IT dev squads (it's in our DNA) but using Scrum with huge backlog is sometimes – often – **quite challenging to get proper focus**
- **Non-IT stakeholders** are often « okayishhh » to work in Scrum but not that much familiar with the use of structured/layered backlogs to define **how incremental** we should move

Why is splitting useful ?

Beside functional negotiation (with business), other dimensions are also impactful for your progress:

- Organizations do have **distributed contingency** between feature teams or other parts of the organization
- Scope often include **non-functional requirements** that can slower your pace
- **Time-to-market** is key in our fast-paced world => **focus** is key to roll-out faster
- We feel more empowered and efficient as a team when we do focus
- There is no better split than the one that is co-designed and drive **aligned transparency**

SET THE STAGE :

PEOPLE & BACKLOG

Bridging 2 worlds and their logics

BUSINESS

- **Market** watch & Sales opportunities or lead(s) on min 6 months, average 3-5 years plan trends
- Product vision & **strategy** on 1-2 years
- Ideation, **workshops** & customer **exchanges**
- **Yearly** roadmaps, **Programs** & Market launch **campaigns**

IT

- 6 or 3 months tangible **roadmaps** (potentially containing TONS of detailed User Stories ;-))
- Scrum : Backlog **grooming**, US **refinement** cycles
- Non-functional requirements & **DevOps** WoW
- Delivery happens through **sprint** cycles

Backlog structure & OKR links

Source : Atlassian.com

OKR's

Strategic goals & Outcomes

Outputs

Activities & people

Backlog artefacts & lifecycle

Initiatives have often yearly or semester targets that should find incremental translation into roadmap into Epic

A convenient EPIC split can help squad to build and follow focused built progress without disturbing sprints progress too much

An Epic roadmap and pipeline helps to anticipate and cadence upcoming sprint grooming/refinement with proper focus and adapt to changing priorities while still delivering

Item	Lifecyle (built/delivery)	Size	Can be frozen ?
Initiatives	1 to several quarters	Medium to huge	YES
Epic (& features)	1 to several sprints	Several to 1 sprint	YES, but not the goal
User Stories	1 single sprint	Should fit sprint capacity & duration	Can happen upon impediments
Sub_tasks	Less than 1 or 2 days	any	Almost never

Why is Epic the correct level of tactical design ?

EPIC is THE TACTICAL DELIVERY ITEM that helps squad to progress with incremental pace and pivot if needed while still deliverying consisted tailored increments.

Initiative/year	Epic/quarter	US/sprint
Business targets & outcomes Validation is gained through end customer feedback and measured with tangible business KPI's or process improvements	**EPIC's are key artefacts to use to :** **- Deliver transparency between various stakeholders** **- Drive alignment potential between common objectives while involving all upon changing context** **- Can represent a correct balance when estimates are requested** **- Pivot upon impediment occurrence while keeping squad progress**	Built by Dev squad but reviewed and validated by business stakeholders
Hundreds/Thousands of MD built through HL « guesstimates »		Some SP refinement is required before starting full Epic Mixed content (Spikes, US, NFR's)
More strategic & business-driven split		Development complexity vary a lot upon progressive findings
		Might be strongly depending on other development done by other squads

So Epic split help to create alignment from start

- **A convenient Epic split is the most suitable approach to :**
  - Anticipate and align on convenient development pace
  - Feature splitting do invite all stakeholders to align incremental split and MVP definition
  - Define incremental focus of all stakeholders (your dev squad, other dev squad, business)
  - Take external contigency (IT or business) into consideration while targeting tangible outputs from start
  - Deal with changing priorities without loosing proper focus, neither fully revamping your project progress

SO, WHAT CAN BE AN EPIC SIZE AND SCOPE ?

- Group of consistent development work around same (sub_)feature
    - Can help your overall plan to reach tangible intermediate dev milestone
    - Can be fully refined with your development squad (focus)
    - Can be covered by E2E consistenty test coverage (quality)

- Consistent effort (not only dev) around a same goal (business goal included ;-))
    - Can trigger proper alignment with all potential stakeholders (dependency)
    - Can deliver first tangible business value in a balanced way (Initiatives VS US)
    - Can allow reactive focus upon impediment

HOW TO SPLIT AT FIRST ?

Main Split dimensions

Dimension	Epic single achievement criteria	Advantages	Risks if used solely	Key input
Architecture of applicative/technical components	Development of full applicative (micro) service(s) flow	• Squad focus on whole component dev • knowledge spread	Not delivering immediate business value	HL arch Design, Technical grooming
Functional increment	E2E uses cases or features	• Can drive negotiation and alignment with business on MVP • Is testable and can be validated with business	Not delivering immediate business value (if only part of your process)	Story mapping with business users
Business value	KPI improvement	Market impact	Not granular enough	Business analysis and KPI's
External dependency (NFR or integration) with external functional component	Integration context or dependency on other component availability	• Defining squad pace & dependency with other stakeholders to reach common targets	Unstable development pace (too depending)	Alignment with external stakeholders

OTHER SUB_DIMENSIONS

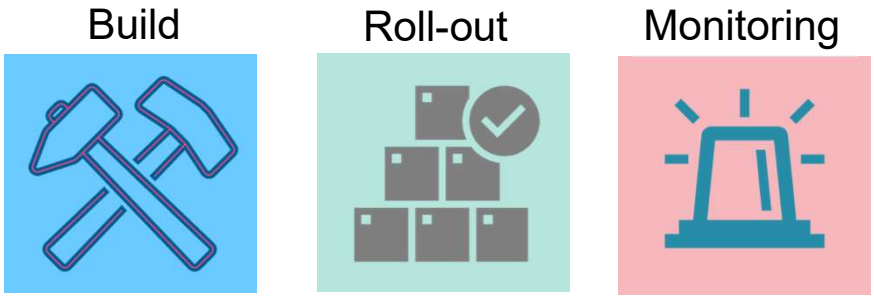
Dimension	Epic single achievement criteria	Advantages & goals
Technical Complexity	- New components to build from scratch - Complex E2E technical changes	Increasing complexity allows your squad to learn on errors and fix it along the way
Dev workload size	N/A	Similar size allows some rythm & predictability
E2E Testability	Epic scope is covered by automated testings	Incremental E2E is the essence of agile
Built-in Monitoring	KPI are measurable and understood/shared with all stakeholders	Facts & figures allow transparency
Underlying Business process steps	Intermediate BP step is improved	Testing should already be possible, even partially

Epic split drivers inception : questions toolkit

- Alignment with Solution Architecture & Tech Leads about High-level Solution Design and linked complexity/size
    - Are you gonna build new application components or adapt/enrich existing ones ?
    - What's linked development complexity and workload ?
    - Is there some new technical WoW you'll bring on your way ?
    - Do we need other teams development work ?
- Roundtable with Business about functional changes & increment
    - Identify your impacted flow, stakeholder audience and MVP first main use cases
    - Are there various features that can be delivered incrementally ?
    - Are there various KPI's that can be used to measure improvement ?
- Is it possible to proceed with incremental roll-out ?
    - Do you have an MVP feature that can be delivered before project closure ?
    - Are there some pilot stakeholders that can be involved for first roll-out ?
    - Is there some opportunity to use toggle features to roll-out new features incrementally ?
- Testability and monitoring inception
    - Can we build E2E test coverage distinctly on some sub_features ?
    - Can we build dedicated monitoring on these sub_features ?

SOME EXAMPLES

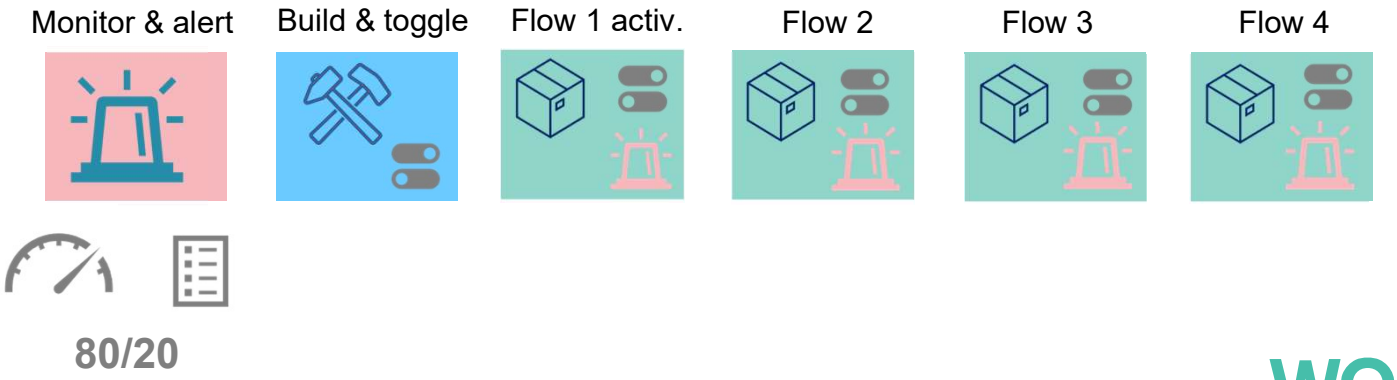
Critical Platform BE process re-engineering



Key split drivers

- New dev from scratch but replacing existing legacy same flow
- New Tech stack (queue engine)
- Flow to decompose & monitor thru KPI's + strangler pattern approach

Integrate these dimensions into a new Epic split

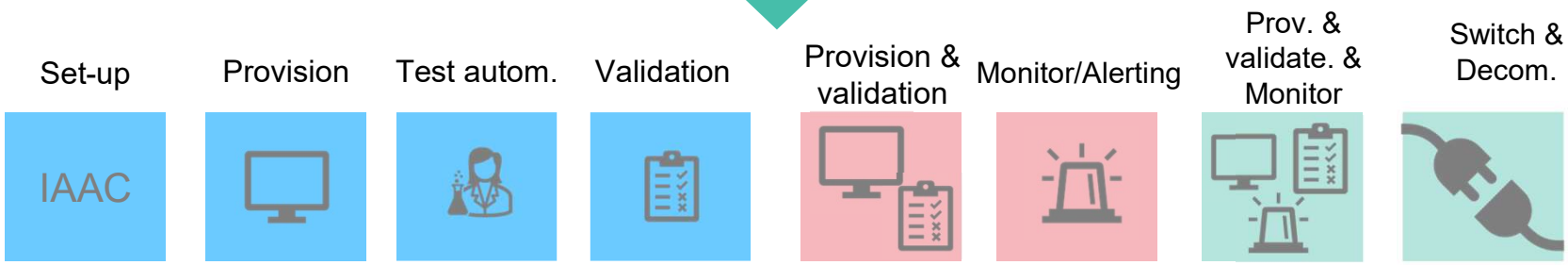


Whole Platform migration (15 feature dev teams + Infra/Ops)

Key split drivers

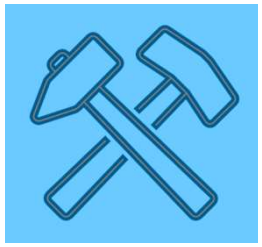
- New technology
- Distributed team (15)
- Huge Infra NFR's
- No MVP : iso-functional migration

Integrate these dimensions into a new Epic split



Web portal (additional FE use cases)

DEV BUILT



PRE-PROD

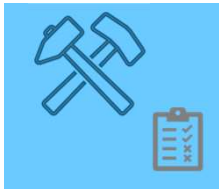
PROD

Key split drivers

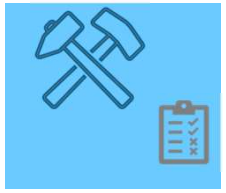
- Existing MVP Portal
- Existing CI/CD flow
- MVP is possible for split + toggle
- Huge customer population

Integrate these dimensions into a new Epic split

Feature 1



Feature 2



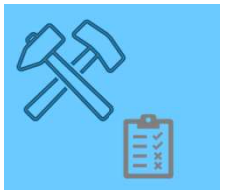
PROD (inactive)

Pilots

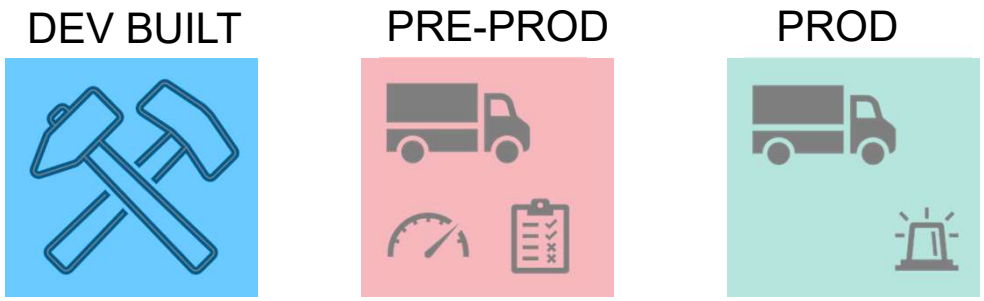
Wave 1

Wave 2

Feature 3



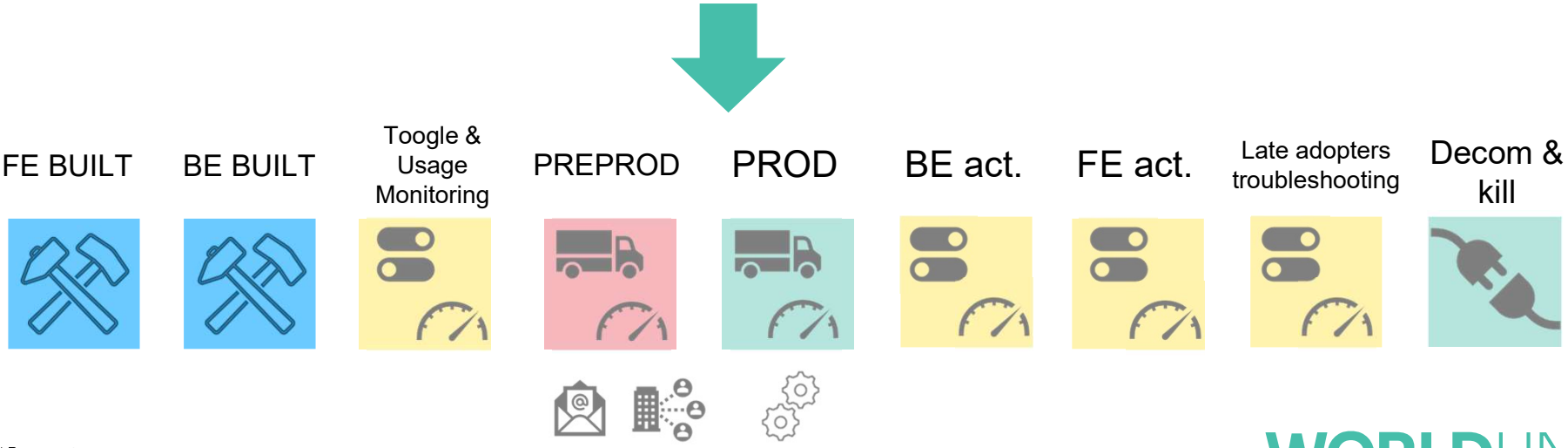
TLS 1.2 compliancy



Key split drivers

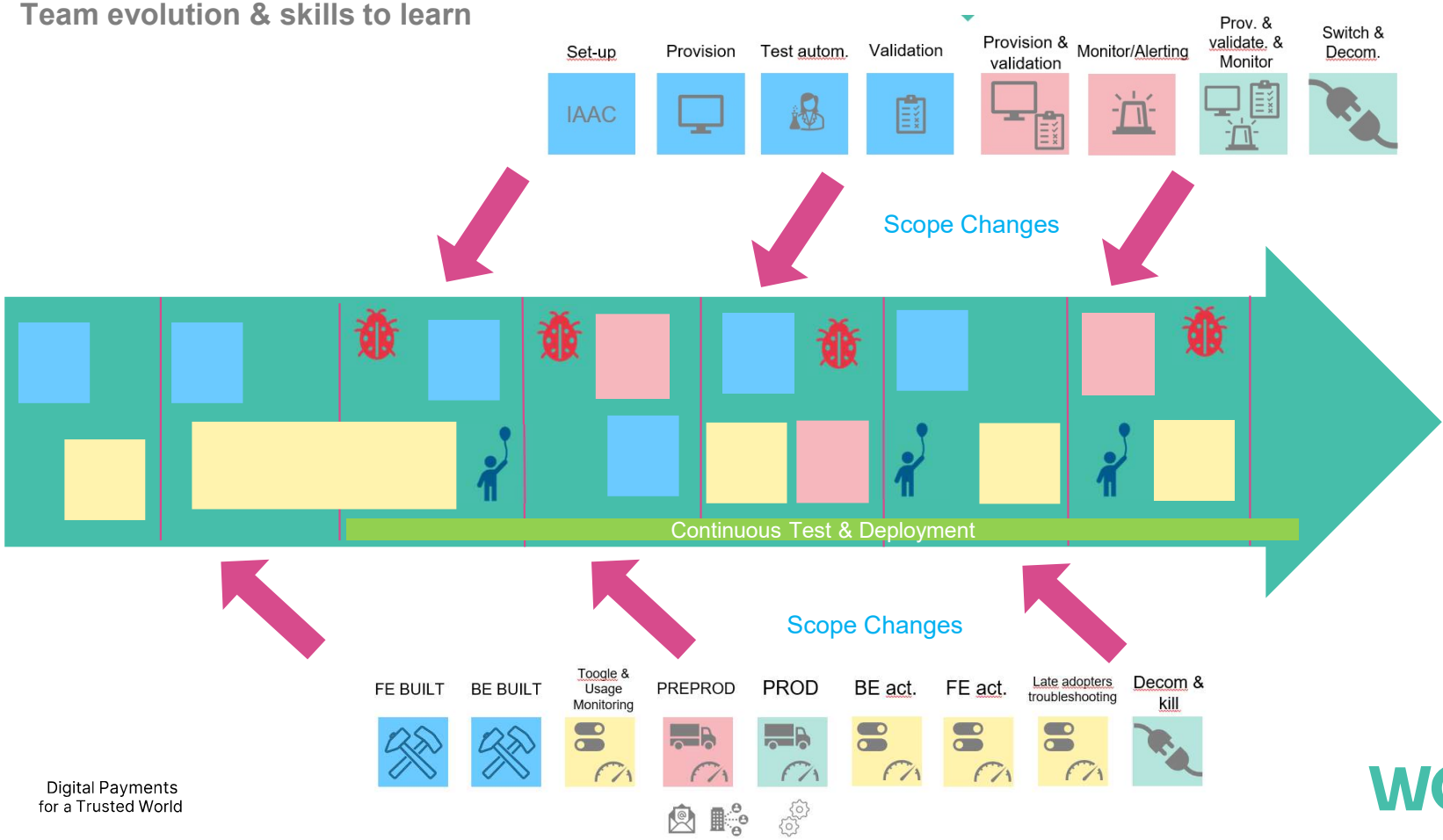
- Thousands of stakeholders (merchants, partners, value-chain actors, third-party tools)
- Early toggle exposure was key to let them test and prepare + monitor
- External dependency but long Compliancy timeline

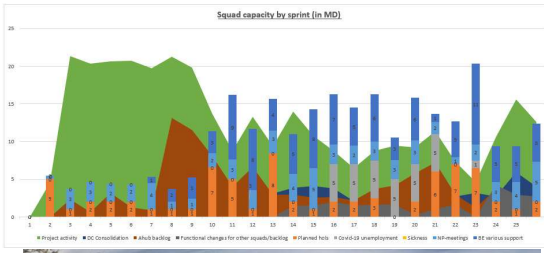
Integrate these dimensions into a new Epic split



And then you add the “real life” context of DevOps squads

- **Competing roadmap requests to prioritize (+ potential scope changes)**
- **DevOps flavor (incidents, CI/CD trains management)**
- **Team evolution & skills to learn**





Digital Payments
for a Trusted World

WORLDLINE

Epic split is not a fixed plan, neither a waterfall approach but

- It’s a « focus-enabler » to allow ALL stakeholders to align on shorter short-term challenges & targets
- Can still be changed / reshuffled easily upon priority changes, without disturbing too much development squad
- Let your squad focus their grooming effort in a convenient pace, while anticipating next steps as well
- Still allows adaptative process & refinement inside single Epic
- Support high-level workload estimates (T-shirt sizing)

PO has a chapter with 20+ existing skilled PO’s + Worldline is hiring several PO ☺

careers.worldline.com

Q & A ?

THANK YOU ☺

Digital Payments
for a Trusted World

WORLDLINE