February 2024

# Become 20% more efficient with one tool

Enate Architecture and Roadmap

enate

# Agenda

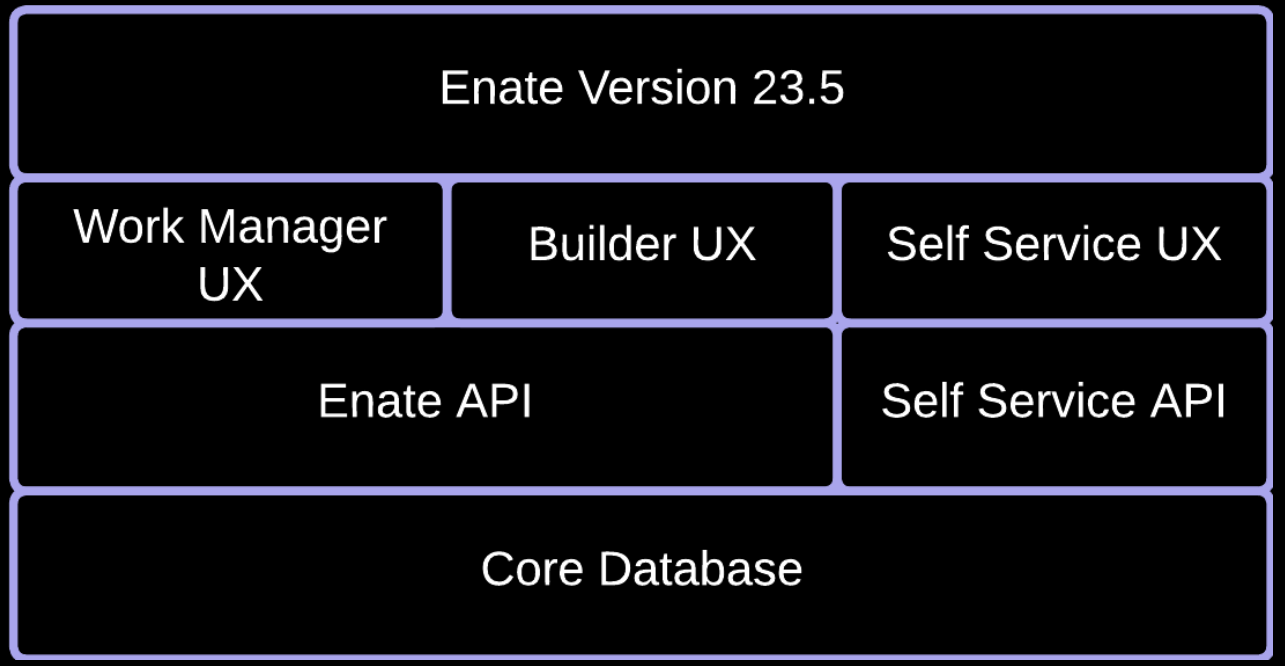
- Where are we now?
- What does this mean for our Roadmap?
- Where are we going architecturally?
- How to we take the first step?
- How do we move forward with confidence after that?

enate

# Where are we now?

enate

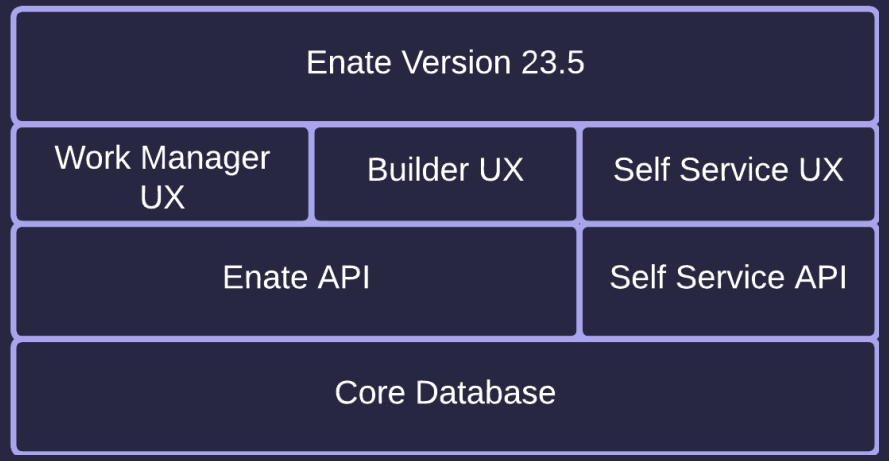
# Where Are We Now?




## Enate is a Monolith

Meaning the whole product is released together, UI, API, Database schema. The only true exception to this being the Enate Marketplace.

# Where Are We Now?




## This has drawbacks

We cannot be agile, the fastest we can respond to anything is every 2 months.

We cannot get real user feedback during the development process.

We can only ‘patch’ the entire monolith making the break/fix cycle slow.

Makes it hard to hit precise release dates because we cannot pull part-completed features.

Every new version includes a lot of simultaneous change that users have no control of.

# In 2023 we decided:
# Enate will become a SaaS Only platform
# On Microsoft Azure

## We have ambitious plans

enate

### Operationalise Generative AI

Deliver extraordinary new productivity capabilities making GenAI useful for service delivery

### New User Experiences

Deliver new user experiences that take advantage of the cloud native technologies now available to us e.g. Intelligent Search and Reporting

### Be Agile

Work with real end users throughout the engineering process to test and validate new experiences before mass release.

We now have exciting opportunities for our roadmap!

# What this means for our Roadmap

enate

# The Enate Roadmap

Our Roadmap has two fundamental areas that reflect how we will deploy our resources to delight our customers. We will continue to be a platform for business users and business technologists.

## New User Experiences

Deliver great new user experiences taking advantage of Enate as a SaaS only platform.

## Operationalise AI

Operationalise AI for service delivery, massively increasing productivity and quality of service delivery.

## Extending Enate through Partnerships

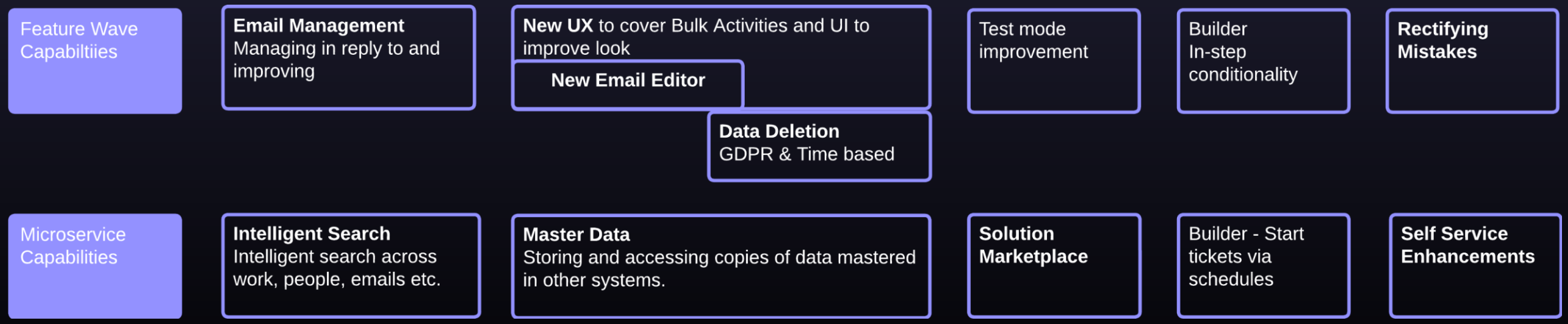
Partner with other leading technology vendors to resell their products alongside Enate as part of a wider solution. Integration Platform as a Service partner is our first target.

The next slides go into each of these areas in more detail. These are shown in priority order, but are not aligned to feature waves yet.

enate

# New User Experiences

The diagram below sets out the key Themes that we will cover in New User Experiences, it broadly breaks these down into which will be delivered primarily through new feature waves and which through new Microservices.




We will be keen to work with customers on feature requests that align with these themes, but not those that don't.

N.B. 'New Email Editor' involves us validating new 3rd party components to use for this capability. This may not involve us sticking with the current one.

enate

# Operationalise AI

The advances in AI in the last 12 months have been spectacular. Our vision is to make it extraordinarily simple to operationalize generative AI for service delivery.

**Operations Copilot Phase 1**
Reason on transaciton data against policy

**Email Copilot**
EnateAI Email Writing and Tone Change Copilot and Pattern

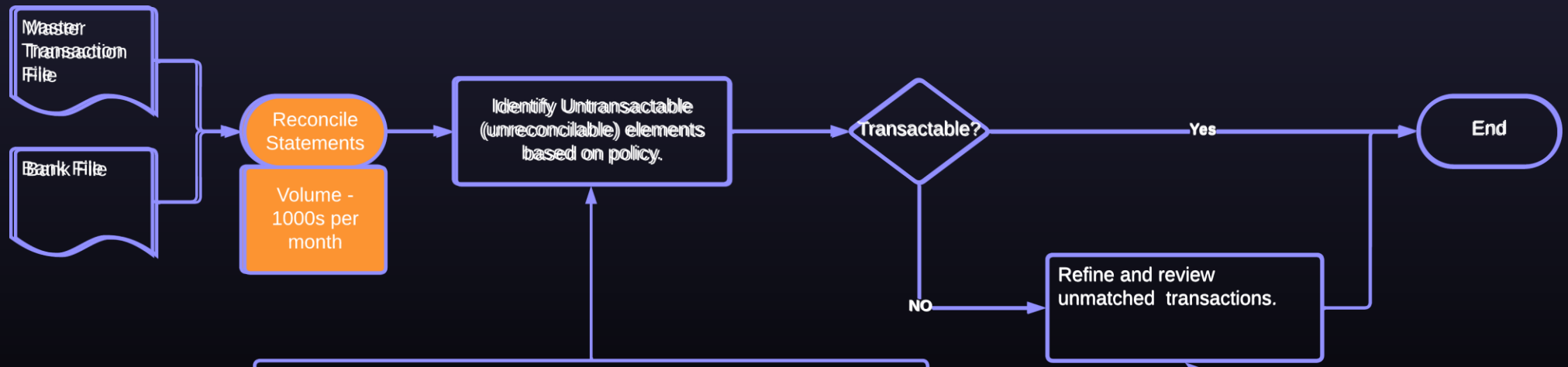
**Operations Copilot Phase 2**
Reason on transaction data and take action.

**Operations Copilot Phase 3**
Reason on transaction and and master data and take action

enate

# Operations Copilot Phase 1

This is based on needs identified in conversations with many other customers and partners.
This is an example of payment reconciliation.

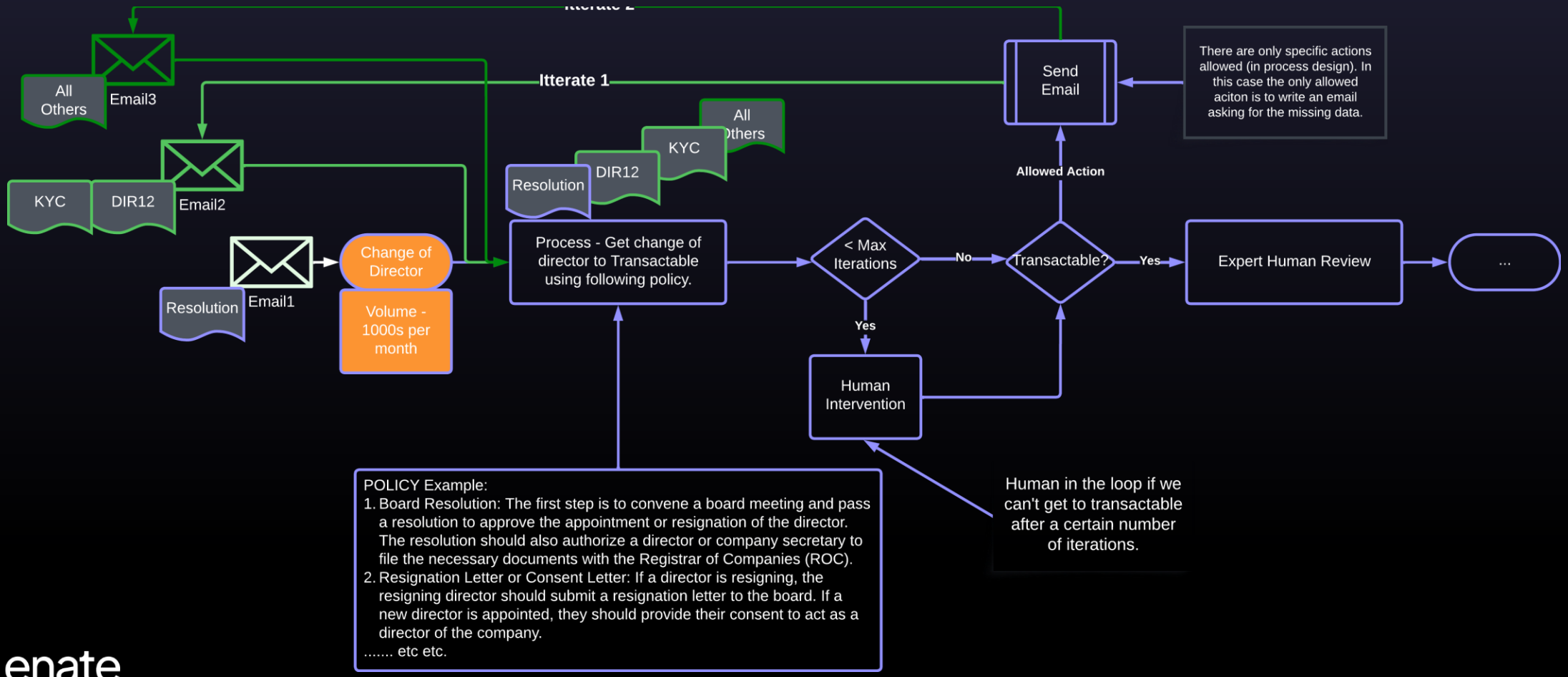



This is currently being done manually every day reconciling multiple bank accounts against extracts from the master transaction system and then processing those reconciliation differences.

enate

# Operations Copilot Phase 2

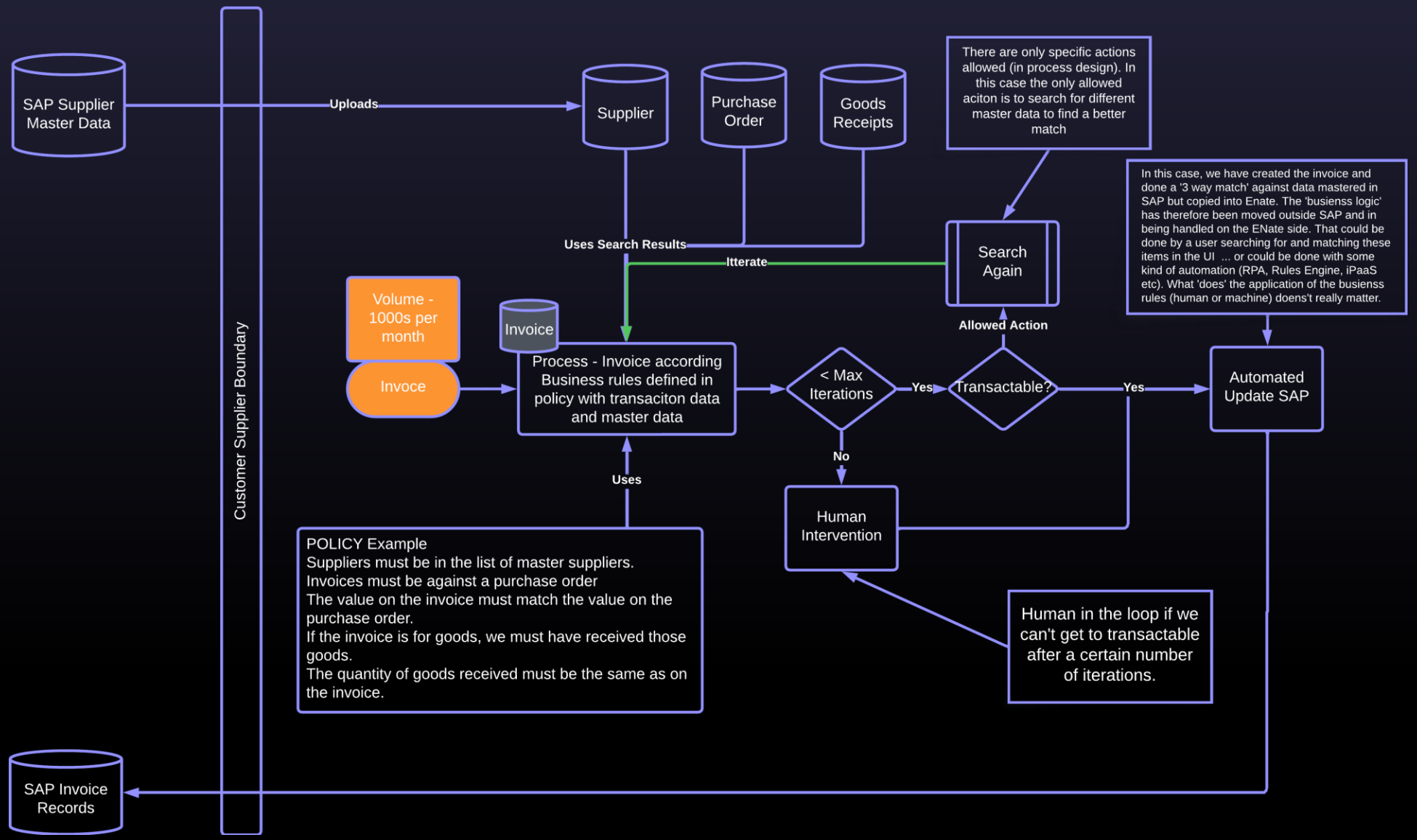
The same ‘Pattern’ is used in this example for changing directors. Except this time there is a new capability to ‘take action’ in this case, send an email through Enate.




enate

# Operations Copilot Phase 3

Then this example from a major customer, again the pattern is the same but with the addition of the Master Data feature that we will add.




enate

To deliver, we have to make some changes to our Architecture

# Where are we going?

enate

# The significant changes

## Versioned APIs

The Enate APIs will become versioned APIs meaning that we will be presenting a stable interface for integration with the core Enate platform. This is how the Enate Marketplace works (for example)

## Microservices

Many of the new capabilities in the product will be created using a 'Microservices' architecture meaning that they can be developed, tested, tweaked, improved and released independently.

## Alphas and Betas

New user experiences will (wherever possible) go through Alpha and Beta phases with real users being able to try them out and give the product team feedback throughout the process.

## More SaaSy

In 'pure' SaaS platforms there is only ever one version of the code in production at any time. We won't be going to that full extent, but there will be some aspects of Enate that will be delivered this way. Everything delivered this way will be patchable without 'taking a new release'

## Feature Waves

The concept of an 'Enate Version' won't truly exist anymore. Instead, changes to the public APIs will be released in Feature Waves, with each feature wave potentially bringing new versions of APIs while continuing to support old versions.

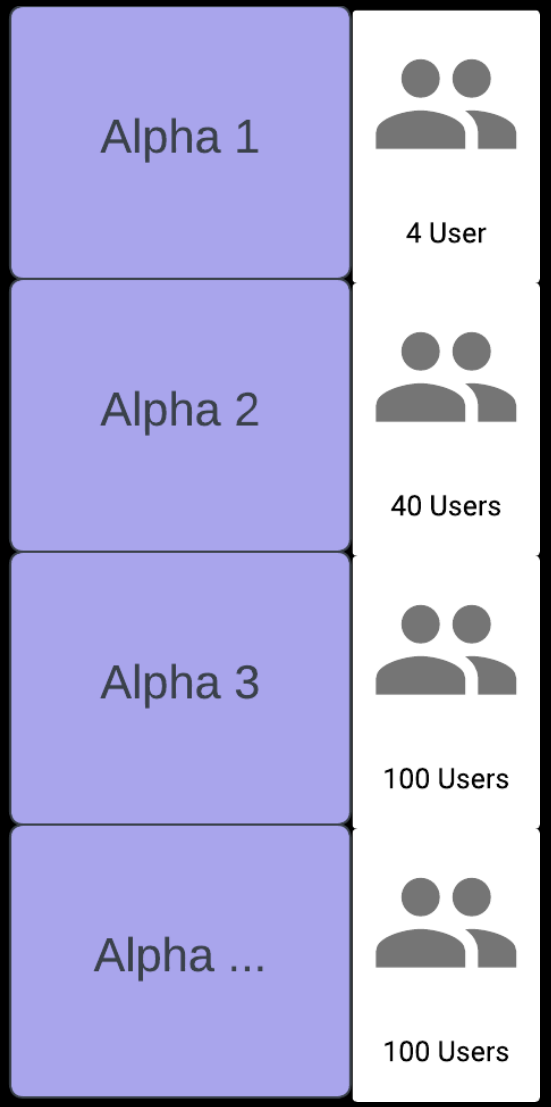
# Alphas

**What is an Alpha?**
Alpha means that we (Enate) do not think the feature is yet functionally complete, but it has been delivered to a point where we want to see and understand end user feedback. We will closely monitor Alpha users use of a feature and seek their feedback. There will be multiple Alphas available to a progressively wider audience as the capability of the feature increases.

**Do all features go through Alpha?**
No, only features where we want / need user feedback on the user experience or logic and where the feature 'works' if only a handful of users have it. There will be some features where we cannot release alpha versions, e.g. a new action type like Approvals.

**Who gets Alphas?**
Alphas features will be released to handpicked users behind a 'Feature Switch' that the user activates, meaning that they can switch the feature on or off.

**Enate will control which users the feature is released to.**
We will talk to you about who would be good candidate users to work with on an alpha feature. But we will decide who to make the 'Feature Switch' available to because numbers will be small.

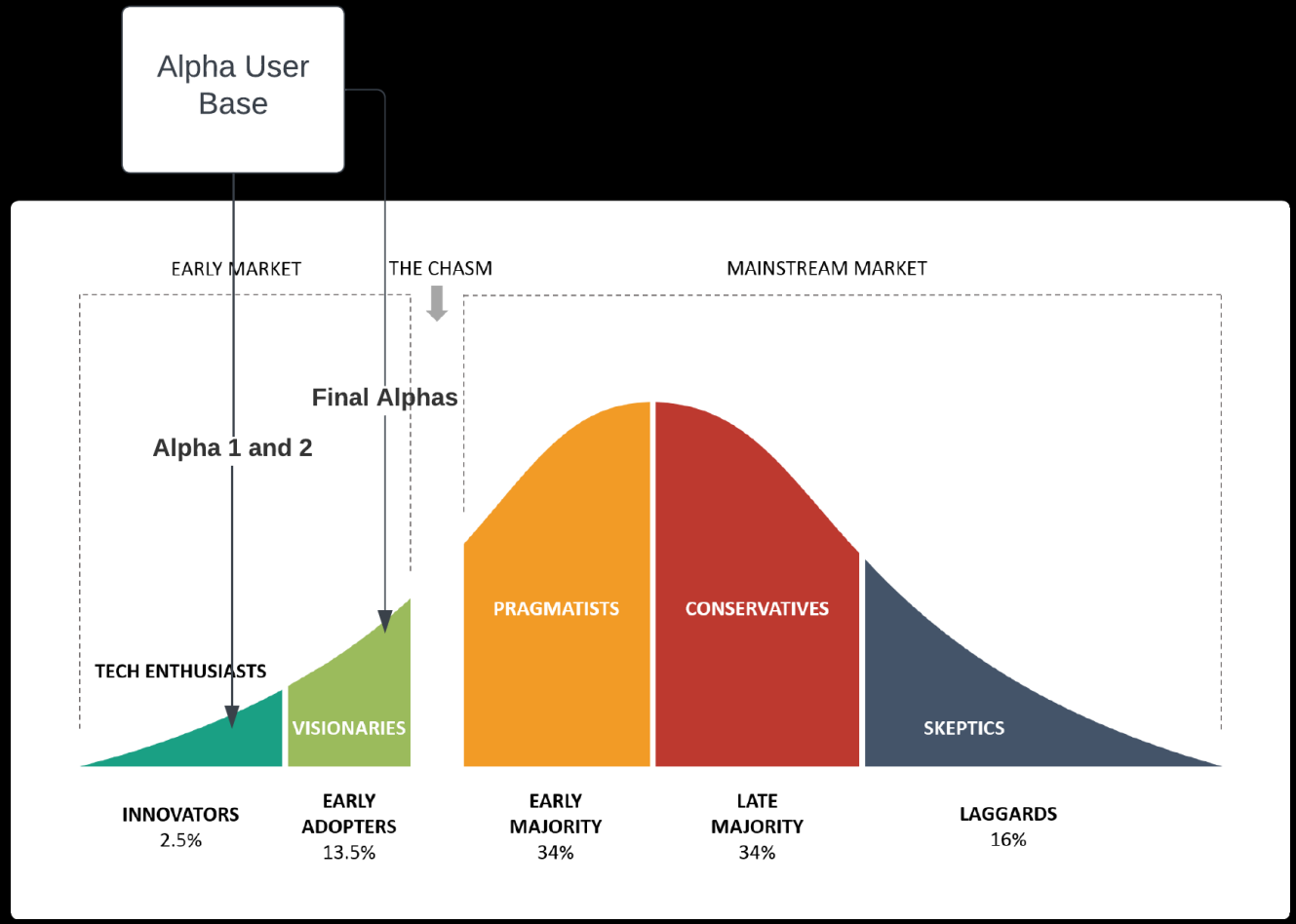


# Alpha Users

The Alpha users should be innovators and early adopters and be comfortable giving feedback on something that by definition isn't finished.

There won't be documentation available for alpha features, but for early Alpha's the product team will look to engage personally with the users.

We may also release UserPilot flows for later Alpha releases.



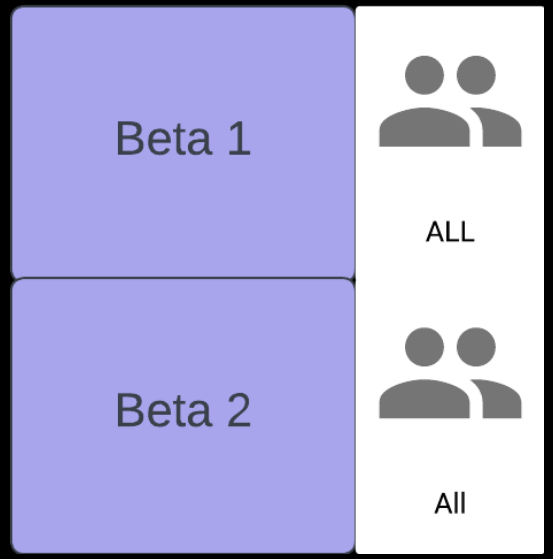
# Betas

**What is a Beta?**
Beta means that we (Enate) believe that the feature is complete, and it has had successful feedback through the Alpha phase to shape it to real users' needs and desires. We will continue in Alpha until we are happy with the feedback.

**Do all features go through Alpha / Beta?**
No, only features where we want / need user feedback on the user experience or logic. It is possible to release some new features as Beta that can't be released as Alpha e.g. a new action type could be released behind an administrator switchable 'Feature Switch'.

**Who gets Betas?**
Beta features will be released to all users behind either a user switchable 'Feature Switch' or an administrator switchable 'Feature Switch'. There may be more than one beta release, which will all be minor tweaks.

**When will features move on from Beta?**
Features will move to 'released' as part of a Feature Wave. Released features can no longer be switched off via a feature flag.



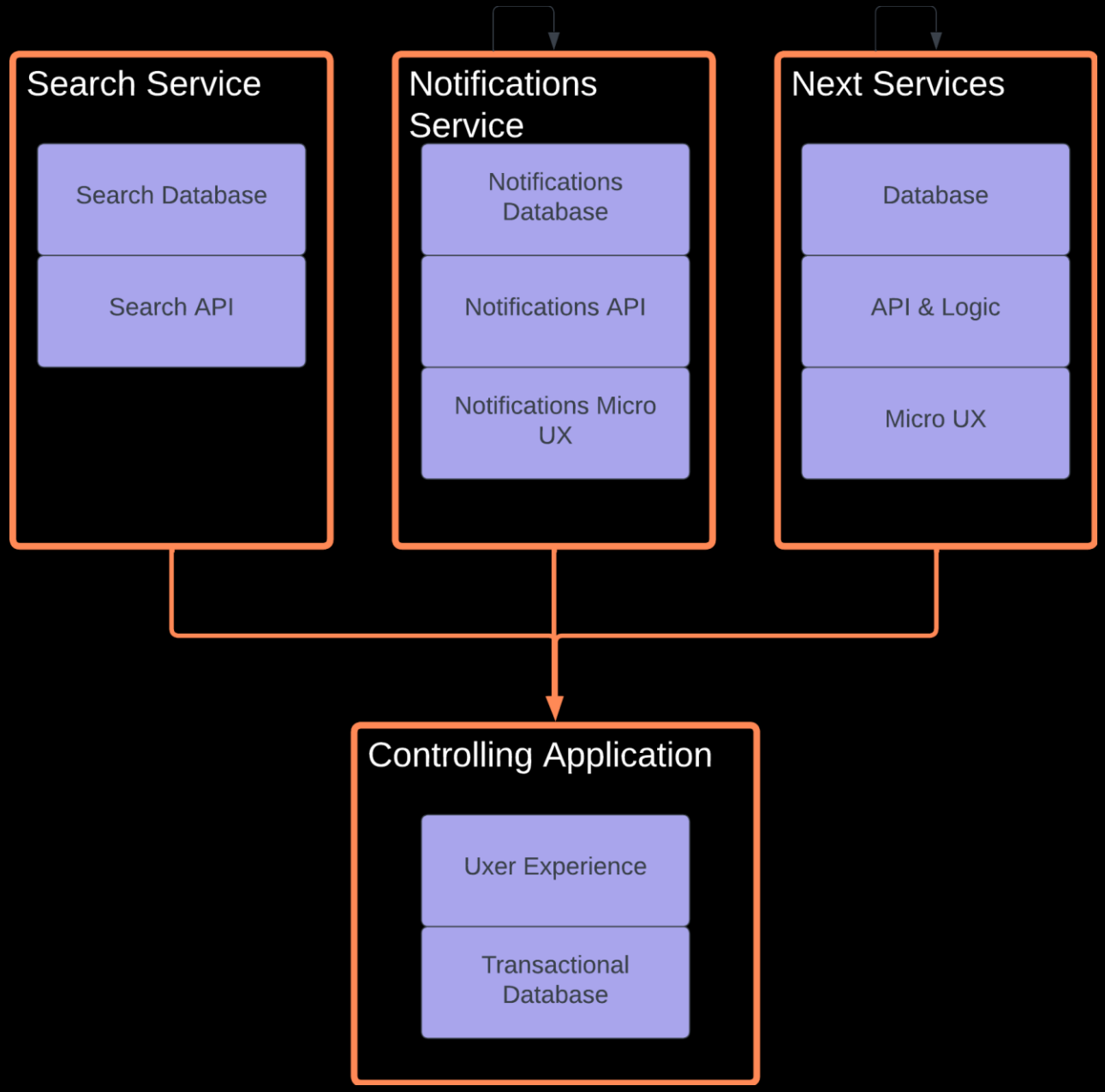
# Microservices

**What is a Microservice Architecture**

A microservice architecture is where you create joined-up user experiences by bringing together lots of smaller 'services' that can be thought of as little applications. Each will have its own business logic and API and may have its own datastore and micro UX.

It is the approach used to deliver most large scale SaaS platforms like Hubspot, Notion, Service Now and many others.

Each 'Service' can be developed, tested and released independently of the other services.



# Moving to Feature Waves

enate

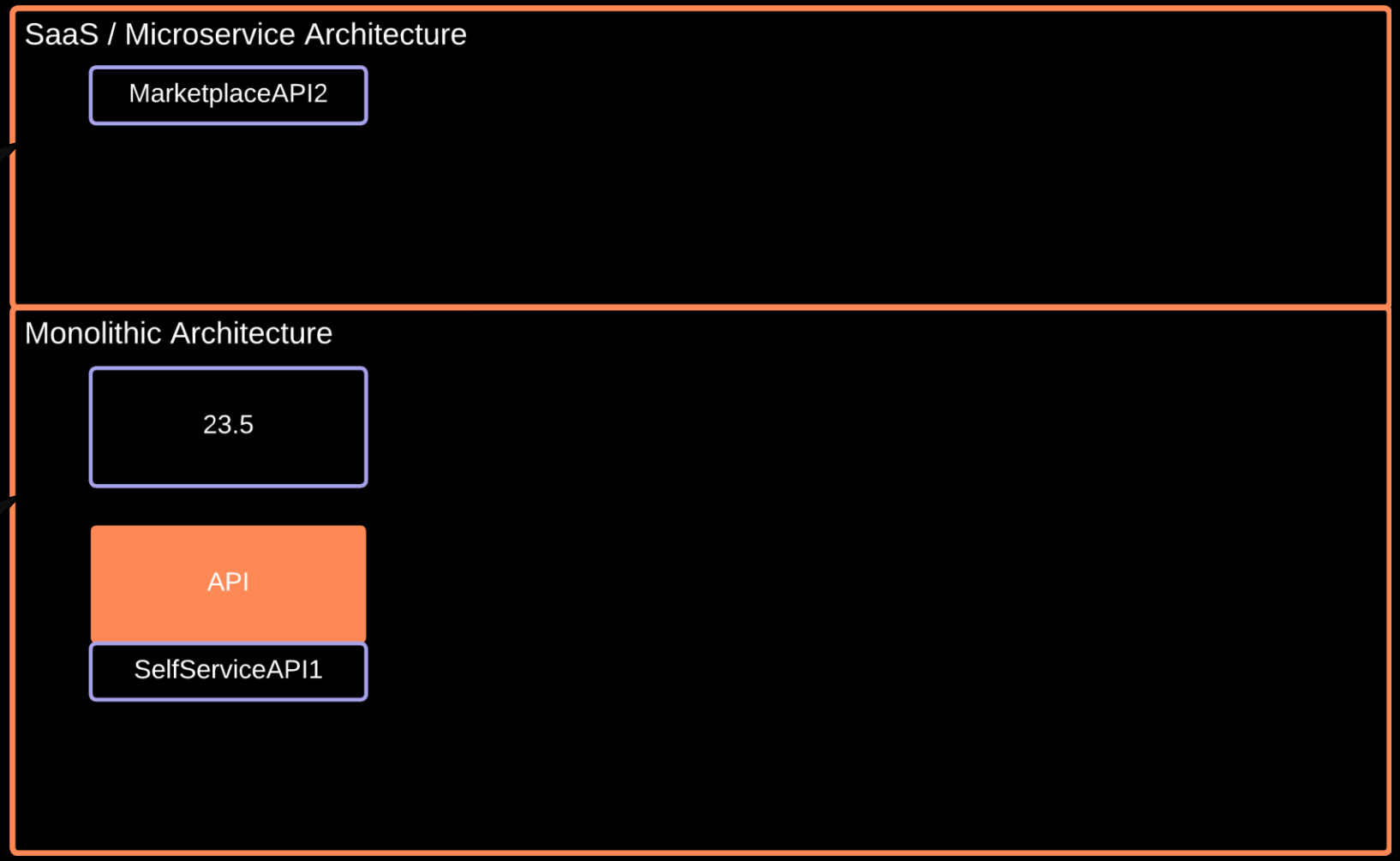
# We are here …

## Microservices / SaaS Architecture.

One version of code in production at any time

## Monolith Architecture

Released in a similar way to current Enate product releases

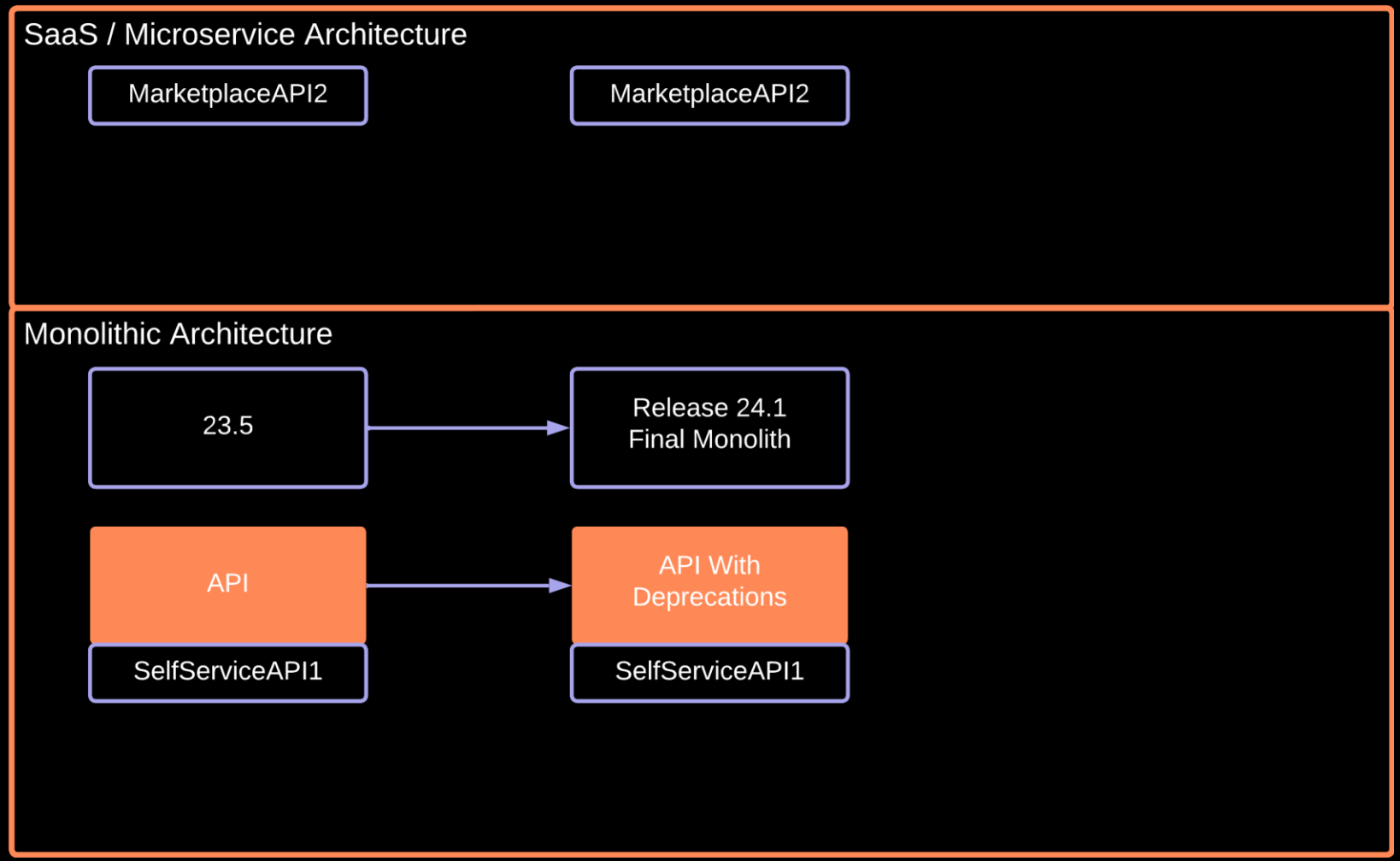


# Moving to Feature Waves…

Enate 24.1 will be the LAST monolithic release of Enate. ALL customers must take 24.1

24.1 will for example deliver new Email Management functionality for end users.

It will also make private many of the Builder APIs that are not designed for integration.

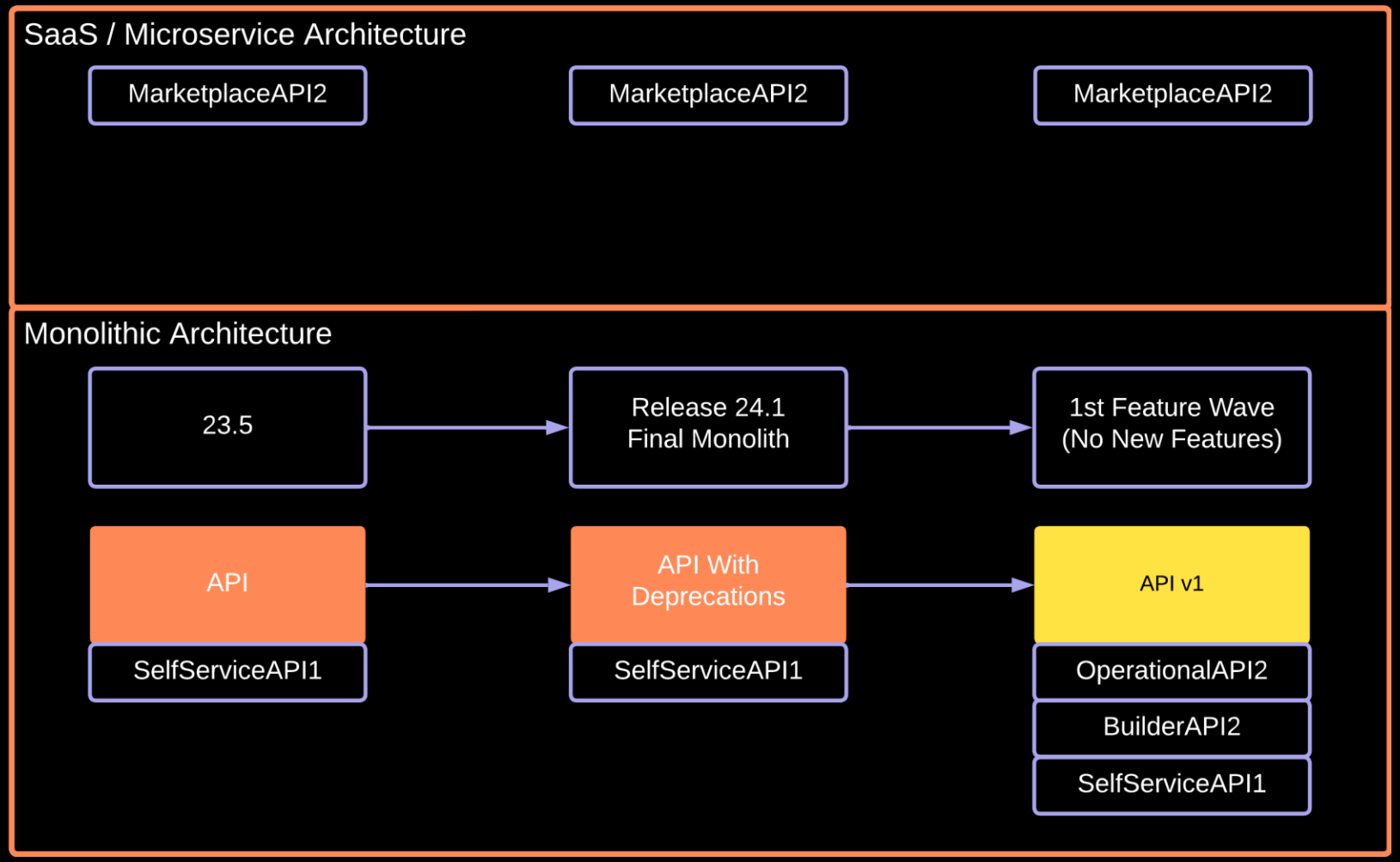


# Moving to Feature Waves…

We will deliver the first Feature Wave of the new architecture shortly after the release of 24.1.

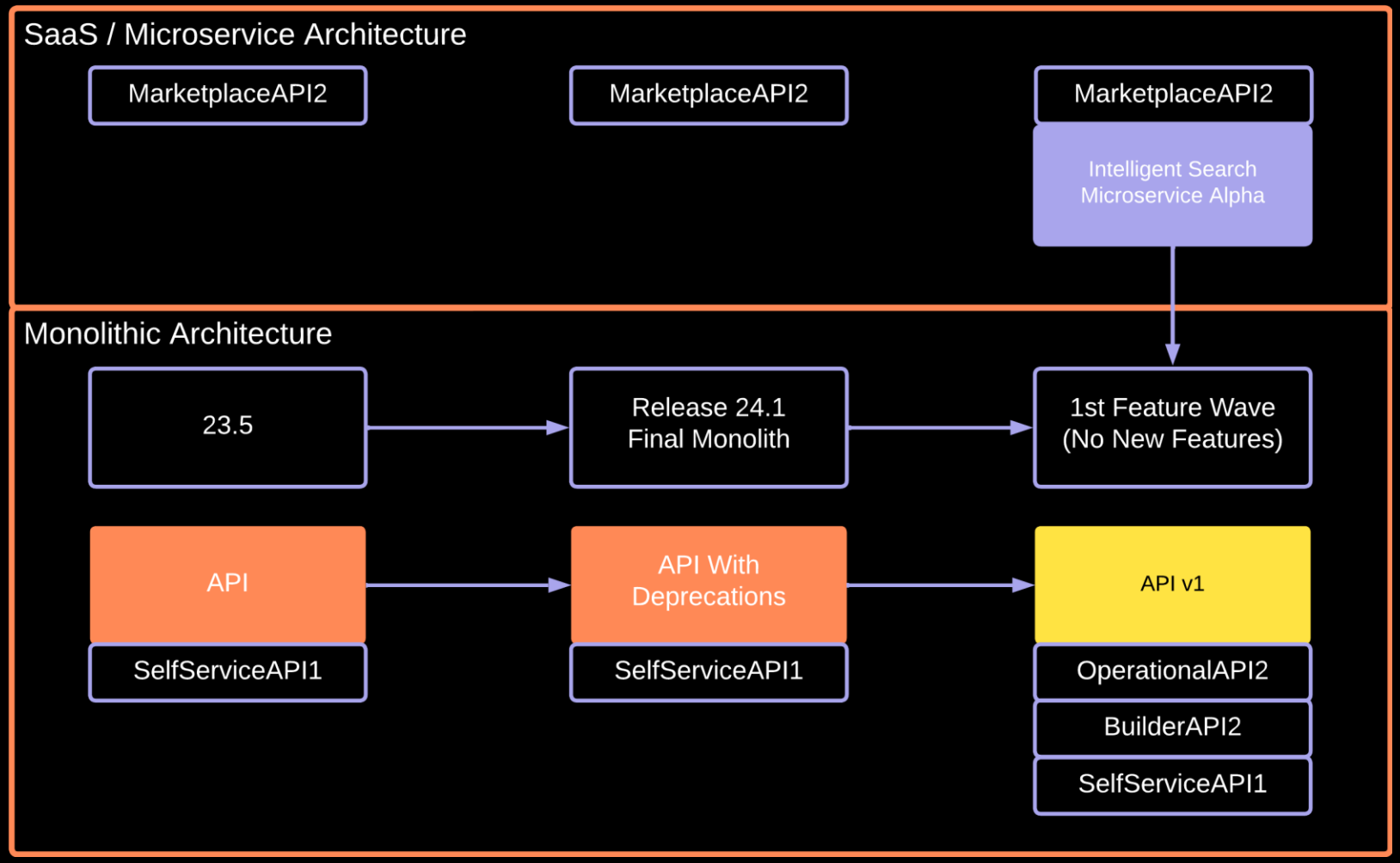
This will have NO new end user features so will be seamless for users.

This feature wave will introduce separate versioned APIs for Work Manager and Builder.



# Moving to Feature Waves…

Once you are live on the first Feature Wave, we will be able to enable Intelligent Search for Alpha 1 users.

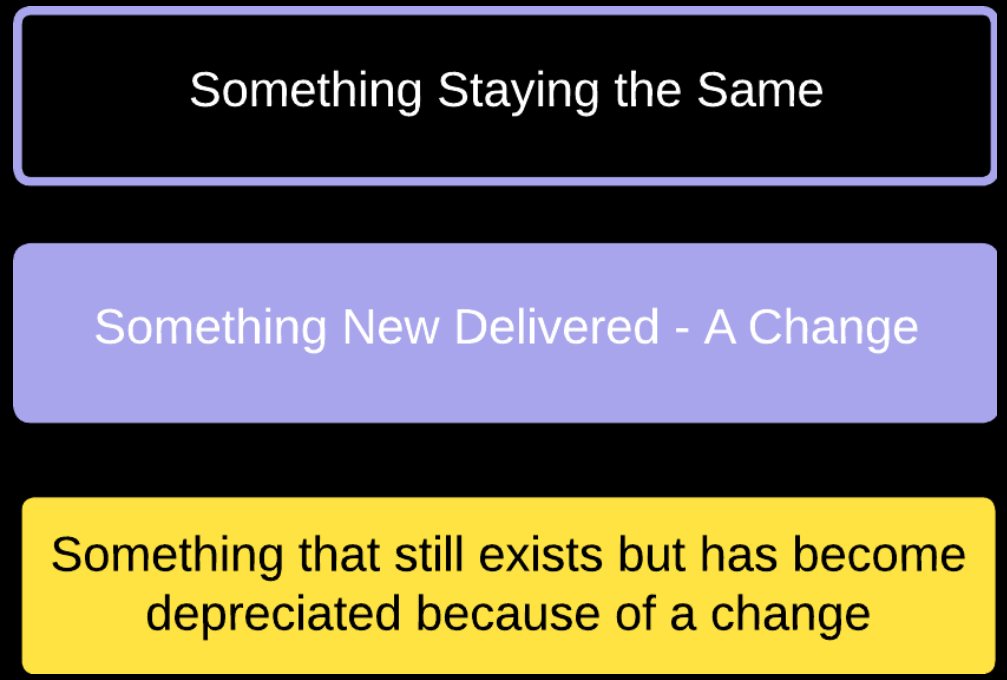


# Examples of Progression Without Feature Waves

enate

# A Key for these diagrams...

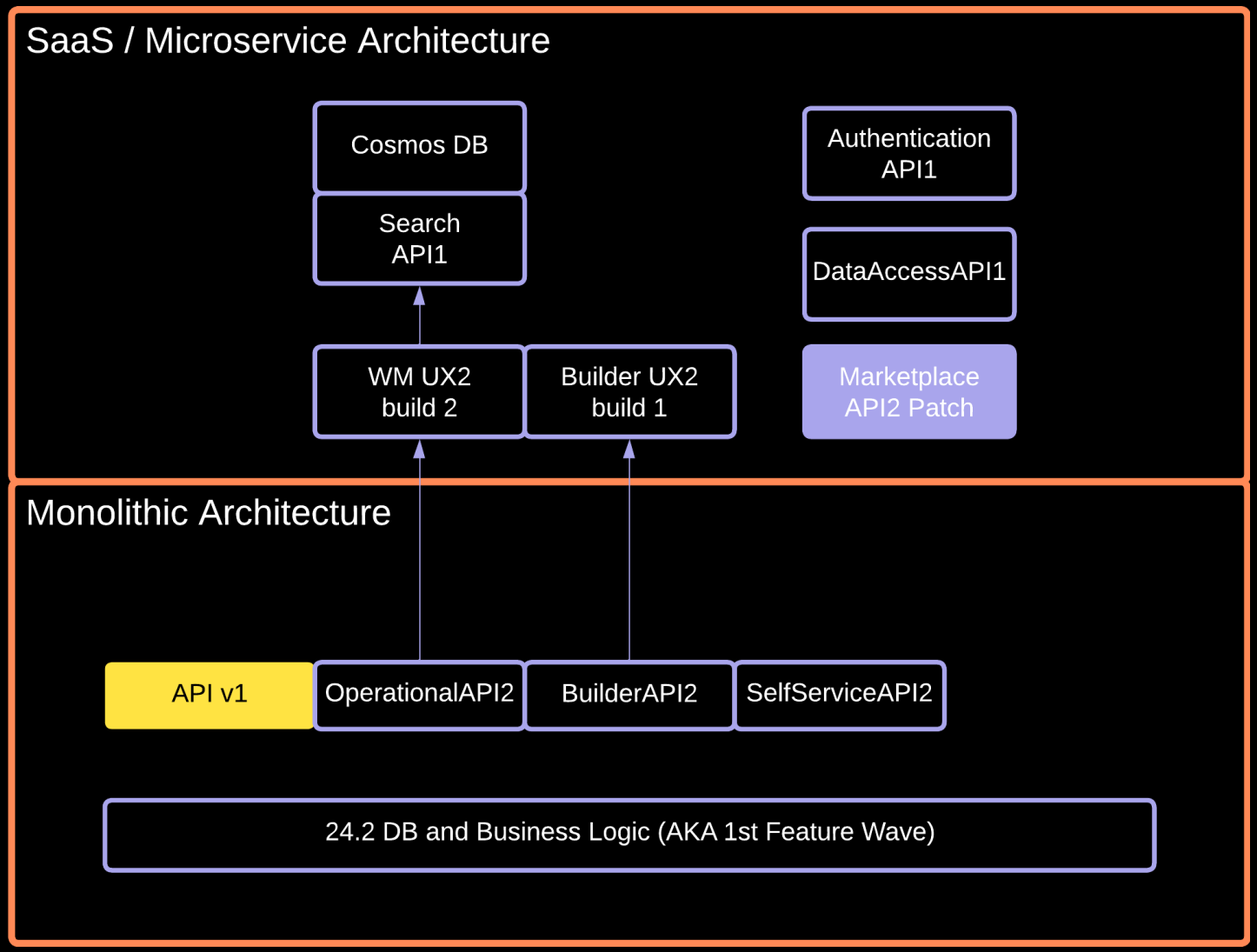
This is the key for the diagrams on the following slides



# Patching a Digital Worker

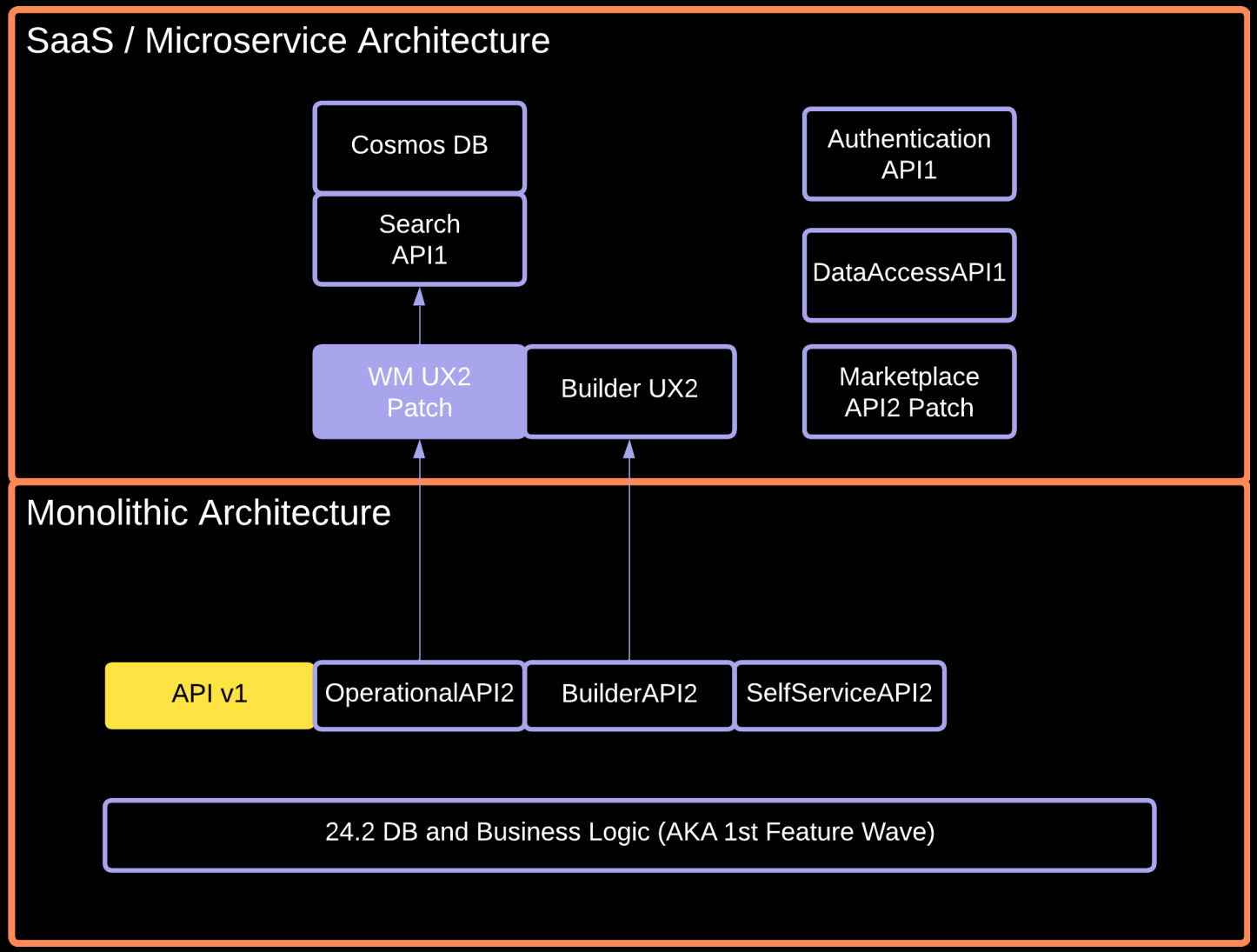
Example: A new build of the Marketplace is released fixing some bugs and enhancing some capabilities.

This is automatically deployed to production through the Enate pipelines having been through our full test cycle.



# Patching a UX Only Bug

Example: A Work Manager UX only bug can be patched without releasing automatically to all customers through the Enate pipeline.



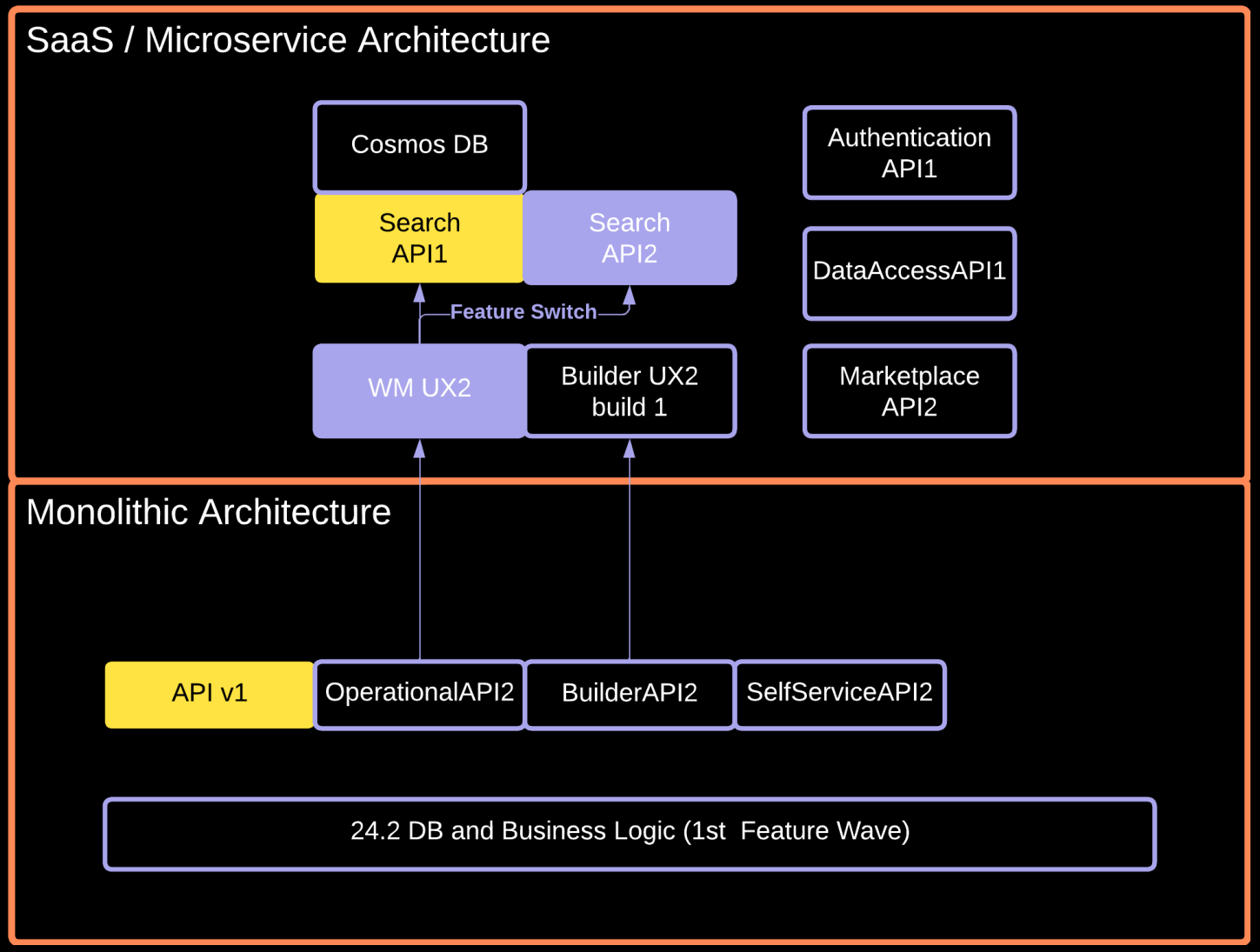
# Updating Search

A new version of the Search Service is released with a patch update build of the UX to accompany it.

The UX includes a feature switch to switch between the updated search and the old search.

These updates are delivered to all customers simultaneously.

Search API1 is depreciated.



# Examples of Moving to a New Feature Wave

enate

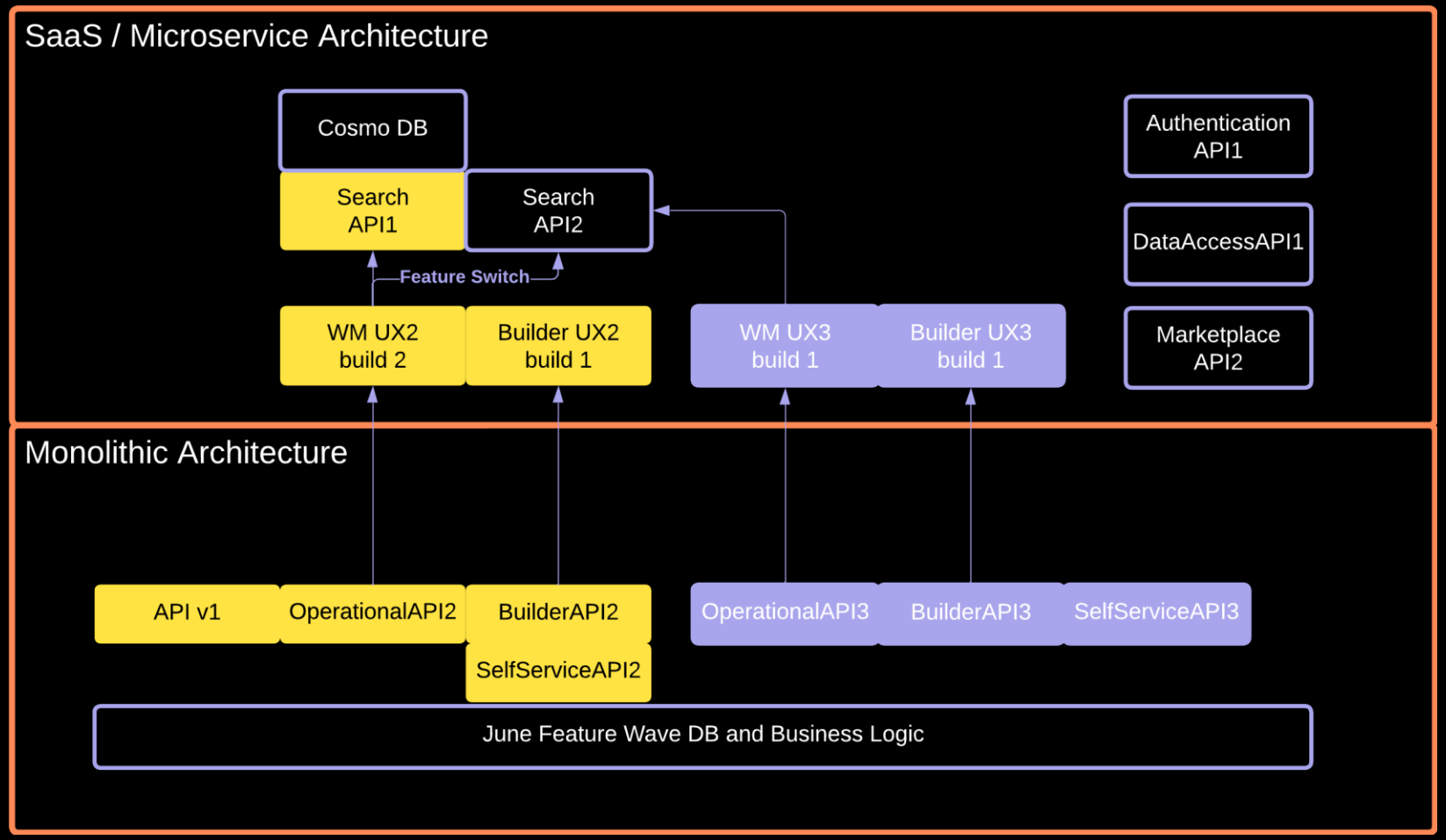
# Releasing the next feature wave

Example: The June feature wave is released.

All customers are updated to this feature wave within 4 weeks of its release. (More on this later)

But … the June feature wave still supports UX2 and Builder UX2 from the previous wave … so end users don't see any change!

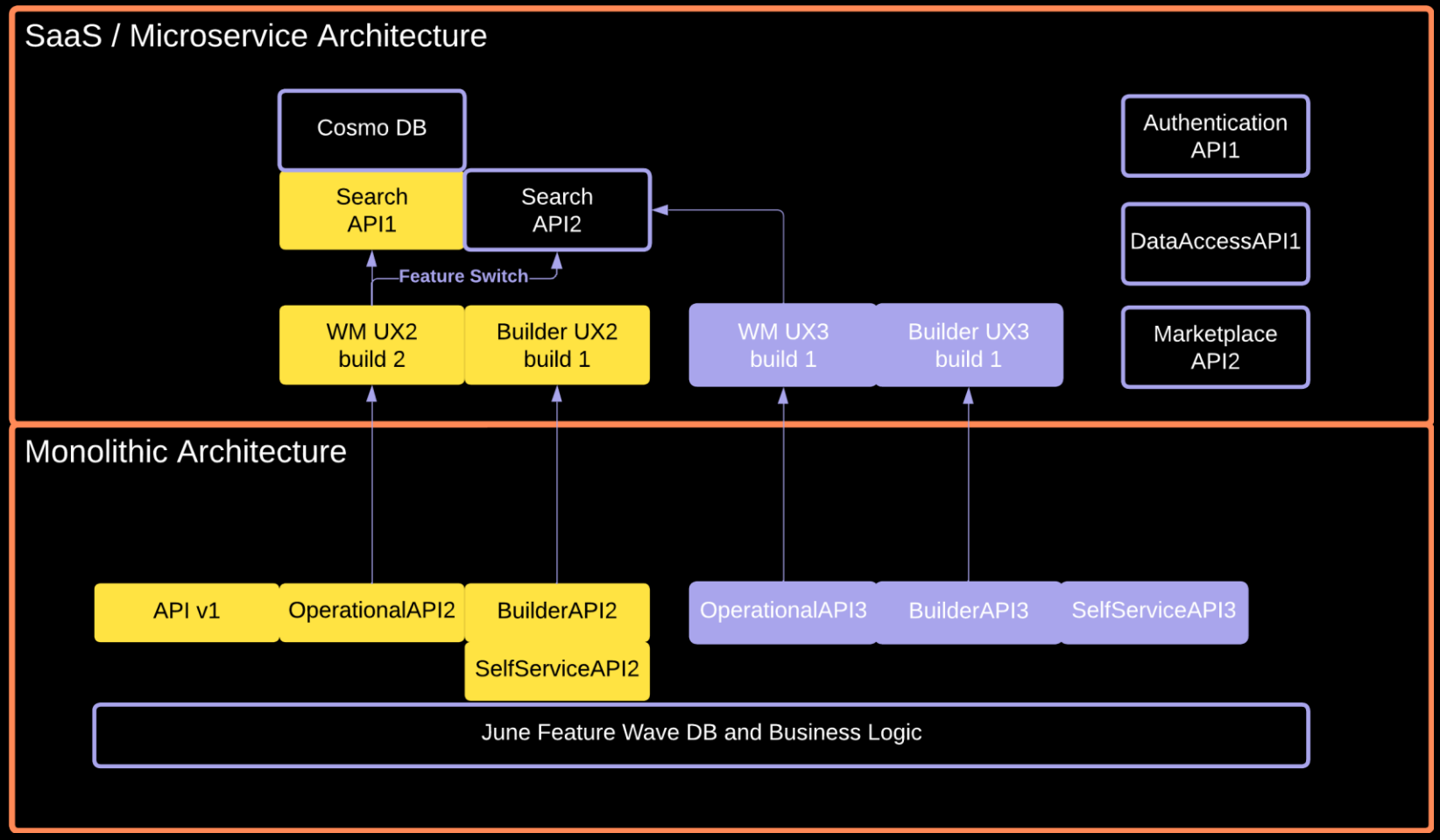
How do users switch to the new UX and experience.



# Releasing the next feature cont…

**Builder Users:**
Will be switched from UX2 to UX3 based on a platform level feature flag that is controlled by system administrators.

You're in control BUT all builder users will switch to the new UX simultaneously. This avoids conflicts between users (for example two users tyring to edit a process but only one of them can add the new action released in the feature wave.

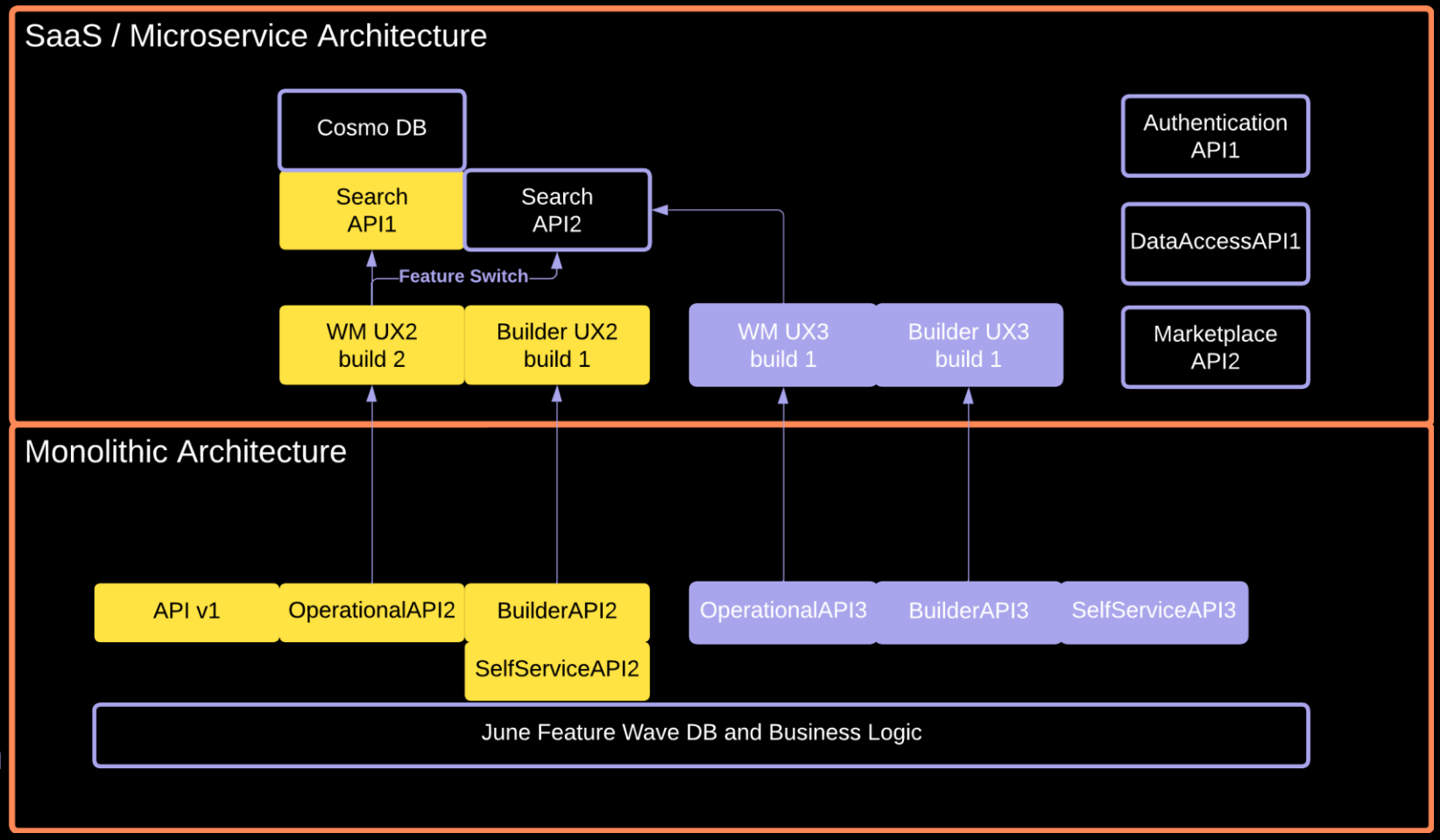


# Releasing the next feature cont…

**Work Manager Users:**
Wherever possible Work Manager users will be in control of the feature flag to switch across to the new user experience.

They can move at a time that suits them.

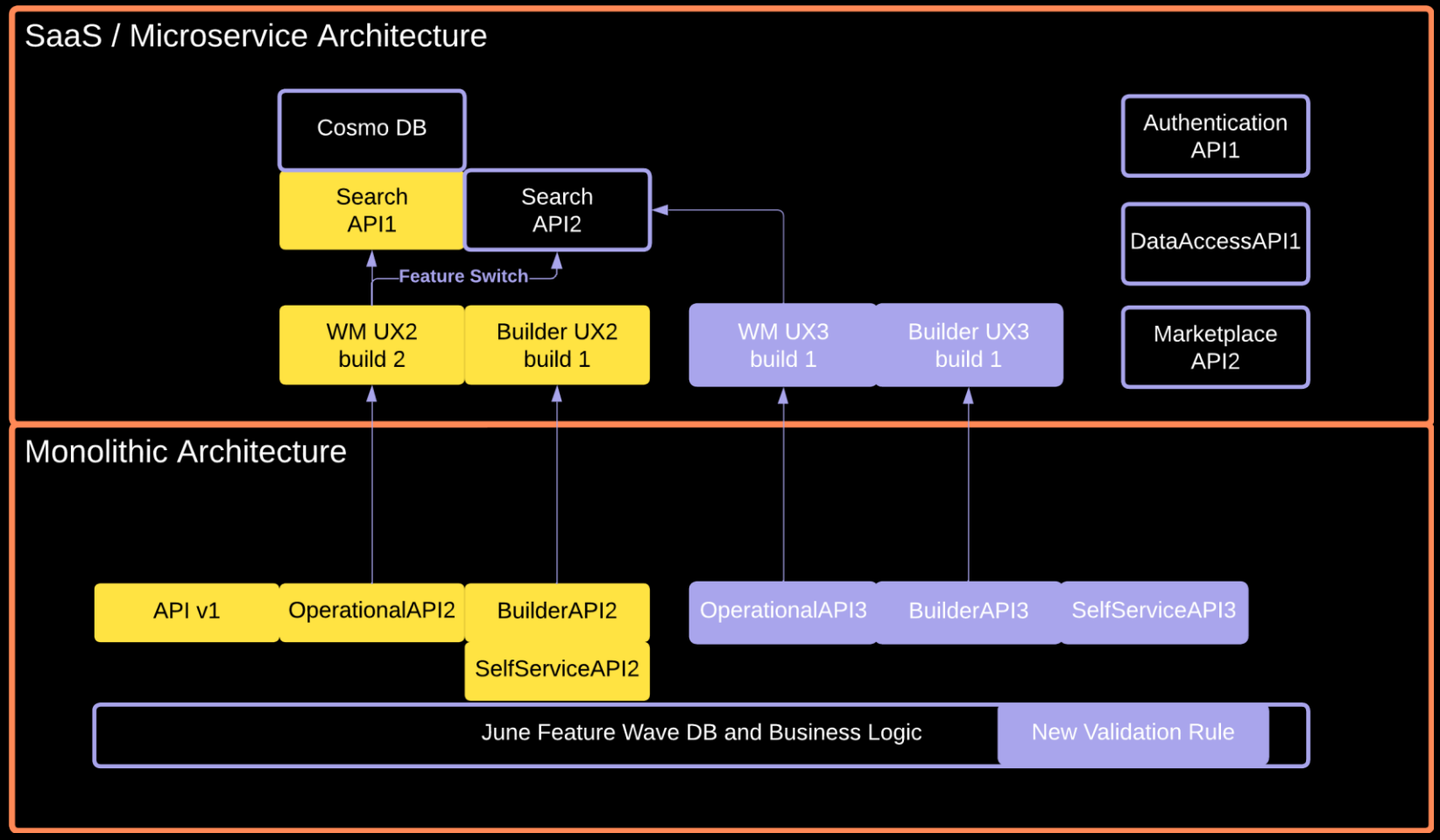
System administrators however will be able to switch the flag for users forcing them onto the new UX.



# What might change between feature waves?

Whilst the June feature wave fully supports the API and UX from the previous feature wave, if the business logic for a feature has changed then this WILL be experienced as a change when up push the new feature wave.

For example, if new logic for processing inbound emails has been delivered, then this will apply from the moment the feature wave is installed.
This is regardless of whether the change is due to a bug fix or a feature change.



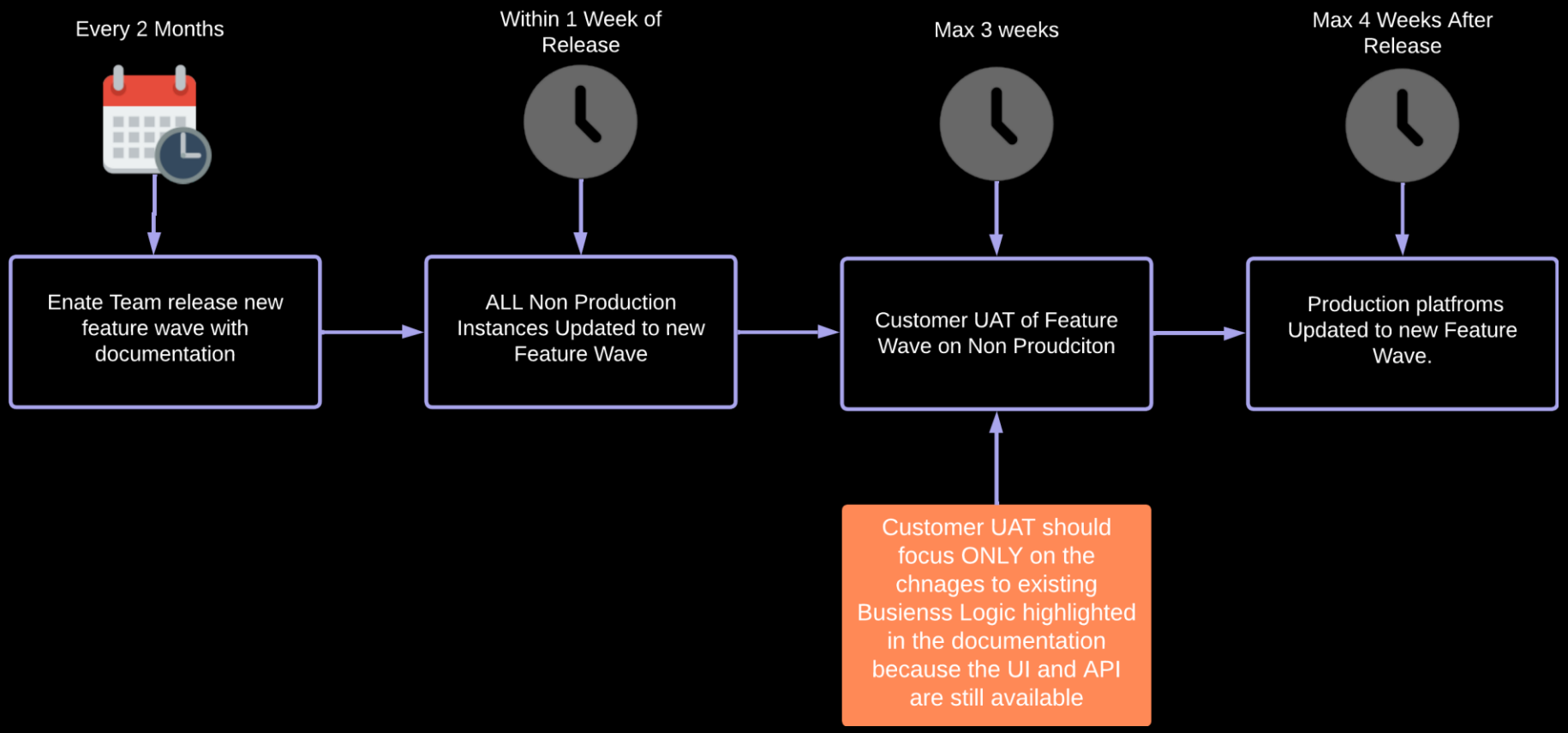
# How will you receive feature waves?

enate

# Releasing Feature Waves

Currently we envisage releasing feature waves every 2 months. However, if we find that most new features are being delivered through microservices then we may reduce the frequency of feature waves to every quarter.

Each Feature Wave will include the TWO previous versions of the API and the TWO previous versions of the UX.



# Updating to A Feature Wave

## Q&A

Will the there be downtime required to update to a new feature wave?

*Yes … but not much because we won't be doing significant data migrations anymore.*

Does this approach restrict us or you in any way?

*Yes, it does mean that we can't make radical changes that would break existing APIs such as remove columns from the schemas.*

Do we still 'back port' bug fixes?

*No. All customers will move onto the new feature wave quickly meaning that there is no need to fix previous feature waves. This may mean fixing previous versions of the API and UX that are available with that feature wave (which is a bit like back porting). This is good for stability because we are only fixing forward.*

Does this mean there will Never be any breaking API changes?

*No, although they should be extremely rare, we cannot guarantee 'never'. If there is a breaking API change then we will give a longer window before updating production platforms to the new FW.*

# Updating to A Feature Wave

## Q&A

How many previous versions of the API and UX will be supported?

*Each feature wave will include the Two previous versions of the API and UX .*

How will API changes manifest given that we are supporting previous versions of the API?

*Most API changes will be additive and many API signatures will be identical between versions (in which case the change to use the new API is as simple as changing URL).*

How do things get resolved if bugs / erroneous code is released?

*With microservices we have the option to both roll back or patch a code change meaning VERY short fix times. For core API and Business Logic code released in the monolithic way, bug fixing and patching will be similar to current approaches.*

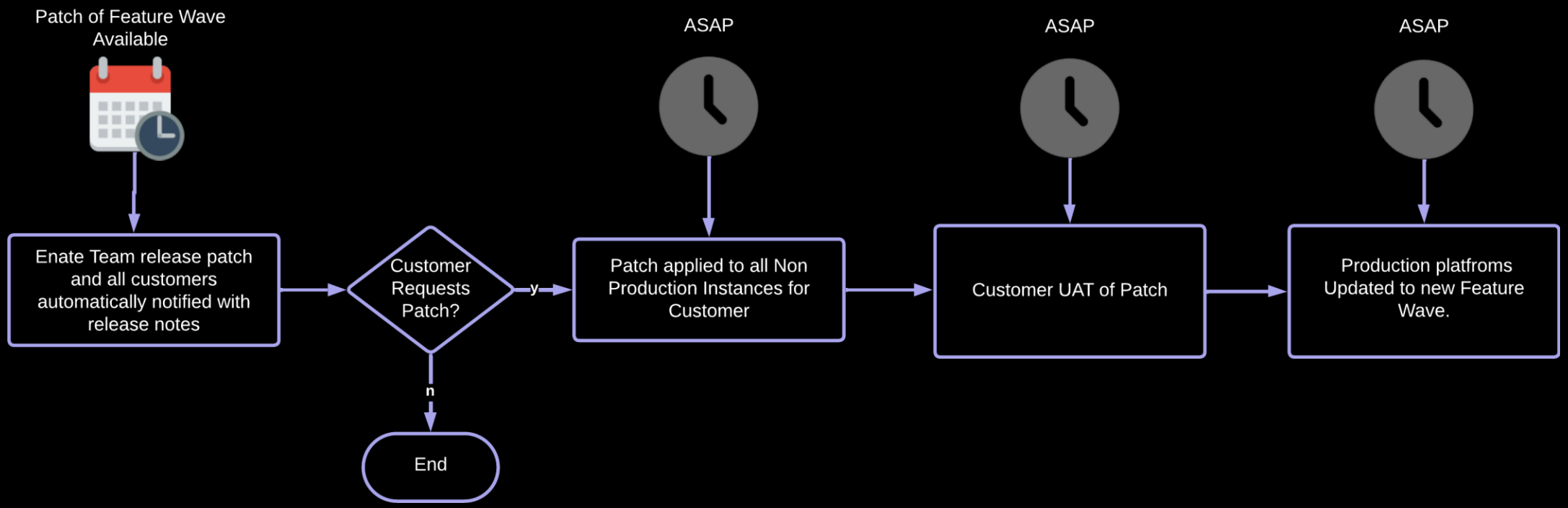
How long do we get to produce our own explainer material?

*You can start creating explainer material as soon as you receive the feature wave on your non-prod environments. But remember users don't have to switch across to the new UX immediately so you've got more time.*

# Patches to a Feature Wave

Taking a patch to a feature wave will remain optional while we release feature waves on a 2 monthly cycle. The only exception to this will be patches to address security or data integrity issues, in which case we will work with you to apply the patch in a timely manner.

For normal patches, the process will be as follows:



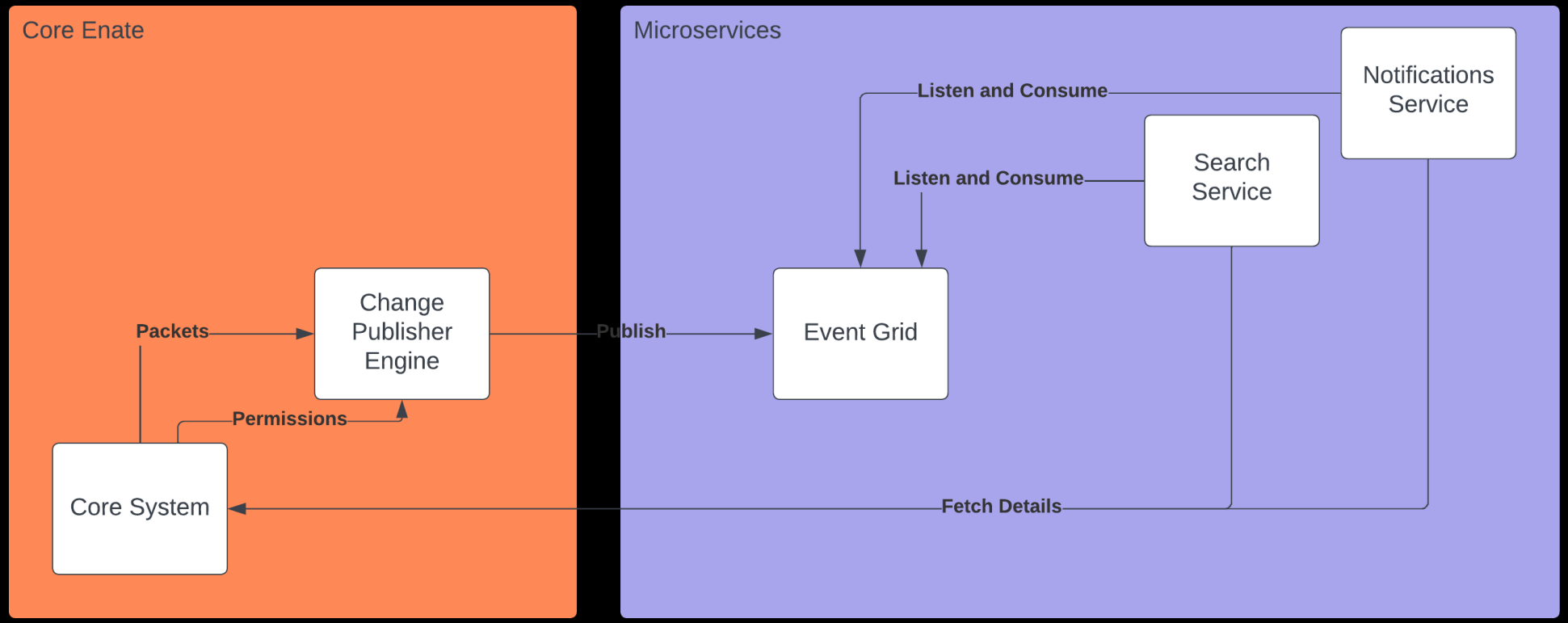
# Deep Dive …managing microservice data

enate

# Microservices

## Populating Microservice Data

The core Enate instance publishes changes to an Azure Event Grid. The microservices listen for events they are interested in and fetch appropriate data into their data stores. Microservice datastores are segregated in the same way as the core database, this is already fully described in the Where's My Data and How is it Stored and Protected document.



# Q&A

For any questions after the session – please send these to:

2024roadmap@enate.net

enate