



Making & Keeping IT Simple and Smart

Rocky Woestenborghs: IT Lead Domestic Products

June 2024



do your thing

Context

- 150 years of banking history
 - Using IT for more than half a century and becoming increasingly more important.
 - Doing the **opposite of keeping it simple and smart.**
- Became technology hoarders:
 - Priority to adding new things instead of removing the old.
- Now IT landscape is part of a larger group eco-system
 - More complexity
- Technology evolving at breakneck pace:
 - Old way of working is not sustainable and not advised for the future.



How to proceed?

In 2018, triggered by our datacenter strategy to migrate our applications to our ING Private Cloud, we started our **simplification and modernisation journey**

Our approach can be divided into 4 area's :

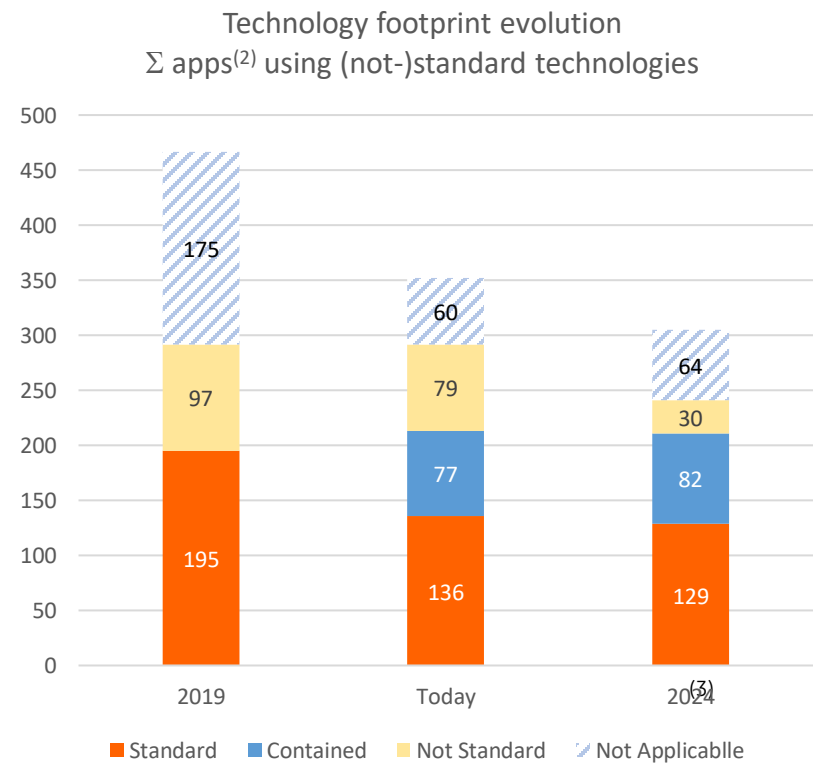
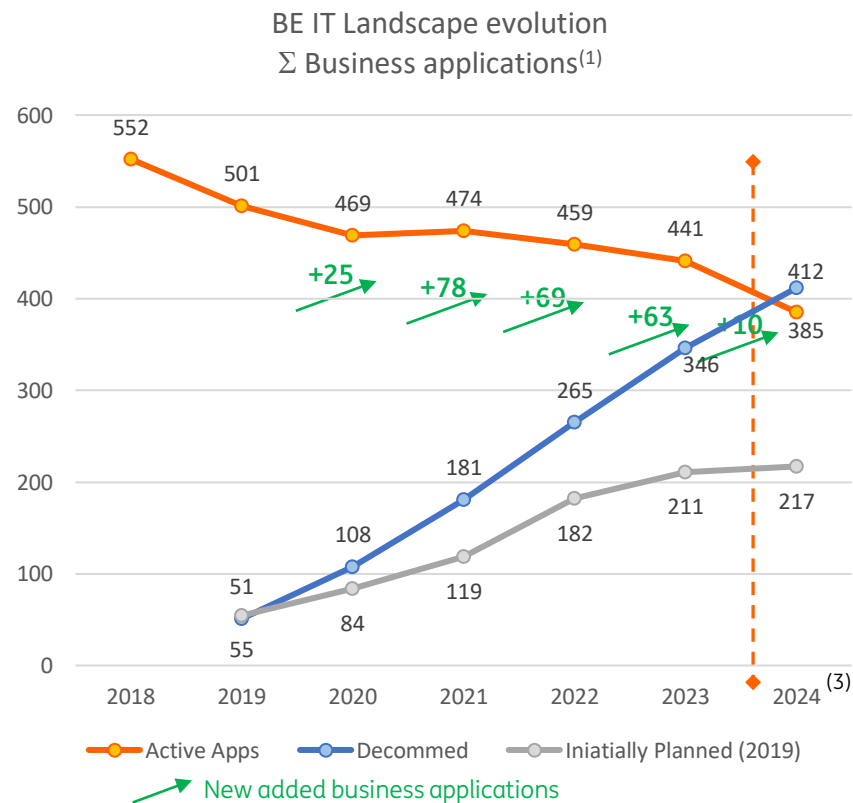
- Remove functional duplication leading to a decrease in number of assets
- Significantly reduce the number of supported and allowed technologies
- Automate RUN
- Automate Change



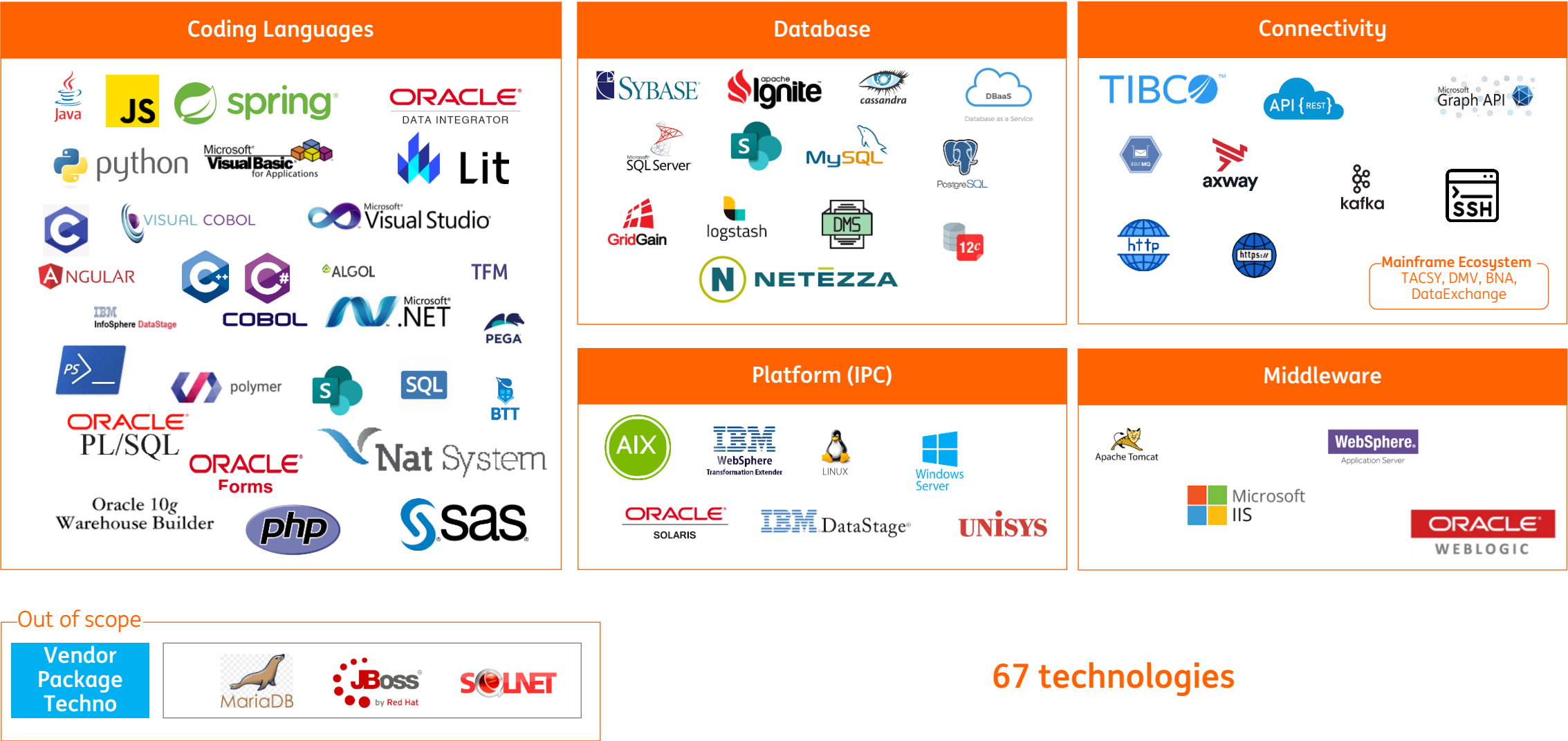


Simplify & Modernise

Simplify & Modernize | Application Landscape & Technology Footprint evolution

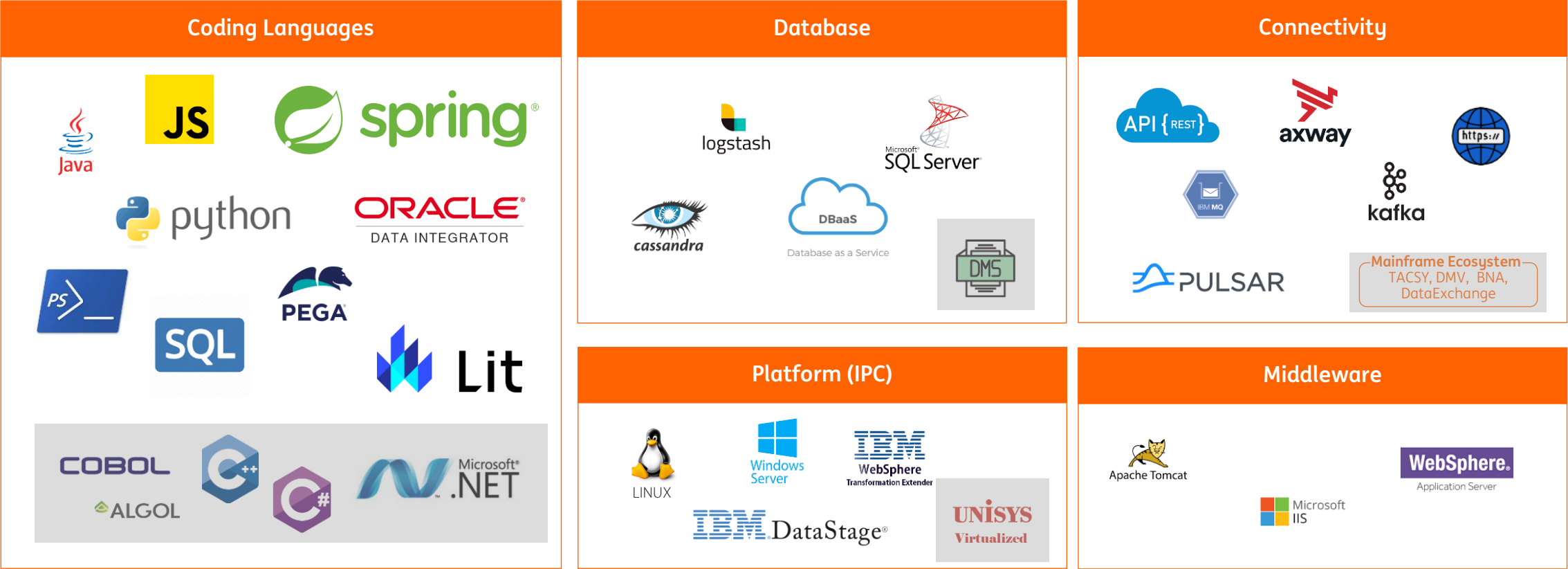


Technology Footprint | At the beginning of 2022



67 technologies

Technology Footprint | Target state




Out of scope



38 technologies

Subject to NBP for decom
(2032)

A photograph of a modern industrial manufacturing environment. Several large, orange robotic arms are visible, mounted on white bases. They are positioned over a complex assembly line with various metal frames and components. The background shows more industrial equipment and a clean, well-lit factory floor.

Automation & Re-use

Automation initiatives

Change

Synthetic test data

Release Automation

Infrastructure As Code

AI Change Impact
Assessment

Identity and Access
management

Run

SRE

Capacity Management
As A Service

AutoCert

AutoPatching

Disaster Recovery As A
Service

Reuse

Digital Factory

Bakery

Pega

Touchpoint
Architecture

Example 1 : Synthetic test data (powered by Datacebo)

Problem statement :

Long lead time test data

Limited coverage of edge cases

One time use

Loading & masking of PROD data into ACC

Solution :



Impact

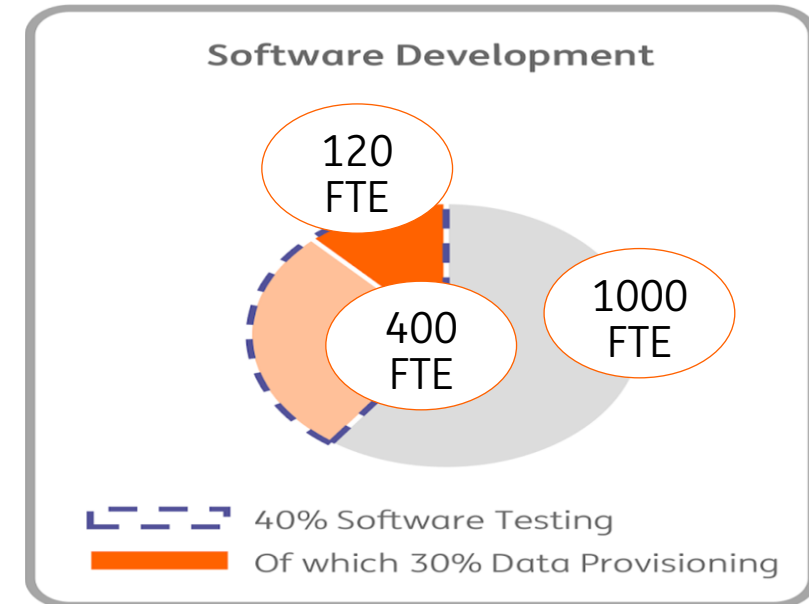
Significant time gained

No sharing of test data

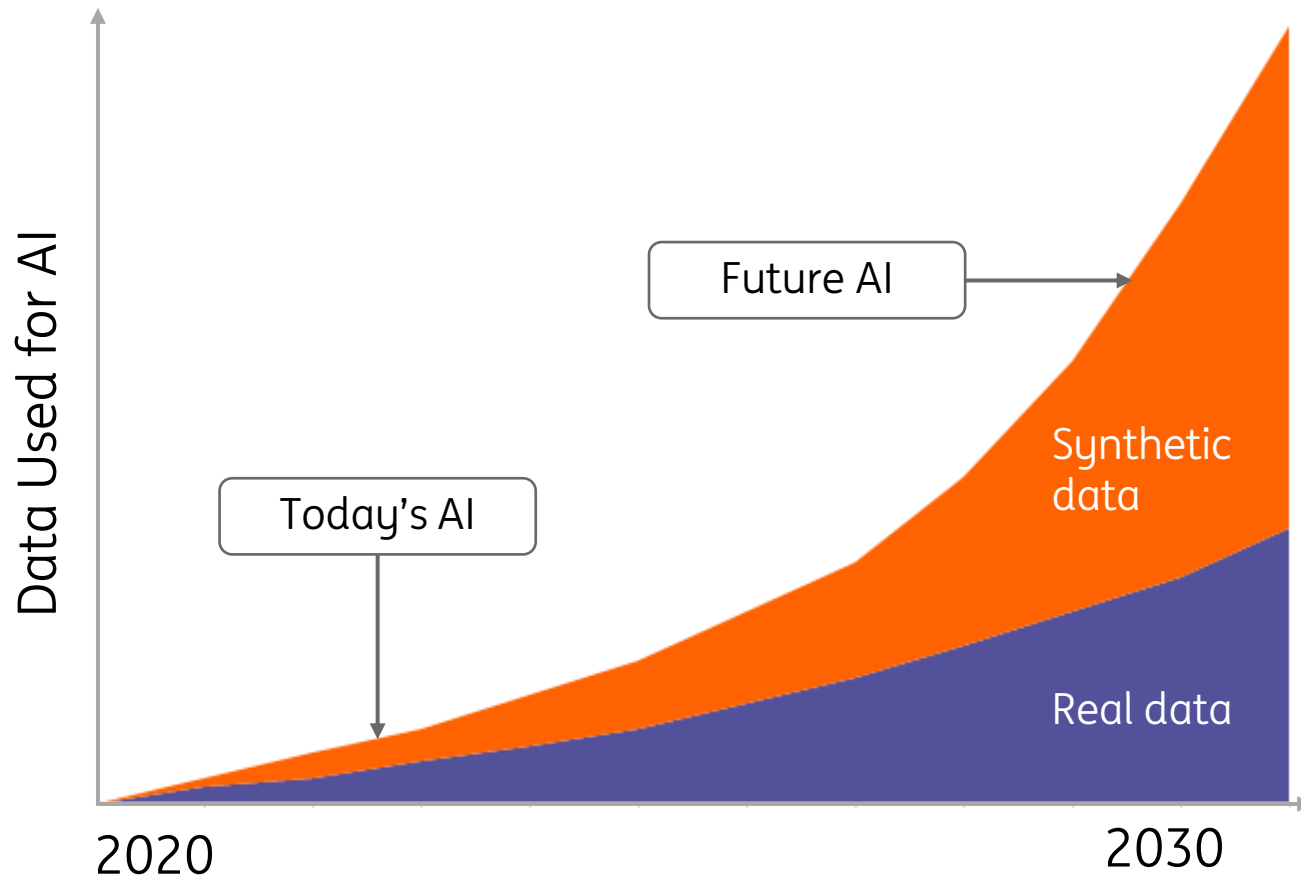
Better test coverage

No privacy risks

Improved time to market



By 2030, Synthetic Data Will Completely Overshadow Real Data in AI Models*

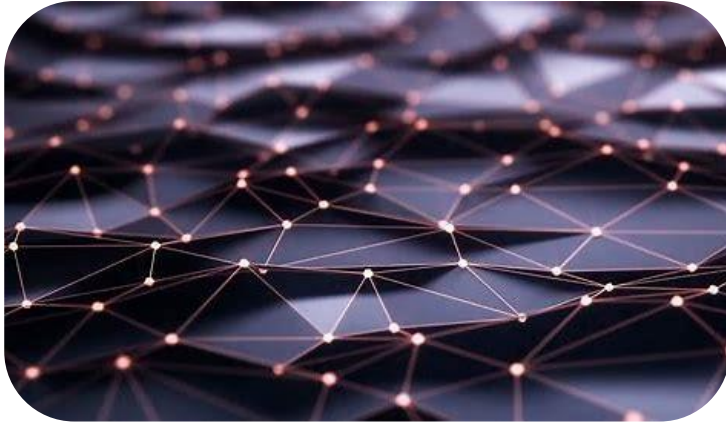


- Artificially Generated Data
- Generated From Simple Rules, Statistical Modelling, Simulation and Other Techniques

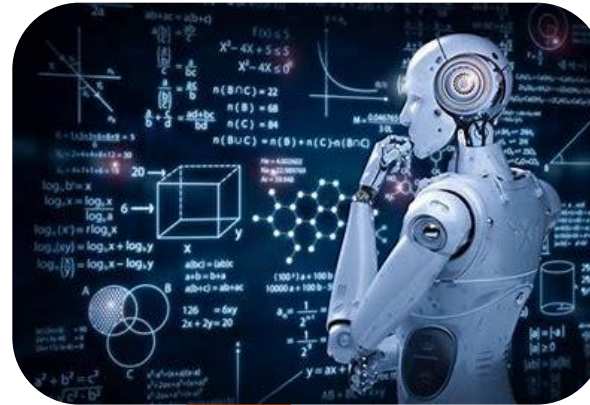
- Obtained from Direct Measurements
- Constrained by cost, Logistics, Privacy Reasons

How does it work?

Neural network (Deep Learning)



Machine learning (model / sample)



Training set

	INSTRUMENT_ID	INCEPTION_DATE	SETTLEMENT_DATE	ORIG_MATURITY_DATE	SRC_AMORTISATION_TYPE	INTEREST_RATE_SPREAD
708	5210304320170731	20170728	20170731	20170831	BLLT_RPAYMT	0.95
709	3443691920170731	20170727	20170731	20171031	BLLT_RPAYMT	1.50
710	3397808620090505	20090506	20090505	20360505	CUSM	0.28
711	3402123320040430	20090505	20040430	20290430	LVL_PAY	0.10
712	5183396820170731	20170727	20170731	20170831	BLLT_RPAYMT	2.25
713	3159562720170717	20170713	20170717	20170815	BLLT_RPAYMT	1.00
714	4081002120170731	20170727	20170731	20170831	BLLT_RPAYMT	1.90
715	3784960220170731	20170727	20170731	20170831	BLLT_RPAYMT	1.10
716	5243395420170331	20170329	20170331	20170929	BLLT_RPAYMT	1.20
717	4949495520170731	20170731	20170731	20170831	BLLT_RPAYMT	0.85

Synthetic data

	INSTRUMENT_ID	INCEPTION_DATE	SETTLEMENT_DATE	ORIG_MATURITY_DATE	SRC_AMORTISATION_TYPE	INTEREST_RATE_SPREAD
235	3661620163891808	20170506	20170703	20191008		0.48
236	5021342811413070	20180211	20180507	20180205	B	0.02
237	3805474417239341	20180201	20161116	20171110	LVL_PAY	1.35
238	5048794391269236	20171213	20180301	20180817	BLLT_RPAYMT	1.54
239	4199038482039758	20170927	20170515	20180515	BLLT_RPAYMT	0.52
240	5518870250638483	20170630	20170702	20191016	CUSM	0.82
241	2641388838396848	20170917	20170514	20181109	BLLT_RPAYMT	0.00
242	5123725580571533	20180227	20170711	20180816	BLLT_RPAYMT	0.67
243	5242188874334071	20170702	20170713	20190816	BLLT_RPAYMT	2.11
244	3695777015957606	20160130	20170508	20180614	BLLT_RPAYMT	1.47

FAKE

Statistics (copula , ...)



Example 2 : Digital Factory

Problem statement :

150 Digital journeys to be build in 12 months

Solution :

Factory Model

Bakery Framework

Impact:

Time gained in project delivery

Clear focus for all involved

Improved time to market

Scalable

Continuous Improvement

Discovery

Start Discovering in full alignment and cooperation with every party involved

Keep Software delivery clean from dependencies

In Discovery we aim to never block Delivery

We prioritize in Discovery, not during Delivery

Delivery

Software Development Process is industrialised and highly automated (supported by CI/CD Pipeline)

In Delivery we start finishing, by focusing on Flow and the lean principle of Pull

In the quality we deliver we trust our process, pipeline and craftsmanship. Quality improvements along these axes

Impediments not only block stories but also the individual working on it, so enabling this person to unblock the impediment

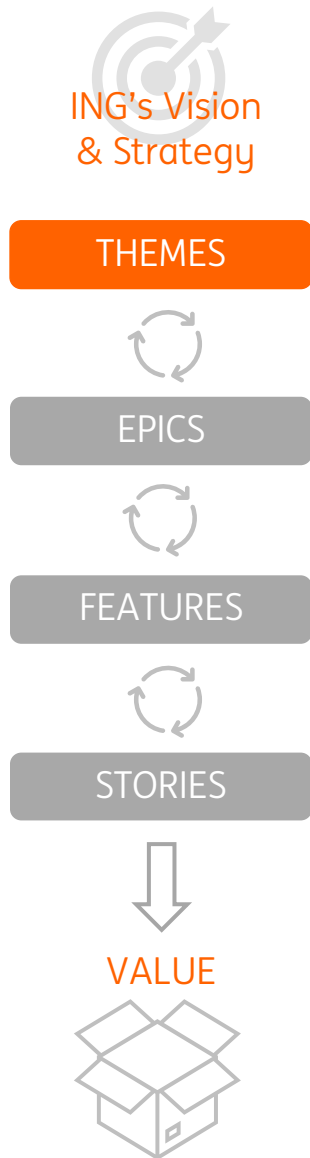
Generic

Focus on Lean Principles

We work Shift Left...

Coaching Leadership

Themes to stories



THEMES:

Bridges Strategy and Execution and serves as guide in prioritization:

- Derived from targets of the Retail Bank
- Represented across 1-5 years
- Commonly translated into **Objectives and Key Results** (OKRs)*

EPICS:

A functional chunk of value that is directly derived from a theme:

- Has a lead time of approximately a quarter.
- Delivers value that discussed and prioritized at Tribe level.
- Usually has cross-tribe dependencies which need to be agreed upon with other tribes in the QBR cycles.

FEATURES:

A functional chunk of value that is directly derived from an EPIC:

- Has a discovery cycle time of approximately 10 working days (1 sprint).

STORIES:

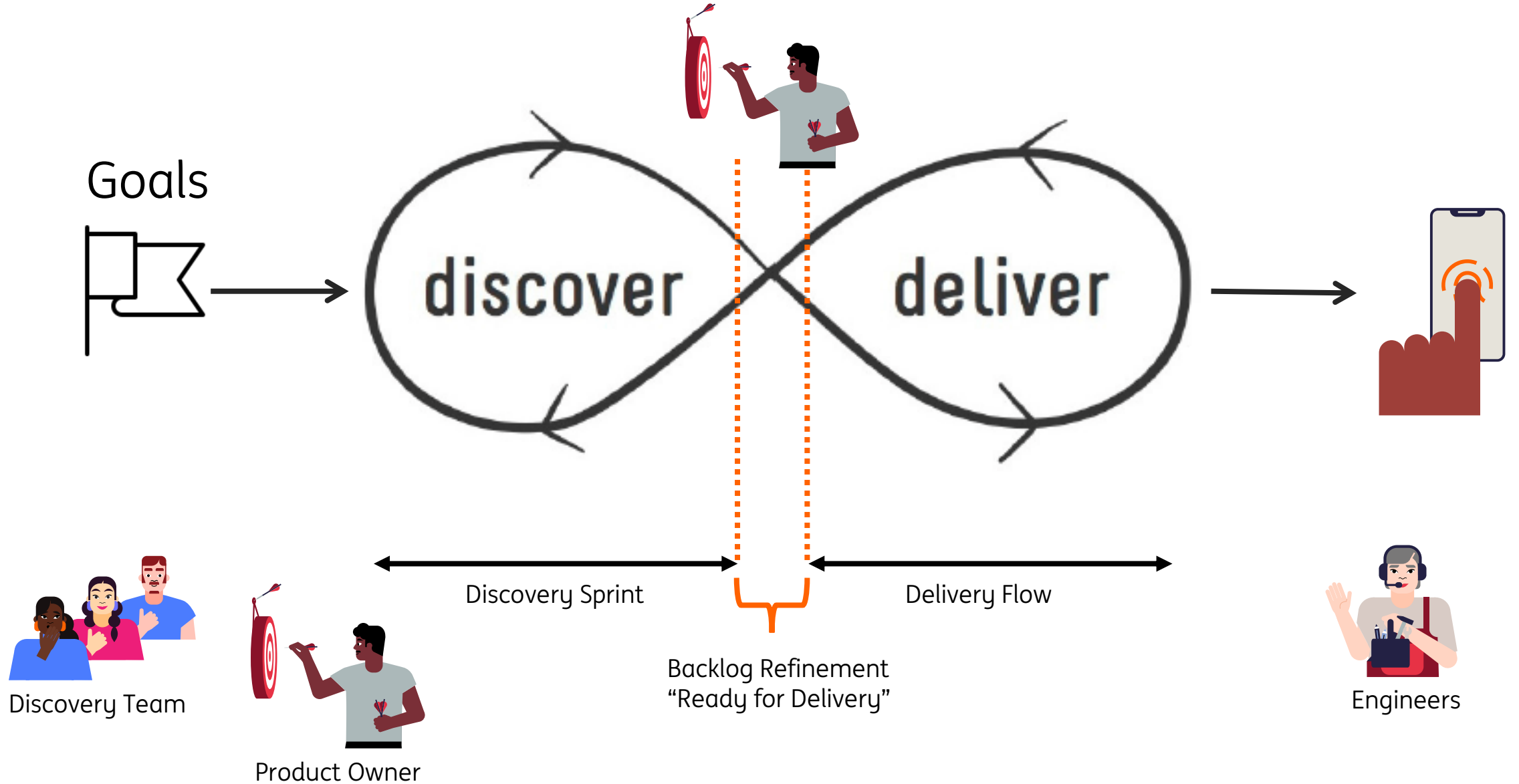
A chunk of work that is directly derived from a FEATURE that complies to the Definition of Ready:

- Has a Built effort of approximately 8 hours (1 day).
- Are considered done only once delivered in production.

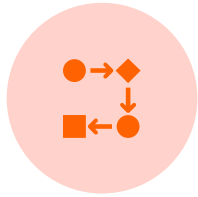
* Objectives – Qualitative and Aspirational Goals

Key Results – Quantitative (SMART) indicators of realizing Objectives

Discovery and Delivery Flow



Principles applied



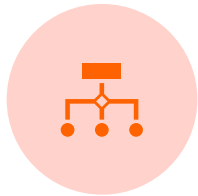
Shift left : we move things that we typically tend to do in later stages to an earlier point.,



We focus on Lean Management principles to spot the things that make us slow and improves upon waste



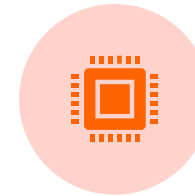
We only start discovering a feature in full alignment and cooperation with every party involved



Keep Software Development Process clean from dependencies : In discovery a feature results in stories that are refined in such a way that all dependencies with other parties are cleared, solved or managed in a way that engineers can focus on building it and moving it to done without having to wait for other parties.



We set priorities in Discovery not during Delivery



Our software development process is industrialized (Baker(y) and supported by our Continuous Integration and Delivery Pipeline.



In the quality we deliver we trust our process, the pipeline and the craftsmanship the individual peers we work with. Quality improvements are performed on these axes.



Enable & Empower : Impediments not only block stories but also the individual working on it, enabling this person to unblock the impediment.

Example 3 : Baker(y)

Problem statement:

Global push for digitalisation of journeys

Low velocity

Low standardisation

Teams re-inventing the wheel & doing the heavy lifting

Observability and resilience not accounted for

Solution:

API based open-source orchestration layer
<https://github.com/ing-bank/baker>

Impact:

Increased speed of delivery

High predictability of flow

Improvement of quality

Global reuse

Open child account

Verify customer rules

Register individual

Send a present

Send a message to customer

Open a payments account

Enable billing

Youth savings account

Verify customer rules

Register individual

Send a present

Send a message to customer

Open a savings account

Enable billing

Mobile onboarding

Verify customer rules

Register individual

N/A

Send a message to customer

Open a payments account

Enable billing

What is Baker & Bakery?

Baker is an ING developed library that reduces the effort to develop (micro)service-based process flows.

Developers declare the orchestration logic in a **recipe**. A **recipe** is made out of

- **interactions** (Code to call (ING) systems providing a service; i.e. Datalake, Databases, other APIs),
- **ingredients** (data),
- **events**. (Client input, API call results, etc)

A visual representation of the recipe allows product owners, architects and developers to have a clear understanding of a process as it is generated from the code.

Bakery is a service that is hosting the feature teams' **recipes** and **interactions**.

Built as a cloud native platform from the ground up.

Bakery is built on ING standard products

- ING Container Hosting Platform (OpenShift)
- Pipelines based on Azure DevOps



How does it work for the teams ?

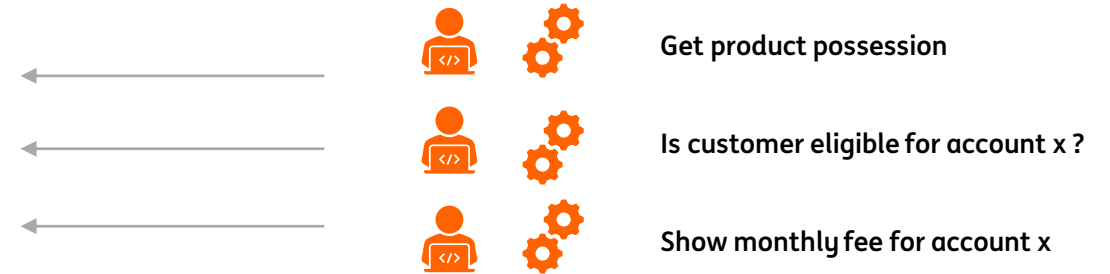
Baker platform team :

- Maintains core Baker Scala library
- Maintains Bakery cloud service in OpenShift,
- Maintains pipelines in Azure DevOps
- Owns reusable recipes in git
- Provides support, standby to consuming teams



ING teams creating digital flows :

- Write their interactions (recipes) in Java/DSL
- Deploy them to Bakery cloud



ING Cloud



Pipelines



Core Library



Consumers



