
>>> network .toCode()
What does an FTS look like?

WHY

WHO

WHAT

HOW

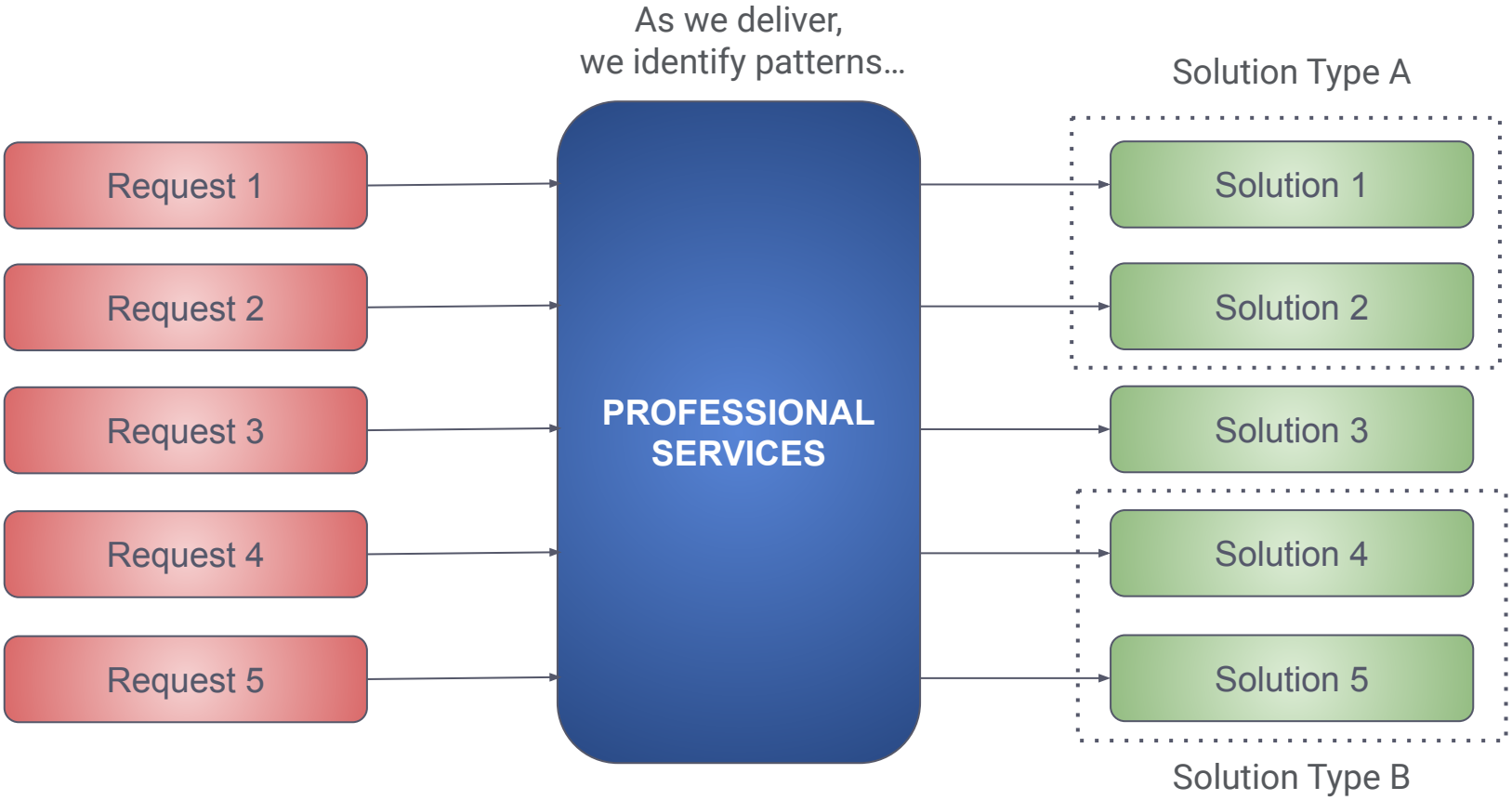
WHEN

Disclaimer

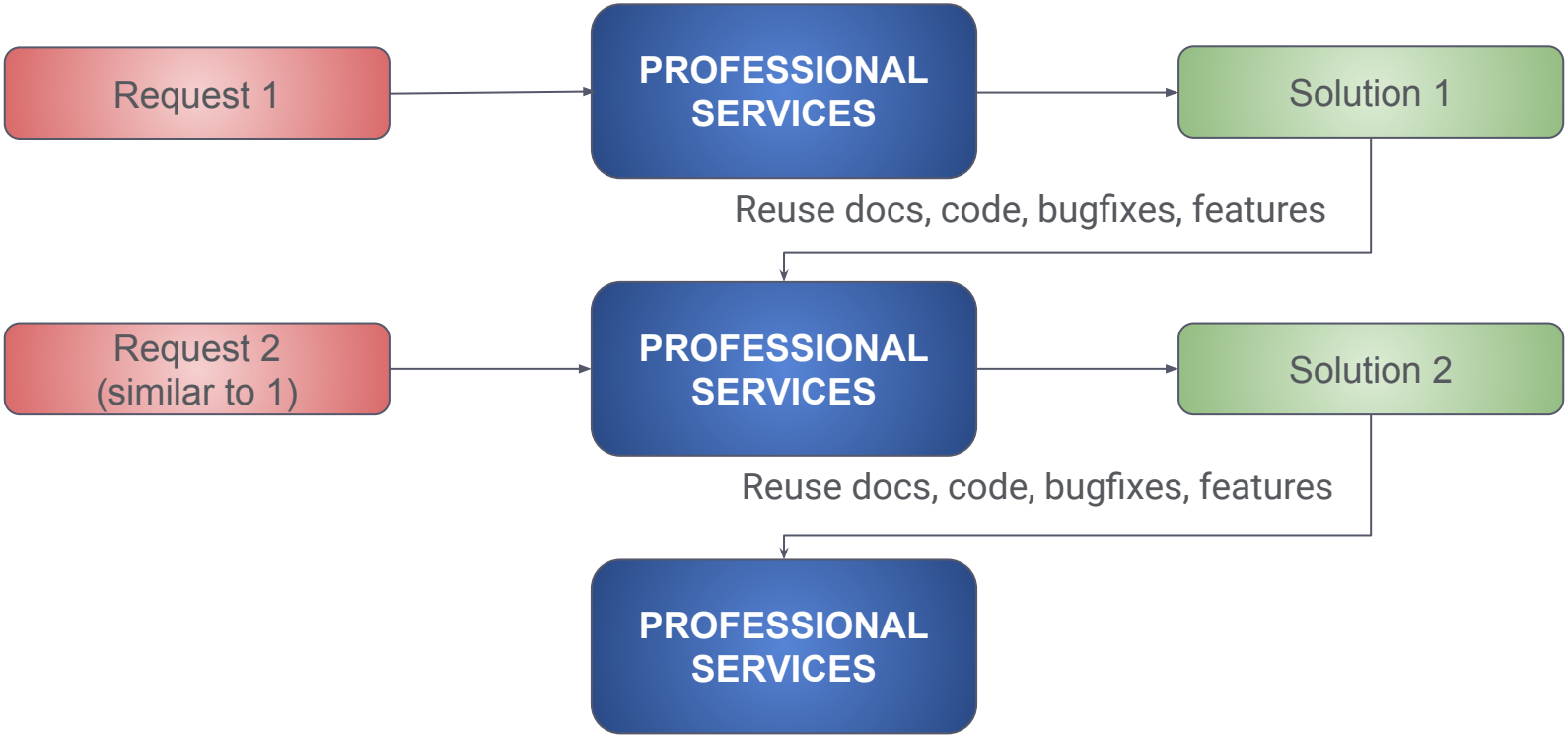
This is just an educated summary of all the info available in Confluence
https://networktocode.atlassian.net/wiki/spaces/CTO/pages/1736671235/Fast+Track+Solutions

WHY

Professional Services evolution



>>> "An FTS is a journey, not a destination"

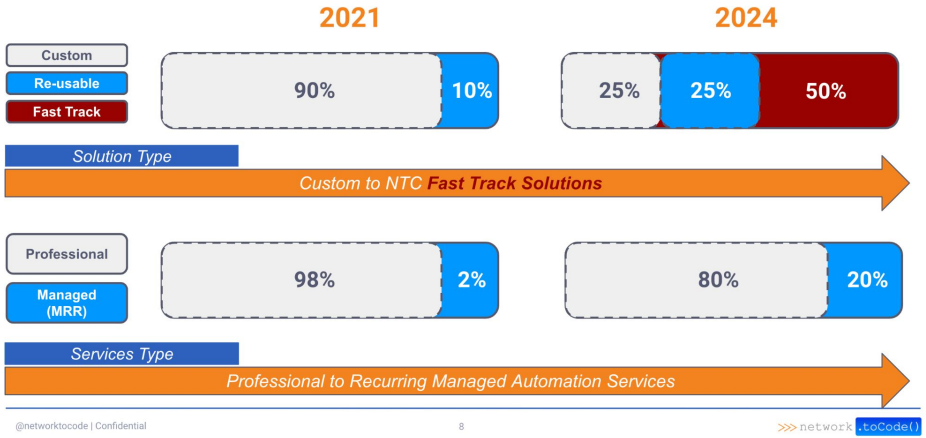


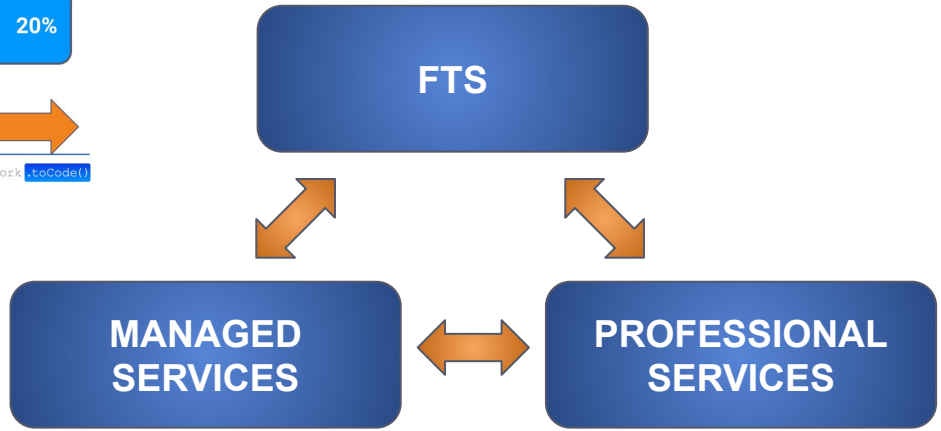
More value for customer (new features) + **Less** internal **cost** (efficiency) = **Increase Business Value**

>>> Provide Solutions

>>> Launching Productized Services

Building NTC to Scale





WHO

>>> Who are involved on FTSes?

STAKEHOLDERS

Understand which solutions
align with the business strategy

SALES

Elaborate Budgets and SOW
Demonstrate solutions
Identify new market trends

ENGINEERING

Develop coding artifacts
Deploy solutions for customers
Create technical documents
Define discovery process
Determine what is possible

Technical Product Owner

MARKETING

Create marketing content
Define marketing strategy

AGILE GOVERNANCE

Define Delivery plan
Prepare Customer interactions

WHAT

>>> FTS Artifacts

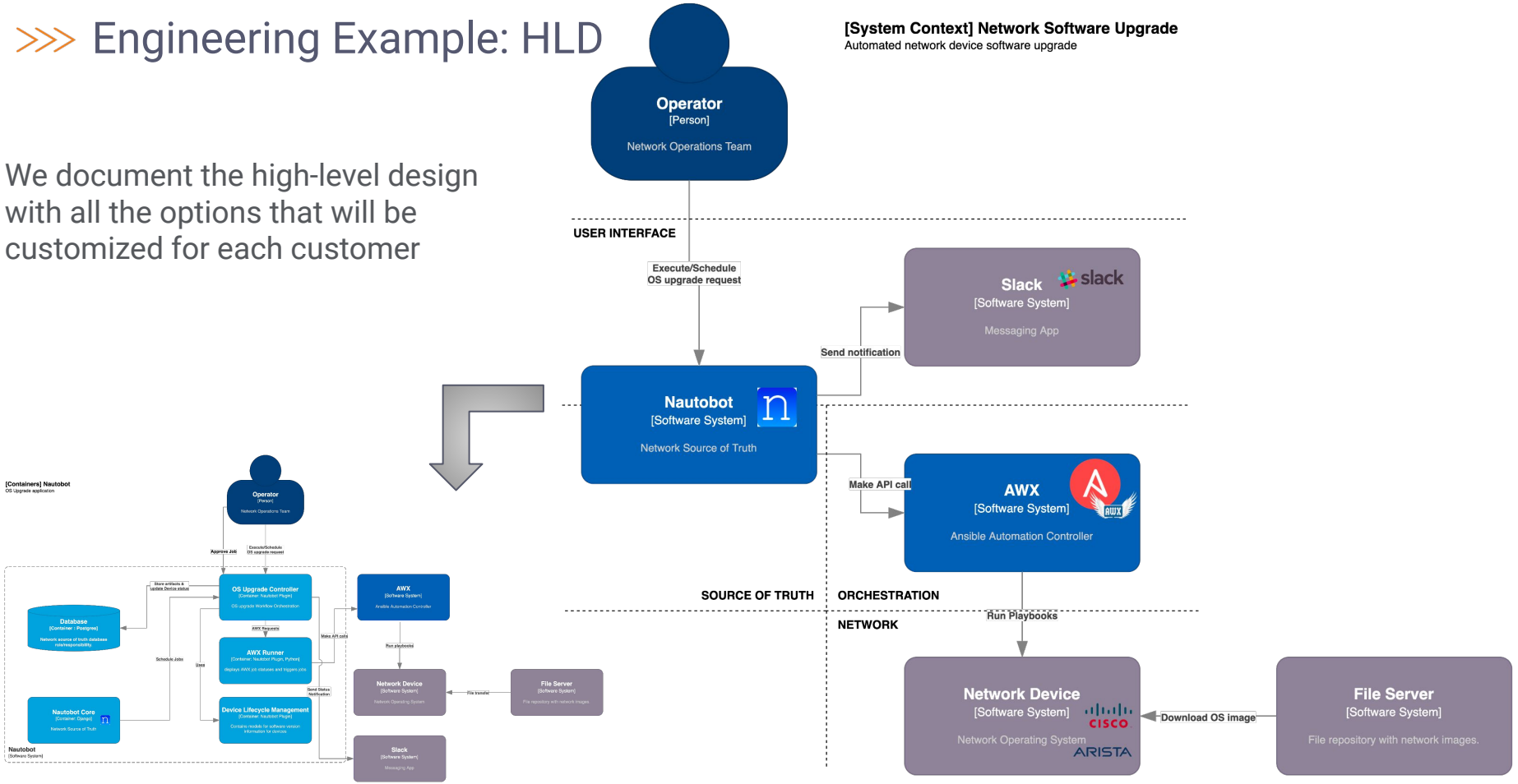
MARKETING	SALES	ENGINEERING	AGILE GOVERNANCE
MARKETING CONTENT	SLIDES	HLD	PROJECT PLAN
	SALES QUESTIONS	LLD	PROJECT MGMT
	BUDGET	ACCESS & DISCOVERY	
	SOW	HANDOFF DOCUMENTATION	
		DEMO	
		OPEN SOURCE PROJECTS	

>>> Sales Example: Budget

FTS - Config Compliance				
Phase	**Tasks**	**Inputs**	**Hours**	**Notes**
Planning & High-Level Design	Customer Profile (1-5): - 5 being large Enterprise (>25000) with Heavy Process. ex Greek State - 3 being Medium Enterprise (>1000) with not a heavy process. ex - 1 being Small Enterpreise < 1000, with simple process. ex	4	1.5	SA to define based on complexity and difficulty of customer **THIS HAS A BIG IMPACT IN ALL THE OTHER ITEMS** we use a factor to apply
	Existing PS Customer (y/n)	n		Will drive hours below for access/onboarding
	Existing Nautobot Deployment (y/n)	y		if not, we should add the FTS Nautobot Deployment a part
	NOSes (qty): Cisco IOS, Cisco NXOS, Cisco ASA, Arista EOS, AIRE OS, Linux, F5, Junos, Fortinet	2	6	2 hrs per NOS, for discovery work
	NOSes (qty) not in supported in Netutils parsers	1	3	2 hrs per "non core NOS", for discovery work
	Network Roles (qty)	3	9	2 hrs/role, for discovery work
	Migrating from another tool (y/n)	n	0	If 'y' add hours depending on the customer profile
	Existing configuration standards & Jinja template (y/n)	y	6	Jinja Templates
	HLD documentation		24	takes into account all the previous parameters
	Additional Tasks			
	Used to add tasks that aren't yet in the budget			
	Planning & Design Engineering Subtotal		49.5	
Deployment & Integrations	**Solution Deployment**			Customer Profile affects to all the parameters here
	NTC Nautobot Deployment (y/n)	y	0	If 'n'; add 16 hours for adopting our deployment standards (risk)
	Access & Onboarding		6	factors -> customer profile, existing customer
	Nautobot Golden Config installed (it assumes Nautobot installed and data populated)	n	6	if 'y' 2 hrs if 'n' add 4
	Known Version Control System: Git, GitLab	y	3	if 'y' 2 hrs if 'n' add 8
	GC to perform backups (y/n)	y	3	if 'y' 2 hrs if 'n' add 8
	GC to perform intended configs (y/n)	y	3	if 'y' 2 hrs if 'n' add 8
	Access from Nautobot to devices (y/n)	y	3	if 'y' 2 hrs if 'n' add 4 , only if GC performs backups
	Repos already created (y/n)	y	0	if 'n' add 2
	Data Population missing but with an existing SSoT	n	0	if 'y' 4 hrs , doens't apply if not Intended config
	Data Population missing without an existing SSoT	n	0	if 'y' 20, doens't apply if not Intended config

Engineering Example: HLD

We document the high-level design with all the options that will be customized for each customer

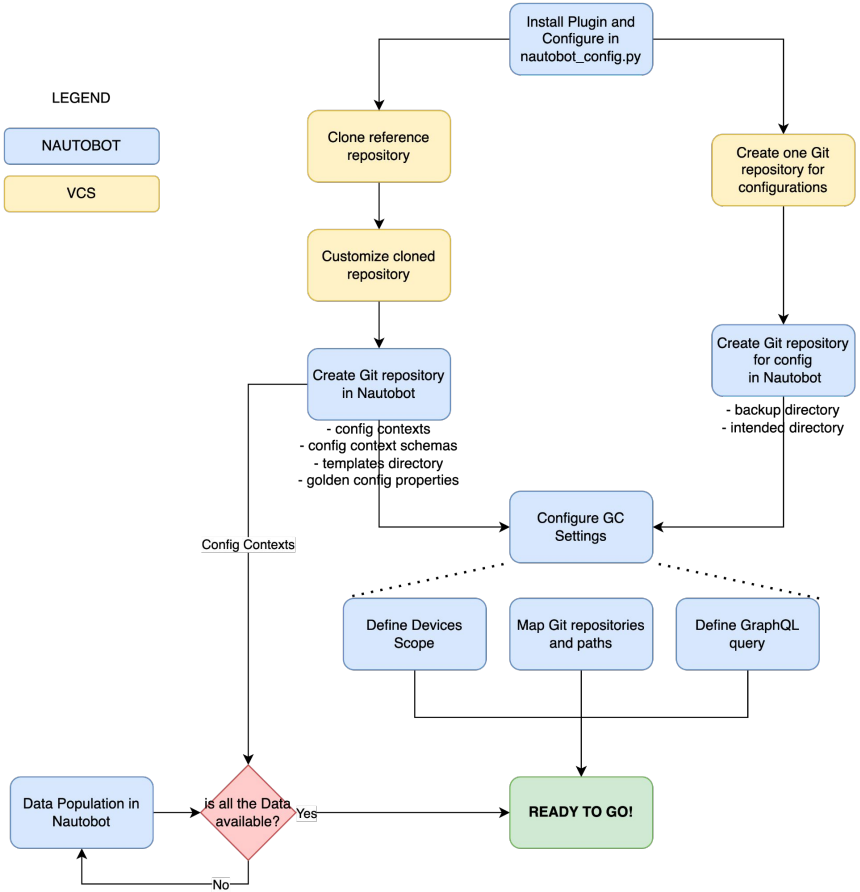


Engineering Example: LLD

The LLD should contain the information you would like to have when you start the project, in a NTC opinionated way.

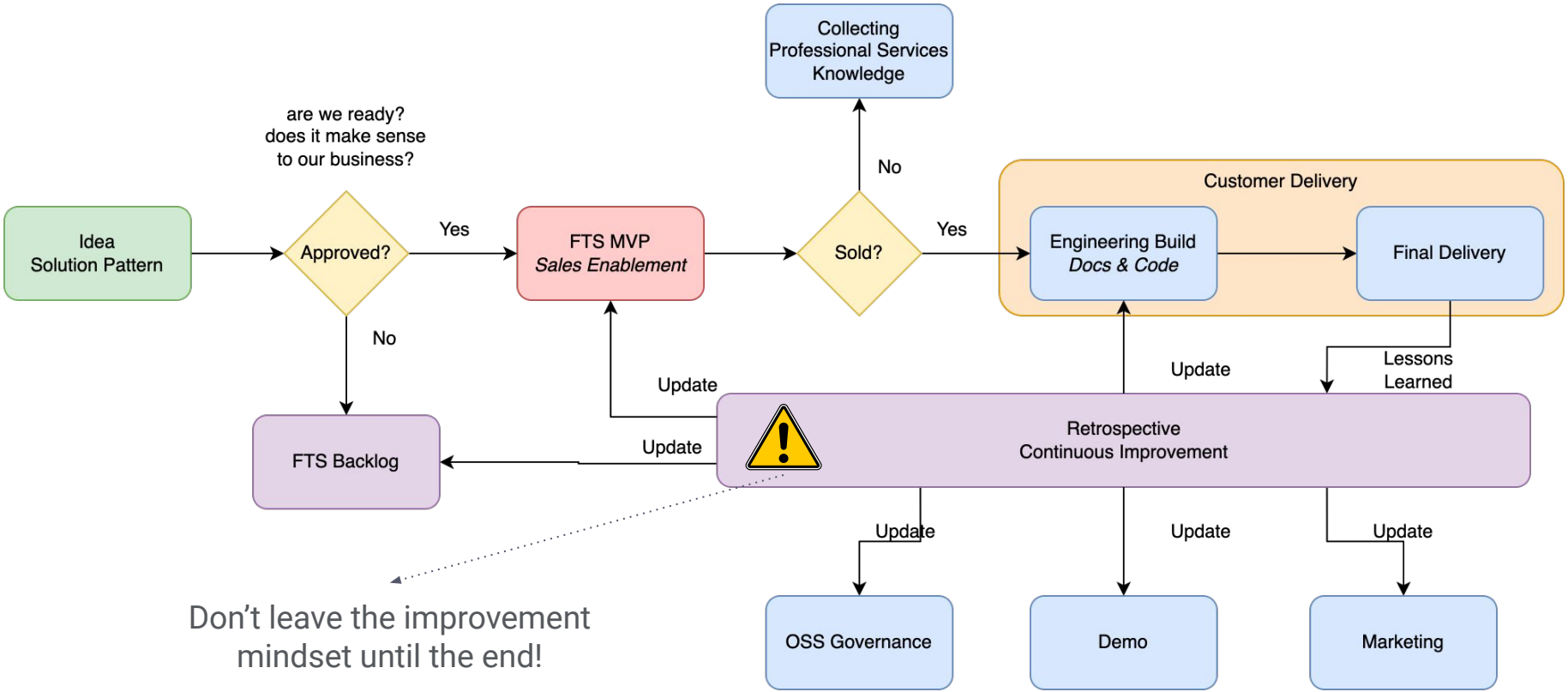
We can add extra sections to enhance and cover corner cases, but the goal is to cover the deployment process in a straightforward way.

Imagine it as a **cheatsheet**



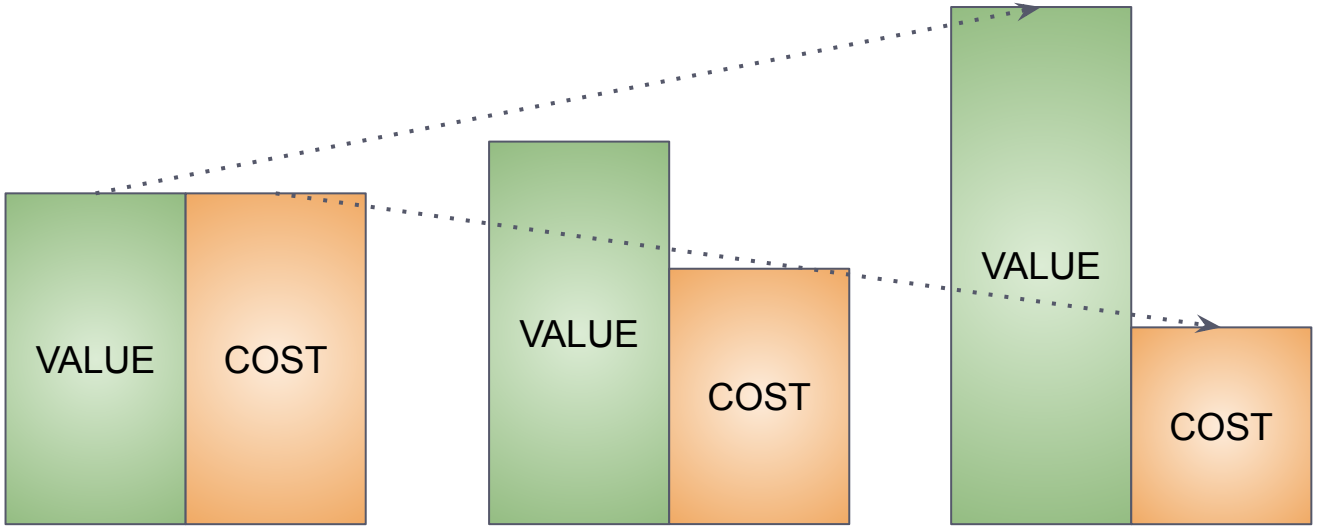
HOW

>>> Simplified Workflow

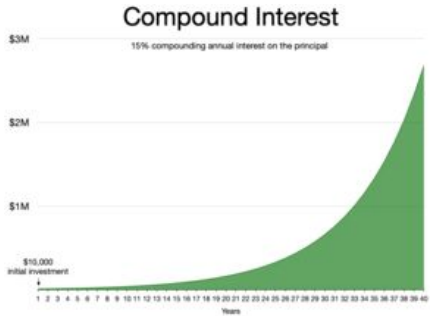


>>> Continuous Improvement

VALUE => Benefit the customer takes from the solution
COST => How much effort takes from NTC to deliver the solution



Simply, every iteration, we add some new value and we simplify our execution process



Constant Improvement
Exponential Growth

WHEN

>>> When should I start thinking about FTSes? NOW

- Everyone should start thinking with the FTS mindset
  - What part of this engagement could be **reused**?
  - What could make the delivery of this solution more **efficient**?
  - Which killing **feature** will add value to the solution?
  - Which where the main **blockers** and how could be avoided?
  - Why the **documentation** didn't include this situation?

- Contact the Architecture team (#ask-architecture) with ideas, suggestions to be considered as seed of ongoing/future FTS


>>> network .toCode()
Questions