Absa – BIAN ADOPTION

Table of Contents

Context

About Absa

Absa is a truly African brand committed to finding local solutions for uniquely local challenges and everything they do is focused on adding value. To this end Absa offer all clients across the continent a range of retail, business, corporate and investment, and wealth management solutions as well as ensure a positive impact in all the countries where it operates in.

Vision : To become digital bank of choice.

Primary Drivers for BIAN Adoption

- The aim of the bank is to become the digital bank of choice across the African continent with customer centricity at the core. Absa wanted to achieve this by ensuring the faster launch of digital products and features for its customers. The bank had a vision to Lego-fy its banking services by repackaging services into reusable building blocks which would guarantee them success in their digital transformation journey.
- Decompose existing legacy middleware into API, Microservices.
- Introduce new age digital channels and establish DevOps, CI/CD for faster Time To Market.

Our new Microservices/API platform - Way Forward - Principles

The new platform is not a like-for-like replacement for existing ESB middleware

- Service will not be built to retrofit existing channels.
- Not just a protocol change i.e. SOAP to REST.

The new platform is more than a “gateway”, it provides :-

- Decoupling and abstraction from the core banking and other backend services.
- Standardization and governance via its use of the BIAN Framework.
- Orchestration where it is required.

Vendor applications and in-house developed solutions needs to comply with the BIAN-based Absa standards that have been adopted when competing for business.

Designed with a Microservices architectural style to :-

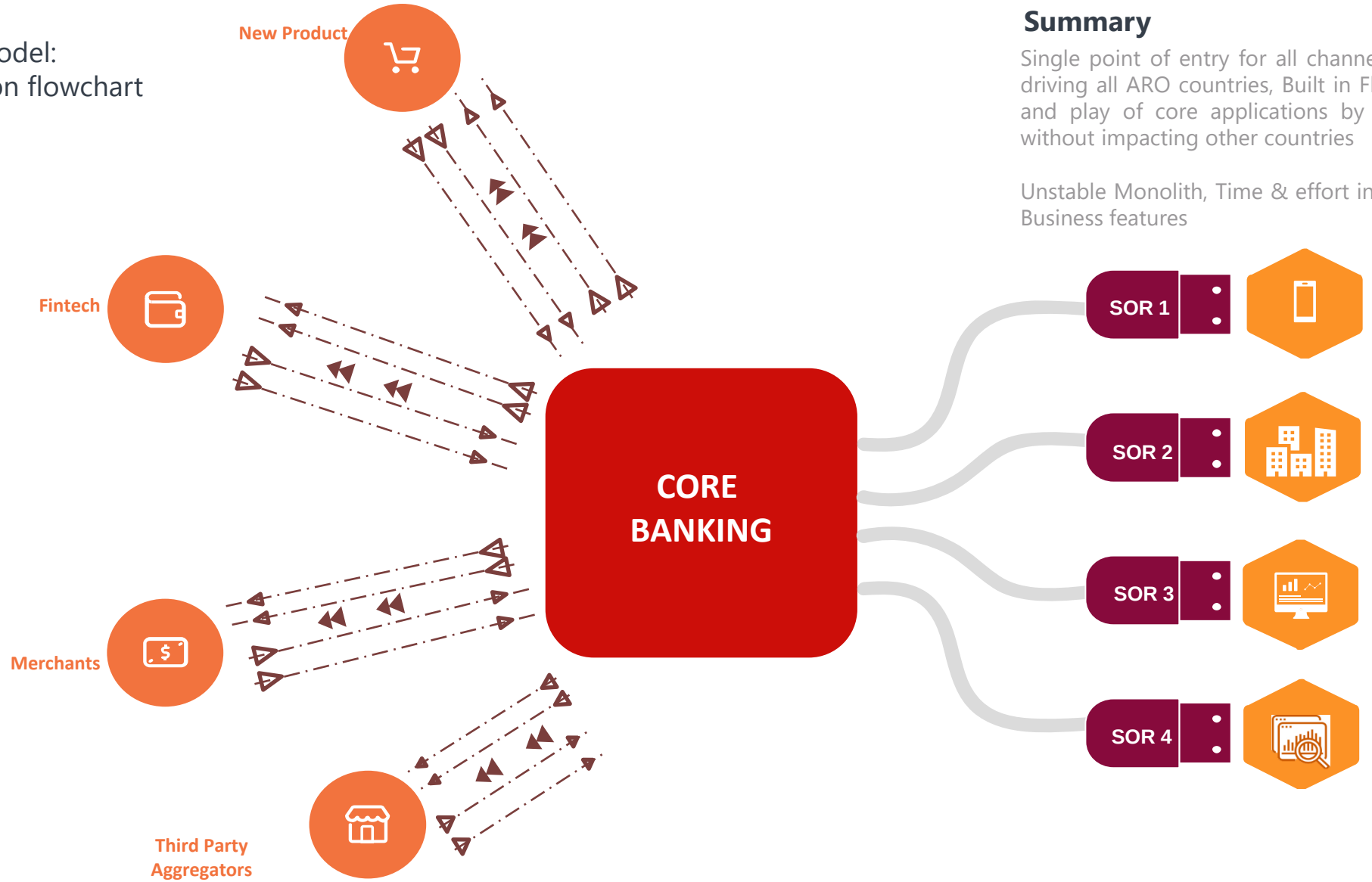
- Decouple complex systems from business functions like with SOA (Service-oriented Architecture) but with more service independence and more resilience.
- Leverage of AGILE and best practice principles coming out of our industry.

The new platform seeks to adopt a standardized approach by implementing ‘common’ architecture principles across the Absa Regional Operations landscape.

The new platform is designed to decouple back-end solutions so that that a “plug-n-play” approach can be adopted to replacing back-end applications.

Existing Architecture

Core banking model:
Single transaction flowchart



Summary

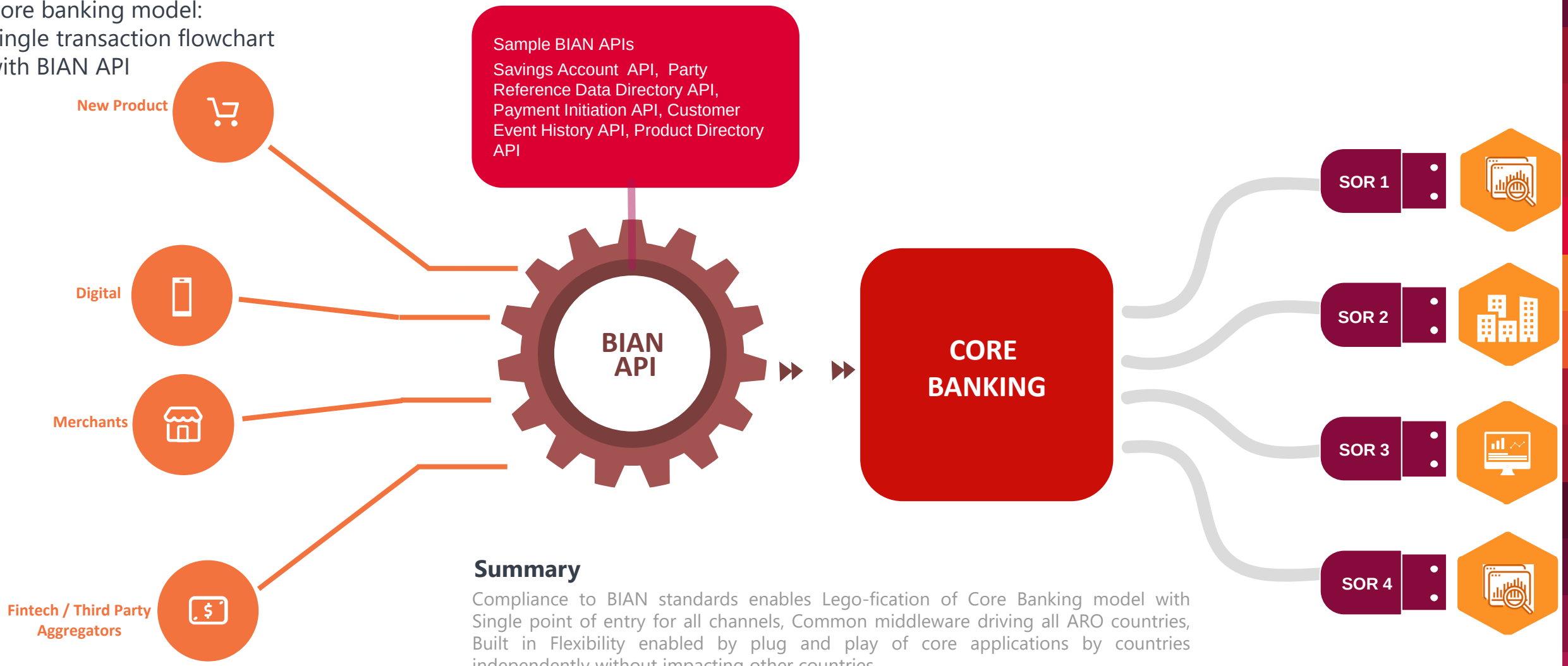
Single point of entry for all channels, Common middleware driving all ARO countries, Built in Flexibility enabled by plug and play of core applications by countries independently without impacting other countries

Unstable Monolith, Time & effort intensive to introduce new Business features

Applications | Core Banking | ARO Apps | Multiple Integrations with Core banking | Integrations through multiple languages | Core Banking Integrating with internal applications

BIAN Adopted Architecture

Core banking model:
Single transaction flowchart with BIAN API



Summary

Compliance to BIAN standards enables Lego-fication of Core Banking model with Single point of entry for all channels, Common middleware driving all ARO countries, Built in Flexibility enabled by plug and play of core applications by countries independently without impacting other countries

Secure, Reusable, Standardized enabling cost and effort efficient introduction of Business features

Applications | Core Banking | ARO Apps | SINGLE Integrations with Core banking using BIAN API | Integrations through SINGLE BIAN language | Core Banking Integrating with internal applications

Program Approach

MVP – Current Approach

- Focused on onboarded channel only

Pros:

- Faster TAT (Turn around time) for channels
- Zero wastage

Cons:

- Additional cost due to continuous engagement with SORs and Channels
- Lack of reusability. Need enhancement every time

Big Bang

- Consider all channel needs at once

Pros:

- Zero rework as all requirements analysis completed before start of development

Cons:

- Can lead to wastage due to un-used APIs
- Increase in time to market of new channels
- Disconnect between business needs and solution offered
- Not able to gain competitive advantage

Hybrid – Suggested Approach

- Consider requirements for existing key channels and uplift APIs for new requirements on need basis

Pros:

- Build right first time
- Zero or minimal re-work
- Reduce cost for SMEs support
- Faster Time to market of channels introduction

Cons:

- None

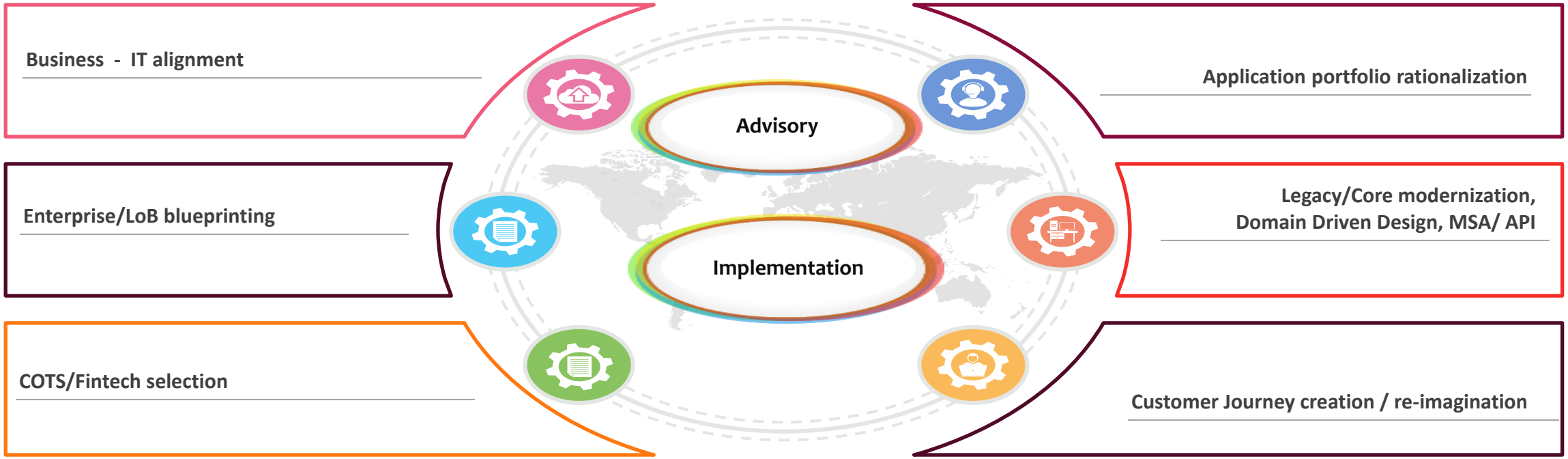
- Adopted green field implementation approach to maximize benefits of agile architecture.
- Targeted new Channels to be onboarded first on the new platform followed by transformations of existing channels.
- MVP approach followed for quick onboarding of transactional Chat Banking channel and now we are looking at analysing existing capabilities of all digital channels.

BIAN Adoption trends & recent experiences

Multiple customers
Few customers
Very Few customers

Speed to Market – Launch products, features, capability faster

- **Business - IT Alignment & Enterprise/ LoB Blueprinting** (Multiple customers)
- **Application portfolio rationalization to identify gaps, overlaps and misaligned applications** (Multiple customers)
- **Match and select vendor solutions/COTS/Fintech for business capabilities** (Very Few customers)
- **During M&A process to select competing capabilities from the merged organization** (Very Few customers)
- **Moving away from ESBs to Standardized APIs and Microservices on Cloud** (Multiple customers)
- **Legacy modernization / Core systems repurposing and moving towards coreless banking** (Multiple customers)
- **Create DDD based right sized services for any greenfield / brownfield implementations** (Multiple customers)
- **Innovation - Customer journey creation / re- imagination** (Few customers)



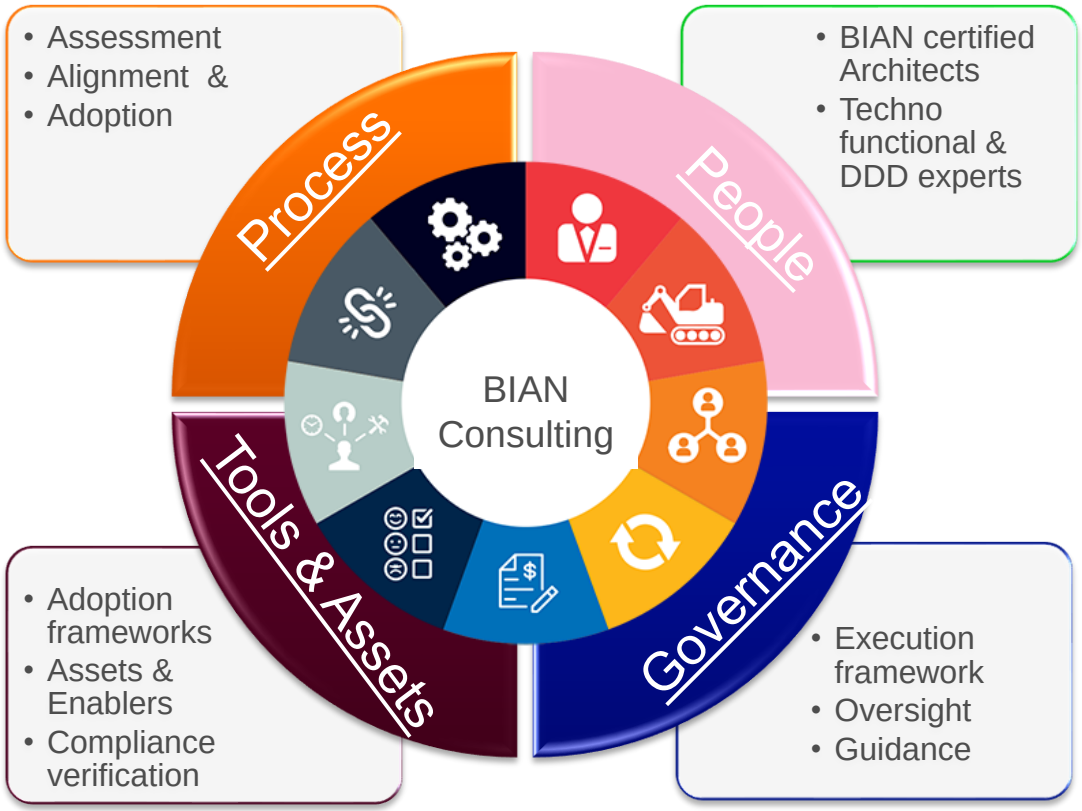
TCS BIAN Consulting

TCS BIAN Consulting
(BIAN.Consulting@tcs.com)
Establishing future readiness in Banks

Participation in 16 BIAN working groups | Unique assets empowering adoption | Multiple adoption pattern expertise.

BIAN Consulting

Enable BIAN adoption, Adoption assessment, Adoption pattern identification, BIAN compliance, Contextualization and extension of BIAN business Scenarios, Accelerate deployment through TCS Enablers



The Journey – We started with an execution approach …

Post PoC Implementation, a detailed execution approach was put together for the entire program

TCS BIAN Assets | Templates | Enablers | Best Practices

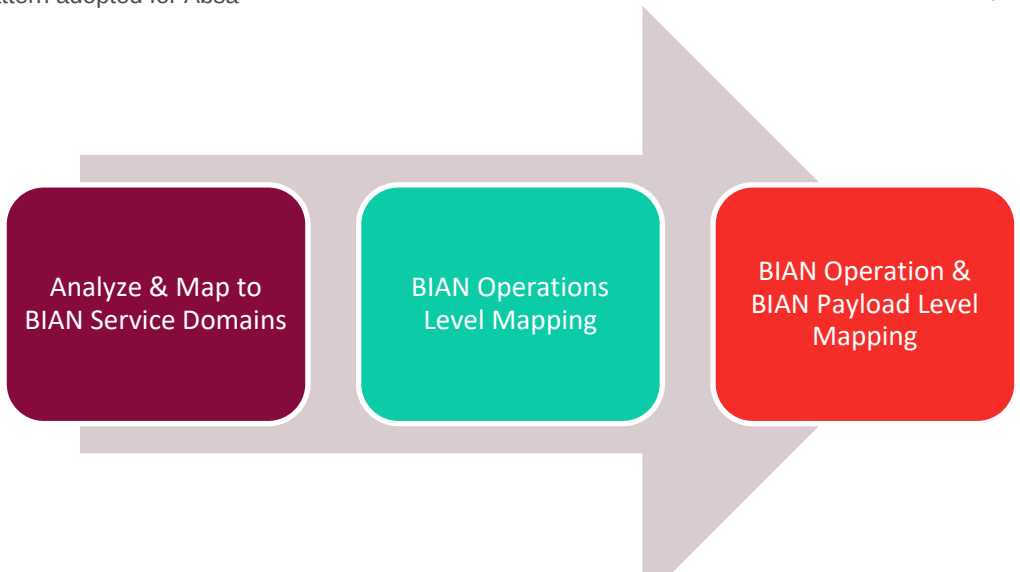
	Pre- work	Analyse	Align & Adopt
Activities	• Understand back end applications & services in scope • Identify stakeholders – All SMEs and reviewers needed for the project. • Confirm level of BIAN API adoption. • Agree on a timeline for execution.	• Understand the business processes in scope- entire process flow from channel to SORs. • Collaborate with stakeholders for thorough analysis. • Review Understanding	• For BIAN aligned DDD services, create Service URIs & Contracts. • Present to Stakeholders for review and signoff
Deliverable	• Detailed proposal (scope, execution plan, teams involved and timeline)	• Mapping of existing services to DDD based BIAN service Domains and create service catalog.	• BIAN based service specifications along with URIs and contracts .

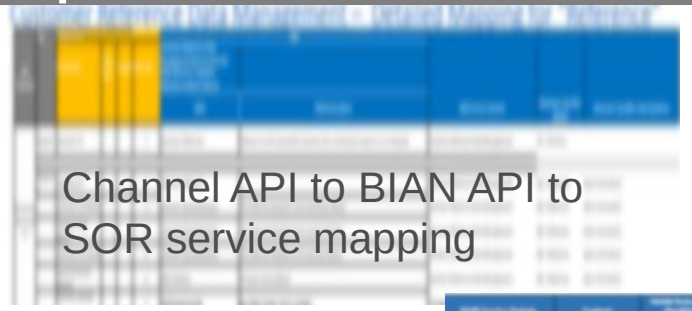
Identified the right API Adoption pattern & delivered artefacts…

API adoption pattern	Align Service Domain & URI	Align Service Domain, URI and Control record	* Align Service Domain, URI and Control record as well as BIAN BOM
Impact	Consumers will have to be made aware of changes in the URI.	Request response will have to be changed, mapping dictionary to be maintained for data elements.	Entire data model has to be aligned with BIAN.

Increasing Level of alignment

* Pattern adopted for Absa

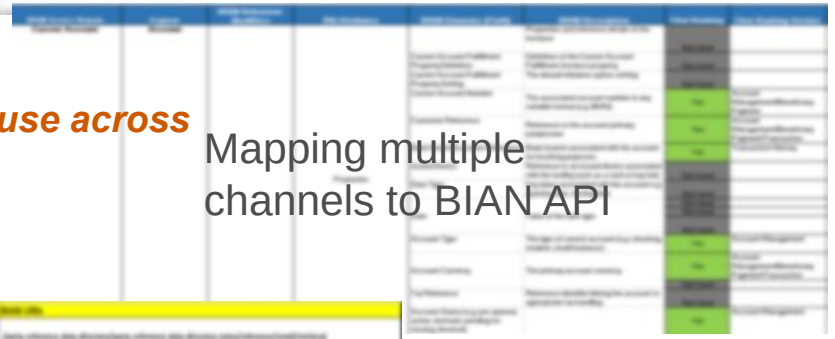




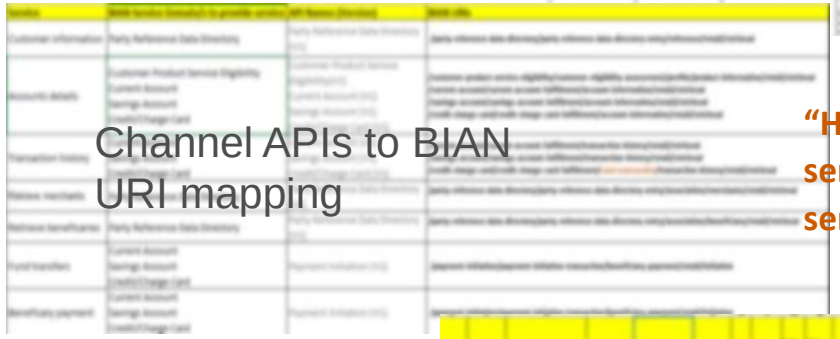
Channel API to BIAN API to SOR service mapping

"Helped consolidate multiple services providing same capabilities. Eg. 20 payment services were consolidated to 4"

"Helped identify reuse across channels "



Mapping multiple channels to BIAN API



Channel APIs to BIAN URI mapping

"Helped create future ready services, channel agnostic services "

"Helped provide the right level of abstraction thereby ensuring SOR changes are not impacting the channels."

BIAN APIs & request response mapping to SOR services request / response

Handling few challenges faced during adoption….

BIAN provided Payment initiation semantic API has this URI for initiation operation -
/payment-initiation/{sd-reference-id}/payment-initiation-transaction/initiation

While implementing for Absa we added multiple sub qualifiers for the various business scenarios as shown
/v1/payment-initiation/payment-initiation-transaction/fund-transfers/domestic/initiation
/v1/payment-initiation/payment-initiation-transaction/fund-transfers/international/initiation
/v1/payment-initiation/payment-initiation-transaction/bill-payments/initiation

Challenges

- BIAN semantic APIs will serve as reference APIs hence during implementation journey, for all the needs specific to banks we need to introduce the right sub-qualifiers in the service operations so that they are discrete and non-overlapping and rightly mirror the banks context.
- Most of the Bank's data elements were mapped to BIAN BOM, for the ones which were not available in BIAN BOM were added and Bank's own custom data model extending BIAN BOM was created. This exercise needs a lot of domain expertise as well as in-depth knowledge of BIAN BOM. It is quiet time consuming, so we have created a framework to automate some aspects of this data mapping exercise.
- Synchronizing with latest BIAN releases have to follow a well -defined process so that there is no impact to execution timeline and also ensures the latest BIAN updates are incorporated.

Sample updates in BIAN version 9

Service Domain Name Changes

- Fraud AML/Resolution => Fraud Resolution
- Credit/Charge Card => Credit Card
- Customer Product/Service Eligibility => Customer Product and Service Eligibility
- Document Services => Document Library
- Party Data Management => Legal Entity Directory
- Customer Reference Data Management => Party Reference Data Directory
- Contact Dialogue => Session Dialogue

Critical success factors that helped ….

Collaboration with all stakeholders like Business team, technology team, operations team and EA team.

Roadmap & right execution approach –start with PoC or MVP for single LoB or entire LoB before scaling at a larger level.

BIAN implementation requires a thorough knowledge of BIAN framework so we deploy a team with prior experience that regularly connects with BIAN.

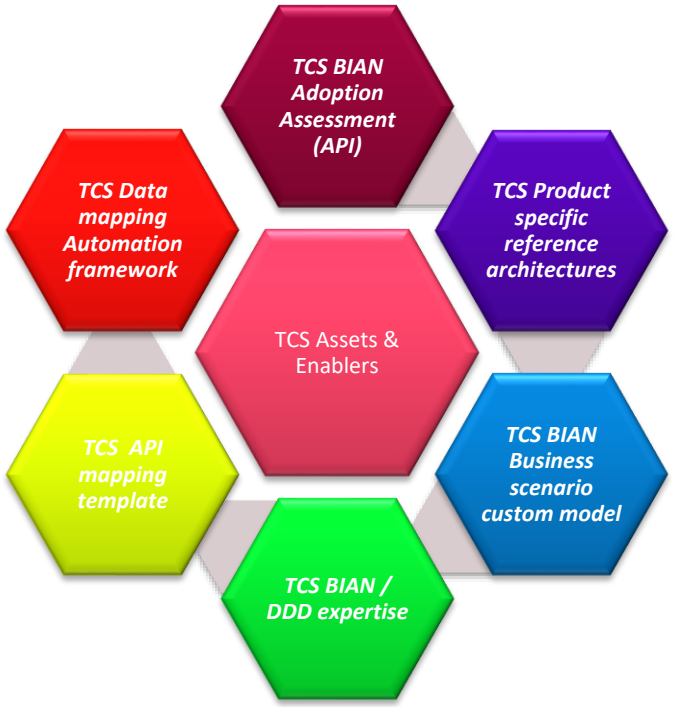
Accelerate adoption by usage of templates, assets, automation frameworks etc.

Sample feedback shared with BIAN

For a business scenario, we have a situation to handle Memo for transaction processing within a Current/Savings account service domain. We have not come across memo handling capability. Can it be added?

In BQ Association instance record of Party reference Data directory SD, we have 'Proxy/Representative/Power of Attorney Reference' element to capture associated reference details . Can we extend it to capture address, email etc.

We need a field in credit card control record to store reference to 3rd Party who acquired this customer. Can this be added?



Delivered the below benefits to the bank…

Speed To Market

- Quicker Integration due to standardization.
- Plug & play integration capabilities.
- Avoidance of Vendor –lock in.
- API catalog re-used across the enterprise leading to quicker development
- CI/CD to improve TAT
- Enabled API Marketplace business model

Increased revenue

- Capability to leverage ecosystem players leading to new revenue stream
- Seamlessly integrate with fintech and partners to deliver new features & products

Standardization & Scalability

- Standardization of architecture capability definition & information model,
- Leaner application stack, Optimized calls to the core
- Discrete Business functionalities driven by Modular service domain design
- BIAN aligned APIs with clarity of purpose
- Scalability and availability as per business needs
- Enabled Cloud readiness

Cost reduction & Reuse

- API re-use across the enterprise.
- Cost reduction due to reduced API footprint.

Customer satisfaction

- Improved customer satisfaction due to quicker launch of products.

) Thank you (