BIAN
Banking Industry
Architecture Network

Banking Industry
Architecture Network

BIAN

How-to Guide

Introduction to BIAN

Organization

Authors		
Role	**Name**	**Company**
BIAN Architect	Guy Rackham	BIAN

Status			
Status	**Date**	**Actor**	**Comment / Reference**
DRAFT	21st Feb 2014	Magnus Thorburn	Restructure, Diagrams
Approved	17th Mar 2014	Architectural Committee	

Version		
No	**Comment / Reference**	**Date**
0.90	First edited version	21st Feb 2014
0.91	Updated diagrams	27th Feb 2014
0.95	Editorial review and added comments	10th Mar 2014
1.0	Comments added	17th Mar 2014

Copyright

Table of Contents

Table of Figures

1 The BIAN How-to Guide

1.1 Introduction to BIAN

The Banking Industry Architecture Network (BIAN) is an association of banks, solutions providers and educational institutions with the shared aim of defining a service operation standard for the banking industry. BIAN's expectation is that a standard definition of the business functions and service interactions that describe the general internal workings of any bank will be a significant benefit to the industry. When compared to a proliferation of proprietary designs such an industry standard provides the following benefits:

- Enable the more efficient and effective development and integration of software solutions for banks
- Improve the operational efficiency within banks and provide the opportunity for greater solution and capability re-use within and among banks
- Support the adoption of more flexible business service sourcing models and enhance the evolution and adoption of shared 3rd party business services

BIAN refers to the collection of designs that make up its industry standard as the BIAN Service Landscape. The BIAN Service Landscape's development is iterative, relying on the active contribution of industry participants to build consensus and encourage adoption. BIAN coordinates the evolution the BIAN Service Landscape on behalf of its membership with regular version releases to the industry and seeks feedback to help continually expand and refine its content.

The main BIAN documents that make up the BIAN standard include:

- The high level BIAN reference map: the BIAN Service Landscape (and supporting service domain definitions)
- The BIAN How-to Guide series (a collection of documents targeting different audiences)
- The BIAN Metamodel and Supporting Definition
- BIAN Business Scenario Definitions
- BIAN Service Domain Definitions (these include the BIAN standard semantic service operations)

The BIAN standard is published in a UML repository, an HTML read-only version of which is freely available on the BIAN website (https://bian.org/). In addition a collection of supporting documents is maintained and released with each published version of the BIAN standard including this 'How-to Guide' series.

Note that the collection of BIAN design documents is often referred to as the 'BIAN Service Landscape' by the membership. The more formal name is the BIAN SOA Framework and it is more fully described in the second document of the BIAN 'How-to Guide – Developing Content.'

The following process is in place to receive and process your feedback:

1. BIAN members are encouraged to provide feedback by using the BIAN Wiki, to the Architectural Committee, Service Landscape Committee or via their Workgroup representatives
2. Non-members are invited to post their suggestions by using our website www.bian.org
3. Feedback can also be posted to how-to.guide@bian.org

1.2 The BIAN How-to Guide - Contents

The BIAN How-to Guide describes BIAN's working practices. It is presented as a series of three main documents in addition to this introduction and overview document:

BIAN How-to Guide - Design Principles & Techniques – This is intended for business and technical architects. It explains the theory and design practices for those wishing to understand and review the BIAN approach

BIAN How-to Guide – Developing Content – This is intended for BIAN working group members. It explains the working approach and the various tools and templates used to capture BIAN standard content

BIAN How-to Guide – Applying the BIAN standard – This is intended for members and other financial institutions wishing to apply the BIAN design content in various technical deployment situations

Each document targets a specific audience. This introduction summarises the the goals of the BIAN standard and provides a general context for reviewing the more detailed documents. It also presents an outline of the three main documents of the series so that different audiences can identify the correct document to review for their particular needs.

Disclaimer

The BIAN How-to Guide is a collection of working documents that BIAN maintains to reflect the current design principles, techniques and approaches in use by the BIAN membership. The content of the BIAN 'How-to Guide' is constantly updated as new insights are obtained and the working practices within BIAN are improved.

BIAN does not certify the accuracy or suitability of these documents for any specific purpose. The guides are informal papers that are published to support and explain the BIAN standard and to seek constructive comments/feedback from the industry.

1.3 A different Approach to a Well Established Problem

Many banking industry participants including the founding members of BIAN have frequently observed a common and enduring problem: excessive complexity in most Banks' application portfolios. This complexity results in inflexible/unresponsive systems, inflated enhancement, maintenance and operational costs; and an inability to leverage rapidly evolving advanced solutions, technologies, approaches and business models.

BIAN set out to address this issue by developing a common industry standard to define functional partitions and service operations that could be used inside any bank with the anticipated benefits already noted. However, the BIAN objective raises a key question: why should the BIAN model and approach be any better than previous attempts to address application portfolio complexity?

At the core of BIAN's proposition is the adoption of a service oriented approach to architecting the systems that support the bank. This approach is fundamentally different from the prevailing 'process–centric' designs. To underscore this critical difference a comparison can be made with architectural disciplines when applied to the highly tangible problem of designing the layout of a city as opposed to the much less tangible design of a commercial enterprise such as a bank.

Any design is a combination of the *ingredients* that are used and the *behaviours* that the design is intended to support. The ingredients relate to static or persistent things that are 'deployed' and the behaviours refer to more dynamic patterns of desired responses to anticipated events or triggers. An architect understanding how the ingredients need to be configured to support the intended behaviours will develop an overall design. In the case of the city designer, this is a town plan. The ingredients seen in the town plan are the buildings, parks and communications infrastructure that need to be in place to support the anticipated behaviours of the town's inhabitants. These behaviours could be traced as journeys or 'days in a life' on the town plan.

Comparing building architecture as practiced by the city planner and business architecture as might be used to design the applications for a bank reveals an important shortfall in the arsenal of tools for business architects. This is illustrated in diagram 1.

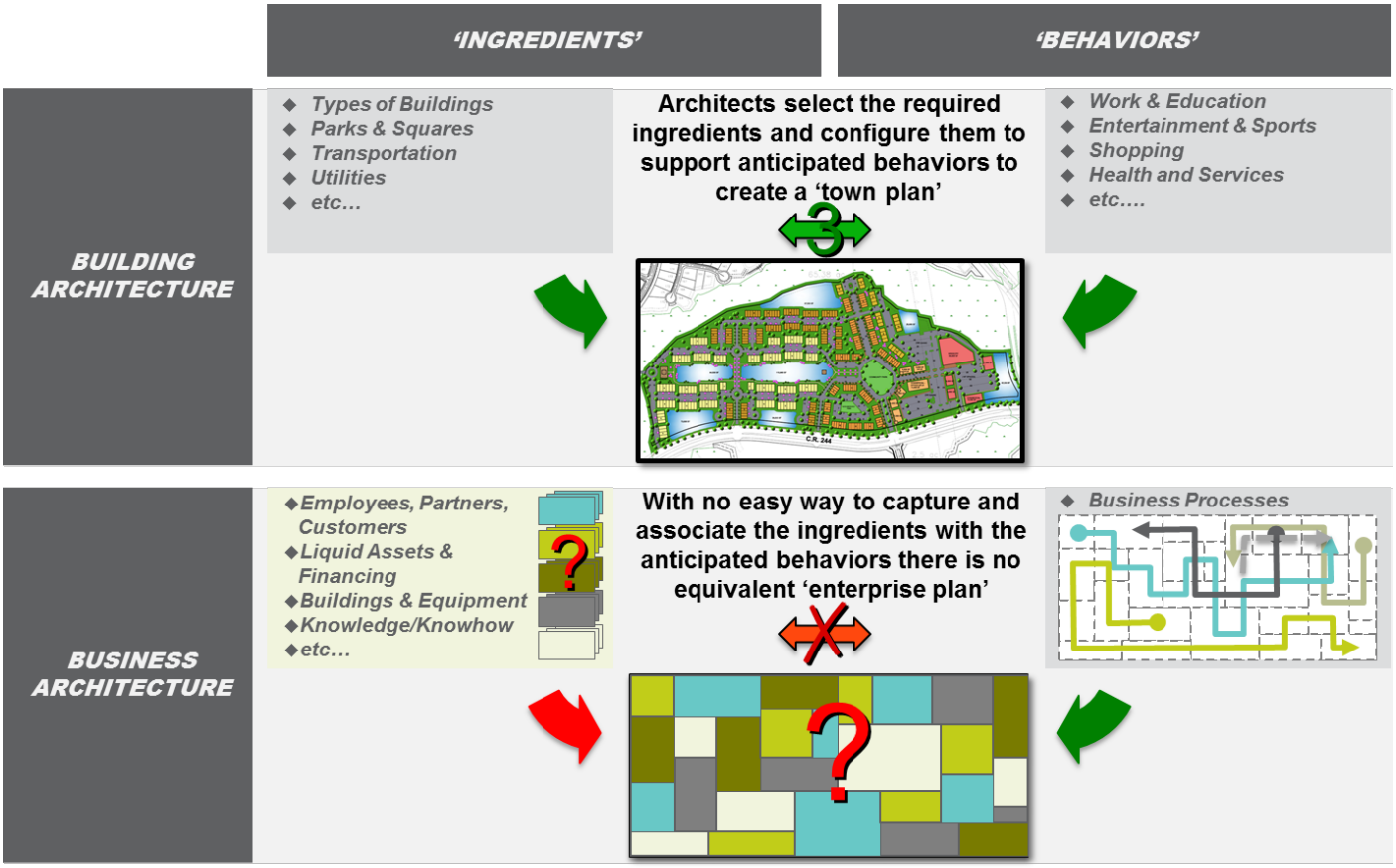

Diagram 1: Comparing business & city planning

The ingredients that make up the bank are not tangible things like buildings and roads, they are the far less tangible business capabilities that a bank must establish to be able to execute business. The behaviors that are modelled as journeys through the town are the business processes that the bank supports. Business architects have extensive experience modeling processes, but only in isolation. The gap for the business architect is defining the generic capability building blocks that they would select and configure to create the town plan equivalent for the bank. This in turn would then support those more familiar processes.

The result of building in a city without a governing town plan is a shanty town – buildings and roads are put up as and when they are needed and over time chaos is inevitable. Without a town plan for business, systems built to meet the immediate needs of the processes as they are today, over time leads to the same inevitable chaos in terms of overlapping and redundant applications, as shown in diagram 2.

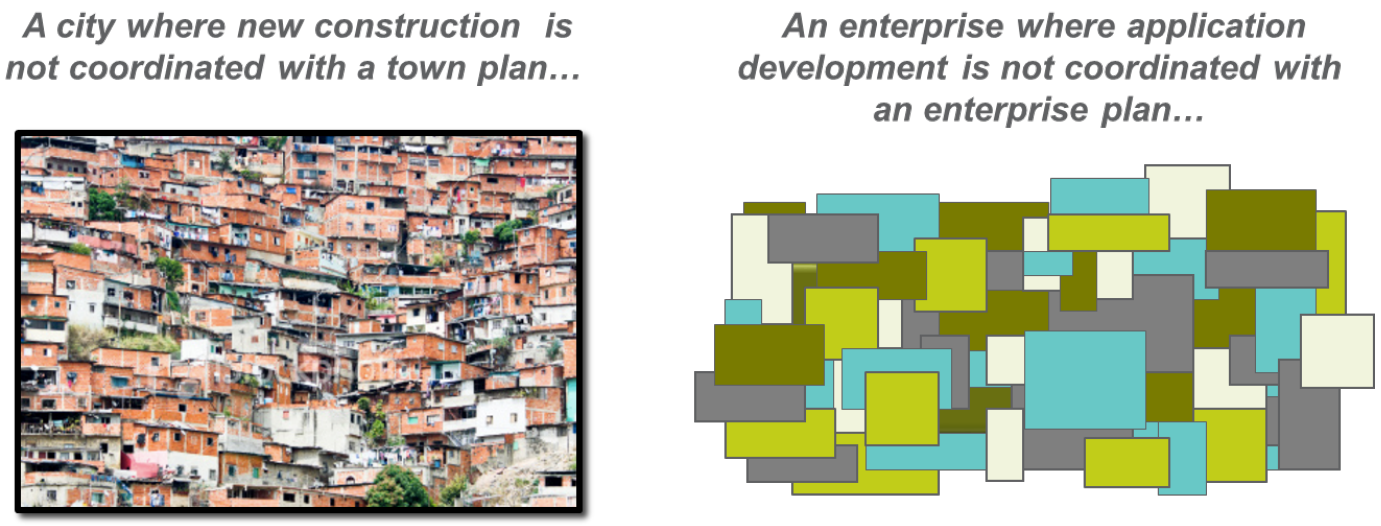

Diagram 2: Building without a plan – shanty town and application portfolio

The problem of application complexity goes much further than the obvious problem of redundancy in the overlapping applications. It is greatly exacerbated with the need for the applications to be connected to each other. As every application has its own specific scope and boundary, every connection is unique and point-to point. It doesn't take much imagination to see how as the application portfolio grows to several hundred overlapping systems (each with their own set of point to point connections), that adding or enhancing any system becomes an exercise in tracing highly complex dependencies.

The functional partitions that support the BIAN standard define discrete non-overlapping business capabilities. The BIAN Service Landscape seeks to identify all possible 'elemental' business capabilities that might make up any bank. A plan for part or all of a bank made using the BIAN partitions can provide the same organizing blueprint as the town plan – eliminating overlaps and defining standard connections. Diagram 3 shows how BIAN anticipates that as the standard is established and adopted, banks will be able to progressively rationalise their application portfolio to eliminate the redundancy and the associated operational complexity.

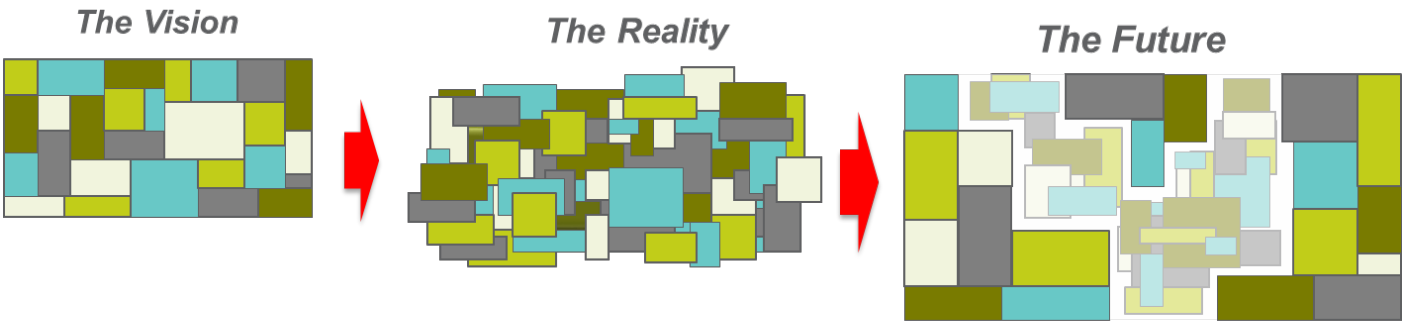

Diagram 3: Migrating to a well architected application map

The BIAN 'How-to Guide' series explains the BIAN approach to service oriented architectures (SOA) in detail. In particular it explains how banks can adopt the service based approach incrementally, targeting those areas where existing complexity is constraining the business most, or where more flexible and responsive systems are needed to exploit new business opportunities.

The three documents of the series are now described. In each case there is a summary diagram followed by brief content descriptions for the document's main sections and sub-sections.

2 BIAN How-to Guide - Design Principles & Techniques

2.1 Document Introduction

BIAN has developed an approach to business architecture design that identifies business capabilities and service operations that can make up any bank (or financial institution). The BIAN designs are 'canonical' meaning they can be consistently interpreted by any bank in many different implementation situations. In order to define canonical designs the BIAN approach has to be fundamentally different from more traditional service oriented architecture (SOA) techniques that tend to follow process-based designs and to be pitched at a more detailed implementation level.

This document describes the key design concepts and techniques employed in the BIAN approach as set out in diagram 4:

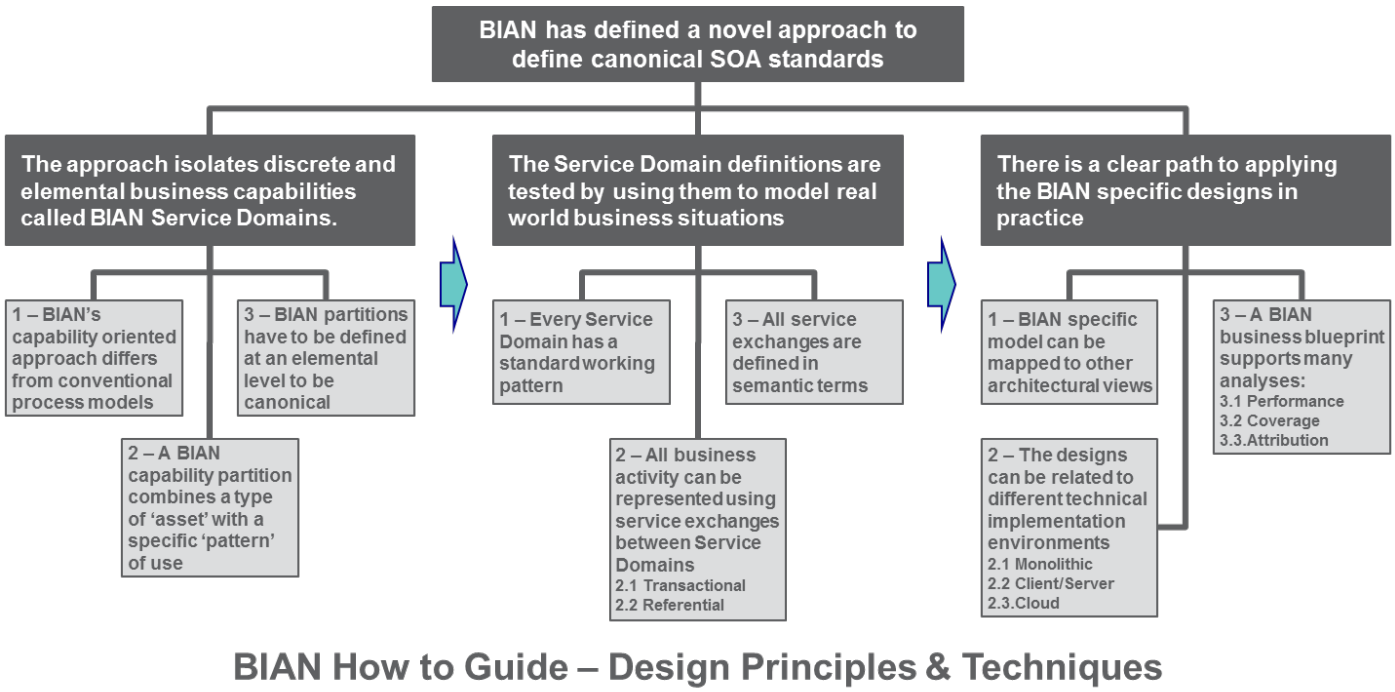

Diagram 4: Design principles & techniques content

As shown in Diagram 4, the BIAN design principles and techniques are explained in three main sections:

2.1.1 Business Capability Partitions

BIAN's approach is based on breaking all banking activity into a collection of discrete business capabilities called BIAN Service Domains. The collection of BIAN Service Domains is intended to be comprehensive so that any and all business activity can

be supported by a suitable selection of Service Domains interacting through their associated service operations. This section has three sub-sections to fully explain the BIAN Service Domain concept:

1. **Comparing BIAN's Capability View to a Process Representation** – More conventional business models use a process description of activity. A process view represents business activity as a linked series of (pre-defined) steps or tasks that are worked through to get to a specific goal. BIAN models the same business activity by identifying discrete business capabilities that need to be involved to reach that same goal without prescribing any specific sequence of interaction. An analogy can be made by considering route and town planning for a city – where a business process is analogous to charting a journey or route through the city. Conversely, BIAN business capabilities are equivalent to a town plan showing all the different types of buildings and infrastructure that could be 'visited' or 'involved' in this and indeed any journey.

2. **A BIAN Service Domain Combines an Asset and a Use** – The technique used to isolate a BIAN Service Domain defines a business capability to be the combination of a type of action or use applied to a type of asset or entity. BIAN has identified a standard list of uses (called functional patterns) and has developed a deterministic decomposition of the assets or entities (tangible and intangible) that may make up any bank. Each Service Domain combines a single primary functional pattern (for example 'maintain reference details', 'define and execute a plan') with an asset or entity type (for example 'a piece of equipment', 'a customer relationship').

3. **BIAN Service Domains are Elemental in Scope** – In order to define canonical capabilities each Service Domain must fulfill a single/elemental business role. If a Service Domain covers multiple functions then different combinations could apply in different deployment situations and the behaviors would cease to be canonical/standard. The functional patterns and asset decomposition mentioned in the previous section help identify 'elemental' business roles, but some additional considerations are necessary to ensure the designs are indeed canonical for some specific banking activities.

2.1.2 Modeling Real World Behaviors

The specification of the BIAN Service Domains are tested and refined by modeling business behaviors to test that the business function each performs is indeed well defined and discrete and to reveal the necessary interactions between Service Domains (service operations).

1. **BIAN Service Domain's Share a Common Structure** – All Service Domains fulfill a unique business purpose acting as a 'service center' providing access to their business capability through offered service operations and drawing on the services of other Service Domains as they may require. Every Service Domain has an operating pattern characterized by the handling of its 'Control Record'.

2. The Control Record reflects the combination of the Service Domain's functional pattern applied to its asset/entity type. A Control Record instance is created each time a Service Domain fulfills its role from inception to completion so for example the Service Domain with the functional pattern 'maintain reference details' for the asset/entity type 'customer relationship' will maintain a control record instance detailing the reference details for each customer for as long as they are known to the bank.

3. **Business Activity is Modeled as Service Domain Interactions** – Anything that goes on in a bank can be represented using a suitable selection of Service Domains and capturing the pattern of service interactions between them. The primary model representation captures transactional activity using the BIAN Business Scenario, similar in purpose to a high-level business process. The Business Scenario is a simple diagrammatic representation of the involved Service Domains and the archetypal flow of service interactions involved in handling/responding to a business event. A second representation of business activity currently being considered represents the background 'synchronization' activity between Service Domains.

4. **Service Operations are Defined in Semantic Terms** – The interactions between the Service Domains represent the core of the BIAN industry standard. They are described in semantic terms, covering the main business concepts involved in the interaction in sufficient detail to provide an unambiguous definition that can be consistently interpreted in implementation.

2.1.3 The BIAN Standard can be Interpreted in Different Situations

The BIAN standard is an aspect of a business architectural SOA model that can be related to different conventional technical architecture views and applied/interpreted in different technical implementation environments. The BIAN standard can be used in two broad ways – one as a high level design specification for targeted solution implementation, the other to define a stable blueprint of the enterprise for business and technical planning and analysis activities.

1. **BIAN is a Business Architecture that can be Mapped** – As a business architecture the BIAN Service Landscape bridges between the high level business model/strategy and the many underlying implementation level architectural views. BIAN's Service Landscape semantic service operations can be consistently mapped to established industry messaging standards and proprietary message definitions. The functioning and role of the BIAN Service Domain can also be related to conventional implementation level architectural views such as process and data models.

2. **Applying BIAN Designs in Different Technical Environments** – The BIAN Service Domain and associated service operations define business functional partitions and the interfaces between them. This high level specification of business behavior can be interpreted as a high level design for a range of technical environments. Three main environments are considered:

    - As a framework to better structure monolithic legacy technologies.

- As a design for aligning a client server based process orchestration capability
- As a schema for ‘container’ type service enabled partitions for advanced ‘cloud’ type technologies.

3. **Using BIAN to Define an Enterprise Blueprint** – Using the BIAN Service Domains as the building blocks of an enterprise blueprint. A key property of the Service Domain is that its business purpose/role does not change over time. The way a Service Domain works or achieves its purpose can change as practices and enabling solutions evolve, but its core business purpose is stable. As a result, a business blueprint defined using Service Domains is also highly stable and suited to different types of analysis. Three general categories are defined:
   - Setting and tracking performance
   - Mapping and evaluating coverage
   - Associating behavioral attributes to better specify requirements.

3 BIAN How-to Guide – Developing Content

3.1 Document Introduction

BIAN has established an internal organization with central oversight and support functions, and a collection of specialist Working Groups that develop the BIAN standard's content. For consistency a common approach to content development is used across all teams. This second document in the How-to Guide series describes the BIAN approach, guidelines, supporting templates and tooling. It is set out as summarized in diagram 5:

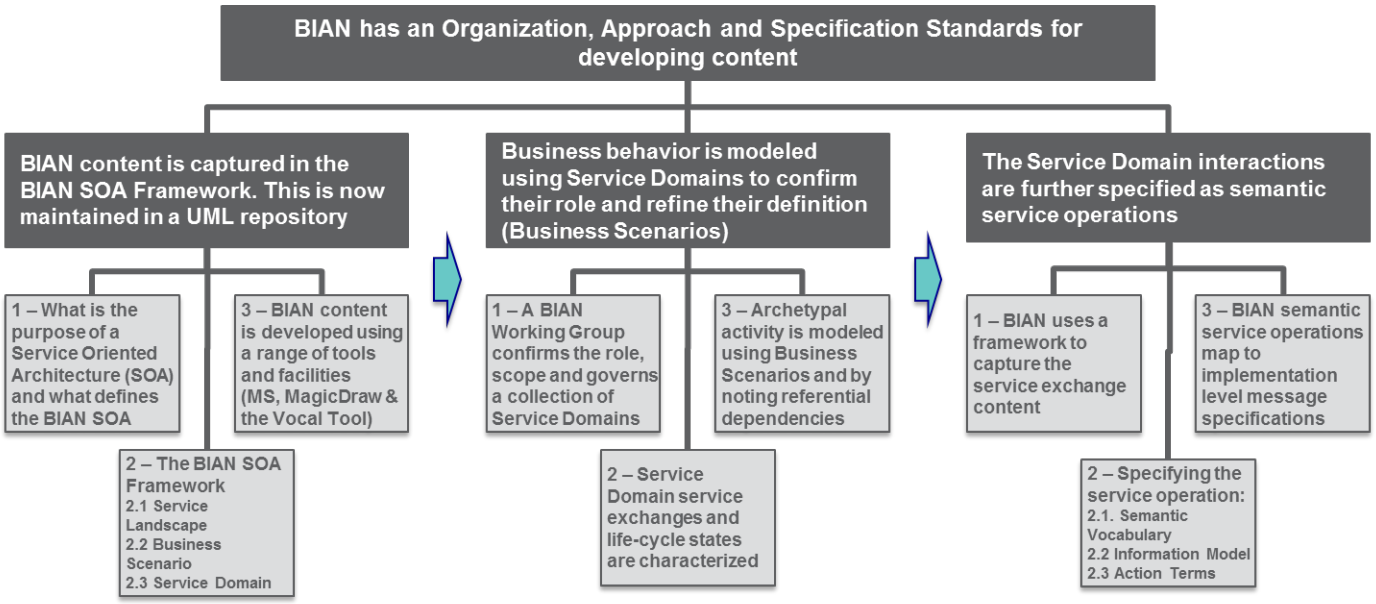

Diagram 5: Developing content

The main teams providing central oversight and support for the content development Working Groups are:

1. ***Architecture Framework & Foundation (AF&F)*** – is a unit responsible for defining the design techniques, guidelines, naming conventions and standard terms used by the content definition Working Groups. A team within the AF&F unit is responsible for developing UML based tooling and repository support for content capture. A second AF&F team provides central coordination and tool-based support for the maintenance of a semantic business vocabulary

2. ***Service Landscape*** – is a unit that oversees the general layout of the top level Service Landscape diagram and assigns definition and access rights to the Service Domains for the Working Groups

3. ***Architecture Committee*** – in addition to a range of general architectural oversight responsibilities, the Architecture Committee coordinates with the two prior named units to approve the definition of new and any updates to existing Service Domains in the BIAN Service Landscape

As shown in the above diagram the BIAN content development approach is explained in three main sections:

3.1.1 The BIAN Standard is Captured in the BIAN SOA Framework

There are common reasons for adopting of a Service Oriented Architecture (SOA). BIAN has developed a specific approach to SOA that is needed to define canonical designs (designs that can be consistently interpreted by any organization). The BIAN designs are assembled in a framework using a range of supporting tools and facilities.

1. **What is the purpose of SOA and how has BIAN applied it -** Service Oriented Architectures model business activity in a particular way that is intended to leverage service based operational approaches and technologies. BIAN has extended the general SOA design concept in order to identify generic capability partitions that can be service enabled, and more importantly that represent the elemental building blocks of any bank as is needed to establish an industry standard

2. **The BIAN SOA Framework** – BIAN's designs are captured in a framework that consists of a high level reference landscape that captures all of the generic business capabilities referred to as BIAN Service Domains. The framework also records the nature of the service exchanges between Service Domains defined as service operations that are required to support business execution. A modeling technique referred to as the BIAN Business Scenario is a mechanism used to identify and specify the involved Service Domains and associated service operations

3. **Content development is supported by tools & facilities -** Content development originally done in productivity tools such as Excel and PowerPoint is now captured in an interactive UML repository. BIAN hopes to add an integrated vocabulary tool in the near future

3.1.2 Business Behavior is modelled using Service Domains

The general behaviors that might be found in any bank are used to refine the definition of Service Domains and the interactions between them. It is important to note that the developed models of behavior (Business Scenarios) are archetypal and only used to clarify the workings of Service Domains. They are not intended to represent desired behaviors and are not part of the BIAN standard

1. **Working Groups govern Service Domains -** Each Working Group has an associated area of business expertise. The scope covered by individual Working Groups is defined in their charter so that collectively Working Groups cover the whole landscape with no overlaps between them. (Note that there are some areas of the landscape assigned to Working Groups that are not currently active.) The governance for Service Domains within a business area is assigned to a Working Group. The Working Group is then responsible for the initial specification and any subsequent updates to its assigned collection of Service Domains.

2. **Standard actions and states are associated with Service Domain exchanges** – All Service Domains have a common operating behavior: they perform a single dominant function to a single type of asset or entity – for example one might handle the 'operation' of a 'piece of equipment'. One instance of a Service Domain fulfilling its assigned business role from start to finish is managed using a pattern or structure called a 'Control Record'. The design approach is fully described in the 'How-to Guide – Design Theories & Techniques.' In this document a number of standards/checklist items are used to define the behavior of Service Domains:

   2.1 Control Record States – based on working of the Service Domain general states are defined for their control record instances

   2.2 Service operation types – a small range of exchange types can be associated with the service operations

3. **Archetypal activity is modelled using Business Scenarios** – The main mechanism used to model interactions and clarify the nature of the service exchanges between Service Domains is the Business Scenario. This simple technique identifies the involved Service Domains and the service operation exchanges associated with handling some kind of business event or transaction. The modeled flow is archetypal, using a representative example to clarify the roles of Service Domains. A Business Scenario is not intended to define a standard process but is simply one viable example of possible behavior. It is also not intended to be necessarily exhaustive or complete, it merely needs to include sufficient context to expose the targeted actions of the Service Domains being considered. The scenarios will often need to reference Service Domains that are handled by other Working Groups, and the central BIAN administration helps coordinate these dependencies to ensure the referenced service operations are developed by their respective host Working Groups.

In addition to transactional interactions, there is background coordination between some Service Domains that may not be readily represented using Business Scenarios. BIAN is developing techniques to capture this kind of 'referential consistency' service operation traffic.

3.1.3 Service Domain Interactions

Finally the specification of the service operation exchanges between Service Domains is broken down to a level of detail that unambiguously defines the interaction, and that is sufficient to match to underlying system message exchanges where appropriate

1. **A framework defines the exchange content** – BIAN uses a simple template that defines generic types of input and output parameters to structure the description of the service operation. The business concepts represented by the Service Domain's control record and the content of the information exchanged through service operations can be modeled using conventional conceptual information modeling approaches.

2. **A vocabulary and other standard terms are applied** – BIAN has defined a generic set of action terms that characterize the possible range of service operations needed to access any business capability to improve consistency in their specifications. The naming of the service operations and the content descriptions use terms that are cross referenced to a BIAN business vocabulary. BIAN has developed a semantic business vocabulary tool that is integrated with the UML based content repository to help capture and maintain this vocabulary

3. **The semantic service operation is mapped to messages** – the intent behind the semantic service operation is that it can be mapped to the underlying message specifications for machine to machine and person to machine interactions. The BIAN service operations are mapped differently depending on the technical environment and nature of the messages themselves.

4 BIAN How-to Guide – Applying the BIAN Standard

4.1 Document Introduction

The BIAN standards define generic business capability partitions (Service Domains) and their semantic service operations. In order to map these standard designs to a specific organization they need to be selected, adapted and assembled to match the operational scope and structure of organization. BIAN's high level conceptual definitions must then be mapped to more detailed implementation level technical designs. This third document of the BIAN 'How-to Guide' presents initial guidelines for applying the BIAN designs as summarized in diagram 6:

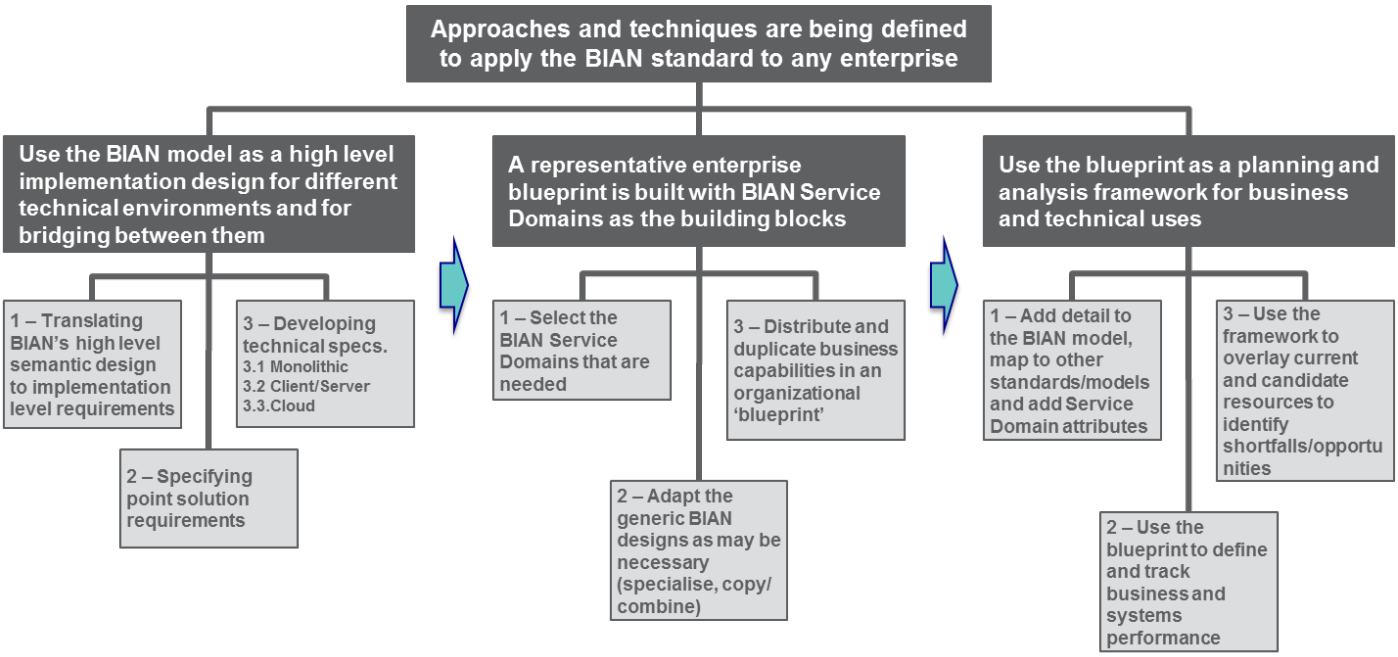

Diagram 6: Applying the BIAN standard content

The guidelines presented in this document will be continually expanded and refined based on the experiences and feedback of BIAN members and other industry practitioners. As shown in the above diagram, the evolving approach to applying the BIAN standard is explained in three main sections:

4.1.1 Using BIAN Specifications as a High Level Implementation Design

The BIAN conceptual designs provide a starting point for specifying solution implementation. This can be used to re-purpose/re-align existing systems, to select and configure commercial packages or to develop bespoke solutions. The BIAN

conceptual designs need to be interpreted differently depending on the target technical environment. Outline guidelines are provided for three discrete situations:

1. **Translating BIAN's high level semantic designs into software specifications –** the high level semantic designs need to be extended in detail to provide requirements definitions that can guide solution development

2. **Specifying point solutions** – explains a general approach for developing implementation requirements for a point solution

3. **Dealing with different technical environments –** the BIAN designs are interpreted differently for different types of technical environments

    3.1 Dealing with monolithic structures – applies typically to established host mainframe solutions

    3.2 Dealing with client service structures – applies to application orchestration/process assembly solution environments that often wrap legacy hosts systems

    3.3 Dealing with advanced 'cloud' type environments – applies to solutions built using the evolving advanced Internet and cloud technologies

Note that by aligning solutions to an enterprise blueprint it is not only possible to support development in a range of technical environments but also better to integrate solutions built in different technical environments that are aligned to the same model

4.1.2 Assembling a Representative Enterprise Blueprint

The BIAN Service Landscape contains one of each identified Service Domain organized in a reference framework. The landscape's coverage is intended to include any and all capabilities that may be required in any bank. To interpret the BIAN standard for some types of analysis it can help to first assemble a 'blueprint' of the organization using BIAN Service Domains as the building blocks.

1. **Select Service Domains that Match the Enterprise Activity** – the first step involves filtering away Service Domains that support capabilities that are not needed by the target enterprise (for example excluding some kind of product/service capabilities or some channel specific activity).

2. **Adapt the General BIAN Specifications as Necessary** – the BIAN Service Domain specifications cover the core/general actions performed. These specifications may need to be adapted to reflect the prevailing situation at a specific bank. Features of the Service Domain (and its service operations) may need to be specialized to reflect geopolitical considerations, enterprise specific scale/performance considerations or to handle unique advanced/differentiating behaviors. Furthermore the scoping decisions applied

to define a Service Domain's control record (and the 'granularity' of its operation) may need to be adjusted by either combining or duplicating and specializing Service Domains.

3. **Assemble Service Domains in a Structure Matching the Enterprise** – the final and most complicated step in assembling the enterprise blueprint involves organizing the Service Domains in a structure that matches the enterprise (or its target operating structure). This involves handling considerations such as recognizing parallel business activities (for example multiple lines of business), capturing centralized operations/activities and reflecting the legal entity/reporting structure.

4.1.3 An Enterprise Blueprint is a Framework for Analysis

The roles played by Service Domains tend to be highly stable. As a result an enterprise view built using them is also highly stable, so such a blueprint can be used as a framework for a wide range of business and technical planning and analysis activities. This representation is referred to as the M4Bank model within BIAN and is an area for future development.

1. **The BIAN Specifications can be Augmented** – The BIAN high level conceptual designs can be augmented to define a 'target state' model retaining the organizing framework of the Service Domain capability partitions and their service operations. BIAN is developing repeatable mapping techniques and has examples linking BIAN designs to established implementation level industry standards (e.g. for messaging: ISO 20022 and IFX).

   Another important use of the enterprise blueprint is to associate a wide range of behavioral attributes with Service Domains – for example comparative cost, security requirements, operational criticality. The range of possible attributes is practically unlimited and in combination the attributions provide insights into business and technical planning and evaluation.

2. **Track Business and Technical Performance** – An enterprise blueprint provides a stable and comprehensive view of the constituent parts of an organization (resolved to individual Service Domain capabilities). This framework can subsequently be used to set goals and track business and technical performance

3. **Overlay Resources to Identify Shortfalls** – As the enterprise blueprint provides a non-overlapping view of the capabilities that make up an organization it can be used to overlay resources (such as staff assignments, production applications) to identify shortfalls in coverage – e.g. gaps, duplication, misalignment.