Banking Industry Architecture Network

BIAN

How-to Guide

Developing Content

Organization

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

Status			
Status	**Date**	**Actor**	**Comment / Reference**
DRAFT	January 2018	Guy Rackham	Restructure, Figures
Approved			

Version		
No	**Comment / Reference**	**Date**
6.0	First edited version	January 2018

Copyright

Table of Contents

Table of figures

1 BIAN How-to Guide – Developing Content

1.1 Document Introduction

This is the second of three main documents making up the BIAN 'How-to Guide' series. It describes how BIAN members develop content. The intended audience for this document is the content development Working Groups within BIAN. It describes the working practices, the collection of design artifacts of the BIAN standard and the tooling support available. In particular it describes how BIAN is re-organizing to support the creation of extended Service Domain and service operation definitions. It also explores how BIAN can expand its coordination with member's own solution development initiatives to ratify the BIAN designs in practice

As BIAN evolves so do its working practices. The organization, tooling and approaches described in this guide differ significantly from the procedures described in earlier guides. The earlier guides are available on the BIAN website www.BIAN.org for reference. This guide describes the current and proposed working practices.

Some of the topics covered here from a content creation perspective are revisited in the other documents of the 'How-to Guide' from their respective viewpoints.

1.2 Document Summary

The goal of BIAN is to improve application interoperability within financial institutions (with the main focus on banks) by defining a standard description of the generic business capability building blocks or partitions that might make up a typical bank. BIAN describes the main business interactions that occur between these business capability partitions in the execution of business. More recently a focus area for BIAN has been on defining how the BIAN standard can be used to organize and help define application programming interfaces (APIs)

BIAN Service Landscape Content

BIAN has developed an approach formally to define general business partitions called BIAN Service Domains. The exchanges between the BIAN Service Domains are defined by the 'service operations' that are offered and consumed by the BIAN Service Domains. BIAN is seeking to specify all possible Service Domains with their associated service operations that may be needed to support any and all banking activity. The BIAN Service Domains are presented in a reference structure called the BIAN Service Landscape. The BIAN Service Landscape and its constituent Service Domains form a type of 'Service Oriented Architecture' (SOA).

Examples of possible business activity are represented in an informal design artifact called the BIAN Business Scenario. Alongside the Business Scenario, the BIAN wireframe is a framework that shows the key service connections between a collection of Service Domains. BIAN Business Scenarios can be traced as a flow across a suitable wireframe.

The Service Landscape, Service Domains and service operations make up the canonical BIAN standard. The BIAN Business Scenarios and wireframes merely

represent possible business behaviors and are used to explain the designs by providing business example and context. Though they are published with the Service Landscape, the BIAN Business Scenarios and Wireframes are not technically part of the standard and in particular are not intended to represent standard/prescribed behaviors.

BIAN Content Development

With its latest release BIAN is starting to extend the definition of the Service Domains and service operations adding an additional level of detail in order to provide more comprehensive content. This has proven to be necessary to ensure the role/definition of the Service Domains and their service operation connections can be consistently interpreted in different deployment situations. Sample extended definitions have been developed and initial ratification has been gained in coordination with BIAN semantic API initiative. These have been included in the latest release.

In this document BIAN's approach to SOA is explained. Descriptions are provided for the key content of the BIAN Service Landscape release: the BIAN Service Landscape; the BIAN Service Domain with its underlying service operations; and the explanatory BIAN Business Scenarios and BIAN Wireframes. BIAN maintains a comprehensive UML model of all of its designs and design concepts to ensure their integrity and to facilitate tooling support and content export.

BIAN also maintains an integrated business vocabulary for BIAN specific terms and has recently embarked on creating its own business object model (BOM) to define the service operation information content. BIAN is also building out and exploring the use of more flexible presentation/browsing tools. These facilities are cross-referenced as necessary throughout this document but are fully described in other dedicated BIAN documents.

BIAN has continued to develop and ratify a comprehensive collection 'First Order Interactions' across the landscape. A First Order Interaction is a simple/constrained form of business scenario. It shows how a selected Service Domain (the primary Service Domain) responds to a business event in terms of the Service Domain that might call it triggering its response to the event and any Service Domains it may delegate to in order to be able to handle the event. Over 1000 provisional First Order Interactions were published in the prior Service Landscape release and many of these have since been ratified and new First Order Interactions added with the latest release.

This main use for First Order Interactions is to identify the required service operation connections between Service Domains. These patterns of connections are used to help with the development of more complete/sophisticated Business Scenarios and Wireframes. The current procedures BIAN uses to develop First Order Interactions, to generate more detailed Business Scenarios and Wireframes are set out in this guide.

The layout of the remainder of the document is shown in the next figure

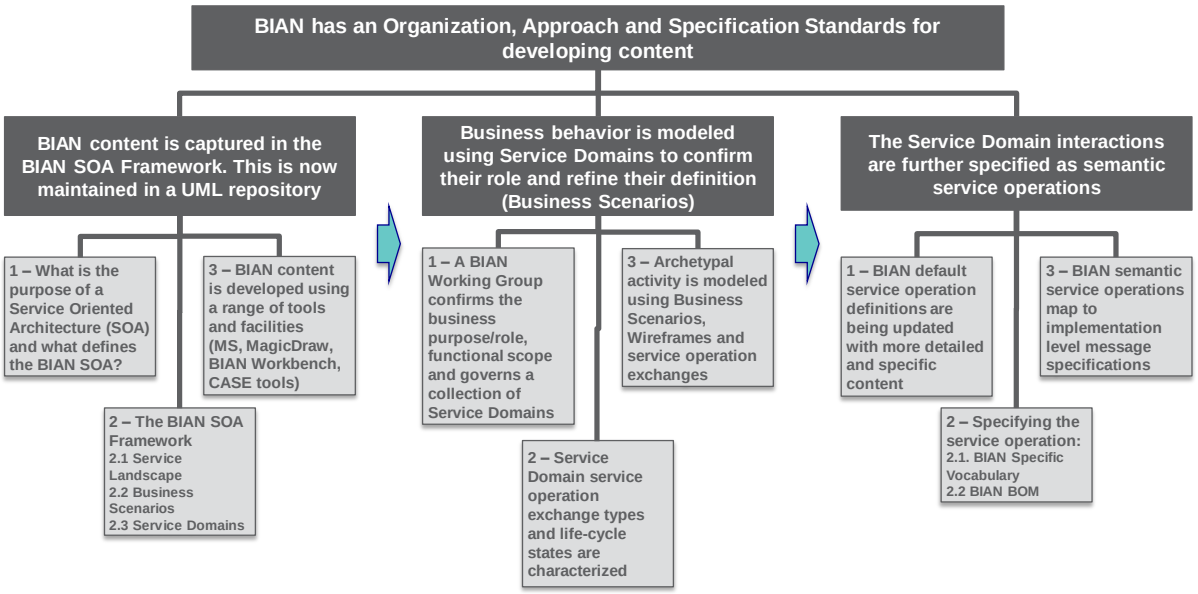


Figure 1: Developing Content

The BIAN content development approach is explained in three main sections in this guide.

The BIAN Organization – BIAN is a non-profit body staffed mostly by individuals from its member banks and solution providers with a small team responsible for various centrally managed activities. The make-up of the BIAN organization and the BIAN specific approach towards service oriented architecture is outlined

The BIAN SOA Framework – BIAN's content is captured in a structured set of design documents referred to as the 'BIAN SOA Framework' or less formally by the top-level representation, the 'BIAN Service Landscape'. This includes the Service Landscape, Service Domains, their service operations and the known service operation connections (first order interactions). It also includes less formal design artifacts including the business scenario and wireframe. The theory behind the design artifacts are more fully explained in the How To Guide – Design Principles & Techniques. Their descriptions are included in this guide for ease of reference

Content Development – Describes the specific content creation approaches employed by the Working Groups and in particular explains the working approach for the BIAN Semantic API Initiative.

2 The BIAN Organization

This Section describes the BIAN Organization, describes how BIAN approaches Service Oriented Architecture and outlines the membership's general working approach to developing content. The current tooling and support capabilities are outlined along with a brief summary of past and future working approaches.

2.1 BIAN Organization

The BIAN organization structure is more fully described on the BIAN website (BIAN.org). The roles of the key central support teams and the content development approach are outlined here. Content is developed through a coordinated effort between the central team and specialist content Working Groups. Each Working Group has a specific area of business expertise.

The central support and oversight teams as follows:

Architecture Framework & Foundation Team (AF&F) – is responsible for defining and supporting the design principles and techniques used to develop BIAN content. This includes defining standard terms and formats that are used in the specifications and the maintenance of the BIAN metamodel. AF&F is also responsible for the implementation and operation of the BIAN content repository and a specialist sub-team of AF&F maintains the BIAN business vocabulary, the BIAN business object model and all associated tooling/environments.

Service Landscape Team (SL) – is responsible for the layout and content of the BIAN Service Landscape. The Service Landscape Team also assigns Service Domain 'governance' responsibilities to Working Groups and is responsible for ensuring the completeness and consistency of the landscape. The Service Landscape team is currently merged with the AF&F team.

Architecture Committee (AC) – is a group of member senior architects with review and acceptance authority for content before it is sent to the BIAN Board for final sign-off and publication. The AC reviews all Service Landscape amendments and new submissions before changes are approved. The AC is supported by a specialist Quality Assurance (QA) team.

Central BIAN Resources – the Working Groups are supported on a day-to-day basis by the BIAN program manager, technical architect team, content generation support resources and the BIAN administrator as necessary. Additional specialist resources are engaged as needed.

The BIAN Board – a senior group of Banking and Solution provider representatives is elected by the membership to oversee and approve all business and technical activities

2.2 What is the purpose of SOA and how has BIAN applied it?

Some of the general benefits of adopting a service-oriented architecture (SOA) are outlined by the Open Group. Below some of the key benefits they describe are paraphrased, (note this list provides a few interpreted highlights for indicative purposes only, the Open Group's own publications should be referenced directly for their complete explanation - www.opengroup.org.):

- *Service* – the adoption of services in the systems architecture can improve information flow, help expose otherwise embedded or hidden functionality and result in greater organizational flexibility.

- *Service re-use:* service based software leads to lower software development and management cost.

- *Messaging* – has a wide range of positive impacts including configuration flexibility, better monitoring and business intelligence, greater control and security.

- *Complexity and Composition* – services can simplify the software structure enabling the development of more complex, adaptive and more easily integrated software solutions.

The many benefits described by the Open Group for SOA tend to relate most directly to the efficient development, resulting performance and 'fit-to-purpose' of software solutions. BIAN has extended the SOA design concept into the realms of business architecture. The BIAN approach further constrains the specification of the business functional partition that is then 'service enabled' in a number of key ways:

- *BIAN Service Partitions are Discrete* – the business role/purpose of a service partition is unique, non-overlapping and discrete. If in fulfilling its specific purpose a service partition needs to access other business capabilities, it does so through calling the services of other service partitions as necessary.

- *BIAN Service Partitions are collectively comprehensive* – BIAN seeks to define a complete set of service partitions such that any and all possible banking activity can be represented by a suitable selection of service partitions collaborating through service calls as necessary.

- *BIAN Service Partitions are 'elemental'* – the business role or purpose supported is a single business purpose or function, in simple terms BIAN service partitions are not made up of smaller service partitions, they form a comprehensive 'peer set'.

BIAN refers to a service partition as a BIAN Service Domain. The fairly complicated design techniques required to conform to the above design principles are briefly outlined later in this document and explained in detail in another document of the 'How-to Guide – Design Principles & Techniques.' In essence a BIAN Service Domain more closely represents a business or operational capability partition rather

than a software utility and as a result the BIAN SOA provides additional opportunities when used better to align the underlying business applications:

- *Service Domains can be widely accessed increasing operational capability re-use*: through its offered service operations the business capabilities supported by a Service Domain can be accessed across the enterprise to maximize operational re-use. It can potentially concentrate scarce and/or specialized resources improving utilization/leverage for the enterprise.

- *Increased operational flexibility*: as more business capability building blocks are made available through shared services, changing business needs can more readily be supported through service realignment/re-use. Furthermore as solution providers and financial service organizations align increasingly to a common definition of operational service partitions their underlying solutions can be more easily integrated and flexible sourcing arrangements and business models that leverage them can better be supported across the industry.

- *The reduction of business information inconsistencies and fragmentation:* the partitions are each intended to fulfill a unique and discrete business role and so act as the single source for the services they provide and the business information that they 'govern'. The BIAN SOA therefore provides insights into function and information governance and usage that can be used to reduce inconsistency and fragmentation and rationalize service and message use across an enterprise.

- *Performance can be optimized*: because each service partition fulfils a narrowly defined business purpose its operational performance and supporting systems can be internally optimized.

BIAN Service Domains are defined independently of any particular organizational structure and technical implementation approach. A specific business enterprise can select and assemble service domain designs from the landscape in the BIAN SOA as needed to support its own particular processes and organizational layout and the BIAN designs can be interpreted for implementation in different prevailing technical environments (as described in more detail in another document, the 'How-to Guide – Applying the BIAN Standard.'

2.3 Content development is supported by tools & facilities

The design elements of the BIAN SOA Design Framework are supported and enabled by some key standard specifications, guides and tooling. These include:

1. The BIAN Metamodel
2. The BIAN UML Repository
3. The BIAN Business Vocabulary
4. The BIAN Business Object Model
5. The BIAN Business Scenario Generation and Browsing Tool
6. BIAN How-to Guides and other training materials

7. The BIAN Wiki and Working Group tools and facilities.

2.3.1 The BIAN Metamodel

The BIAN Metamodel captures all of BIAN's design and specification concepts in a central UML model. The model is fully documented in its own How-to Guide available on www.BIAN.org. Points of note include:

- *ISO 20022 compliant* – the BIAN Metamodel is an extension of the Metamodel defined by the ISO 20022 financial services industry standard. The BIAN Metamodel adds features to support BIAN's unique semantic designs of the Service Domain and associated structures. By virtue of the fact that the BIAN Metamodel extends the ISO 20022 Metamodel, BIAN adopts the ISO 20022 structures for lower-level design elements, avoiding the need to create additional, potentially conflicting definitions.
- *Alignment for Tooling* – by ensuring all BIAN design concepts align to the BIAN Metamodel, options for providing tooling support are enabled for the range of Service Domain related designs and standards, including the BIAN Business Vocabulary. BIAN currently maintains the standard in a MagicDraw UML repository
- *Cross-Standard Mapping* – the BIAN Metamodel defines structures that support mapping between different vocabularies (both standard and proprietary). This feature is revisited with the outline of the BIAN Business Vocabulary that follows shortly.

2.3.2 The BIAN UML Repository

BIAN has implemented a UML repository using No Magic Inc's MagicDraw UML repository product. All of the BIAN SOA Framework content is now maintained in this repository. It is accessible to members (with suitable version and access management capabilities) and is used to generate a 'read only' HTML version of the published Service Landscape for a wider audience. An Excel extract of the content is also available exclusively to BIAN members

Training materials, usage guidelines and content description for the repository is documented elsewhere, though example references to content are made extensively throughout the BIAN 'How-to Guide' series.

2.3.3 The BIAN Vocabulary

The BIAN Vocabulary is used to capture all BIAN specific terms. BIAN's vocabulary capabilities are fully documented in their own guides available on www.BIAN.org. A comprehensive semantic business vocabulary is needed to support the consistent interpretation of Service Domain and service operation specifications and naming conventions.

The BIAN vocabulary has extended concepts defined in the ISO 20022 model. The vocabulary tool is integrated with the BIAN UML repository. This is to enable BIAN Working Groups to link and cross-reference all content to the vocabulary definitions.

2.3.4 The BIAN Business Object Model (BOM)

The BIAN business object model (BOM) is intended to define specifications of the business information content for the Service Domains and service operations. BIAN's policy is to adopt established industry standards where available but at this time there is no single comprehensive business information model covering the banking/finance industry.

BIAN hopes to assist the evolution of an industry standard model through providing a facility that allows BIAN to map to prevailing terms and introduce new terms as necessary. The semantic definitions of terms maintained in the BOM facility can be used to track and display the similarities and differences between equivalent terms used in selected existing information models.

BIAN has started to develop its own business object model (BOM) that covers the business information content governed by the Service Domains and exchanged through the service operation connections. Most importantly, the BIAN BOM is 'informed' by the prevailing industry standard ISO 20022 BOM. A BIAN Working Group is responsible for the coordinated development of the BIAN BOM in support of the content development Working Groups. The BIAN BOM is mapped to the ISO20022 model at the level of the Service Domain control record and any related additions and links needed to align the two models are being documented and fed back to a dedicated team at ISO to help coordinate.

2.3.5 The BIAN Business Scenario Generation and Browsing Tool

BIAN has developed an internal tool to assist with the content development – the 'BIAN Workbench'. This tool helps with the recording of Service Domain business events and First Order Interactions. An extension also supports the capture of more complex business scenarios. The tool has been designed to be able support future enhancement to support more flexible review/browsing of the Business Scenarios and service operation connections as might be shown in a wireframe model. This facility may be of use to members and non-members when they reference the standard.

A decision whether to further invest in the Workbench as a browsing/reference facility is likely to be made during 2018.

2.3.6 The BIAN How-to Guides

This document and others in the How-to Guide series are intended to provide the main reference overview of the BIAN approach for both internal BIAN and external audiences. BAIN also maintains training materials in the form of presentations and recorded sessions for its membership.

The ‘How-to Guide’ series is a collection of working documents that is updated with each major release of the Service Landscape as BIAN develops and refines its working practices. With this release an additional guide has been added to the collection – the BIAN Semantic API How to Guide.

2.3.7 The BIAN Wiki and Working Group tools and facilities

BIAN operates a range of web-based team working and publication facilities for use by its members and the general community. These facilities are constantly being refined as new design and publications tools are made available. The BIAN external website, internal WIKI and the many internal Working Group tools and facilities are fully described in their own supporting documents available on the BIAN WIKI.

2.4 Summary Goals and Approaches for the Next Cycle

The latest BIAN Service Landscape Release (V6.0) continues with the rapid expansion of content. In particular the release introduces an additional level of detail to the BIAN specifications with specific focus on the use of the BIAN standard to support API development. Key aspects of the release include:

- Extensions to the Service Domains and service operation specifications with the introduction of ‘behavior qualifier types’ – a way to further break down the Service Domain’s behavior as defined by its functional pattern. The additional detail is used primarily to add specificity to the purpose of service operations and their semantic information payload
- Introduction of and initial development of a BIAN business object model BOM. Cross referenced to the industry standard ISO20022 model where appropriate
- Additional First Order Interaction definitions and ratification of a proportion of the provisional definitions published in the prior release
- Additional Business Scenarios
- Specific content in support of the initial design cycle for BIAN Semantic API initiative (Wave 1)

Though not included in the latest release, there has been significant progress in a number of architectural alignment and other additions to the BIAN standard. Two ongoing activities of particular note that will impact future releases are:

- ***BIAN’s business capability model*** – augmenting the Service Landscape, the business capability view will provide an access path to the BIAN standard more suited to business practitioners that may not find business activity representations limited to Service Domains the most intuitive
- ***Vendor Agnostic Application Architecture*** – a Working Group is exploring how the business activity related BIAN Service Domain designs can be best related to application architectures with emphasis on commercial application development

The content development activity for the coming cycle divides between several threads of related activity:

- On-going development of extended BIAN specifications to support the BIAN Semantic API Initiative. The approach for this is described in the last Section of this guide
- In coordination with the Semantic API Initiative there will be on-going expansion of the BIAN BOM, including activities to ensure mapping back to the ISO20022 industry standard is defined where appropriate
- On-going expansion of the First Order Connections – as described earlier a key aspect of the BIAN specification is identifying the allowed service operation connections between Service Domains. Provisional first order connections will continue to be defined across the Service Landscape by the central team. These provisional definitions are later ratified when they are used to assemble business scenarios representing known business activity or that are used in solution development (including API work). The approach for defining First Order Interactions is outlined next in this guide
- Continued development of the BIAN Vendor Agnostic Application Architecture and Business Capability view

3 The BIAN Standard is captured in the BIAN SOA Framework

The BIAN SOA Framework is an integrated set of design artifacts that are captured in the BIAN UML repository. In this section the main components of the BIAN SOA Framework are described for reference. The associated design concepts are described in more detail in the How To Guide – Design Principles & Techniques. The components of the Framework are presented here in MS Office document form (Powerpoint/Excel) and/or as report views generated from the BIAN Workbench and BIAN UML repository as appropriate

3.1 The BIAN SOA Framework – an overview

BIAN's goal is to define standard semantic service operations with an initial stated emphasis on the internal operations of any bank (as opposed to inter-bank exchanges) in order to help improve the bank's internal operational performance. With BIAN's Semantic API Initiative the scope of interest has arguably been expanded to include the externally accessed banking services. Indeed since the formation of BIAN the distinction between internal and external exchanges has become increasingly blurred. BIAN standard service operations are also intended to cover any and all types of business service exchange with specific focus on those aspects of the exchange achieved through some aspect of information systems involvement/support.

The BIAN SOA Design Framework is the working name given to the collection of formal and informal BIAN design artifacts used to specify the semantic service operations that make up the BIAN standard. It is more commonly referred to within BIAN by the name of its highest-level design component: the BIAN Service Landscape.

The BIAN SOA Design Framework is intended to cover all of the business capabilities any bank might employ. (Typically any one bank will only need a subset of this collection, for example because it supports only certain banking products). BIAN has developed a design rationale and supporting techniques to break up banking capabilities into non-overlapping partitions called BIAN Service Domains. The Service Domain is the fundamental building block of the BIAN standard. The collection of Service Domains is arranged in a reference framework called the BIAN Service Landscape.

Any and all possible banking business activity can be modeled as a pattern of collaboration involving service exchanges between a suitable selection of Service Domains taken from the Service Landscape. BIAN models examples of Service Domain interactions using an informal representation called the BIAN Business Scenario. The BIAN Business Scenario is used to clarify the roles of BIAN Service Domain and their service exchanges by providing a contextual illustration of behaviors. The BIAN Business Scenario is not a formal design but merely an archetypal instance of one possible pattern of collaboration.

Another view of the Service Domain interactions is the BIAN Wireframe. The Business Scenario models the interactions as they relate to a specific requirement or event, similar in many ways to a conventional process view. The Wireframe conversely shows all allowed/known service connections between a selection of Service Domains in a static framework. Business Scenarios can be traced as instances of flows that traverse this framework in the same way a journey can be charted on a map.

The exchanges between Service Domains are modeled as 'service operations' that are offered and consumed. In practice a Service Domain can be involved in any number of business scenarios but because it always fulfills a unique and discrete business purpose the service operations it offers (and consumes) is defined as a finite or 'bounded' collection.

The BIAN Service Domain's specification contains the semantic definitions of all of its offered service operations and provides references to the service operations it consumes from other Service Domains, along with an outline of its business purpose or role. The Service Domain semantic service operation specifications define the core of the BIAN industry standard.

BIAN's Standard Semantic Service Operations are 'implementation agnostic'

Though BIAN's goal is to improve application interoperability, in order to be canonical (i.e. consistently interpretable by any bank in any technical environment) the specifications themselves must be system implementation agnostic. The BIAN service operation specifications include nothing specific to nor reliant on some feature of any particular technical architecture or solution approach. BIAN does however provide guidelines as to how its standard can be applied/interpreted in different prevailing systems architectures. These are presented the third document of the 'How-to Guide series – Applying the BIAN Standard.'

A service operation's specification defines the Service Domain exchange in narrative terms as might be described and understood by a business practitioner. With prior releases the service operation content was defined using checklists of possible information content that was pre-filtered based on specific design properties of the Service Domain (its 'control record') and the service operation itself (its 'action term'. These Service Domain design concepts are explained later in this guide.

With the latest release BIAN has made two significant changes to the way it defines the service operations:

1. ***Additional Detail*** – the *functional pattern* that defines the main behavior of a Service Domain has been further broken down to define 'behavior qualifiers'. This additional level of detail is used to enhance the specification of the Service Domain's working, its offered service operations and the business information it governs as outlined later in this Section
2. ***BIAN Business Object Model (BOM)*** – BIAN has embarked on the definition of a Business Object Model that provides Service Domain specific definitions of the key business information content governed and exchanged. The BIAN

model is cross referenced to the industry standard ISO20022 where appropriate

In addition to the BIAN BOM, BIAN continues to maintain its own vocabulary of definitions for BIAN specific terms.

The BIAN service operation specification is intended to be sufficiently comprehensive such that where appropriate it can be interpreted for supporting systems implementation without the need to specify additional business requirements. (Note that significant additional design effort will typically be needed to translate BIAN's high-level semantic service operations into the far more detailed code level message designs.)

The BIAN Service Landscape covers all banking activity including Service Domains typically having a high dependency on underlying systems support and others that have minimal need for highly integrated information systems. Given BIAN's priority on improving application to application interoperability, the designs and associated techniques described relate primarily to Service Domains with a high systems dependency.

Each artifact of the BIAN SOA Framework is now described in more detail. Note that these artifacts can be presented in different formats: as documents in standard productivity tools (e.g. Microsoft PowerPoint, Excel & Word); and as structured report extracts from the BIAN Workbench and UML MagicDraw repository. Different formats are shown for ease of presentation and explanation in this document. The artifacts that make up the BIAN SOA Framework or 'Service Landscape' described below are:

1. The BIAN Service Landscape
2. The BIAN Service Domain & Service Operations (High Level)
3. The BIAN Business Scenario
4. The BIAN Wireframe
5. Service Domain Service Operations (Detail)

3.1.1 The BIAN Service Landscape

The BIAN Service Landscape is a reference framework that contains all identified BIAN Service Domains. Its purpose is to provide a mechanism for quickly identifying and selecting Service Domains. The landscape uses a hierarchical decomposition of general banking industry capabilities at three levels as described below. As noted, it is the goal of BIAN that the BIAN Service Landscape will eventually contain all possible Service Domains. Any and all business activity can then be represented by a suitable collection of one or more Service Domains working together in collaboration. BIAN uses a primary 'Matrix' Service Landscape view based on agreed categorizations that have been refined in use over several years by the BIAN membership.

Note that the BIAN Metamodel is a detailed and comprehensive (UML) model that defines all of the BIAN design structures – it is fully documented elsewhere in its own

guide (The BIAN Metamodel). The Metamodel has three elements that detail the structure of the BIAN Service Landscape.

1. *Business Area* – is the highest-level classification. A *business area* groups together a broad set of business activities/capabilities. In the case of the BIAN Service Landscape they are defined to be aspects of business activity that have similar supporting application and information-specific needs. In cases business areas may contain finer grained/nested business areas

2. *Business Domain* – at the next level, *business domains* define a coherent collection of capabilities within the broader *business area*. In the BIAN Service Landscape the *business domains* are associated with skills and knowledge recognizable in the banking business.

3. *Service Domain* – is the finest level of capability partitioning, each defining a unique and discrete business operational capability partition. The Service Domains are the 'elemental building blocks' of the service landscape

The Service Domain relates to generic capabilities that do not vary in their scope, but the definitions of the Business Domain and Business Area are classifications that are specific to a particular Service Landscape layout. The BIAN Service Landscape Version 6.0 presents two Service Landscape layouts:

- *The historical 'Matrix' Service Landscape* – this is the tabular layout that BIAN has evolved over recent years. There have been changes over time in the scoping of some Business Domains to reduce anomalies with product categorization
- *The 'Model Bank' or 'Value Chain' Landscape* – this is a complete re-scoping of the Business Areas and Domains and a reformatting of the layout to better align to an organizational classification of Service Domains. This format is used in the development of a bank 'enterprise blueprint' as explained in the How-to Guide – Applying the Standard

The original 'Matrix' Service Landscape Business Areas have been defined corresponding broadly to types of systems use. It has the following Business Areas working from left to right:

- *Reference Data* – contains Business Domains (and their contained Service Domains) that handle access to both internally and externally sourced information that is widely accessed by different parts of the business
- *Sales & Service* – brings together Business Domains that support the interactions with the bank's customers through all channels for the purposes of selling and servicing in-force products and services
- *Operations & Execution* – is a large area that combines all transaction processing oriented aspects of product and service fulfillment, including product specific activities, 'vanilla' capabilities that can be integrated within many products and shared supporting operational services. Due to its

large size, this business area is subdivided into two regions. One corresponding to groups of product specific activities and the other to shared product fulfillment support capabilities

- *Analytics & Risk* – consolidates the Business Domains that support and perform detailed analysis functions. These cover product and customer related analyses, business unit performance assessments and all dimensions of risk (e.g. credit, market, instrument, operational & compliance)
- *Business Support* – combines the wide range of general management and support activities common to most enterprises, including the executive, finance, staff, systems and facilities

Within these five broad Business Areas, the approximately 40 more finely grained Business Domains represent generally recognizable banking functional groups. The approximately 300 'elemental' Service Domains have then been mapped into this two/three tiered reference framework based on their specific business roles, as shown in the figure below:

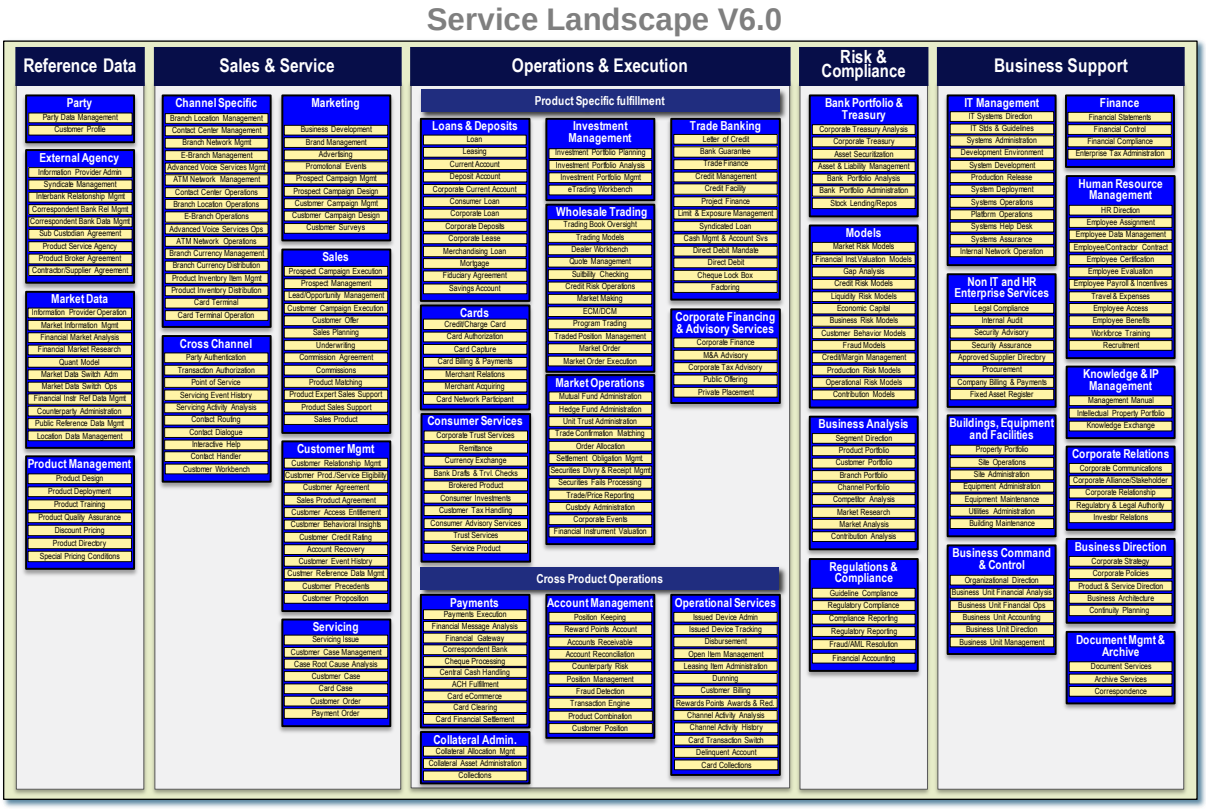


Figure 2: BIAN Landscape V6.0

High-level Service Domain Definition

Standard high-level definitions are maintained for each Service Domain in the UML repository (some of the technical terms described here are revisited later in this document and more fully explained in another document of the 'How-to Guide - Design Principles & Techniques'):

- *Name* – the descriptive name of the Service Domain.
- *Business Role* – a brief description of the business role
- *Example of use* – a brief example of some business event/context that involves the Service Domain
- *Control Record* – a control object or 'pattern' that is used to track an instance of the execution of the Service Domain's business role from start to finish.
- *Functional Pattern* – the dominant type of business function performed Each functional pattern has an associated *generic artifact*. The artifact represents the type of document or record that might be used to manage/track the execution of the function. In the latest release the functional pattern has been further decomposed to define its 'behavior qualifiers' as described later
- *Asset/Entity* – the business asset or entity type that the Service Domain acts upon in the manner as characterized by its associated functional pattern
- *Comment* – general clarification of the Service Domain's role and any open considerations arising from BIAN's internal design discussions.

The primary use of the Service Landscape is as a reference framework to organize the full collection of Service Domains. The current layout has been driven primarily by the need to discover and develop content within the BIAN membership.

Different criteria to those just described can be used to define alternative Business Domains and Business Areas and create different layouts (that typically also contain the complete collection of identified BIAN Service Domains). One such alternative arrangement noted above – the *'value chain view'* has been developed to support deployment of the BIAN standard and is shown in the the next figure:

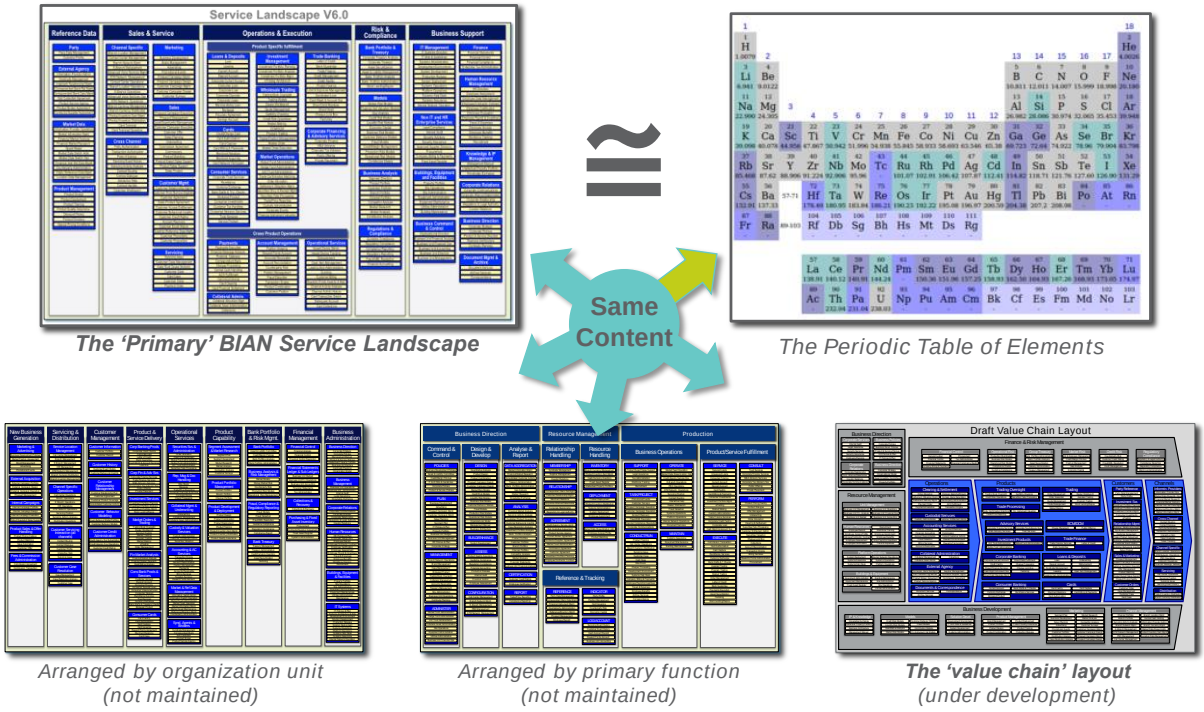


Figure 3: Periodic table and different BIAN Service Landscape views

As the above examples in the figure show, many possible layouts of the Service Domains can be defined to group and highlight different properties and associations between Service Domains. The BIAN Service Domains represent the elemental business capability building blocks of business – the Service Landscape in one way can be can be thought of as the 'Periodic Table' of business capability elements.

Two additional draft layouts are shown for explanatory purposes only:

- **Arranged by organizational unit** – in this layout Business Areas and Business Domains have been specified in a way that aligns to the possible divisional and operational units of a typical bank.
- **Arranged by functional pattern** – in this layout the Business Areas and Business Domains cluster Service Domains based on their associated Functional Patterns (the BIAN Functional Patterns have been refined since this example was developed). This is a type of grouping that could help match service domains to shared/common technical platforms and applications.

3.1.2 The BIAN Service Domain

The fundamental building block of the BIAN standard is the Service Domain. In this section the Service Domain is described here from two perspectives. First some more general design properties and considerations are provided. Second its specific design structures and standard properties are listed for reference purposes. More

detailed explanation of the design concepts behind these artifacts can be found in the How to Guide – Design Principles & Techniques

3.1.2.1 General Design Considerations & Properties of the Service Domain

The basic building block of the BIAN Service Landscape is the BIAN Service Domain. The BIAN standard semantic service operations at the heart of the BIAN standard are each uniquely associated with a Service Domain. The specification of the Service Domain and its service operations is intended to be generic or 'canonical', meaning that its business role or purpose can be consistently interpreted in different banks. This property is clearly critical for the definition of an effective industry standard.

The Service Domain can be thought of as the design/specification of an operational unit of the organization, not unlike a business unit, that provides a unique business service and that is called on by other units in the execution of business. Its design combines people, procedures and supporting systems as may be necessary for its business role. Some key Service Domain design/specification properties include:

- *A Unique Business Purpose* – a Service Domain has sole responsibility for fulfilling a specific and discrete business purpose/role.
- *It is Elemental* – it is not an assembly of other Service Domains. The full collection of Service Domains represents a 'peer' set of non-overlapping business capability partitions.
- *Collectively comprehensive* – any and all possible business activity can be modeled using suitable selections of Service Domains.
- *The Role Combines Action and Entity* – the role of a Service Domain reflects the combination of some asset or entity that the bank owns or has influence over and a specific action or function performed to that entity with the intent of enabling or creating commercial value.
- *Has a 'Control Record'* – the control record is a 'pattern' that is used to track the execution of the business purpose/role of the Service Domain. An instance of a control record is created each time the Service Domain fulfills is business role.
- *Full Life-Cycle support* – the Service Domain is responsible for executing its role for the full 'life-cycle' (e.g. inception, maintenance, operation/execution, reporting and eventual closure/termination). A control record instance is used to trace the different states of an asset or entity that is acted on by the Service Domain as it fulfills its role from start to finish.
- *Single or Multiple Instances* – depending on its role, a Service Domain may need to handle a single active instance or multiple active instances of its control record (for example there would typically be a single business unit plan at any one time but there would be multiple active customer accounts).
- *Short or Long Life-Span* – the life-span of a control record instance can be short such as that needed to oversee a customer interaction, or long such as one governing the life cycle of a product design.
- *Service Based* – all interactions with the Service Domain are realized through service operations and all possible business activity can be modeled as a pattern of service interactions between suitable selections of Service Domains.

Service Domain properties are illustrated in the figure below.

Some defining Service Domain characteristics:

- *A unique business purpose* **– has sole responsibility for fulfilling a specific and discrete business purpose**
- *It is elemental* **– it is not an assembly of other Service Domains.**
- *Collectively comprehensive* **– all possible business activity can be modeled using Service Domains**
- ***Has a 'Control** Record'* **– the control record reflects its business role or purpose (does something to something)**
- *Full Life-Cycle support* **– it is responsible for all possible states of its control record**
- *Single or Multiple Instances* **– can have a single active instance or multiple active instances of its control record (e.g. a single business unit plan, or multiple customer accounts)**
- *Short or Long Life-Span* **– its life-span can be short or long lived (e.g. a customer interaction or a product design)**
- *Service Based* **– all possible business activity can be modeled as a pattern of service interactions between a suitable selection of Service Domains**

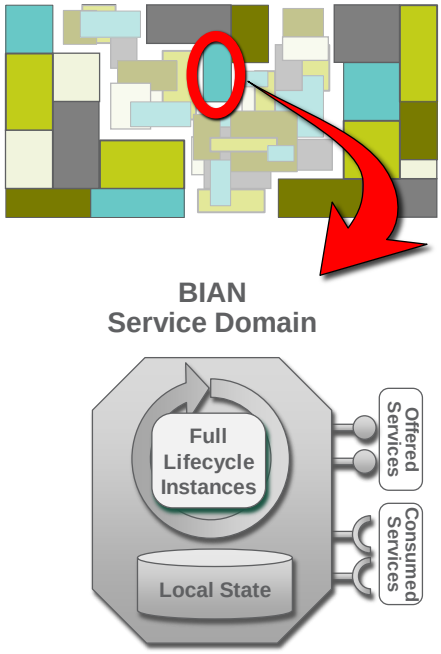


Figure 4: Key properties of BIAN Service Domains

The full rationale behind the specification of the BIAN Service Domain can be found in the 'BIAN How-to Guide – Design Principles & Techniques.' Some considerations are outlined here to provide the necessary context for the explanation of the Service Domain's specification content development covered by this document.

Business Capability Partition Vs Business Capability

There is an important distinction to be made between the business role represented by a Service Domain and the concept of a "business capability" as commonly referenced in business architecture related discussions.

A BIAN Service Domain is most accurately referred to as a business capability partition or business capability building block, sometimes mistakenly abbreviated to 'business capability'. There is a subtle distinction between the capability partition represented by a Service Domain and an aspect of a business that is conventionally referred to as a 'business capability'. The Service Domain represents a discrete and generic business function or the capacity to perform some action such as *maintain reference details about a customer relationship* or *operate a network*.

A formal definition of a 'business capability' typically goes further to describe something that the business wishes to be able to do within a defined business context or accountability for which some associated value and/or motivation can be

ascribed. The business capability combines the capacity to perform with a ***specific business context***.

The function performed by a Service Domain may be leveraged/reused to support different business capabilities with different associated business contexts and associated values and/or purposes. For example BIAN has defined a Service Domain that tracks/determines a bank's credit view for a customer (Customer Credit Rating). Consider when this is involved in two different business capabilities:

1. (The capability to) Match products to customers
2. (The capability to) Negotiate product pricing with customers

The business capabilities can be represented with business scenarios that would both reference Customer Credit Rating. But the value/impact of the bank having an inaccurate credit perspective of the customer varies between the two. If say the credit perspective is overly generous the impact on product matching could be to recommend the wrong product, leading to a missed sale or the sale of an inappropriate product. The impact on the pricing business capability could be to offer too generous terms - a different value measurement.

Having the business capability view allows this context-based distinction to be maintained. BIAN is currently developing a business capability model to augment the current Service Landscape that will be made available in a later release.

The Service Domain Control Record

A Service Domain's 'control record' is used to track one occurrence of the execution of a Service Domain's business purpose or role. BIAN defines the role of a Service Domain to be the combination of some type of influence or control that is exerted over a particular business entity or object (asset). For example applying a contractual agreement to a customer relationship, or executing the operational schedule for a piece of equipment.

To identify elemental and discrete business capability partitions BIAN has used a simple hierarchical decomposition of the possible assets or entity types (both tangible and intangible) that may be found in a bank. BIAN has also identified a finite number of actions or functions that may be performed on those assets/entities in order to foster or exact commercial value from them – these are called 'functional patterns'. Every BIAN Service Domain has the same design property: its business purpose or role combines the execution of one dominant functional pattern on instances of one type of asset or entity.

Furthermore the Service Domain is responsible for exerting this business control for the full 'life-cycle'. For example an 'agree terms' functional pattern can be applied to an instance of the intangible asset type 'customer relationship'. The associated Service Domain 'Customer Agreement' is responsible for the initial set-up, maintenance, for supporting any updates and access to the customer agreement, all analysis and reporting up to and including the customer agreement's final termination.

Rightsizing the BIAN Service Domain

A more involved design consideration relates to the 'right-sizing' of a Service Domain. The critical requirement is that the scope of its functioning is 'elemental' in nature. If a Service Domain is an assembly of many elemental functional components then different combinations of these components may be relevant in different deployments and any attempt to define a canonical specification of its behavior and service boundary is quickly compromised.

In practice BIAN has found that there is a point when decomposing business activity that there is a transition from business functions possessing unique 'business context' to being more utility in nature. When the business role/purpose of a Service Domain is matched to a business function at the threshold of retaining unique business context it is found to be an elemental business capability partition. The concept is fully explained in another document of the guide: How-to Guide – Design Principles & techniques. A simple example is used here to help clarify the idea for ease of reference.

The area of customer management clearly combines a broad range of business capabilities. Within customer management, one of many finer-grained capability partitions is the handling of customer agreements - as already mentioned. To test whether this is a capability partition found at the 'finest level of detail that retains unique business context,' the customer agreement handling capability partition can be broken down further into its constituent actions.

This decomposition results in actions such as reviewing, classifying and filing documents and maintaining customer details. These actions may not be uniquely associated with handling a customer agreement. Such fine-grained actions could be performed in many areas of the bank and so are more utility in nature. Handling customer agreements is therefore confirmed to be a correctly scoped, elemental business capability partition for a 'right-sized' Service Domain.

Translating BIAN Service Domain designs into Software Specifications

The BIAN standard is a 'business architecture' level perspective defining the discrete business capability partitions that make up a bank and the operational service exchanges between them that are defined in semantic terms. These high-level generic business designs need to be translated and extended in implementation. They need to be matched to the specific scope and layout of a particular enterprise and then the high-level business behaviors that define functional requirements must be extended in detail and realized by systems operating in a range of very different technical environments.

The way this can be done is set out in more detail in the How-to Guide – Applying the BIAN Standard. In summary, the semantic descriptions of the role of the Service Domains and the service operations that they call and consume define logical functional boundaries and interfaces that can be mirrored in the underlying systems architecture. When the underlying systems are aligned in this way, the benefits of service-based design outlined at the outset of this document can be realized

(including optimized performance, improved interoperability, better resource leverage and greater operational solution reuse).

The practice of retaining Service Domain boundaries in the derived software design is an important aspect of implementing a service-oriented architecture. It has no obvious equivalent in conventional process based design where business functionality is typically decomposed to a fine-grained and tightly coupled sequence of tasks that can be automated and run as a repeatable script or production process.

In contrast, applications aligned to the BIAN service-oriented design retain the coarse-grained functional boundaries defined by the Service Domains. Top-level application modules match the Service Domain scope, encapsulating its specific business role and providing service based access to its capabilities and delegating services to other Service Domain aligned applications as needed.

In more advanced and highly distributed technical environments (such as the 'cloud') and more recently micro-service architectures the service execution can be event-driven, asynchronous and loose coupled supporting a highly adaptive, flexible and effective operating model. In less advanced technical environments the standard can be used simply to eliminate overlaps and complexity in the application portfolio and rationalize interfaces, optionally leveraging client/server and enterprise service bus (ESB) technologies. (see the How-to Guide – Applying the BIAN Standard for examples)

The BIAN Service Domain Specification – Explaining Behavior Qualifier Types

The BIAN Service Domain specification extends the high-level definition captured at the Service Landscape level (and described earlier in this Section). In addition to this general definition, the Service Domain specification needs to set out the functionality required to support the service operations offered and consumed by the Service Domain and descriptions of the service operations themselves.

The functional description includes only that needed to detail externally visible behaviors. BIAN specifically does not attempt to specify the internal working or functionality of the Service Domain in any great detail. BIAN defines 'What' a Service Domain does by detailing the services it offers and consumes. BIAN does not attempt to define 'How' a Service Domain fulfills its purpose providing only a limited but sufficient description of its internal operation as necessary to support its offered services and highlight those services it needs to call on from other Service Domains.

In earlier releases of the Service Landscape reference is made to Service Domain 'responsibility items' as defined in the BIAN metamodel. Each responsibility is related to a service operation in one of three ways:

1. The responsibility item handles a service operation offered by the Service Domain.
2. The responsibility item delegates activity to a different Service Domain by calling on its service operation(s).

3. The responsibility item describes additional activity associated with the handling of either an offered or called service operation when necessary for definitional clarity.

The third type of responsibility item was intended to be used sparingly where key steps/decisions in the internal business logic need to be explained to clarify the specific business context for offered and consumed services. The concept of a responsibility item has been superseded in practice for now with the more recent use of business events to characterize the behavior of a Service Domain as described earlier in Section 2.4 with the description of First Order Interactions.

Service Domain Specific Behavior – the Behavior Qualifer

With this release an additional design concept has been applied to the Service Domain specification in order to clarify its role, to add precision/definition to the service operations and to help expand the detail of the information content. This concept is the 'behavior qualifier type'. The addition of the behavior qualifier type to the BIAN approach marks a subtle addition/shift in BIAN's overall approach. To this point the specification of content has been centered on the use of general patterns of behavior and/or characteristics to rapidly create candidate content that is subsequently ratified/refined through review.

With the introduction of the behavior qualifier type and work in parallel to develop the BIAN business object model (BOM) more effort is now being made to define the specific and unique characteristics of individual Service Domains in alignment with the more generic patterns of behavior. This process can be seen in the explanation of the behavior qualifier type that follows with the definition of behavior qualifiers that are specific to the Service Domain.

Every Service Domain has an associated functional pattern that characterizes its business behavior. BIAN has defined 18 generic functional patterns that cover the range of activities covered by all identified Service Domains. The *behavior qualifier type* defines the way the functional pattern can be further broken down. Behavior qualifier types have been defined for each functional pattern and these are used to define how the behaviors of a specific Service Domain can be broken down to define its own specific *behavior qualifiers*.

An example of the behavior qualifier type being applied to define a Service Domain's specific behavior qualifiers shows how a specific type is uniquely interpreted for the Service Domain. The Party Authentication Service Domain has the functional pattern 'Assess' with the behavior qualifier type being 'Tests'. The specific behavior qualifiers for the Service Domain are the different tests it may use to authenticate – for example passwords, secret questions, formal document scans, biometric matches.

The behavior qualifiers are used to extend the Service Domain's specification in the following main ways. :

- The list of behavior qualifiers clarifies the functioning/role of the Service Domain

- They can be used as an extended term or parameter for a service operation to provide additional precision to its purpose
- They provide a more detailed basis for defining the business information governed by the Service Domain and also that may be contained in the individual service operations

Examples of these uses can be found in the latest release for the Service Domains in the scope of the BIAN Semantic API initiative.

Another way that Service Domain functionality has been captured in the past is through the use of a more simple and informal template. This has been used on deployment projects to summarize the desired functionality of the Service Domain for reference and comparison purposes. It is called the (Functional & non-Functional) *feature template*. An example feature template with a full explanation can be found in the How-to Guide – Applying the BIAN Standard.

Service Operation Details

The most detailed aspect of the Service Domain specification is the semantic definition of the service operations it offers and consumes. A Service Domain's use of delegated service calls to other Service Domains provides additional insights into its internal working.

The structure and content of a service operation is described in a later section of this document within the context of business behaviors modeled using business scenarios. These specifications capture the key types of information that is exchanged between the involved Service Domains in semantic terms.

3.1.2.2 Service Domain Structures and Properties

The design structures and general properties are described here for reference. As note more detailed explanation is provided in the How To Guide – Design Concepts & Techniques. The elements used in a little more detail are as follows:

- **Functional pattern** – each Service Domain has one assigned dominant functional pattern. This defines the primary working role of the Service Domain. Each functional pattern has an associated 'generic artifact' that can be related to the Service Domain's 'control record'
- **Derived Service Domain states** – proposes a series of (externally visible) main states for each functional pattern. These are subsequently used to define valid pre and post state conditions for each service operation.
- **Standard service operation actions** – a structured list of service operation actions terms is used to characterize the general purpose of the individual service operations.
- **Relating default service operation action terms to functional patterns** – based on the nature of the functional pattern and its life cycle states, a suitable selection of the available action terms has been made to define the default candidate service operations for each Service Domain.

- **Service operation content** – the main payload of the service operation provides input to or is an extract of one or more control record instances. In earlier releases this content has been defined using filtered checklists. In the latest release Service Domain specific content is captured.

The structures BIAN uses to define service operations are now each described in more detail with explanatory examples as necessary.

Service Domain Functional Patterns

A Service Domain combines a type of commercial behavior that acts on a type of asset. BIAN has developed a decomposition of the types of assets that might be found in any bank to a specific level of granularity. BIAN has also defined a standard list of functional patterns that reflect the different commercial behaviors. This list has been iteratively refined in use over many release cycles of the BIAN Service Landscape. There are currently 18 standard functional patterns as listed in the table below.

Functional Pattern		Description	Example(s)
Management & Support Capabilities	DIRECT	Define the policies, goals & objectives and strategies for an organizational entity or unit	Direct a business division of the enterprise
	MANAGE	Oversee the working of a business unit, assign work, manage against a plan and troubleshoot issues.	Manage the day to day activities at a bank branch location
	ADMINISTER	Handle and assign the day to day activities, capture time worked, costs and income for an operational unit.	Administer the time reporting and billing for the specialist sales support team.
	OPERATE	Operate equipment and/or a largely automated facility.	Operate the bank's internal intranet facility
	PROCESS	Complete work tasks following a defined procedure in support of general office activities and product and service delivery functions.	Process the evaluation and completion of customer offers
Resource Management	REGISTER	Capture and maintain reference information about some type of entity.	Register customer reference details in a directory
	DESIGN	Create and maintain a design for a procedure, product/service model or other such entity.	Create and maintain product designs and analytical models
	DEVELOP	To build or enhance something, typically an IT production system. Includes development, assessment and deployment activities.	Build, enhance, test and deploy a major enhancement to a production processing system
	ASSESS	To test or assess an entity, possibly against some formal qualification or certification requirement.	Perform regulatory tests on a proposed financial transaction; check a new offer conforms to an existing contractual agreement
	MAINTAIN	Provide a maintenance service and repair devices/equipment as necessary.	Establish a maintenance and repair program covering the PC technology used in the central offices
Activity Oversight	TRACK	Maintain a log of transactions or activity, typically a financial account/journal or a log of activity to support behavioral analysis.	Maintain a financial journal of transactions processed for a product; maintain a log of customer events and activity for subsequent analysis
	ANALYSE	To analyse the performance or behavior of some on-going activity or entity.	Provide behavioral insights and analysis into customer behavior; analyse financial market activity in order to identify opportunities
	MONITOR	To monitor and define the status/rating of some entity.	Monitor the status and key indicators of a customer to influence on-line interactions; track the status of issued cards for security control
Resource Assignment	AGREE TERMS	Maintain the terms and conditions that apply to a commercial relationship.	Define and maintain the terms govering the contratcual relationship with a customer
	ENROLL	Maintain a membership for some group or related collection of parties.	Administer the memebrship status of a syndicate of investors
	ALLOCATE	Maintain an inventory or holding of some resource and make assignments/allocations as requested.	Track the inventory and administer the distribution of central cash holdings throughout the bank branch & ATM network
Production	FULFILL	Fulfill any scheduled and ad-hoc obligations under a service arrangement, most typically for a financial product or facility.	Perform the scheduled (e.g. statements, standing orders) and ad-hoc fulfillment tasks (e.g.fund transfers) for a current account facility
	TRANSACT	Execute a well bounded financial transaction/task, typically involving largely automated/structured fulfillment processing.	Execute a payment transaction

Figure 5: Functional Patterns, Outlines and examples

The Service Domain also defines its *control record*. This is a mechanism used to track the state of one instance of the asset type acted on by the Service Domain as it applies its functional pattern for the full life-cycle. For each Functional Pattern BIAN defines a generic artifact. The Generic artifact provides a better description of the control record that it easier to interpret. The Functional Pattern generic artifacts are listed in the table below:

TABLE Generic Artifacts (MISSING FIGURE)

The meaning of the Functional Patterns and the associated control record/artifact is obvious in most cases. The distinction between the 'Transaction' and 'Fulfillment' patterns can benefit from some clarification. They both support the day-to-day handling of different banking products and services. Transaction refers to products that typically complete in a single transactional cycle. Fulfillment products represent the on-going support of a financial facility.

The guidelines in a little more details are as follows:

> **Transactional** – the pattern relates to a product or service that has a well-defined processing path, perhaps with different possible conclusions (such as an option that may expire or be exercised). However once the product has been initiated, the pricing/terms, dates etc. are fixed. If a significant change is required the typical action would be to cancel and replace the product instance
>
> **Fulfillment** – the pattern relates to ongoing facilities (such as a current account) where there can be schedule and ad-hoc activity and associated rules/terms that may change over time. Also it covers products that may have a defined life-cycle as with Transactional behavior but for which there can be 'in-flight' changes such as re-negotiated terms, extensions and rollovers.

Service Domain Standard States

As noted each Service Domain has a business purpose or role that is defined to be the combination of a type of function or 'functional pattern' that acts on a type of asset or entity. Typical states and state changes can be defined and associated with the functional patterns providing an opportunity to identify and confirm the desired response from service operations that may trigger state changes. BIAN has identified the key externally visible state changes associated with each functional pattern in the table below.

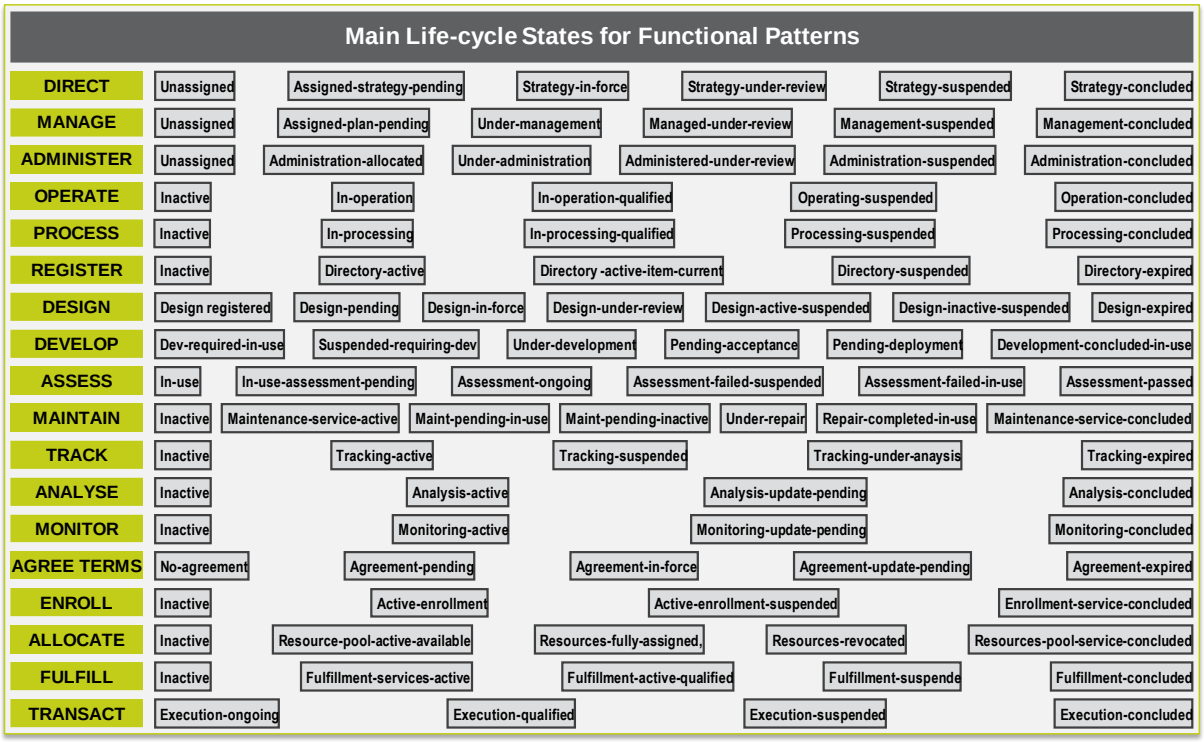


Figure 6: Functional Pattern main Service Domain states

At this stage the use of state analysis is limited to these main external states. It is anticipated that more detailed internal state transition analysis could be added to the standard in future releases.

Functional Pattern Behavioral Qualifier Types

In order to develop an additional level of detail to the Service Domain specification a behavior qualifier type has been defined for each Functional Pattern. The qualifier type defines how the pattern can be further broken down into its constituent elements. The behavior types vary significantly in nature as do the Functional Patterns. The behavior qualifier types are listed in the table below:

Functional Pattern	Brief Definition	Information Profile			
		Generic Artifact	Definition/Description	Behavior Qualifier Type	Behavior Qualifier Type Description
DIRECT	Define the strategy	Strategy	The purpose and mission for the enterprise including its competitive positioning and bases for competing in the market	Goals	A collection of goals and objectives for the enterprise and its main divisions
MANAGE	Oversee activity	Management Plan/Charter	The management and oversight while running an operational unit of an enterprise	Duties	A collection of one or more responsibilities or tasks under management
ADMINISTER	Administer activity	Administrative Plan	The clerical support for an operational unit/function of an enterprise	Routines	A collection of one or more clerical routines that are to be followed to administer the operational unit/function
OPERATE	Operate facility	Operating Session/ Facility	The operation of a technical/automated facility employed/provided by an enterprise	Functions	The collection of operational serivces/functions offered by the operational facility
PROCESS	Process work	Procedure	The performance of a supporting office activity within the eneterprise (not product/service fulfillment specific)	Worksteps	The main worsteps to be followed in th eexecution of the procedure
REGISTER	Register details	Directory	A registry of items recording key reference information and properties relating to each	Properties	The properties/reference details recorded In the registry for items
DESIGN	Design solutions	Specification	A specification of a product or service offering covering all aspects required for its use	Aspects	The main design elements/views making up the overall specification
DEVELOP	Execute projects	Development Project	A descrete or bounded effort with a defined remit and intended purpose/outcome	Deliverables	A collection of one or more deliverables that may be further defined in terms of an approach to be followed to create them
ASSESS	Test compliance	Assessment	A formal evaluation or test of a subject against a predefined set of properties or performance criteria	Tests	A collection of one or more tests'evaluations that can be made to certify a subject
MAINTAIN	Maintain resources	Maintenance Agreement	A service to provide maintenance and repair to operational capabilities/technology	Tasks	A collection of tasks needed to support maintenance and repair work
TRACK	Log events	Log	A mechanism to track and record specific events and if necessary maintain associated derived/accumulated values	Events	A collection of the events/transactions recorded by the log
ANALYSE	Analyse activity	Analysis	A service to apply specific types of analsis against a set of provided data related to an item or activity	Algorithms	A collection of models/calculations/algoritms that can be applied to a subject or activity
MONITOR	Measure resources	Measurement	A mechanism to track and report on the state or dynamic property of some item or activity	Signals	A collection of information feeds/measures that can be used to track the status of one or more items/entitites
AGREE TERMS	Govern activity	Agreement	A service to apply specifc laws and/or rules to define the terms and conditions that govern a business service or activity	Terms	A collection of terms (within some jurisdiction) that can be selected and configured to define a contract /agreement
ENROLL	Register members	Membership	A registry of entities that qualify for membership to a group with a recognised business purpose or catergorization	Clauses	A collection of clauses that govern the eligibility for membership
ALLOCATE	Allocate resources	Allocation	A service to track the availability and allocate business resources (staff and/or facilities) on request	Assignments	A collection of one or more specific assignments of inventory allowing for different allocation types and states
FULFILL	Fulfill agreement	Fulfillment Arrangement	The fulfillment of a financial facility, including customer initiated and internally triggered actionsFeatures	Features	The product features/services available with a financical facility
TRANSACT	Execute transactions	Transaction	The execution of a financial transaction	Tasks/Steps	The sub-tasks involved in the execution of the financial transaction

Figure 7: Behaviour Qualifier Types

The actual 'behavior qualifiers' are specific to a Service Domain. The 'type' defines what the nature of the breakdown should be, but the decomposition has to be defined for the individual Service Domain. For example the Service Domain Party Authentication has the Functional Pattern 'Assess' - its business purpose is to 'assess' whether the identity of an individual is correct. The *behavior qualifier type* for the 'assess' functional pattern is 'tests'. The specific *behavior qualifiers* are the types of tests the Service Domain performs to check the identity (e.g. password checks, biometric tests…)

Service Domain Standards Action terms

Each service operation call results in some kind of action being performed to one or more control record instance by the called Service Domain. BIAN has defined a list of allowed action terms. Note that the action terms are grouped under the three responsibility types that apply to offered service operations (Origination, Invocation, & Reporting):

Action Terms			Description	Example
Origination	Actions to set-up, establish a new control record instance	Initiate	Begin an action including any required initialization tasks	A payment transaction is intiiated
		Create	Manufacture and distribute an item	A new analytical model design is created
		Activate	Commence/open an operational or administrative service	The ATM network operation is actived
		Configure	Change the operating parameters for an ongoing service/capability	The on-line ATM's in the network are changed to take machines out of service
Invocation	Actions to access/update/influence an established instance	Update	Change the value of some (control record) properties	A customer's reference details are updated with a change of address
		Register	Record the details of a newly identified entity	A new customer's details are captured
		Record	Capture transaction or event details associated with a life cycle step	An employee logs time spent working on a project against the plan
		Execute	Execute a task or action on an established facility	A payment is applied to a charge card
		Evaluate	Perform a check, trial or evaluation	The eligibility to sell a product is checked against the customer's existing agreement
		Provide	Assign or allocate resources or facilities	A branch requests an allocation of cash for its tellers
		Authorise	Allow the execution of a transaction/activity	Regulatory compliance authorises a product design feature
		Request	Request the provision of some service	A customer requests that a standing order is set up on the current account
		Terminate	Conclude, complete activity	The use of a product version is terminated
Delegation – no new action terms apply as the called Service Domains offer the same Origination/Invocation & Reporting options described here)				
Reporting	Actions to extract details and subscribe to updates	Notify	Provide details against a predefined notification agreement	A unit subscribes to update notifications from the customer agreement service domain
		Retrieve	Return information/report as requested	An account balance is obtained and a report covering activity analysis requested

Figure 8: Action Terms, descriptions and examples

Service Domain Default Service Operation Mapping

In order to define candidate service operations for Service Domains the action terms have been mapped to the functional patterns taking account of the life cycle states for the Service Domain. Note that the candidate service operations with their draft content descriptions are intended to provide a starting point for ratification and refinement by the Working Groups and in deployment. The default action term to functional pattern mapping is shown in the table below:

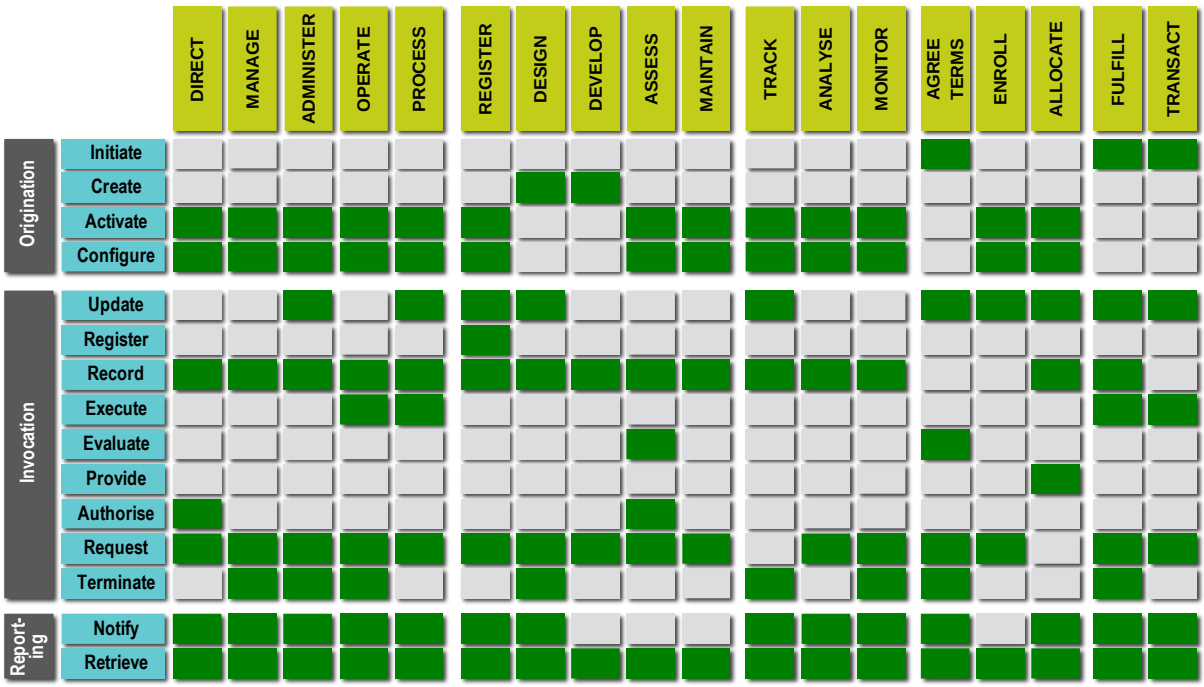


Figure 9: Action Terms, mapped to Functional Patterns for default service operations

More specialized service operations

The collection of candidate service operations with one defined for each of the default action term are in the majority of cases sufficiently narrow in scope/purpose to define a clear business objective for each supported service interaction. For some of the more complex Service Domains, particularly in the area of product/service fulfillment additional, more specialized service operations may need to be defined for some action terms. These more precisely defined service operations use the Service Domain's behavior qualifiers as an additional term in the service operation name.

The service operation in general and the more specialized variety are fully explained in Section 3.1.5 below

3.1.3 The BIAN Business Scenario

The BIAN Business Scenario is not a formal component of the BIAN standard as it does not represent a canonical design. It is included in the BIAN SOA Framework because it provides a useful mechanism to define the roles and interactions that the Service Domains support by example. Without Business Scenarios to provide some explanatory context it can be very difficult to relate individual Service Domains and their service operations to the business activities and supporting systems within an enterprise. A business scenario typically represents the way a business meets a particular business requirement or reacts to some kind of business event.

First Order Interactions

In earlier releases a large number of simple business scenarios were defined based on an event analysis of individual Service Domains. These only considered activity from the perspective of a single 'primary' Service Domain. The scenarios identified the calling and delegated service operation connections for the Service Domain resulting from its handling of the event. The main purpose behind creating these simple event related scenarios was to identify the service operation connections between the Service Domains rather than to provide a full explanation of the fulfillment of some business requirement.

From this point forward a distinction is made between these simple views that are now referred to as 'First Order Interactions' and the more comprehensive Business Scenario as described in this section. BIAN continues to define and ratify the First Order Interactions as these define key properties of the Service Domains. The connections they help identify are referenced in the generation of wireframe views (described next) and can also be used as building blocks to facilitate the assembly of more complex business scenarios. Any service operation connection seen in a Wireframe or Business Scenario can be mapped to the corresponding First Order Interaction. Candidate First Order Interactions are developed by the central team, as described in the last Section of this guide.

Business Scenarios

The BIAN Business Scenario represents how some suitable selection of BIAN Service Domains might work together to handle some identified business event or requirement. It is used to help visualize the roles and service operation exchanges of the involved Service Domains. It defines an archetypal flow but is not required to be complete or exhaustive. Nor does a business scenario prescribe a particular sequence – it merely needs to provide meaningful context for the Service Domain interactions that are of interest.

BIAN uses Business Scenarios internally to help identify and specify Service Domains and their service operation interactions. In the latest release a distinction has been made between a simplified type of business scenario – the First Order Interaction and more complex business scenarios. The First Order Interaction as described in Section 2.4 of this guide is used to identify the required service operation connections between Service Domains. It identifies these connections by considering the way a 'primary' Service Domain responds to a set of business events. This narrow focus does not define the business context in any great detail.

For clarity the term Business Scenario is now used to refer to more complex representations or business activity and the simple scenarios used to model a primary Service Domains behavior are termed a First Order Interaction. BIAN Business Scenarios have a number of material differences from the First Order Interactions already described as follows:

- Pre/Post Conditions – the starting and end points are defined in business terms along with any associated conditions/constraints.

- Goal/Objective – a concise statement as to the business goal associated with the successful completion of the business scenario
- Optionally Performance Measures/Success Criteria – in time additional detail may associated with a Business Scenario that supports its evaluation/impact measurement
- Multiple Events – it is not constrained in terms of the number of business events and 'orchestrating' Service Domains that may be involved. The only guide is that a business scenario should be reasonably concise. More complex processes may need to assemble two or more business scenarios together for practical purposes

The informal term 'orchestrating' used in the last bullet point may need explanation. In a First Order Interaction the primary Service Domain acts as the 'orchestrating' Service Domain. It is typically triggered into action and then 'orchestrates' one or more delegated calls in response to handle the event. Many Business Scenarios may also have only one 'orchestrating' Service Domain. The difference being that the Business Scenario provides a more comprehensive description of the prevailing business conditions for the Business Scenario as listed above. Furthermore as noted, there is no limitation to defining a Business Scenario where there are two or more orchestrating Service Domains where one hands on control to another in some sequence of events/actions.

Another property of the Business Scenario is that is can be used to capture second order or dependent service operation connections. This shows where a delegated call to a Service Domain results in the called Service Domain making its own 'nested' delegated call to a third Service Domain. As described earlier in the description of first order interactions, the principle of encapsulation in service-oriented design requires that any downstream dependencies for an offered service should be completely transparent to the caller.

This is a good and necessary design principle but also has implications on how the design is realized when implemented in a physical system. One aspect of this is in the specification of the logic behind the realized service operation, the other is the performance/response of the service operation:

- ***Service operation logic*** – the definition of the processing logic that defines the required input and expected response to the service needs to avoid any reference to the internal behavior of the offering Service Domain, including any delegation dependencies it may have
- ***Service operation performance*** – where the offering Service Domain may have a downstream dependency it needs to ensure that the committed response it guarantees for its offered service is not compromised by the committed response guaranteed to it by any nested services that it delegates

The above considerations are an aspect of physical implementation design and so it is appropriate that they are captured in a Business Scenario that is modeling actual business behaviors. It also highlights a property of the service operation connections that is not always obvious at first reading. The service connections captured in the scenario show an exchange dependency between two Service Domains, indicating that to be able to perform its role the calling Service Domain is dependent on

information or some action governed and/or performed by another Service Domain. It does not define how or indeed when that dependency is fulfilled in practice.

When reviewing a Business Scenario it is tempting to assume that each step happens in sequence – this exchange, followed by this exchange, etc. as is the case in a conventional business process. The exchanges included in a business scenario however are intended to be loosely-coupled/asynchronous. For example if a called Service Domain needed information from a third Service Domain to be able to respond this would be shown as a 'nested' second order connection in the Business Scenario. In implementation it may be that the called Service Domain goes and gets that information when it needs it in response to someone calling its offered service or it may ensure it has a 'current view' of that information in advance so that it can respond immediately to the callers request without waiting to get the information.

Both approaches would be modeled the same way in the Business Scenario as a dependency, but the timing in physical execution is clearly completely different. This analysis of service dependencies is a complex aspect of service oriented implementation but is not modeled in the Business Scenario, it being implementation agnostic.

The BIAN Business Scenario is captured and referenced using various tools. Below a simple Powerpoint format is shown:

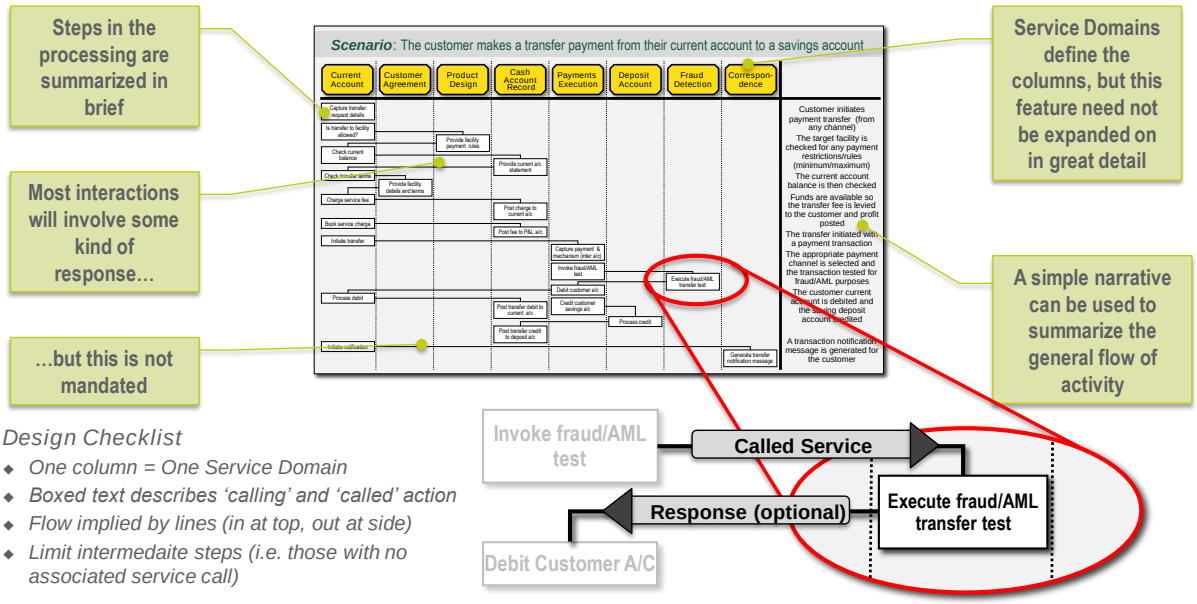


Figure 10: Simple Business Scenario with rules

The loose-coupled archetypal flow of activity in the figure reads from top to bottom. The role of a Service Domain and the actions it performs in the specific business scenario are captured in a single column. The arrows connecting between columns indicate a service operation between the two corresponding Service Domains. In later versions an action term is added to the service operation connection showing the type of service being called.

The simple format is intended to support discussions with business practitioners, avoiding the need to explain complex underlying technical design considerations. As noted the sequence as represented can change in practice, some interactions might be obsolete and others might be missing – it is only intended to provide an effective environment for practitioners to highlight key requirements and to discuss the considerations that need to be captured in the definitions of a Service Domain's service operations.

A second format for the Business Scenario is that captured in the MagicDraw UML repository in the next figure:

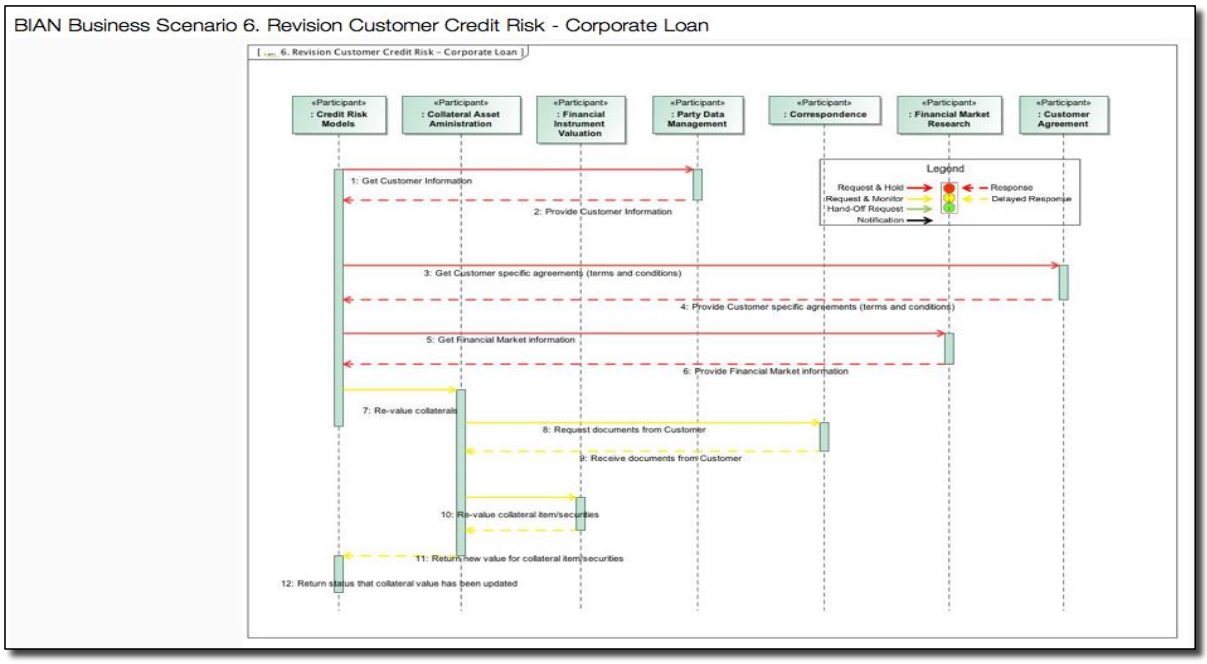


Figure 11: Example Business Scenario in MagicDraw

In the past Business Scenarios have included a definition of the operational properties of the service operation exchanges. Four types of exchange were defined:

- **Request & hold** – the calling Service Domain anticipates a response sufficiently quickly for it to continue with its activities.
- **Request & monitor –** the calling Service Domain knows it will have to wait for a response and so goes off to do other things, monitoring for the response.
- **Hand-off** – the calling Service Domain passes off information and or some instruction to another Service Domain but has no operational dependency on what may happen next.
- **Make Announcement** – the calling Service Domain has previously requested to be kept 'up to date' with information governed by another Service Domain and this is the resulting notification.

In practice it has been found that different combinations of these interactions can sensibly apply in different deployment situations for the same service operation. As a

result the classifications provides little insight and worse can be misleading at this level. The classifications are being phased out and the associated implementation considerations moved to the appropriate guidelines for interpreting the model in solution design and deployment.

The scope of a Business Scenario can be compared to that of a conventional high-level Business Process with one key difference. Both describe action steps and some implicit flow of control, but the Business Process does not formally divide functionality between discrete service based partitions (Service Domains) unlike the Business Scenario. Fortunately this very significant difference is largely transparent to the typical business practitioner who will recognize the actions and implicit sequence of events regardless of any particular grouping and layout.

Presenting Service Domains in the context of a familiar business operation using the Business Scenario format makes the service-based designs easier to understand. Furthermore, presenting different business scenarios each involving a common Service Domain can help reveal how a service-oriented architecture can be used to define highly re-usable and leveraged operational capabilities.

3.1.4 Wireframe Models

With the latest release and in particular the BIAN Semantic API initiative the use of wireframes has been expanded. A Wireframe Model pulls together a related collection of Service Domains and shows the available/established service operation connections between them. These connections can be demonstrated using one or more Business Scenarios. There are many different reasons a collection of Service Domains may be represented in a wireframe model. In the API initiative wireframes have been developed for functional areas of business covered by related collections of business scenarios (such as mobile access and payments). These wireframes have also been adapted to show the flows within the bank and between the bank and external third parties and customers..

The Wireframe is a ‘static’ model view, showing the Service Domains and (all pertinent) available connections. Conversely a Business Scenario is a dynamic model view that shows the temporal pattern of a collection of interactions that are typically triggered by some business action, requirement or event.

A business scenario can be overlain on the Wireframe figure as shown in the example below. The Wireframe shows the Service Domains and Major service operations available for handling the external connection to the S.W.I.F.T network and a simple payment business scenario is mapped over the framework.

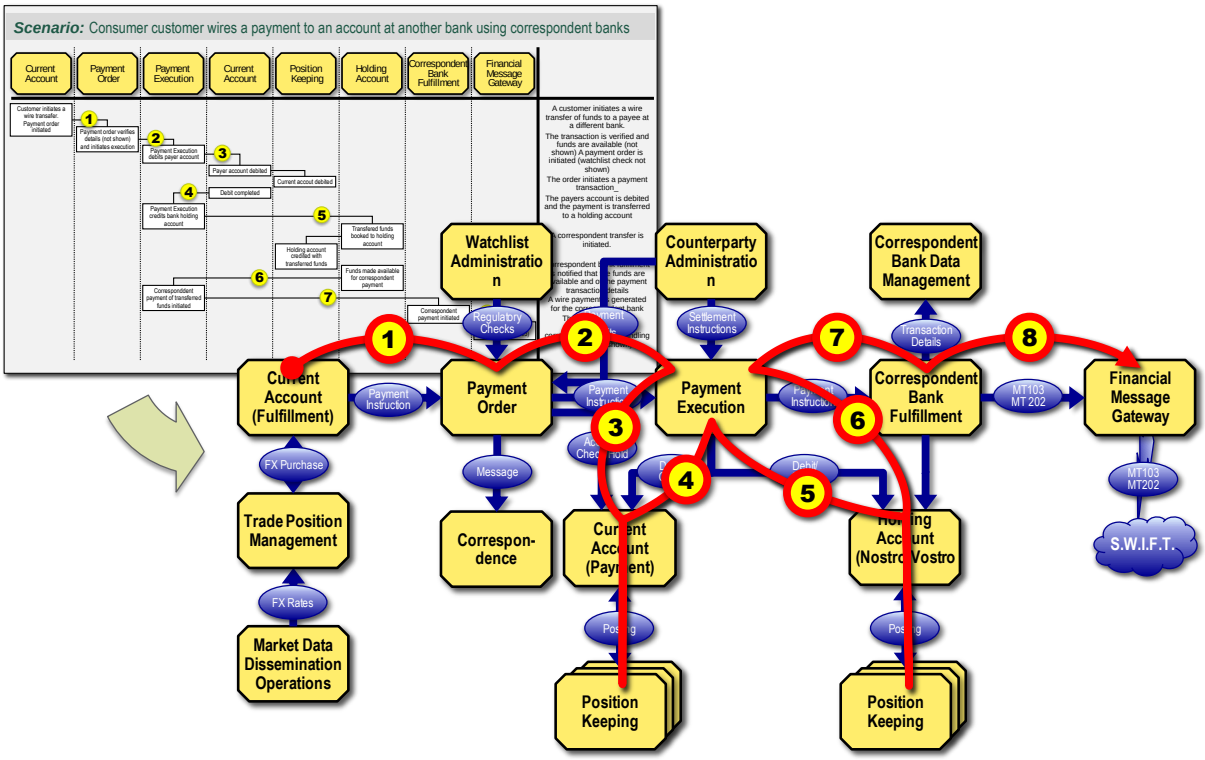


Figure 12: A payment transaction mapped on a Wireframe view.

With the latest release minor improvements have been added to the informal wireframe model shown above. In particular the annotation on the service operation connections now references the associated action term for the service operation. In addition as noted some versions of the wireframe have been adapted to show the internal operational structure and external connections between the bank, third party providers and customers.

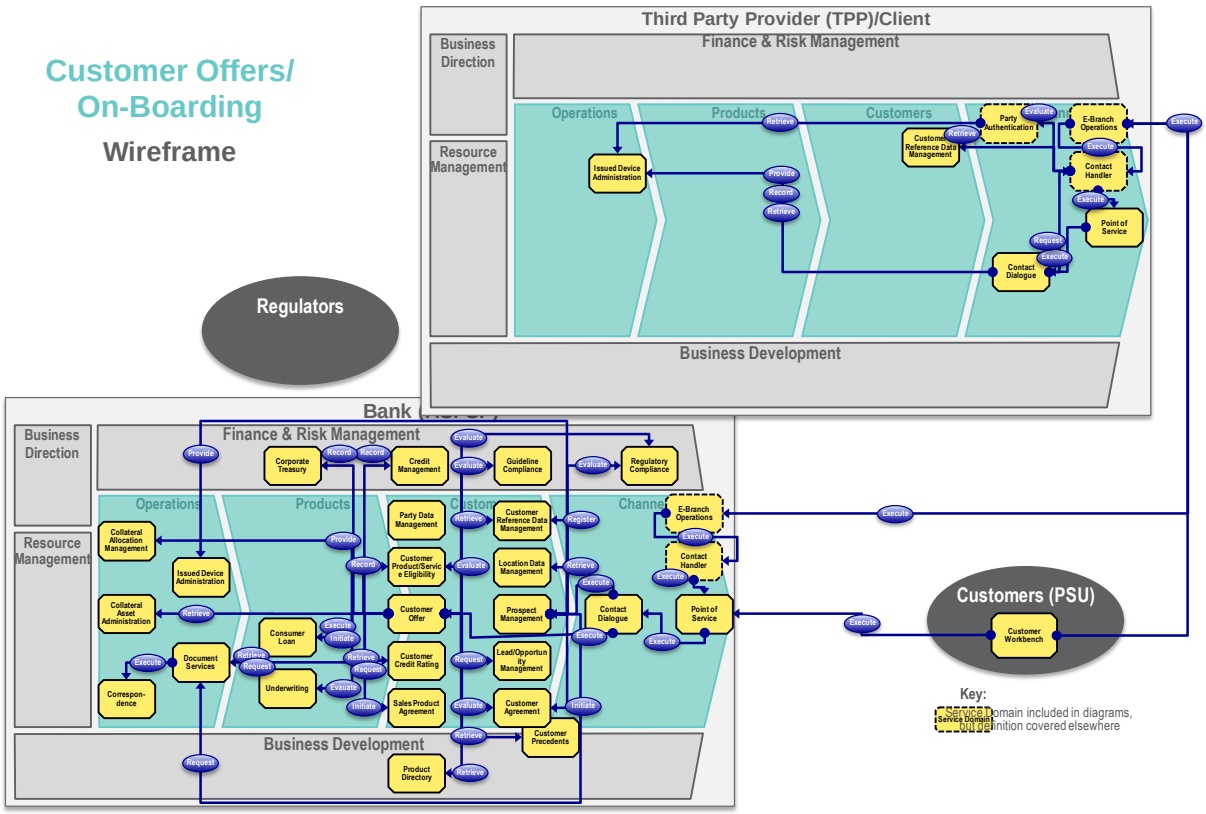


Figure 13: Example of a Semantic API Initiative Wireframe

A more formal notation has been developed for the wireframe that captures the types of state changes associated with the service operation calls on the control record. This more complex representation is not needed for the current use of wireframes and so has not been adopted. It may be used in later releases if work is done to model the internal state transitions of the Service Domains. The more formal representation includes the following notations:

- The point of contact on the Service Domain aligns with five points that reflect the type of service operation interaction. These are described in the figure below that shows how different service operation action terms align to the called and offered service connections
- Offered services connect to the left and top of the Service Domain, delegations are made out of the right and lower edges of the Service Domain.
- Service connections to the left and right sides of the Service Domain (horizontal) act on existing instances of a Service Domain's control records – i.e these services act in a steady-state environment. Vertical connections result in the creation of a new control record instance and/or termination of an existing control record instance
- As with the current wireframes, the elipses on the service connections refer to the associated action term for the service operation, and
- The protocol for the connections is that the arrow-head points to called Service Domain. If the call has a dependent response the root of the connection has a circle connection.

A view of the Service Domain representing these properties and an example of a wireframe conforming to these notation standards is included in the figures below:

The diagram shows the main offered and called service operation connections for the Mortgage Loan fulfillment Service Domain

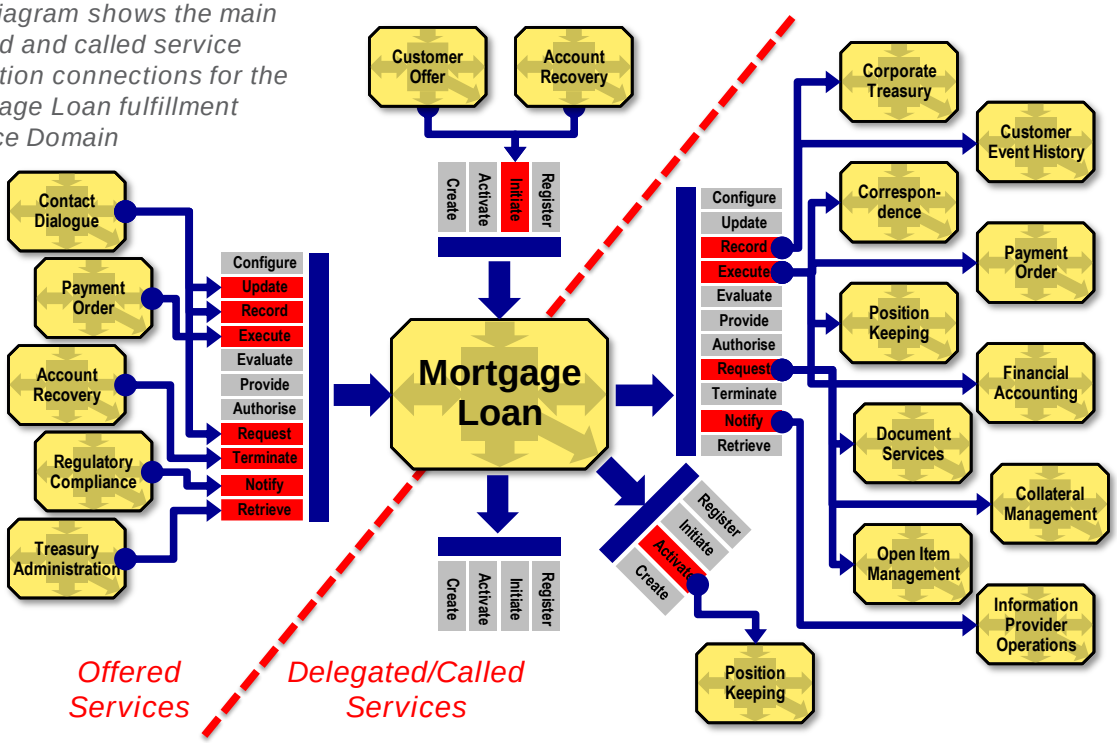


Figure 14: Five prong Service Domain boundary

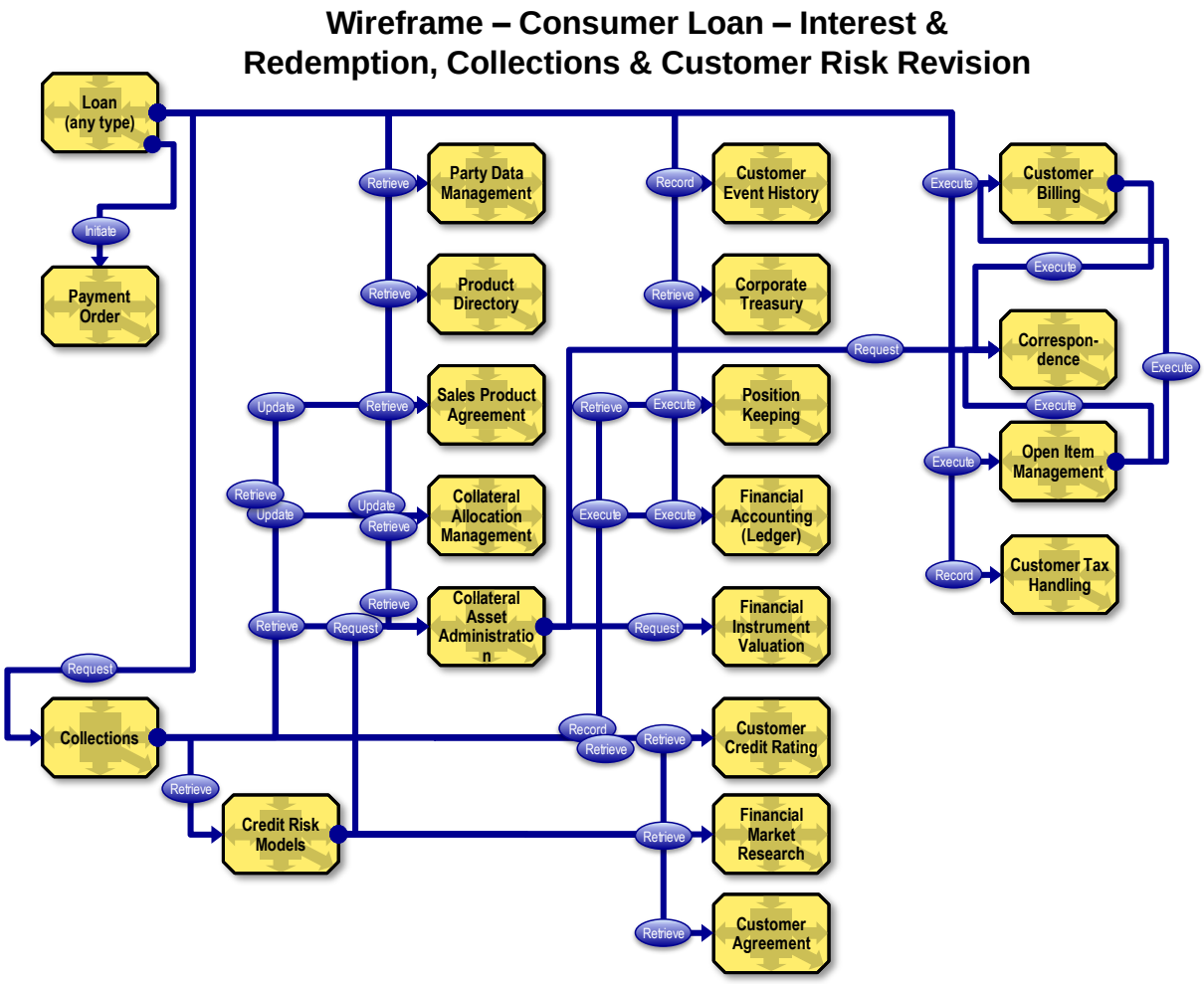


Figure 15: Example more sophisticated Wireframe

3.1.5 Service Domain Service Operations

As noted earlier in this guide with the introduction of behavior qualifier types and the BIAN BOM in the latest release cycle BIAN has started to develop highly specific Service Domain and service operation content. With this addition there are now two ways that the service operation is defined. One that has been used in earlier releases is through the use of an information checklist that is filtered based on properties of the Service Domain and the individual service operations.

The checklist-based content is now being progressively replaced. In the new approach the business information profile of the individual Service Domain is developed using the behavior qualifiers and also the related development of the BIAN BOM. From this profile the pertinent business information is extracted to define the content of individual service operations. The two approaches are outlined below.

Checklist Based Service Operations

The approach used to generate checklist based service operation content descriptions was as follows. First a comprehensive list of possible business information types was defined (by reverse engineering existing examples). This list differentiated between information items, structured information records and unstructured information (reports). Definitions of the checklist business information types are included in the BIAN vocabulary.

The list was then filtered twice – first a selection was made to align to the different functional patterns of a Service Domain, second to align to the different action terms for that Service Domain's service operations. The result was a description of the type of information likely to make up the service operation payload. The approach taken was inclusive, meaning that any possible content that might sensibly be of use was recorded in the draft descriptions. As a result there is sometimes redundancy in terms of service operation parameter content that is excessive to requirements for a particular Service Domain. The checklist based service operation descriptions have been applied across the complete Service Landscape.

Behavior Specific Service Operation

With the latest release BIAN has introduced behavior qualifiers for Service Domains and started to create a comprehensive business object model (BOM) – informed by the industry standard ISO20022 BOM. With these additions the checklist content for the service operations is being progressively replaced with more specific and detailed information definitions. The updated specifications are being developed as an aspect of the BIAN Semantic API Initiative. This approach is described in the final Section of this guide.

In summary business scenarios and wireframes are first used to model key business activity and the associated service exchanges. This perspective provides a more specific business context for defining the service operations. Behavior qualifiers break down the types of activity relating to the Service Domain's specific functional pattern to another level of detail. Based on this more detailed definition of the Service Domain's behaviors a correspondingly more detailed view of the information governed by the Service Domain can be produced. The content of the individual service operations is taken from this more detailed specification of the governed information available.

The same behavior qualifiers can also be used to add specificity to the individual service operations when appropriate. An additional more specific service operation can be defined, adding the behavior qualifier term to the service operation name (as described at the end of this section).

Regardless of the approach taken to define the content of the service operations, their basic layout/structure remains the same, as set out in the remainder of this section.

3.1.5.1 Service operation content

The BIAN service operation covers the key semantic information exchanged between Service Domains. In practice such an exchange may include the assignment or physical movement of resources, person to person conversation as well as machine readable information exchange. As BIAN is focused on the application to application aspects of this exchange the descriptions address only these information aspects.

Furthermore a service operation exchange may in its physical implementation involve the simple one or two way movement of information/data or it could be a complex dialogue over an extended period. As the physical properties of the exchange are implementation specific they are not defined in the BIAN standard.

Some reference to interpreting the BIAN semantic services is provided in the How-to Guide – Applying the Standard. In addition guidelines for interpreting the BIAN designs for API development specifically are provided in the BIAN Semantic API How to Guide.

The service operations and their content is defined as follows:

- Service Operation Responsibility Category and Service Type
- Service Operation Name – a formal structure is used
- Service Interaction Type – likely types of operational exchange – these have been discontinued but are included here for reference
- Service Domain - pre & post states
- Input & Output - descriptions of the four main service operation parameter types

3.1.5.2 'Responsibility' category and service type

The 'responsibility' categories (as defined earlier in this guide) are used to categorize various aspects of a BIAN Service Domain's specification. This includes the service operations for a Service Domain. The categories are:

1. *Origination* – service operation requests that create a new control record instance, or to register a new entity.
2. *Invocation* – service operation requests that may update the details or initiate a task that acts on one or more existing control record.
3. *Delegation* – refers to delegated service operation calls made by a Service Domain.
4. *Reporting* – service operation requests to provide reports on individual, collections and/or analyses of control records.

The categories are currently simply used to group service operations. They will have more significance when additional state based specifications are developed for Service Domains and service operations in future releases. Each service operation has an action term taken from a standard list that reflects its main purpose. The action terms and their default mapping to Service Domains based on the functional patterns were described earlier in Section 3.1.2.2. of this guide.

3.1.5.3 Service Operation Name

The naming convention for service operations uses a simplified subset of the full definition as defined in the BIAN UML specification. The full definition for reference is as follows:

> **Service Operation (in Extended Backus-Naur Form):**
> *[<qualifier terms>]""<action term>""[<qualifier terms>]""<object class term>""[[<qualifier terms>]""<property term>]*
>
> *Example:*
>
> *updatePaymentClearingAgreement (action term = update, object class qualifier term = Payment, object class term = Clearing Agreement)*

The simplified format used in this latest release is as follows:

> **V6.0 Service Operation (in Extended Backus-Naur Form):**
> "<action term>""<control record>""[<qualifier term>]"
>
> The last field <qualifier term> is optional and used to define a finer grained service operation when necessary as described shortly
>
> Example:
>
> initiatePaymentExecutionTransaction (action term = initiate, control record = PaymentExecutionTransaction and there is no optional qualifier term)

For reference the 'control record' for a Service Domain is fully described in the How-to Guide – Design Principles & Techniques and was outlined earlier in this guide. In summary the Service Domain's control record combines the generic artifact (associated with its functional pattern) with the asset type is acts upon.

The introduction of behavior qualifiers to the specification of a Service Domain adds an additional level of detail. This can be applied to the description of its internal working, its governed information and of specific relevance here, can also be used to define a more specific/specialized service operation by adding the behavior qualifier as the optional service operation's qualifier term

The approach is best explained by means of an example. For the Current Account fulfillment Service Domain there are clearly many possible service requests: ordering a statement; requesting a funds transfer; setting up a standing order; etc. The associated candidate service operation for the Service Domain that would be used has the action term 'request':

> ***requestCurrentAccountFulfillmentArrangement***

Normally parameter fields within this service operation would be used to select the particular type of request. But this approach masks the particular purpose of the service operation when it is referenced in business scenarios and wireframes. Having

a more specific service operation in these model views would help remove this ambiguity.

The behavior qualifiers for the Current Account Fulfillment Service Domain break down its functions, listing options such as accessing the standing order facility and making payments/deposits. By adding a behavior qualifier to the service operation name, a new more specific service operation can be supported e.g.:

requestCurrentAccountFulfillmentArrangementStandingOrder.

At the time of this release there is no hard and fast rule as to when a type of service exchange is to be handled using a parameter field in the general default service operation or when a behavior qualifier is used to define additional finer grained service operations. It is likely to be driven by an analysis of the service operation payload. When the content is fundamentally different depending on the behavior qualifier there is a strong argument to define the next level of more specific service operations using the qualifier field.

3.1.5.4 Service Interaction Type

The BIAN service operations are defined to be implementation agnostic. However, in earlier releases four interaction types were used to characterize the nature of the service exchanges as a simple guide. In practice it has been found that in different deployment situations different types of interaction might sensibly apply and so even these high level characteristics were of little value and worse could be confusing. As a result these properties are no longer captured for service operations. For reference the types of interaction originally used were:

Request & Hold – the calling Service Domain can reasonably expect to get a response in a timely manner and so should wait for that response.
Request & Monitor – the requested service is likely to take some time and so the calling Service Domain may go off and do other things in the meantime, but also needs to monitor for the anticipated response in the future.
Hand-Off – the calling Service Domain passes off information and perhaps triggers additional actions in the called Service Domain, but anything that happens subsequently is of no interest to the calling Service Domain
Make Announcement – to establishes a notification service that may result in multiple returns

This checklist of properties may be of use for lower level implementation planning

3.1.5.5 Service Domain pre & post states

As described earlier each Service Domain has a primary functional pattern and for each functional pattern BIAN has identified its main externally visible states. The service operation definition includes a list of the allowed pre states that indicates the state that the Service Domain needs to be in for the service operation call to be valid.

In a similar manner, the post state list indicates the possible states the Service Domain can be in subsequent to the processing of the service operation call.

Because the analysis of states is limited to the main externally visible states the design insights provided are currently limited. It is anticipated that further analysis and insights may be obtained by expanding the definitions to include internal states and state transitions in later versions of the standard.

3.1.5.6 Input and Output Parameters

Four parameter fields are defined for both the call and response services of a service operation exchange. As described above, the content may take two forms. Past definitions used filtered checklists to describe the types of information. More recently the use of behavior qualifiers and the BIAN BOM results in more detailed and Service Domain/service operation specific content.

Content for the four parameter fields, both the original checklist descriptions and in selected cases the more specific content can be reviewed directly in the UML database and using various BIAN access tools. The structure for this content is as follows:

1. **Identifiers** – information items associated with a control record instance that can be used to select/identify that record. For example customer, location, product type, date/time, transaction reference…
2. **Depiction** – a broad range of information content that might be extracted from/recorded against a control record instance. The depiction content information for the checklist based service operations has been categorized into three general types – information items, information records (structured) and information reports (unstructured). For the more specific content being defined now these general categories no longer apply. The specific information elements each have their own associated information type
3. **Instructors** – provides control parameters that are used to govern the way the service operation is executed. For example they could define the time at which to perform an action and/or qualify the specific type of action to be performed and provide the result of the service operation call result if appropriate. (Note: the use of behavior qualifiers described elsewhere to define more specialized service operations is an alternative to using an instructor parameter to define the particular request.)
4. **Analysis** – references different historical and analytical views of individual control records and the whole portfolio of control records that can be maintained by the Service Domain. This analytical information is in addition to the control record instance information covered by the *depiction* parameter already described. Continuing with the *customer agreement* example, the Service Domain may maintain and provide on request analytical views of the make-up of the complete portfolio of all of the active customer agreements it maintains in terms of their properties, usage and processing status…

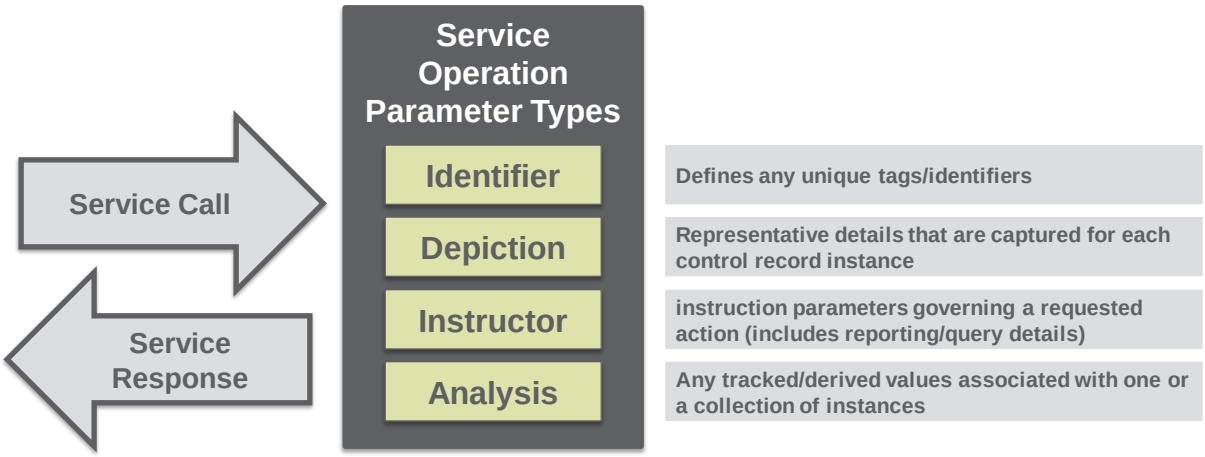


Figure 16: Input and output parameters for a service operation

The service operations in the BIAN Service Landscape contain descriptions of the parameter content at a level of detail that is intended to define the main information elements required to fully satisfy the purpose of the service operation. It does not include any optional, advanced or specialized features that may apply more selectively. It is also only intended to be sufficient for an architect or analyst familiar with the subject area to make an unambiguous assessment of the information needs and where appropriate be able to map this to underlying systems readable message specifications. The content does not attempt to be exhaustive/comprehensive as might be expected in training materials for example.

As noted, with the latest release BIAN has started to develop its own business object model (BOM) that is aligned to the Service Domain control records and where appropriate cross-referenced to the industry standard ISO20022 model. As the BIAN BOM is expanded techniques and guidelines will be defined to assist with the information extraction and mapping to service operations.

3.1.6 The Evolving BIAN SOA Framework

BIAN will continue to build out and refine the content of the BIAN SOA Framework. As it does so the number of Service Domains covered may increase and additional detail will become available in terms of extended service operation definitions, an evolving business information object model (BOM) and the overall scope of coverage of the Service Landscape. In addition to the standard canonical design specifications (Service Landscape, Service Domain and service operations). BIAN will continue to enhance and extend the supporting example materials such as the business scenarios and wireframes.

A portion of content is typically released with the classification ‘provisional/candidate’. This reflects the general working practice within BIAN where the BIAN techniques are used to define initial specifications that are then iteratively ratified and refined in practical use. The current classifications of stages of completion related to Service Domains and service operations are defined as follows:

- ***Provisional/Candidate*** – BIAN design techniques and mechanisms have been applied to create a complete set of default business events, first order business scenarios and candidate service operations with draft descriptive content for the Service Domain.

- ***Reviewed*** – the Service Domain's specifications, business events, first order interactions and available business scenarios/wireframes have been reviewed and amended as necessary by the owning Working Group and/or selected business specialists

- ***Ratified*** – the content has been applied in one or more production initiatives. This may result in further amendments to the specifications

Statement of Coverage by the BIAN standard

Note: the service operations defined by BIAN aims to reflect the mainstream behaviors of a Service Domain that would be common in the majority of deployments. The way some service domains operate in practice will evolve. New differentiating features may be developed by advanced organizations that in time may be adopted by the mainstream. Furthermore location variations may be required to deal with considerations such as geopolitical requirements and variations of operational scale. The purpose of the BIAN Service Domain partition is to define the core working of a discrete and generic business role. It recognizes that there may need to be site-specific adaptations and refinements in deployment, but the implicit service boundary/role should be stable regardless of these local enhancements.

In the future BIAN may consider ways it can capture and share prevailing practice details for Service Domains that include implementation level detail that may also highlight optional features as those just described. These informal design details would not be part of the formal standard but as with other example materials would be used to help adoption amongst the membership and beyond.

4 Content Development

The Section is set out as follows:

- Working Group Assignments – describes how the overall BIAN Service Landscape is assigned to different specialist Working Groups. It outlines how the Working Groups are supported by the Central Team and how their interactions are coordinated where interests overlap. It also notes that provisional First Order Interactions are currently defined by the central team
- Building Content in the Working Groups – describes the steps followed in the definition of content and the current state of tooling and specialist support
- Semantic API Initiative – describes the working approach for a major initiative that is driving the development of content across the landscape in coordination with the Working Groups where necessary

4.1 Working Group Assignments - Governing Service Domains

The central BIAN Service Landscape team assigns the governance responsibility for Service Domain content development to BIAN content Working Groups. BIAN's internal organization, including its support teams make-up and the individual charters of the Working Groups can be found on the BIAN WIKI. Each Working Group has responsibility for a collection of Service Domains, service operation descriptions and the associated Business Scenarios/wireframes matching their specific area of business expertise.

BIAN has agreed Working Groups with collective coverage for the whole landscape (though not all may be active at any one time) and more importantly has provisionally and uniquely assigned all identified Service Domains. It is not unusual that some Service Domains found on the 'cusp' between areas of specialization subsequently get 'reassigned' between Working Groups based on their preferences and with the oversight and authorization of the central BIAN Service Landscape team.

The Working Groups are supported by a central BIAN team as described at the start of this guide. The central team provides administrative and logistical support, technical and architectural guidance and advice. In addition the central team helps coordinate between Working Groups when design decisions impact multiple Working Groups. Members of the central team attend all Working Group meetings to provide this support and ensure the necessary coordination is provided.

More recently the central team has taken on providing support for the input of content using the different BIAN facilities. This function has several advantages including:

- It reduces the manual workload on the Working Group volunteer members allowing them to focus on business specific review and definition activities
- It provides a mechanism to ensure the BIAN designs and techniques are correctly interpreted as the content can be reviewed for compliance as it is entered

- It enhances the consistency across the Working Groups as all content passes through the same central process, eliminating unnecessary variations

First Order Interactions

The first order connections can be considered to represent an aspect of the Service Domains foundational specification. The development of the First Order Interactions had been assigned to Working Groups in the past. The central team now generates the provisional First Order Interactions. This approach makes more effective use of the scarce resources of the content Working Groups.

Provisional first order connections can be quickly defined using business event analysis as described below. The provisional specifications are then ratified and enhanced when they are used to assemble more sophisticated business scenarios and wireframes by the content Working Groups or in actual deployment projects (such as those linked to the BIAN Semantic API initiative)

The approach for defining the provisional first order connections is a streamlined version of the approach used in prior releases with the following five steps:

1. **Define a reference 'wireframe' for the target business area/domain** – the wireframe captures the anticipated service operation connections between a group of Service Domains (this can be ratified as the events are modeled)
2. **Identify Primary and Secondary Service Domains** – primary Service Domains represent those for which events will be defined. Secondary Service Domains are any peripheral capabilities that may be accesses by the primary Service Domains
3. **Define Business Events for the Primary Service Domains** – four established BIAN categories are used to classify the business events :
    a. Origination – results in a new control record instance
    b. Invocation – acts on an active control record instance
    c. Reporting – provides information about one or more active instances
    d. Delegation – results in service calls to other Service Domains
4. **Capture service operation connections** – The service operation connections are codified and captured in the tooling environment for each business event
5. **First Order Interactions** – the final step uses the BIAN Workbench tool to define a simple/constrained business scenario view of the interaction for each event for reference purposes

The definition of First Order Interactions will first be targeted to support the Semantic API initiative and then prioritized across remaining areas of the Landscape as time permits

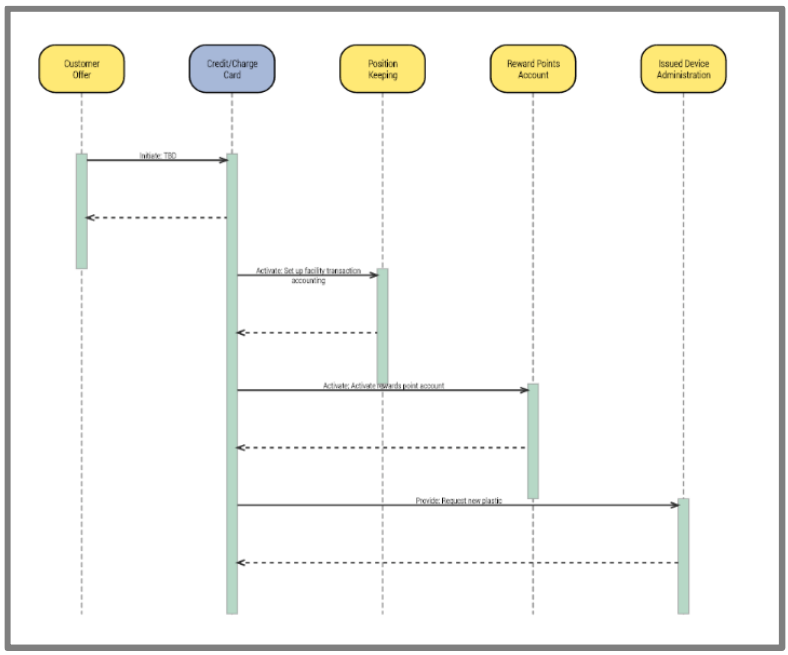
First Order Interactions have a standard representation. They all have an associated primary Service Domain. A business event defines where either some external call or an internal trigger causes the primary Service Domain to act. In responding to the business even the primary Service Domain may invoke the services of one or more Service Domains through delegated service calls.

The First Order Interaction does not capture how any of the delegated to Service Domains may themselves need to delegate to other Service Domains – i.e. it does not show any second and higher order delegation dependencies. This is because

- The First Order Interaction is only intended to show how the primary Service Domain responds to an event
- The First Order Interaction assumes full encapsulation. SOA principle of encapsulation is that a called service is provided in a manner that hides any service dependencies of the called service provider. The caller should not know nor care how the service provider fulfills its offered service responsibilities.
- More complex business scenarios will often reveal second and higher order dependencies and these expose implementation level considerations for scheduling and performance but these will tend to be site/situation specific

The representation of a complex business activity in a business scenario and/or wireframe view will typically combine a collection of several first order interactions and the triggering of events and responses can be traced to ensure the proper encapsulation of each Service Domain. In this way business activity is better represented as loose coupled, networked interactions between multiple Service Domains, each responding to service calls as appropriate. The model view created is not a sequential process that in effect anticipates and imposes behaviors that can contradict the foundational SOA principle of service encapsulation.

The way Business Scenarios and Wireframes are defined is covered in earlier sections of this guide. An example of a First Order Interaction as documented using the BIAN Workbench is shown below:



The diagram shows a simple business scenario as developed using the BIAN web tool. Credit/Charge Card is the Primary Service Domain - called by Customer Offer to initiate new product set up, resulting in a number of immediate delegated calls

Figure 17: Example First Order Interaction captured on the BIAN Workbench

4.2 Building Content in the Working Groups

This Section outlines the working practices followed by the Working Groups when creating content. As working practices constantly evolve the approaches described here can differ significantly from those described in earlier versions of this guide. There is a formal quality assurance and release process for subsequently publishing the Working group content. These procedures are fully documented on the BIAN Wiki and in associated BIAN procedures.

BIAN usually initiates an iterative specification process with the initial definition of 'candidate' content by dedicated central resources. In this way business specialist/practitioners can focus on confirming and refining designs rather than having to create them from scratch, making more efficient use of scarce resources. Candidate/provisional content is reclassified as 'reviewed' once it has been assessed and refined as necessary by the Working Groups.

In general terms the content development approach in BIAN combines two parallel design activities as shown in the '2-Cycle' model shown simply in the figure below:

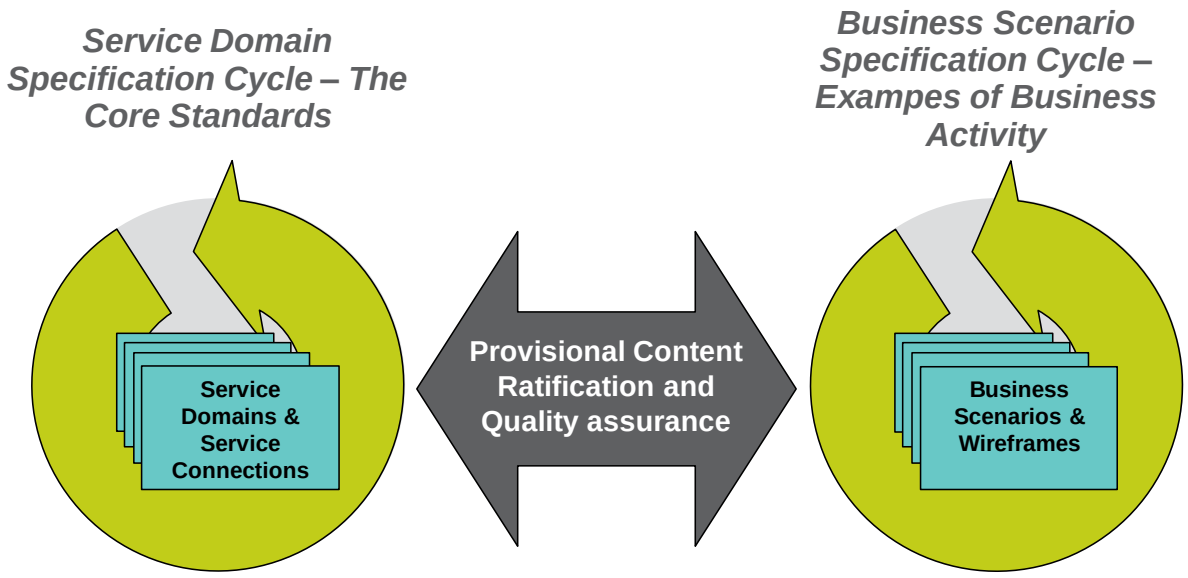


Figure 18: Simple 2-cycle Model

On the left the emphasis is on developing provisional designs for the foundational elements of the BIAN standard – the Service Domains and their service operation connections. On the right the emphasis is on modeling known business activity using these foundational elements in order to ratify the designs and to provide examples that can guide adoption. These examples are captured in Business Scenarios and the associated Wireframe models. The content of the designs then maps to the evolving BIAN BOM for the detailed information specifications.

This balance of effort between the provisional foundational elements and worked examples is increasingly reflected in the working practices of the BIAN membership. The central architectural teams tend to spend more time driving the definition of

provisional designs that conform to the BIAN design principles and the experienced business practitioners in the Working Groups then spend more time testing out and refining these provisional designs in the context of real world examples.

The result of the iterative exchange between the two activities is the ratified specification of the service operation connections between the Service Domains, mapped to the semantic information definitions captured in the evolving BIAN business object model. The activity on the left ensures that these design conform to the BIAN design principles so that the specifications are canonical and properly partitioned. The activity on the right as well as ratifying the designs produces a key by-product: examples of use in the context of real business situations that helps with the correct interpretation and deployment of the BIAN standard.

As described in more detail in the How To Guide – Design Principles & Techniques and outlined earlier in this guide, the artifacts making up the foundational elements broadly define the 'static' elements or building blocks/components of the BIAN standard that are then assembled to support the 'dynamic' behaviors that represent practical examples of real world activities that systems aligned to the BIAN standard can effectively support.

The more specific design steps are shown in a more complete representation of the 2-Cycle approach.

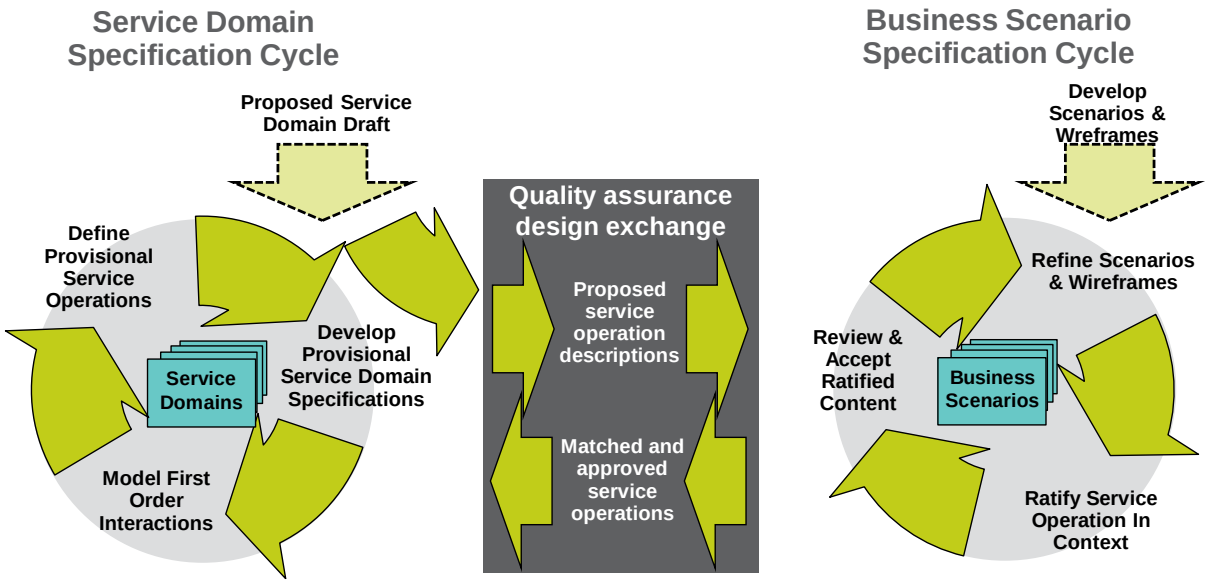


Figure 19: Detailed 2-cycle model

In practice the precise approach varies from Working Group to Working Group but in general terms the two perspectives (specifying the foundational building blocks and testing and refining these in worked examples) are developed iteratively and in parallel with tasks coordinated as outlined in the primary flows shown below.

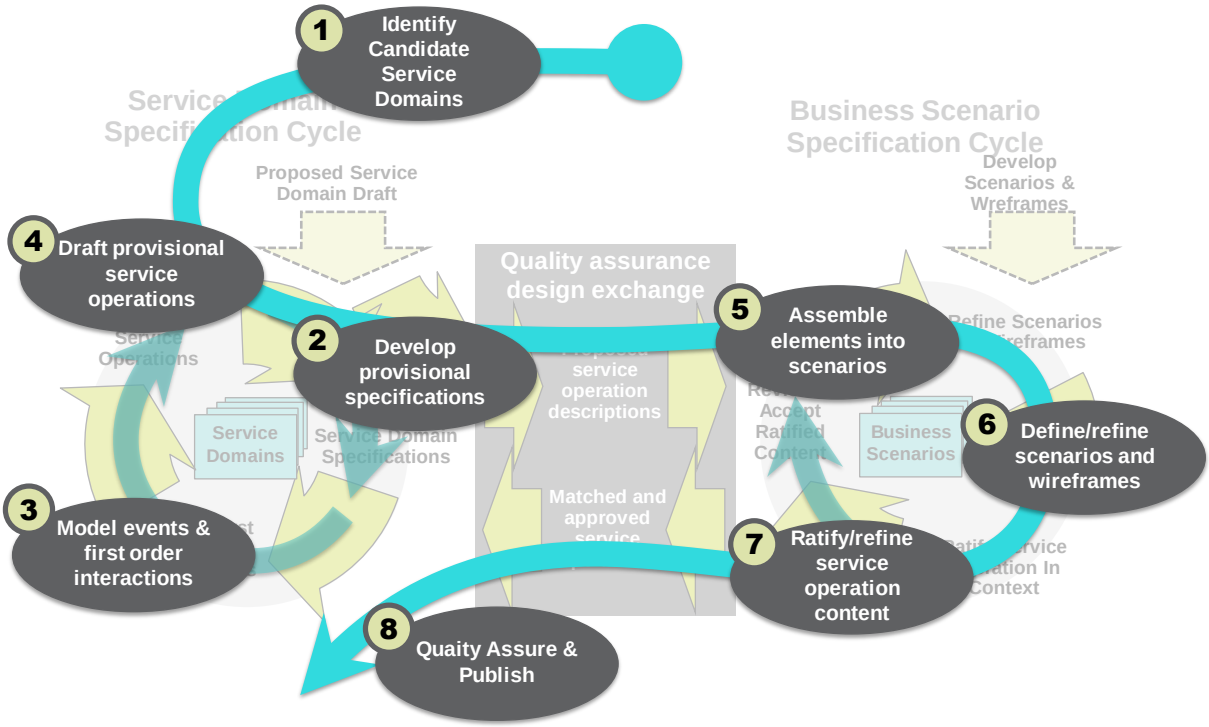


Figure 20: 2-Cycle model with steps

The central team and architectural Working Groups focus on steps 1-4 (and 8) and the domain/content Working Groups focus on steps 5-7:

Step 1 – Identify Candidate Service Domains – less frequently, new Service Domains are identified or existing Service Domain roles are reviewed/refined.
Step 2 – Develop Provisional Specifications – applying the BIAN design principles, define the Service Domain specification including selected functional pattern, asset type, control record, behavior qualifiers and information profile
Step 3 – Model Events & First Order Interactions – list likely triggering events and resolve first order connections.
Step 4 – Draft Provisional Service Operations – list key semantic information content for the service operations – map to BIAN BOM where content is available
Step 5 – Assemble Elements Into Scenarios – select and outline real world business scenarios, assembled using the foundational BIAN components
Step 6 – Define/refine Scenarios & Wireframes – ratify and refine/augment business scenario definitions and the underlying BIAN artifact specifications. Develop wireframe views for related collections of business scenarios when appropriate
Step 7 – Ratify/Refine Service Operation Content – ratify and refine the semantic information content of the service operation connections. Note this may be based on actual implementation work where possible.
Step 8 – Quality Assure & Publish – consolidate, quality assure, capture and publish content

4.2.1 BIAN Vocabulary and Business Object Model

BIAN maintains a vocabulary that defines all BIAN specific terms and has started to develop a business object model (BOM) that captures the business concepts referenced in the Service Domain and service operation specifications. The BIAN BOM is informed by the industry standard ISO20022 model.

The Working Groups are supported by the central team to ensure that the BIAN terms and definitions used align to the vocabulary and that business content is correctly represented in the BOM

4.2.2 Developing Business Scenarios & Wireframes

The business scenario remains the key mechanism for reviewing the context and purpose of service operation exchanges between Service Domains. The eight-step process described above follows the development of content. The Working Groups engage in this process primarily by selecting and developing the business scenarios that address some aspect of interest within their area of the Service Landscape as reflected by steps 5-7.

The make up of the Business Scenario specification and the associated Wireframe view is described earlier in this guide. The focus of the Working Group is to list the business scenarios of interest and then document these scenarios that can increasingly be assembled from First Order Interactions where available. When appropriate the Working Group may also develop a Wireframe in collaboration with the central team to capture the working structure of a collection of related business scenarios.

As noted earlier the definition of the Business Scenario includes a description of the start and end points, the business purpose or goal and narrative that explains the rationale behind the various service operation exchanges. As the business scenarios are refined the specification of the Service Domains, service operations and service operation connections are ratified.

Furthermore the Working Groups define and refine the semantic information content of the service operations. This can include referring to and enhancing the content of the evolving BIAN BOM in coordination with the central team (specifically the Information Architecture Working Group).

As noted the Business Scenario is not a formal definition of the required sequence of actions, but instead provides a contextual example of Service Domain behaviors that is useful to better understand the roles of Service Domains and the service operation exchanges between them. The Business Scenarios defined and maintained by a Working Group will typically be the first design artifact that is used to access the BIAN standard

4.2.3 Modeling referential dependencies

The business scenario is a useful mechanism to capture transactional activity such as the response required to handle some kind of business event like a payment transaction or customer servicing request. There is a second type of Service Domain interaction traffic that occurs much more as a background activity that is not always easily captured by analyzing the more dynamic responses to business events.

This type of traffic supports the background coordination between Service Domains where one Service Domain needs to reference information governed by another Service Domain or obtain resources and other support services. Rather than doing so 'in-flight' as part of some response to a single business event the coordination between the Service Domains is more likely to be handled as an ongoing background activity. The business information and services that are more typically exchanged through background exchanges includes maintaining the prevailing bank rates and service charges that may apply, access to the regulatory rules and internal policies it needs to conform to in general, the provision and support of building facilities, equipment and staff.

The more comprehensive business event descriptions used with the definition of First Order Interactions captures these background exchanges and this coverage will be improved as more events and the associated first order business scenarios are developed spanning the complete Service Landscape.

4.3 Semantic API Initiative

This Section outlines the specific working practices of the BIAN Semantic API Initiative. This initiative is defined in greater detail in an associated guide – The BIAN Semantic API – How To Guide that can be found on the BIAN Wiki. Plans and deliverables can be found on the associated workspace of the Semantic API Working Group.

The Semantic API Initiative is a major undertaking within BIAN to develop and package BIAN specifications in a form that can be used for 'aligned' API development. With the latest release BIAN has published extended Service Domain and service operation specifications and business object model views for areas of the landscape covered by the first cycle or 'Wave 1' of this initiative. Wave 1 content covers:

- Mobile access – both direct customer access and via a third party agent using any appropriate channel. This includes various security and routing capabilities
- Customer On-boarding – the procedures followed to establish a new customer and then the offer process followed with the set up of a product or service
- Payments – with specific focus on consumer payment activity as covered by the European PSD2 initiative
- Consumer Loan – the basic access to a simple unsecured consumer loan

Subsequent waves are planed to extend coverage across all main product and service activities of the landscape.

The API initiative combines two deliverables:

1. A BIAN API Directory – that uses the BIAN Service Landscape, Service Domain and service operations to classify available APIs for reference
2. Extended BIAN Specifications – to support API development

Each is outlined briefly here but first there is an explanation of different levels of sophistication for BIAN aligned API implementation.

4.3.1 Three Levels of Sophistication in the deployment of APIs

The implementation of open APIs varies greatly depending on the technical architectural approach adopted. BIAN defines three distinct types of technical solution that corresponds to differing levels of sophistication.

1. ***Direct to Core*** – the service exchange accesses the host facility directly.
2. ***Wrapped Host*** – the host systems are accessed through a control/wrapping middleware such as an enterprise service bus (ESB)
3. ***Micro-service Architecture*** – the applications are implemented as a network of service enabled, discrete capabilities that can support external access directly

Key properties of each level and the business rational for adopting each is summarized in the table:

	Level 1. Direct to Core	Level 2. Wrapped Host	Level 3. Microservice Architecture
Definition	The API routes direct to the core system providing the service. Intermediate channel based access control and 'buffering' is required	integrating service middleware – a service bus – 'wraps' the host systems. The service bus can offer various host access mitigation capbilities/enhancements	The host services are implemented as loose coupled microservices with complex interactions supported by sophisticated middleware
API Service Description	Read only or simple 'atomic' update transactions supported by a single host system. The solution is likely to be host application specific	Enhanced 'simple access' services aligned to establish standards. Wrapping may enhance service capabilities and some hosts may support more complex exchanges	Support for flexible and complex interactions involving multiple business activities and processing/decision chains
Examples	◆ Retrieve a balance/account statement ◆ Reference a product/service directory ◆ Initiate a payment	Message conforms to industry standards (e.g. ISO20022) ◆ Retrieve a balance/account statement ◆ Reference a product/service directory ◆ initiate a payment ◆ Customer on-boarding/offers	◆ Prospect on-boarding and origination ◆ Customer dispute/case resolution ◆ Customer relationship development/up-sell/cross-sell campaigns ◆ Third party service integration
Business Drivers	Provide application based access to an established/existing type of customer exchange	Provide application based access with a high degree of standards alignment. Mask/augment host/legacy system limitations.	◆ Support sophisticated interactions ◆ Support new business models ◆ Support for 3rd party integration ◆ Leverage advanced technolgy/architecture

Figure 21: Summary of the BIAN API Levels of sophistication

4.3.1.1 Direct to Core

The lowest level of sophistication and the easiest to implement involves constructing a front-end capability to manage external access security and then typically re-package existing host interfaces to support an API. A typical arrangement is as shown that shows direct customer access to the Bank (from an API linked to their personal device) or via a third party service provider:

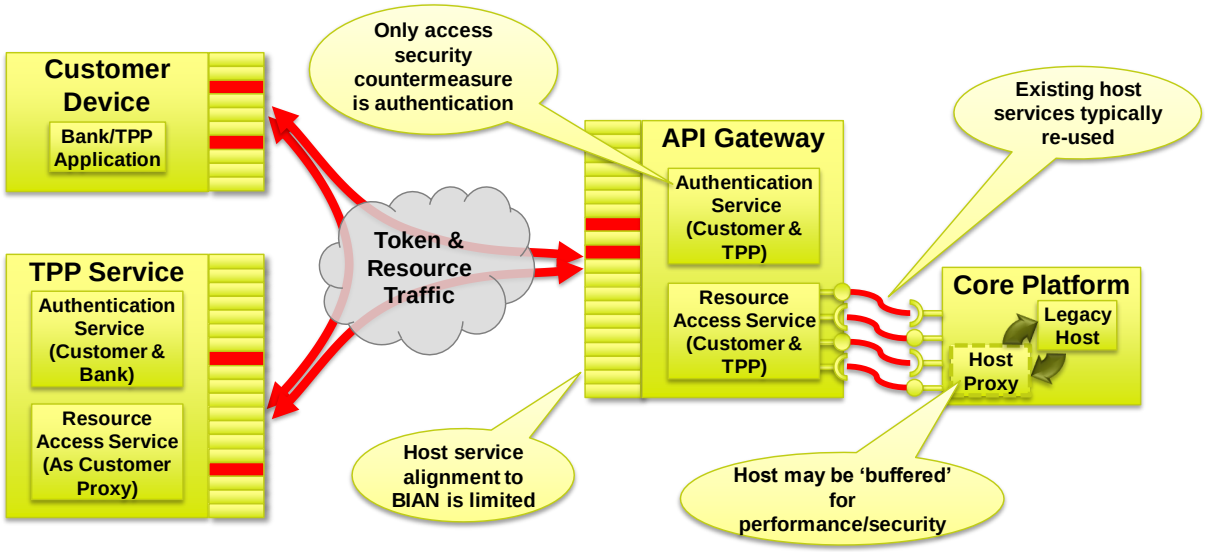


Figure 22: Level 1 layout

Key Aspects of the Approach

At this level the changes required of host systems are kept to a minimum but the facilities that can be supported are limited to repackaging existing services that can be accessed through an API front-end platform.

External access control is implemented using access tokens handled by an authentication service capability. Access sessions will typically be limited to single task exchanges that target an individual host system.

Host access may be direct or host production systems may have a proxy implementation that duplicates aspects of the host system to provide additional access control/security.

API services can be mapped/classified against BIAN Service Domain service operations. It is likely however that there will be significant host system specific features exposed through the API.

4.3.1.2 Wrapped Host

The second level involves the integration of a host access middleware that mitigates host systems shortfalls. The middleware, typically some form of enterprise service bus (ESB) can provide a range of enabling facilities including:

- *Host Access Session Management* – supporting host access 'sessions' that can span multiple external access events
- *Data Caching* – persisting frequently accessed host data to minimize host access traffic
- *Host Wrapping* – adding function and data to mask host system shortfalls
- *Resolve Data Fragmentation* – enforcing master/slave data governance techniques within the application portfolio
- *Advanced Look-up* – using access patterns to anticipate needs and obtain host data in advance to minimize host access latency
- *Transaction Persistence* – provide facilities to track customer 'transactions' between contacts and potentially transactions spanning multiple systems

Again customer access can be direct or via a third party service provider and front-end authentication is the main security countermeasure.

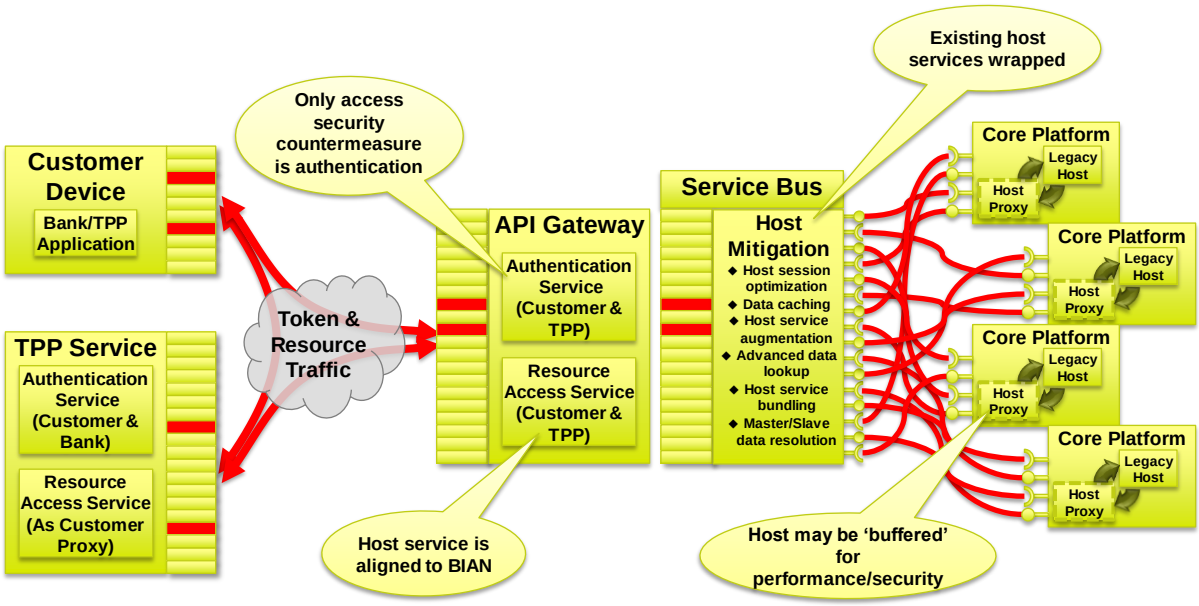


Figure 23: Level 2 layout

Key Aspects of the Approach

The main purpose of implementing a host-wrapping layer is to repurpose or extend the life of existing legacy systems and enable greater re-use of business functionality. In addition to addressing the listed shortfalls and improvements API services are mapped/classified to BIAN and the ESB wrapper can be used to mask host specific features improving the standards alignment.

Wrapped host services can also support front-end (client side) application assembly approaches but this type of solution development is not shown in the figure or considered here in any detail.

4.3.1.3 Micro-Service Architecture

The most sophisticated level is where the host systems conform to a Micro-services architecture with Service Domains (or groups of closely related Service Domains) acting as autonomous service 'containers' in a loose coupled service network. In this configuration a particular collection of Service Domains manage customer access, providing comprehensive services including access security, activity tracking and intelligent routing decisioning.

A micro-service platform that manages external access can link to different host configurations. The figure below shows how a customer access micro-service platform allows managed access to host systems conforming to different levels of API sophistication (Direct to Core, Wrapped Host and Micro-service configurations).

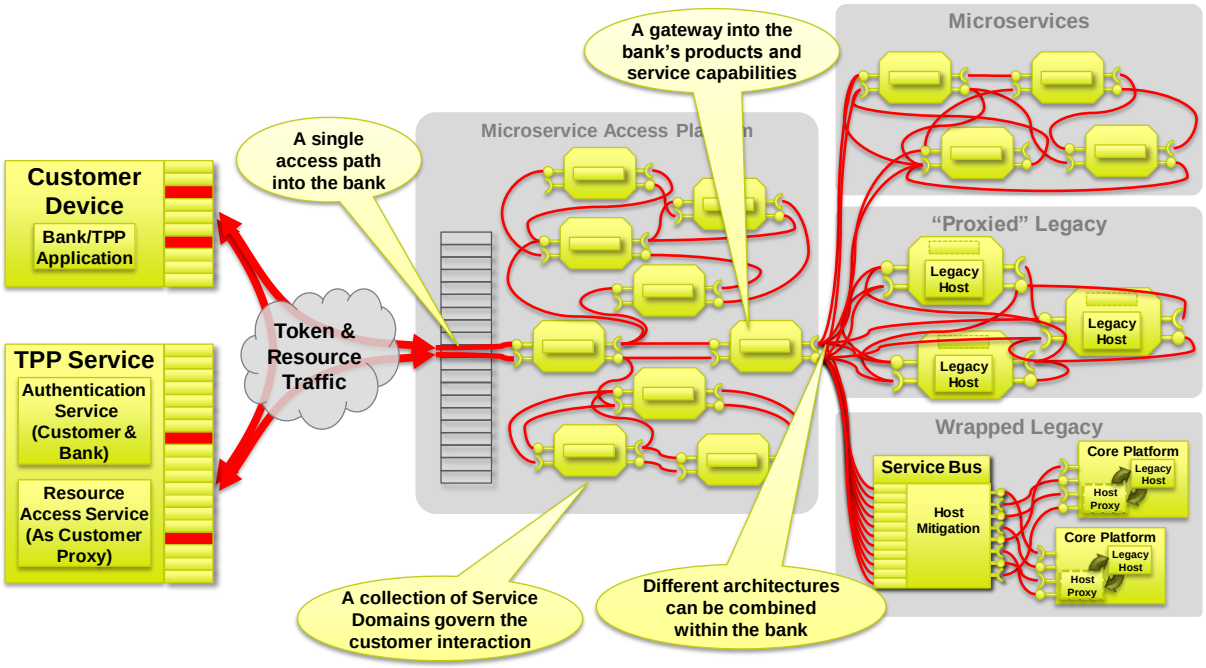


Figure 24: Level 3 layout

Key Aspects of the Approach

The Micro-service architecture approach needs to be considered in terms of two distinct aspects. The first as mentioned is a micro-service based customer access platform that may include a range of facilities and utilities that support external customer access again possibly through third party intermediaries. The second is the bank's product and service capabilities that may increasingly be supported using systems conforming to a micro-service architecture where this is appropriate.

A key advantage of aligning to the BIAN Service Domain and service operation standard for level 1 & 2 solutions is that these interfaces can be later integrated with a level 3 front-end micro-service platform with manageable amounts of re-working.

4.3.2 The BIAN API Directory

The BIAN Service Landscape represents a complete inventory of Service Domains and their associated service operations. It can be used as an organizing framework for categorizing/classifying available APIs. By mapping an API to the corresponding service operation(s) for the providing Service Domain it can be uniquely classified.

As the inventory is populated with references to available open APIs users will be able to identify potential solutions for specific purposes. There is always likely to be implementation and mapping work to do to deal with practical aspects of the API implementation, but the addressed business requirement can be well matched.

An initial review of the Service domains has been made to determine which Service domains are likely to provide external access (through an API). The Service Landscape below has been color coded to show this classification. Service Domains highlight in green represent business functions that provide cross product or utility type services. Service Domains highlighted in red represent business functions that are specific to a particular product.

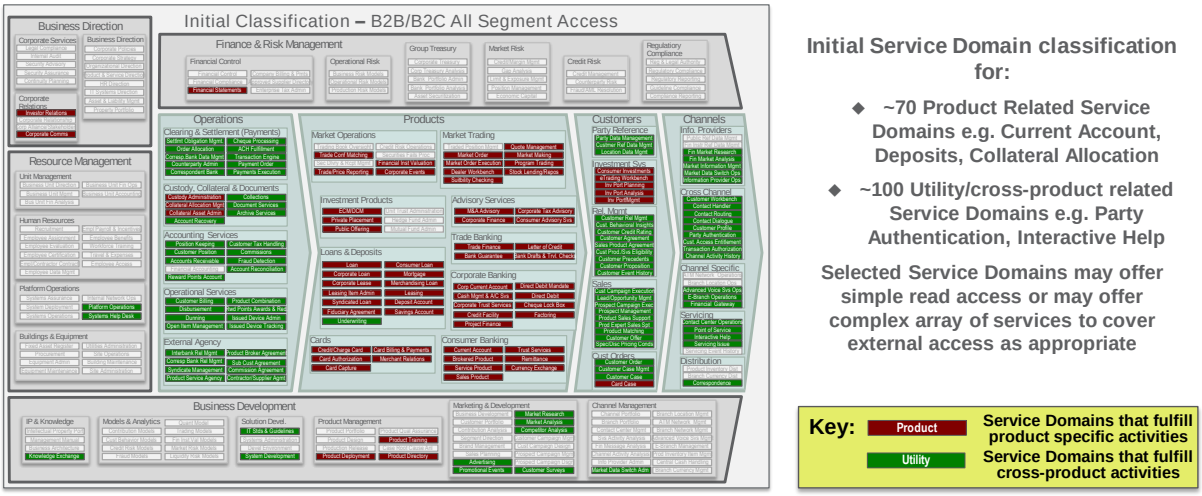


Figure 25: The Service Landscape with Open API candidates

Every Service Domain has an associated set of default service operations. Some of these support command and control activities that are unlikely to be exposed through APIs and so they have not been included. The other service operations are listed with a brief description to help with mapping.

These service operation descriptions are being refined as BIAN develops extended definitions of the Service Domains with this API Initiative. The wave 1 content listed

earlier has these extended definitions that can be reviewed

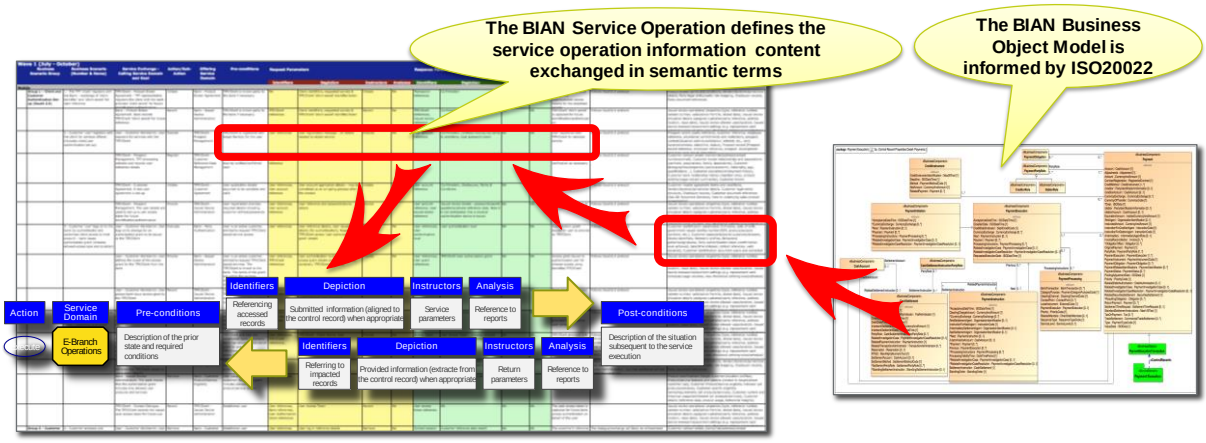


Figure 26: Example Wave 1 service operation description (Excel)

As BIAN expands the coverage of its BOM and the extended specifications are used in deployment the service operation descriptions will be updated with more specific information content.

4.3.3 Developing Semantic API Designs

In this initiative BIAN is developing extended Service Domain and service operation specifications across the Service Landscape. The various design artifacts are outlined below:

- *Extended Service Domain Specifications* – an additional level of design specification has been added to the Service Domains to ensure consistent interpretation of the business purpose behind the service operations
- *Wireframes (showing Enterprise Boundaries)* – wireframes present the collection of Service Domains and their service operation connections that support some aspect of business operation. The wireframes are adapted to show external (3rdparty) activity alongside internal bank flows
- *Enhanced Business Scenarios* – the BIAN business scenario definition has enhancements to clarify the reference to specific business information exchanges (service operation connections)
- *Service Operations (Touch Points)* – Individual service connections are described in more detail to support their adoption in API design for both external (B2B/C) and internal (A2A) traffic. BIAN intends to augment the service operation descriptions with example pseudo code definitions of the semantic information content

Extended Service Domain Specifications

The additional design concept employed is the ‘behavior qualifier type’ described earlier in this guide. The behavior qualifiers defined for a Service Domain are used in two main ways. One, they are used to provide a more detailed definition of the

business information governed by a Service Domain (which feeds into the message content for its offered services). Two, they are used to provide greater precision to the purpose of the offered service operations. The extended definitions for some Service Domains taken from the Wave 1 content is shown in the following Excel extract:

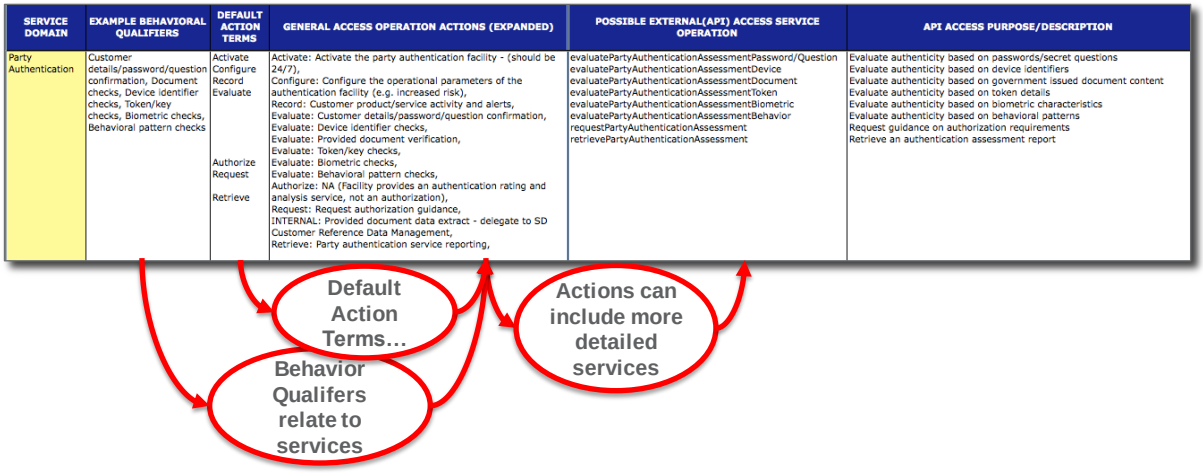


Figure 27: Extended Service Domain specifications (Excel)

Wireframes (showing enterprise boundaries)
For the Semantic API content the wireframe view has been adapted to show the structures within and between entities that may interact with the bank, including the customer, third party solution providers, network providers and regulators.

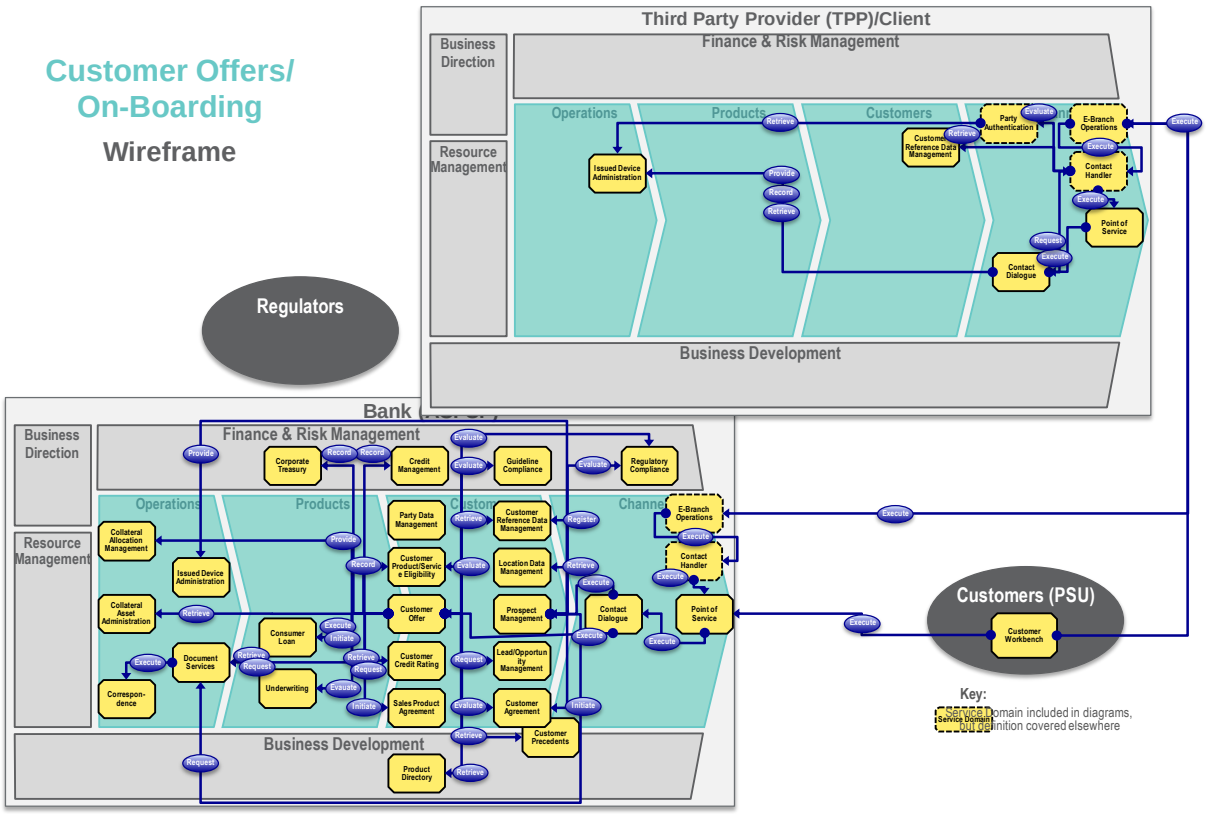


Figure 28: Wave 1 Wireframe example

A further classification of the Service Domains can be considered, showing the time dependency between Service Domains for service operation exchanges. This will usually be an implementation specific property. An example classification of these dependencies is shown in the mobile access wireframe below:

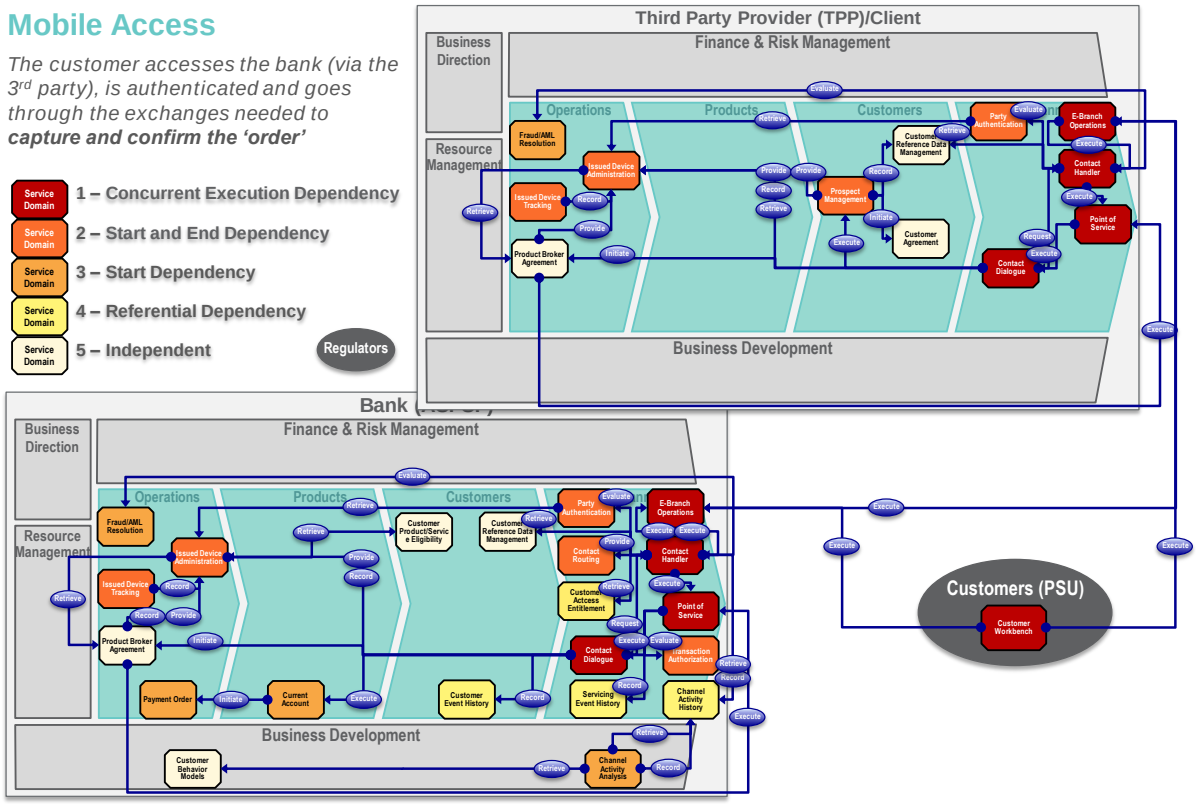


Figure 29: Mobile access Wireframe with time dependencies

Enhanced Business Scenarios

For the Semantic API specification the normal BIAN Business Scenario layout has been extended to show the boundary between the bank and other interested parties (a vertical black line delimits Service Domains within each operating entity).

The scenario template also shows the key business information exchanges between the source (calling) and consuming (offering) Service Domains at the bottom of the figure. These exchanges are tagged to the offering service operation of the called Service Domain. This matched service operation provides the description of the semantic business information content that would need to be exchanged through an API.

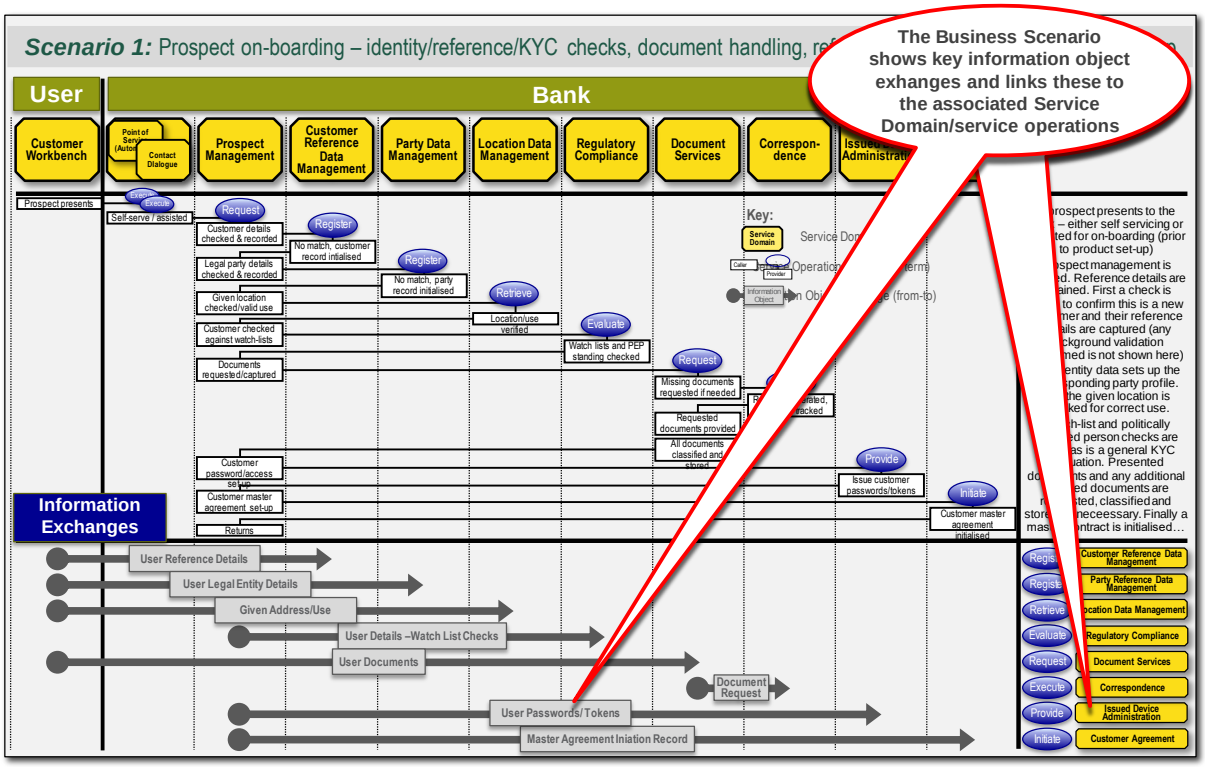


Figure 30: Extended Business Scenario

Service Operations

The final design artifact of interest for API design is the service operation description itself that underlies the wireframe and business scenario examples. As the specifications are used in practice BIAN will provide detailed descriptions of the business information content in a suitable pseudo code form to provide a starting point for API solution development.

At this stage BIAN lists indicative semantic information content based on the business information profile of the Service Domain. Where possible key information elements from the Service Domain's control record have been mapped to equivalent elements in the ISO20022 Business Object Model to provide additional detail.

The initial specification of a service operation is shown in the Excel table:

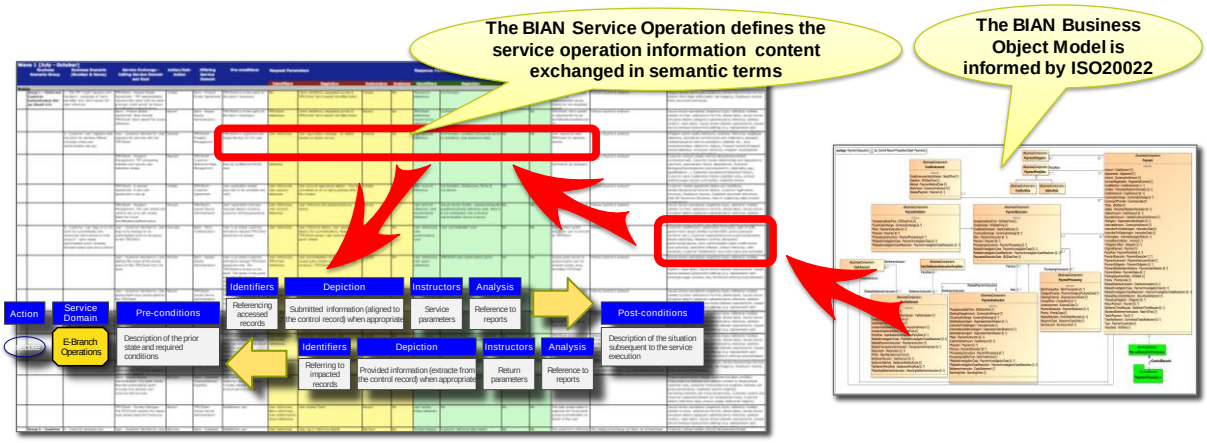


Figure 31: Service operation definition (Excel)

4.3.4 Applying the BIAN Semantic API Designs

The Semantic API Working Group is developing a practitioner guide/cookbook for implementing APIs using the BIAN standard. The BIAN Semantic API guide already mentioned is targeted at technical architects who may need to understand the design concepts behind the specification in more detail.

Key aspects of the approach are outlined for the different levels of sophistication in the How to Guide – Applying the BIAN Standard. These will be updated with the release of the cookbook in the near future.

5 Conclusion

This document that covers the way the BIAN membership develops design content is intended to provide guidance to new BIAN members and Working Group participants. It describes all of the components making up the BIAN SOA Design Framework and explains some of the more detailed aspects of the specification procedures.

In defining the working approaches the document provides brief explanations of some of the concepts applied without going into a level of detail that could be distracting. (The design principles are fully covered in other documents of the 'How-to Guide' series). Members that are interested in understanding the BIAN design concepts in more detail should reference this guide.

The primary approach is described in the sequence that new members would typically work through for content development – from selecting a target business event, modeling a Business Scenario, to identify the Service Domains involved and finally specifying the exchanges realized through their service operations. This guide also describes the specific approach developed to support the BIAN Semantic API Initiative.

The document also outlines several of the tools and support facilities that are available to the working groups. As with all the How-to Guides, BIAN will endeavor to update this document as BIAN's organization, working approaches and supporting tools and techniques evolve.