BIAN Capstone Project
API Classification Guideline for BIAN Architecture

December 17, 2015

Carnegie Mellon University - Heinz College

Project Team:
Chaitanya Eranki
Hideaki Jim Nakajima
Prajwal Niranjan
Tomofumi Ueno
Wei Wang

Faculty Advisor:
Michael McCarthy

API Classification Guideline for BIAN Architecture

Table of Contents

1. Executive Summary …..3

2. Introduction …..4

    2.1. About BIAN …..4

    2.2. About CMU Capstone …..4

    2.3. About Project - Project Goal, Project Scope …..4

        2.3.1. Problems …..4

        2.3.2. Objectives …..5

        2.3.3. Scope …..5

        2.3.4. Steps …..5

        2.3.5. Timeline …..6

3. BIAN Architecture …..7

    3.1. Service Landscape …..7

    3.2. Service Domains …..9

    3.3. Service Operations …11

4. Classification of service operations …13

    4.1. Classification based on communication type …13

    4.2. Classification based on level of detail …14

5. Business Scenarios …15

6. Conclusion …18

    6.1. Findings …18

    6.2. Future Research …18

    6.3. Lessons and Learns …18

7. Appendix …20

    7.1. Business scenario …20

    7.2. Business Scenario Exercise result …25

7.3. Refernces …25

7.4. About CMU Team …26

7.4.1 Team Members …26

7.4.2 Faculty Advisor …26

1. Executive Summary

In Financial industry, many API are released from bank and IT vendors by improving information technology. Especially, mobile application accelerates promulgation of financial API. At the same time, spreading financial API brings increased information security risk such as leaking bank account information due to lack of financial API standards. In this situation, BIAN and CMU are trying to create API guideline by utilizing SOA based BIAN standard, BIAN Service Landscape in this project. The project team determine API as API contents design, such as what information exchanging in the API. The API classification guideline describes API classification type and classification procedure for each banking business process.

Regarding API classification type, the guideline categorizes the banking businesses process from two angles that are a data type of the business information and a communication type with the other process. First, data type, the guideline uses three Tiers approach that focus on the data structure of each information. Tier 1 (Detailed) is all information of the process are structured data. Tier 2 (Mixed) contains both data types that are structured data and unstructured data, in the process. Tier 3 (Generic) is the other end of Tier 1 that all information are unstructured data. Second, communication type, the type also has three groups, “Machine to Machine (MtoM)”, “Machine to Person (MtoP) / Person to Machine (PtoM)”, and “Person to Person (PtoP)”. The communication type looks at the interaction between two banking business processes.

API classification procedure is determined through CMU team business scenario exercise. They conducted the exercise for 5 business scenarios that contained payment transaction business and loan origination business. This 5 business scenario covers 23 business process; the process is called Service Operation in BIAN Service Landscape. Each Service Operation contains data items, and CMU team evaluated the data to classify Service Operation into each data type and communication type. By iterating the business scenario exercise, CMU team standardize the evaluation process into API classification procedure to expand the study for the other business scenarios.

After analyzing the result of the business exercise, CMU team conclude that API should standardize for each Service Operation. Also, there is relevancy among data types and communication types by classifying Service Operation. Classification of Service Operation converges with three groups, “Tier 1 – MtoM”, “Tier 2- MtoP / PtoM”, and “Tier 3 - PtoP”. Also, they recognized this tendency might change by improving text analytic capability. CMU team experienced iteration of business scenario exercise brings new finding in the project.

2. Introduction

2.1. About BIAN

BIAN (Banking Industry Architecture Network) is an independent, member-owned not-for-profit association, founded in 2008. More than 60 entities (Financial Institutions, IT Service Providers, and Educational Institutions) have collaborated as BIAN members. BIAN has defined a Service Oriented Architecture (SOA) based standard of IT services for the banking industry called the Service Landscape. The Service Landscape is published periodically in collaboration with leading-industry entities with the most recent version being 4.0.

BIAN's mission is to support the banking industry using a flexible and semantic architecture for banking businesses. BIAN evaluated current banking industry picture that "Banks are enabled to develop their semantic service definitions on a consistent basis through BIAN to enable internal and commercial SOA-based solutions according to a standardized industry model".[1]

2.2. About CMU Capstone

Students in the School of Information Systems and Management, Heinz College, Carnegie Mellon University, are required to work in a team-based and IT-related project with clients in their final semester. This is called a Capstone Project. The clients vary in type and size: from for-profit to non-profit organizations, from startups to Fortune 100 companies. The project's objectives and size also vary depending on the needs of the client. Through the capstone project, CMU students contribute to the partner organizations to achieve their mission and, at the same time, have an experience where they can apply their classroom study to real-world scenarios.[2]

2.3. About Project - Project Goal, Project Scope

2.3.1. Problems

There are no common realizable API services available among Financial Institutions. The missing services become the business opportunity for financial technology startups. However, the startups solutions increase information security risk due to the accessing bank with the

[1] BIAN Website. Mission. Retrieved from https://bian.org/about-bian/mission-strategy/
[2] Heinz College Website. Student Capstone Project. Retrieved from http://www.heinz.cmu.edu/partnerships/student-projects/index.aspx

non-secure method. In this situation, bank clients and banks need a common standard to prevent the information security issues. [3]

2.3.2. Objectives

The project team sets two objectives to solve the above problems. [4]

- To Provide a ubiquitous set of APIs that would enable the innovation that clients seek while retaining security, as well as the telemetry around user activity for banks to tailor and market new products.
- To Produce a set of "technology agnostic APIs" that can be consumed by members of the BIAN community and external to it.

2.3.3. Scope

Following three points are deliverables to achieve the project objectives. [5]

- Technology agnostic API specification to agreed upon level
- Classification of these APIs
- Methodology guide to drive from business architecture to realizable application

2.3.4. Steps

We conducted these four steps to find recommendations for the deliverables. Details of the steps and result are provided on following sections on this report.

Step 1: Choose sample Business Scenarios from each of Structured and Unstructured scenarios.

Step 2: Analysis of chosen Business Scenarios (Section 5: Business scenario)
2-1. List up Service Domains & Service Operations contained in sample scenarios
2-2. Define meta data elements required for sample scenarios
2-3. Map meta data to existing message standards such as IFX, ISO (if applicable)

Step 3: Classify Service Operations into Tier I, II, III (Section 4: Classification of service operation)
(Example of Classification) Tier I -Detailed, Tier II -Mixed, Tier III –Simplified
3-1. Classifying Service Operations into each Tier based on composed data types like a structured and unstructured.
3-2. Classifying Service Operations into general communication types such as machine to machine, machine to person, person to person.

[3] BIAN API Working Group. (2015). *Working Group Charter version 0.1.* Situation. P.1.
[4] BIAN API Working Group. (2015). *Working Group Charter version 0.1.* Objective. P.2.
[5] BIAN API Working Group. (2015). *Working Group Charter version 0.1.* Deliverables. P.3.

API Classification Guideline for BIAN Architecture

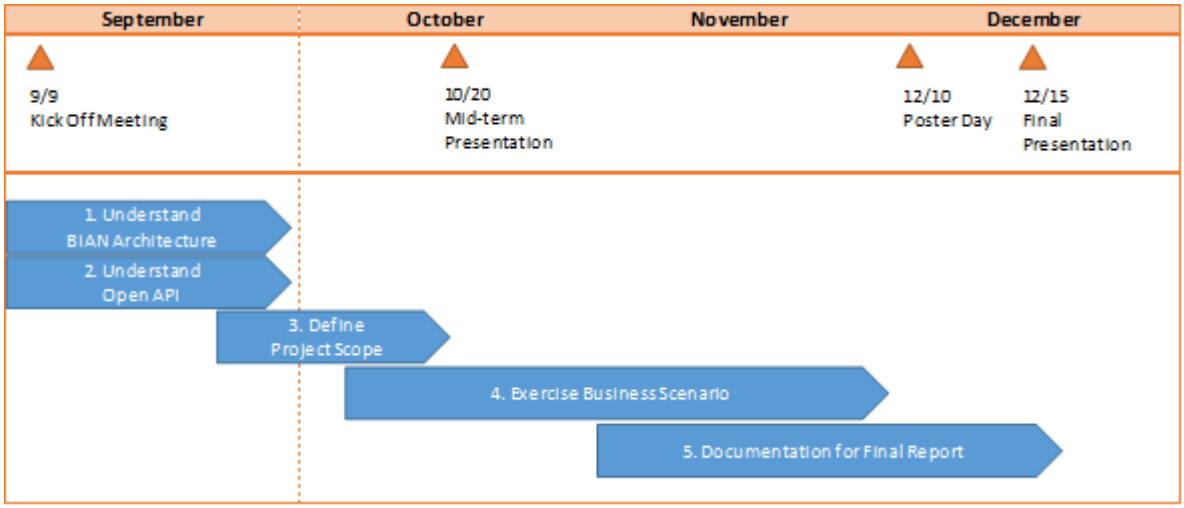
3-3. Map the Service Operations into the 3 x 3 matrix that are Tier and communication data type.

Step 4: Compose Guides for Classification of APIs (Section 6: Conclusion)

2.3.5. Timeline

The project timeline is shown in Figure 2.1 Project Time line.

Figure 2.1 Project Time line



3. BIAN Architecture

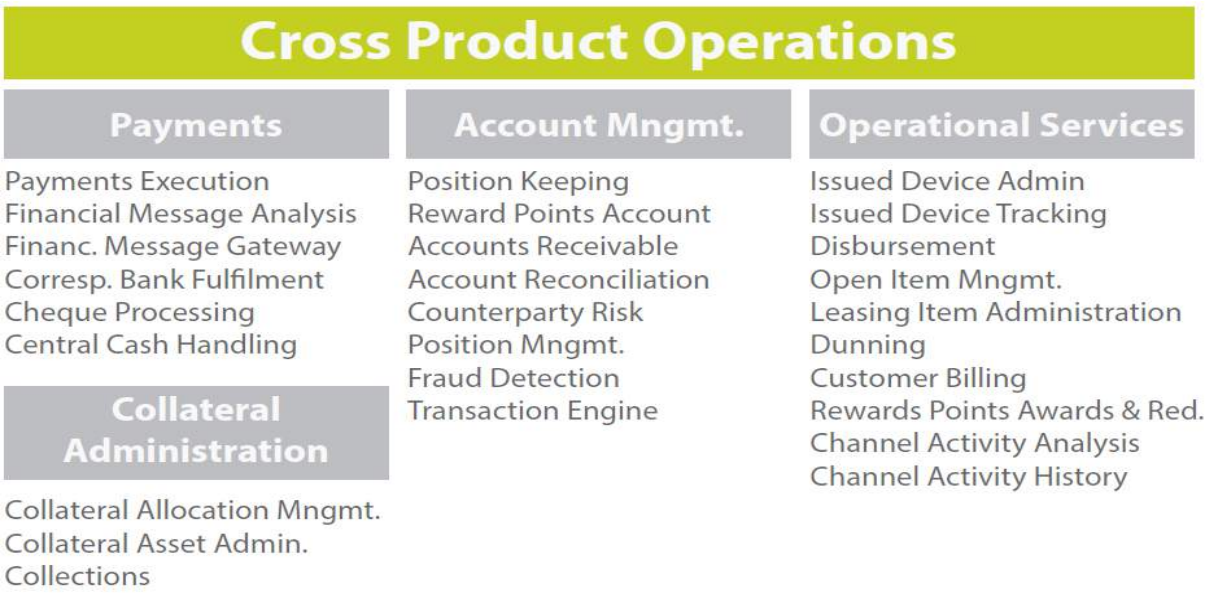
The BIAN Architecture is an elemental capability based canonical model consisting of business functions and service interactions that describes all business capabilities in the banking industry. It allows one to develop a high-level design of a solution and can be used to develop a blueprint of an enterprise for business and technical planning. The model can be applied to different technical environments and consistently interpreted by any bank in different implementation situations. Compared to a traditional proliferation of proprietary design, the BIAN standard provides the benefits such as higher efficiency of developing and integrating software solutions for banks, high operational efficiency and capability re-use within and among banks. The BIAN standard also has more advantage over traditional proliferation of proprietary design in supporting the adoption of flexible business service souring models and enhancing the evolution and adoption of 3rd party business services.

3.1. Service Landscape

BIAN defines the central objectives for IT in the banking industry are to reduce integration costs and utilize the advantages of service oriented architecture (SOA). BIAN's mission is to define "a common yet exceedingly flexible SOA framework for the banking industry with the goal of establishing a common language"[6] which "will enable faster, more efficient strategic and operational changes in banks while helping banks to address the key market imperative to drive cost reductions through greater efficiency and organizational flexibility."
To reach the goals, BIAN's vision is to develop an elemental capability based SOA to design banking systems, which is different and better than the traditional process-centric SOA. BIAN aims at implementing the SOA as the BIAN Service Landscape, which is currently made up of 280 unique Service Domains identified by the BIAN members. The BIAN Service Landscape is canonical so that it can be consistently interpreted by any bank in various implementation scenarios. The picture below shows the current BIAN Service Landscape.[7]

[6] Rackham, Guy. (2015). BIAN Introduction September 2015. BIAN's Mission, P.3.
[7] The BIAN Service Landscape 4.0.1 in poster format. (2015) Retrieved from https://bian.org/wp-content/uploads/2015/07/BIAN_landscape4.0.pdf

On the Service Landscape Service Domains are grouped into Business Domains and further grouped into Business Areas. The picture below shows an example of such kind of grouping through showing part of the above Service Landscape. In this picture, the Service Domains are grouped into four Business Domains, i.e. Payments, Account Management, Operational Services and Collateral Administration. These four Business Domains are then grouped into the Cross Product Operations Business Area.



The development of BIAN Service landscape is a dynamic process, relying on active contribution from all BIAN members. The current version of the Service Landscape is 4.0.1, which was released in 2015.

3.2. Service Domains

BIAN seeks to identify elemental business capabilities that can exist within any bank. The elemental business capabilities are defined at a level of granularity where any further decomposition would cause them to lose unique business context – the capabilities would become utilitarian in nature. Any bank can map any or all of their processes using interactions between these capabilities[8]. The elemental business capabilities identified by BIAN are modeled as Service Domains (SD). SDs represent the finest level of business capabilities of the banking industry. This collection of Service Domains is comprehensive such that any or all business activity of a bank can be modelled using a suitable selection of these Service Domains interacting through their associated interfaces[9].

All Service Domains follow a two-aspect definition[10]

1. **Entity** – It is the object acted upon by the service domain. It can be tangible asset like piece of equipment, card, or an intangible asset such as customer relationship or knowledge. Service Domain handles all the activities of the asset throughout its life cycle.
2. **Functional Pattern** – It defines the type of action that can be performed on the asset. BIAN is maintaining a standard list of patterns that represent these Functional Patterns. Currently identified 20 generic Functional Patterns have been identified after iterative review of Service Domains. Each entity of the Service Domain is associated with one of the 20 identified Functional Patterns. Below is the list of 20 Functional Patterns identified by BIAN[11].

[8] Nishihara, Yasuyuki. Faraco, Felipe S. Gupta, Rohan. Etc. (2014). *Implementation of Financial Message Standards to BIAN Architecture*, P. 11.
[9] Nishihara, Yasuyuki. Faraco, Felipe S. Gupta, Rohan. Etc. (2014). *Implementation of Financial Message Standards to BIAN Architecture*, P. 12.
[10] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.2-2 Defining the Business Role of a Service Domain Interactions, P. 12
[11] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.2-2 Defining the Business Role of a Service Domain Interactions, P. 15

API Classification Guideline for BIAN Architecture

Oversight & Control Functions Activities needed to define the policies and plans to manage and administer the resource/asset and record/register its properties/details	PLAN	Define and direct the execution of a plan with supporting policies, budget, organization, goals etc.
	MANAGE	Manage, oversee and troubleshoot an ongoing activity
	ADMINISTER	Handle the range of associated administrative functions/tasks
	DIRECTORY	Capture and maintain reference details for some entity, collectively details can define a directory
Development & Deployment Functions Activities needed to design, build/develop and enhance the resource/asset, and to deploy and maintain it during its use	DESIGN	Create and enhance a design, model or specification,
	BUILD/ENHANCE	Build and enhance/extend
	DEPLOY	Deploy, distribute (applies to equipment, systems, consumables and inventory)
	ASSURE	Test, quality assure or assess/evaluate against predefined criteria/measures
	CONFIGURE	Define and maintain the set-up/configuration of an operating unit or system
	MAINTAIN	Perform maintenance to a defined schedule and repair as necessary
Assessment & Evaluation Functions Activities needed to track and consolidate activity data, analyse and rate the resource/asset	TRACK	Build and maintain a history, log or account of activity (general activity & financial transactions)
	ANALYSE	Analyse and derive insights, perspectives (includes the data consolidation/manipulation)
	QUALIFY	Determine and maintain status indicators/qualifications/ratings
Agreement and Assignment Functions Activities to estalish usage terms, membership, access entitlement/allocation and manage resource/asset inventories/availability	AGREE TERMS	Establish and maintain a governing terms for some form of engagement relationship (can be legal)
	ENROLL	Establish and maintain a membership group/population
	ALLOCATE	Handle assignment and usage entitlements of the resource
	INVENTORY	Maintain an inventory of a collection of items or data sources that make up the resource
Engagement Functions Activities o use use/apply the resource in business execution	OPERATE	Operate a capability, optionally to a schedule (typically a highly automated capability)
	FULFILL	Fulfill an in-force facility/specialist service delivery commitment – scheduled and ad-hoc activity
	TRANSACT	Complete a task or transaction

The combination of BIAN Service Domain's associated entity and functional pattern is called a Control Record[12]. For instance, this combination could be a single primary functional pattern (for example 'maintain reference details' or 'define and execute a plan') with an asset or entity type (for example 'a customer relationship').

The BIAN Service Domain has the following characteristics: [13]

- **Unique Business Purpose** – the Service Domain performs a discrete business role, and does not represent a grouping of similar business capabilities.
- **Elemental –** they are functions that fulfill a discrete non-overlapping business role/purpose. It may have complex internal processing in order to fulfill this elemental role, but from a business concept perspective, it fulfills a single function.
- **Collectively Comprehensive –** taken together the full collection of Service Domains covers all activities performed in Banking. Any or all activity can be modeled as collections of two or more Service Domains interactions.
- **The Control Record –** it defines the business role or purpose of the Service Domain.
- **Full Life Cycle Support –** the Service Domain handles all activities for the control record from its initiation through to final termination or completion of the role. This is an important distinction from process-based models where the entity can be passed along the process chain as it goes through different states. In the BIAN Service Center, the entity remains with the Service Domain, accessed as necessary through service operations, for its complete life cycle.
- **Single and Multiple Instances –** depending on the business role it can make sense to have one active instance of the focus asset (for example a business unit plan) or multiple

[12] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.2-2 Defining the Business Role of a Service Domain Interactions, P. 15

[13] BIAN Webinar Part 2 –Service Domain, Slide 4

concurrent Control Record in different states of their life cycle (for example customer agreements).

- **Short or Long Lifespan –** again depending on the business roles the control Record lifecycle can be short, for example a customer contact, or very long lived such as a product specification or design.
- **Service Based –** the business role or function supported by the Service Domain can be implemented using a service based construct – where all access to the information and facilities of the Service Domain is by service operation as is all support required by the Service Domain achieved through service call to other Service Domains.

3.3. Service Operations

Service Operations (SOs) are ways by which Service Domains (SDs) interact with one another in a business scenario. Semantically defined Service Operations are offered and consumed by BIAN Service Domains. These ensure that the BIAN Service Operation specifications are implementation agnostic so as to be canonical. Service Operations are discovered using real world business scenarios[14].
The operational dependency between the communicating Service Domains can be categorized into four types, which are mentioned here[15]:

1. **A two-way exchange** – the response is sent immediately to the calling Service Domain.

2. **A request with an anticipated delay in the response** – The calling Service Domain continue its work anticipating the response after some time. Calling Service Domain monitors for the expected response.

3. **A hand-off notification** – No response is expected by the calling Service Domain except an acknowledgement of the receipt from the called Service Domain. Calling Service Domain does not have any operational interest after passing the details to the called Service Domain.

4. **Provision of previously subscribed-to updates** – the calling Service Domain has subscribed to updates from the called Service Domain at some point.

Below are the parameters along with their descriptions, which are associated with service operations[16]:

1. **Identifier** – defines unique tags/identifier that relate to the Control Record instance.

[14] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.3.3 Semantic Definition of Service Operations, P. 24.
[15] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.3.3 Semantic Definition of Service Operations, P. 24.
[16] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.3.3 Semantic Definition of Service Operations, P. 25.

2. **Depiction** – contains the payload captured within the message. It is related to the Control Record instance.

3. **State** – the state of the Control Record instance that is passed within the message.

4. **Control** – defines control 'parameters' that govern the requested action which also includes the specification of reporting/query details.

5. **Analysis** – contains any tracked/derived values associated with one or some combination of Control Record instances.

The Service Operations can be classified into three tiers described in the chart below. The purpose of this classification is to offer a guideline for BIAN members to design their APIs for their business capabilities under the BIAN architecture. Each Service Operation can be classified into one of the three tiers according to its corresponding input & output parameter depictions. The details of the classification approach will be discussed later in this article.

Tier	Description
1	Structured data as output, simple data processing, two-way communication
2	Structured and unstructured data as output, simple data processing, one-way or two-way communication
3	Unstructured data as output, complicated data processing, two-way communication

4. Classification of service operations

In order to design an API, it is very important for us to understand the information that the interface can host. This requires us to look at the service operations from a more granular level and understand the type of information they carry. We have come up with two different classifications of service operations based on communication type and level of detail, that would help us in identifying the information that an API can host.

4.1. Classification based on communication type

1) Machine - Machine: A service operation is labeled as machine - machine when the interaction contains content suited to data fields that may be passed between machines/applications.
   For example: activateCorporateTreasuryAnalysis service operation in the loan origination business scenario mostly deals with research data records and analysis parameters. Service operations of this kind do not require any human intervention. That is why this has been labeled as Machine - Machine

2) Machine - Person / Person - Machine: A service operation is labeled as Machine - Person / Person - Machine when the interaction contains structured forms of data/information that is presented to or completed/provided by a person through the service operation.
   For example: initiateLoanFulfillment service operation in the loan origination business scenario mostly deals with processing requests/updates and amendment forms and parameters, processing activity reporting forms/structured reports. These service operations require human interaction with computers. That is why this service operation has been labeled as Machine - Person / Person - Machine

3) Person - Person: A service operation is labeled as Machine - Person when the interaction contains freeform or unstructured information that may be exchanged between people through the service exchange
   For example: evaluateGuidelineComplianceAssessment service operation in the loan origination business scenario mostly deals with assessment request description/details, assessment analysis report/results, assessment activity historical and analytical reports, assessment portfolio analysis reports. Such service operations cannot be handled by machines. That is why this service operation has been labeled as Person - Person.

4.2. Classification based on level of detail

We listed down service domains and operations for structured and unstructured business scenarios. Then we classified service operations into three tiers based on the level of detail

involved with each. We considered identifiers and depictions of each service operation to come up with this classification.

Definitions for the three tiers are as follows:

1) Tier 1 (Detailed): Service operations are classified as Tier 1 when all the identifiers and depictions can be clearly mapped to a data structure. The level of detail in this case is very high.

   Example: For initiatePaymentOrderTransaction service operation in the internal credit transfer business scenario, the input and output parameters are mentioned in detail and every parameter can be mapped to a data structure. Therefore, we classify this as Tier 1.

2) Tier 2 (Mixed): Service operations are classified as Tier 2 when some of the identifiers and depictions can be clearly mapped to a data structure. The level of detail in this case is medium.

   Example: For retrieveGuidelineComplianceAssessment service operation in the loan origination business scenario, most of the input and output parameters are mentioned in detail and can be mapped to specific data structures. But, there are a few parameters such as GuidelineComplianceAuthorizationRequestDetails, GuidelineComplianceAuthorizationResult etc. that are generic in nature and cannot be mapped to specific data structures. Therefore, we classify this as Tier 2.

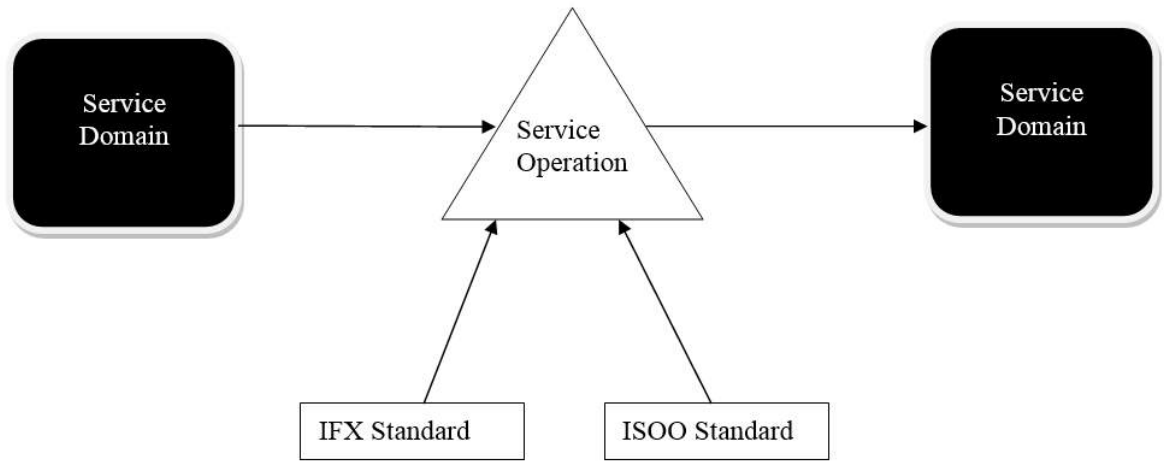
3) Tier 3 (Generic): Service operations are classified as Tier 3 when none of the identifiers and depictions can be clearly mapped to a data structure. The level of detail in this case is very low.

   Example: For evaluateUnderwritingAssessment service operation in the loan origination business scenario, the input and output parameters are not mentioned in detail and cannot be mapped to a data structure. Example – UnderwritingAssessmentEvaluationResultDetails, UnderwritingAssessmentInputInformation/Credentials etc. Therefore, we classify this as Tier 3.

5. Business Scenarios

BIAN defines a Business Scenario as a suitable selection of Service Domains interacting with each other through their associated interfaces representing a business activity of the banking industry. In other words, business Scenarios are archetypal diagrammatic examples of how Service Domains may interact to address some business events. The BIAN Business Scenarios are not prescriptive or canonical. They are examples to provide context or explanation of the Service Domains' roles or behaviors. They do not represent precise logic and sequence of tightly coupled tasks as in the case of process representation; instead, they identify the service domain involved and present the high-level service exchanges possible between those Service Domains.[17] The purpose of business scenarios is to discover and clarify the service operation that is exchanged between Service Domains involved in a business activity.

Structured and Unstructured Data:
Data is exchanged between processes in a structured form or unstructured form. Structured form of exchange is when data is exchanged through a messaging standard. This would typically include an IFX or ISO 20022 mapping Messaging standard. The IFX Object Model defines an **object** as a set of data that is organized according to a consistent pattern, and that supports a well-defined set of operations. ISO 20022 is the ISO Standard for Financial Services Messaging. It describes a metadata repository containing descriptions of messages and business processes, and a maintenance process for the repository content. Both these standards have business processes to map data into messaging format which is exchanged across Services Domains. The following representation is a structured data mapped using a messaging standard.



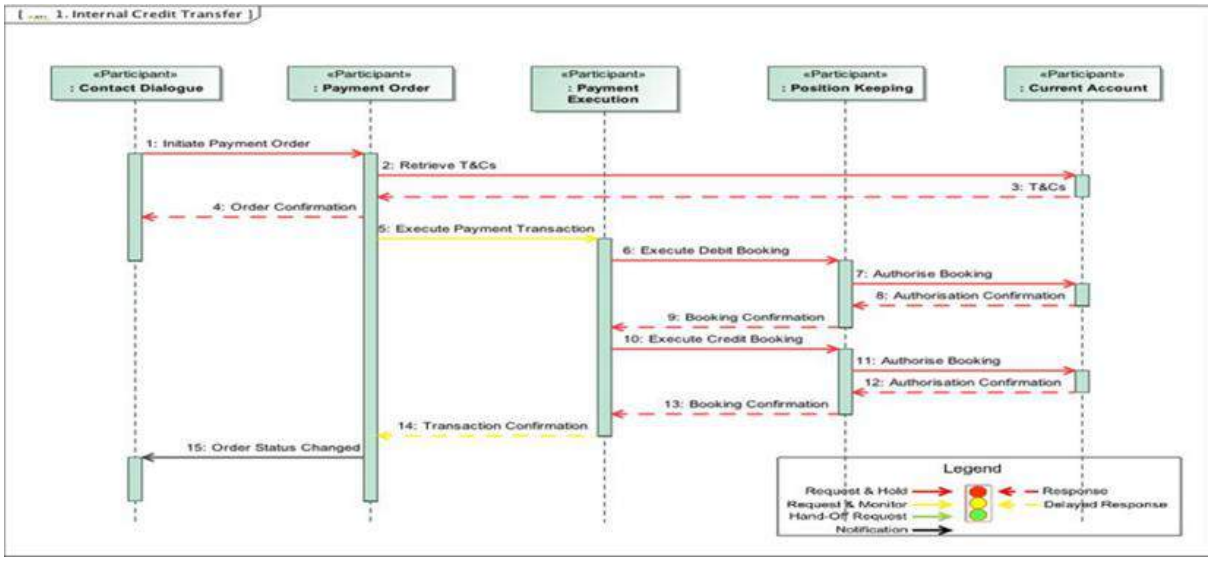
[17] Rackham, Guy. (2014). *BIAN How-to Guide -Design Principles & Techniques*.3.3 Modelling Service Domain Interactions, P. 22.

API Classification Guideline for BIAN Architecture

Unstructured data, is the data that cannot be mapped to a messaging standard to exchange information. It is text heavy and consists of semantic definitions. Example of a unstructured scenario is loan obligation. Many of the information exchanged for this scenario is unstructured and consists of semantic definitions.

Tiered Classification:

1. **Identify Business Scenarios**



The Business Scenario identified here **Internal Credit Transfer**. The business scenario is represented in its granular form of Servide Domains. Each Service Domain interacts with the other service domains through Service Operation.

2. **Identify Service Operations**

Each interaction in a Service Domain is mapped to a BIAN defined Service Operation. THe next step in classification is to identify the correct Service operation. In the above example, the Initiate Payment Order operation is identified as **initiatePaymentOrderTransaction.** This is defined in the BIAN service Landscape.

Each Service Operations has a set of parameters that defines them. The parameters that help identify the type of data that each service operation is used for are **Depictions in input and output parameters.**

For the **Internal Credit Transfer** scenario, consider the **initiatePaymentOrderTransaction** service operation. The following image describes the service operation,

A	B	C	D	E	F
		Input Parameters		Output parameters	
BIAN Service Operation	**Service interaction type**	**Identifiers**	**Depiction**	**Identifiers**	**Depiction**
initiatePaymentOrderTransaction	Request & Hold	PartyReference OfferReference Unit/EmployeeReference Product/ServiceReference	PaymentOrderTransactionType PaymentOrderFeatures PaymentOrderTransactionPricingDetails PaymentOrderTransactionOptions&ParameterDetails PaymentOrderRegulatoryConstraints PaymentOrderTransactionRecord (e.g. InstrumentType; Date/Time; Date/TimeType; Counterparty; CounterpartyBank; Intermediary; Principal; Currency; Rates; Cashflows; CollateralRequirements; Fees/Commissions)	PaymentOrderTransactionReference	NA

API Classification Guideline for BIAN Architecture

3. Classify SOs as Tier 1, 2 or 3 based on input and output parameters

The next step in the overall API Design is to classify the SOs into Tiers. Tiered classification helps to categorize the data that is exchanged between service domains.

4. Classify SOs as M to M, M to P, P to P based on called service domain, functional pattern Each Service operation also belong to one of the following types of interaction, Machine - to - Machine, Machine - to - Person/Person - to - Machine or Person - to - Person.

5. Validate step 4 result by using Likely “artifact” content based on the action term.

6. Set Service Operation into the data tier type and communication type matrix.

	Machine to Machine	**Machine to Person / Person to Machine**	**Person to Person**
Tier 1	activateCorporateTreasuryAnalysis activateFinancialPositionTracking/Log executePaymentExecutionTransaction requestCurrentAccountFulfillment retrieveCustomerCreditRatingMeasurement		
Tier 2	activateInformationFeedOperation	configureDocumentHandlingOperation initiateLoanFulfillment notifyLoanFulfillment retrieveCorrespondenceOperation retrieveDocumentHandlingOperation retrievePartyRegistration retrieveProduct/ServiceAgreement initiatePaymentOrderTransaction notifyPaymentOrderTransaction retrieveGuidelineComplianceAssessment	
Tier 3		recordDocumentHandlingOperation retrieveCurrentAccountFulfillment updateSalesProduct/ServiceAgreement	evaluateGuidelineComplianceAssessment evaluateRegulatoryComplianceAssessment evaluateUnderwritingAssessment; requestSyndicatedLoanFulfillment

6. Conclusion

6.1. Findings

1) Findings from classification based on metadata's level of detail

- Tier 1 (Detailed): Tier 1 service operation should be designed to accommodate "MtoM" communication type.
- Tier 2 (Mixed): Tier 2 service operation should be designed to accommodate "MtoM" or "PtoM/MtoP" communication type.
- Tier 3 (Generic): Tier 3 service operation should be designed to accommodate "PtoM/MtoP" or "PtoP" communication type.

2) Findings from classification based on communication type

- MtoM (Machine to Machine): the messages handled by an "MtoM" type of service operation require at least one 'structured' metadata
- PtoM/MtoP (Person to Machine/Machine to Person): the messages handled by a "PtoM/MtoP" type of service operation contain at least one 'unstructured' metadata
- PtoP (Person to Person): the messages handled by a "PtoP" type of service operation does 'NOT' contain any 'structured' metadata

3) Possible Patterns of classification combinations

		communication type		
		MtoM	PtoM/MtoP	PtoP
metadata's level of detail	Tier 1 (Detailed)	x		
	Tier 2 (Mixed)	x	x	
	Tier 3 (Generic)		x	x

6.2. Future Research

- Need more classification exercises using other business scenarios in order to verify if the current classifications are applicable for the entire BIAN Service Landscape.
- The terms used for communication type classification (i.e. MtoM, PtoM/MtoP, PtoP) might be reconsidered as unstructured data analysis capabilities by machines improve.

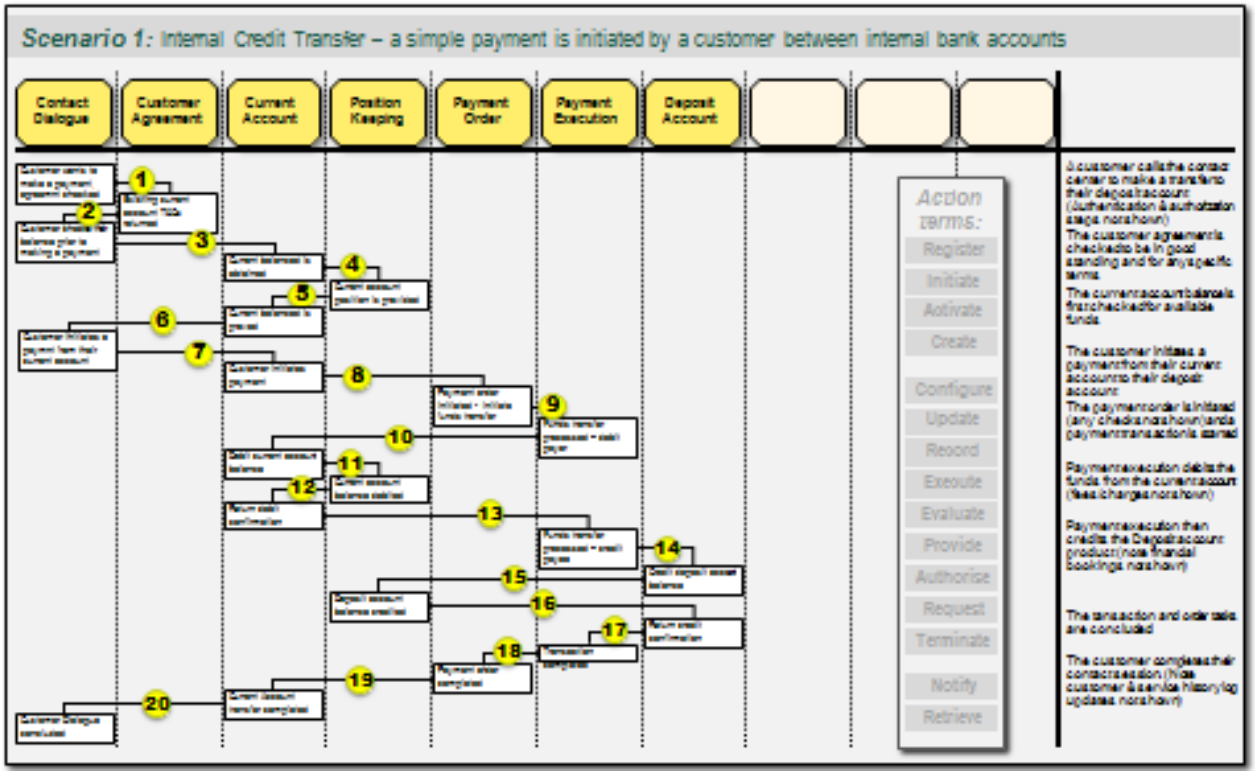
6.3. Lessons and Learns

- Iteration of exercise and analysis bring new finding point
- To make guideline, should focus on apparent common rule, not focus on exception

7. Appendix

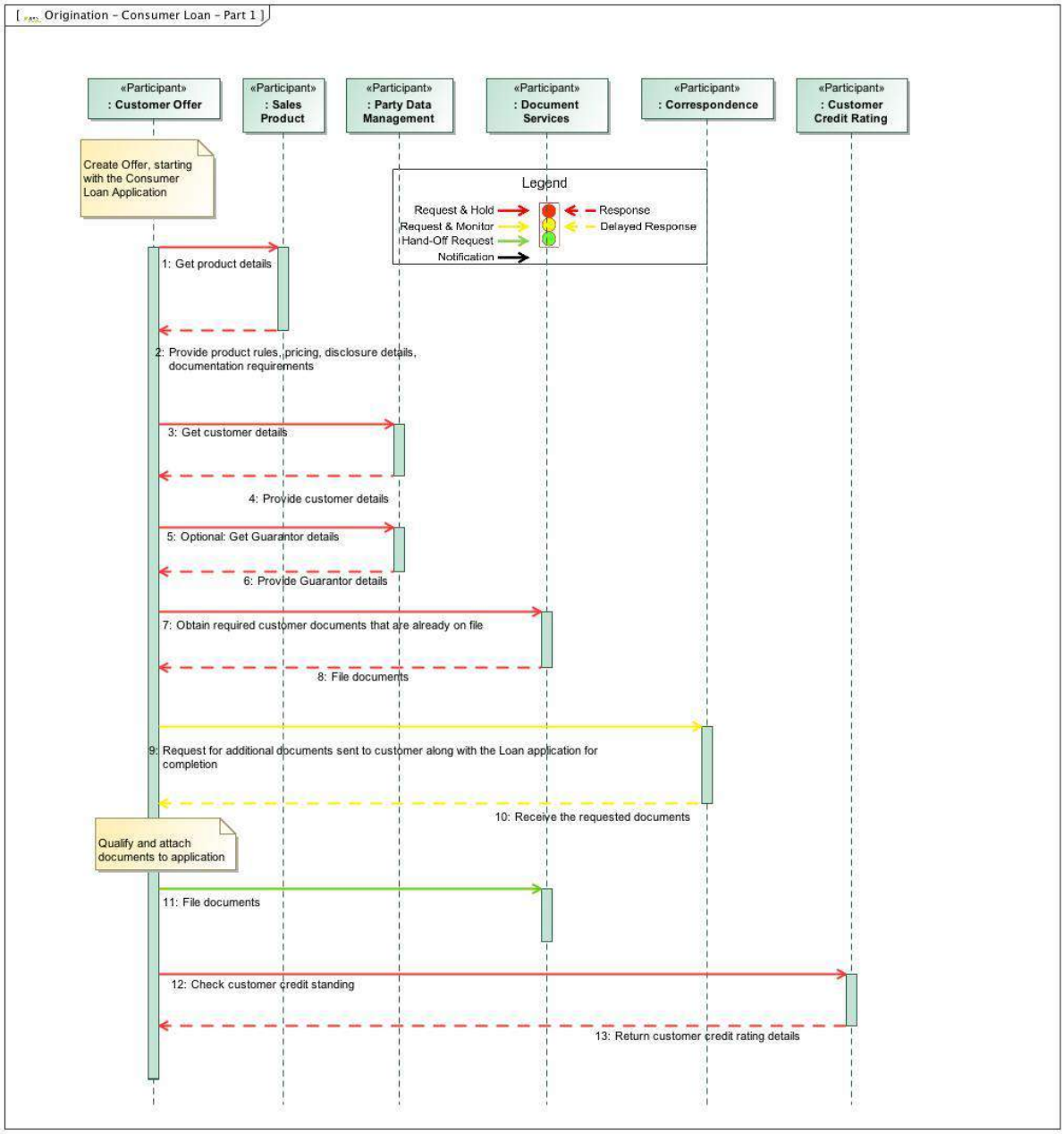
7.1. Business scenario

Scenario 1: Internal Credit Transfer [18]



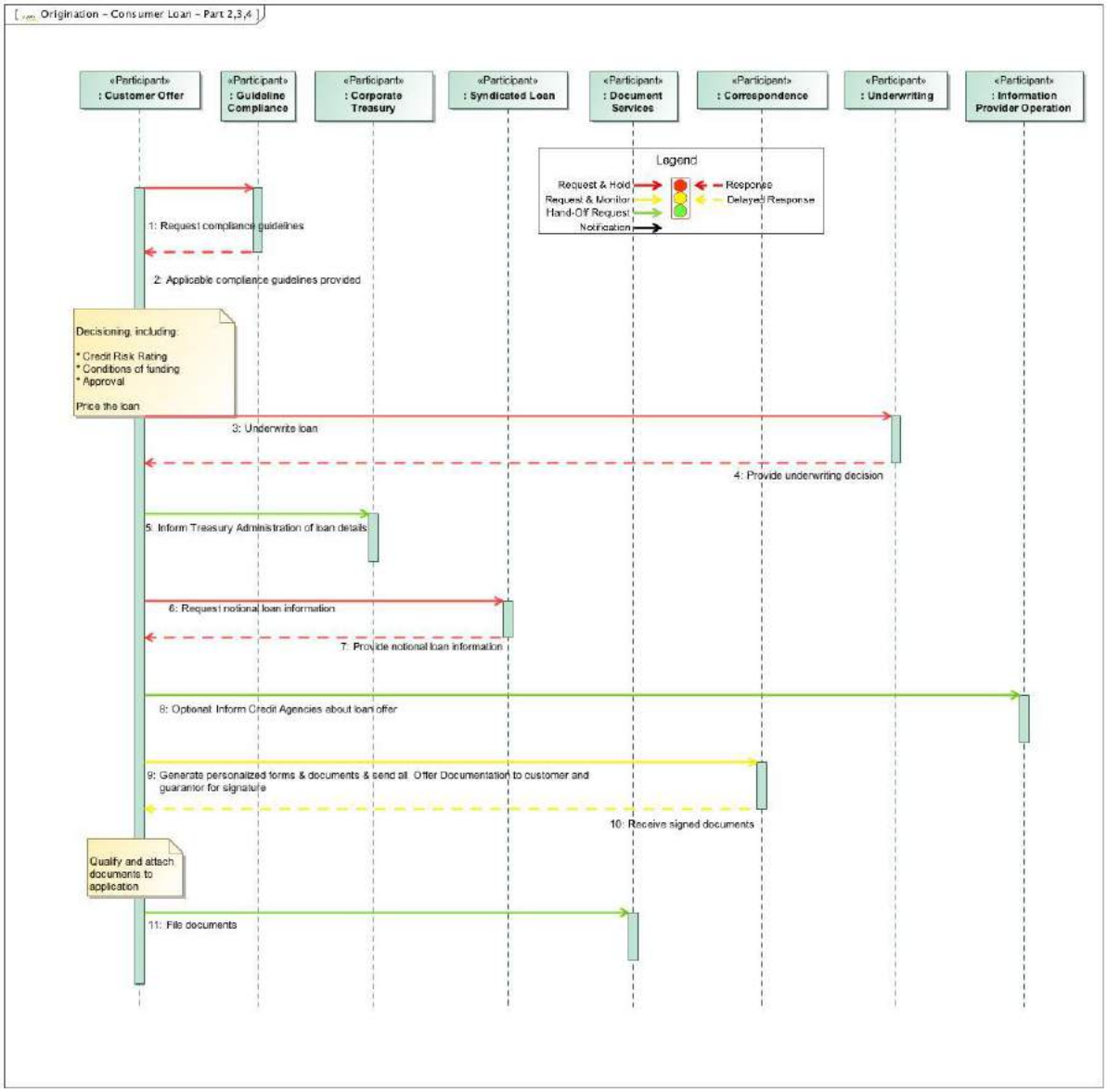
[18] Rackham, Guy. (2015). *BIAN Carnegie Mellon University Updated Business Scenario November 2015*. P.2.

Scenario 2: Origination - Consumer Loan - Part 1 [19]



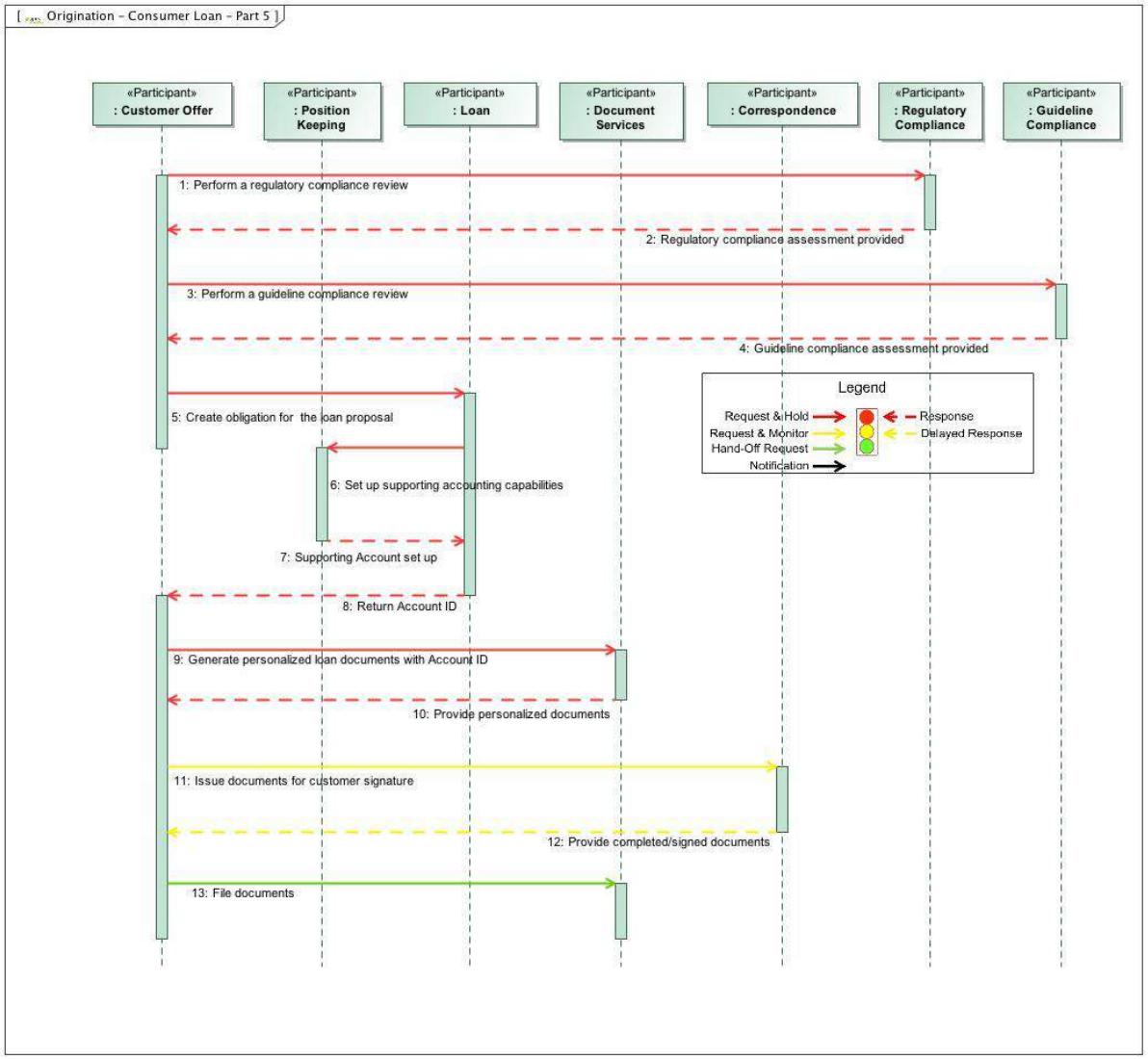
[19] BIAN Business Scenario Origination - Consumer Loan - Part 1. (2015) Retrieved from https://bian.org/servicelandscape/?refid=_17_0_4_1_13d303b9_1381418617975_628156_8417

Scenario 3: Origination - Consumer Loan - Part 2,3,4 [20]



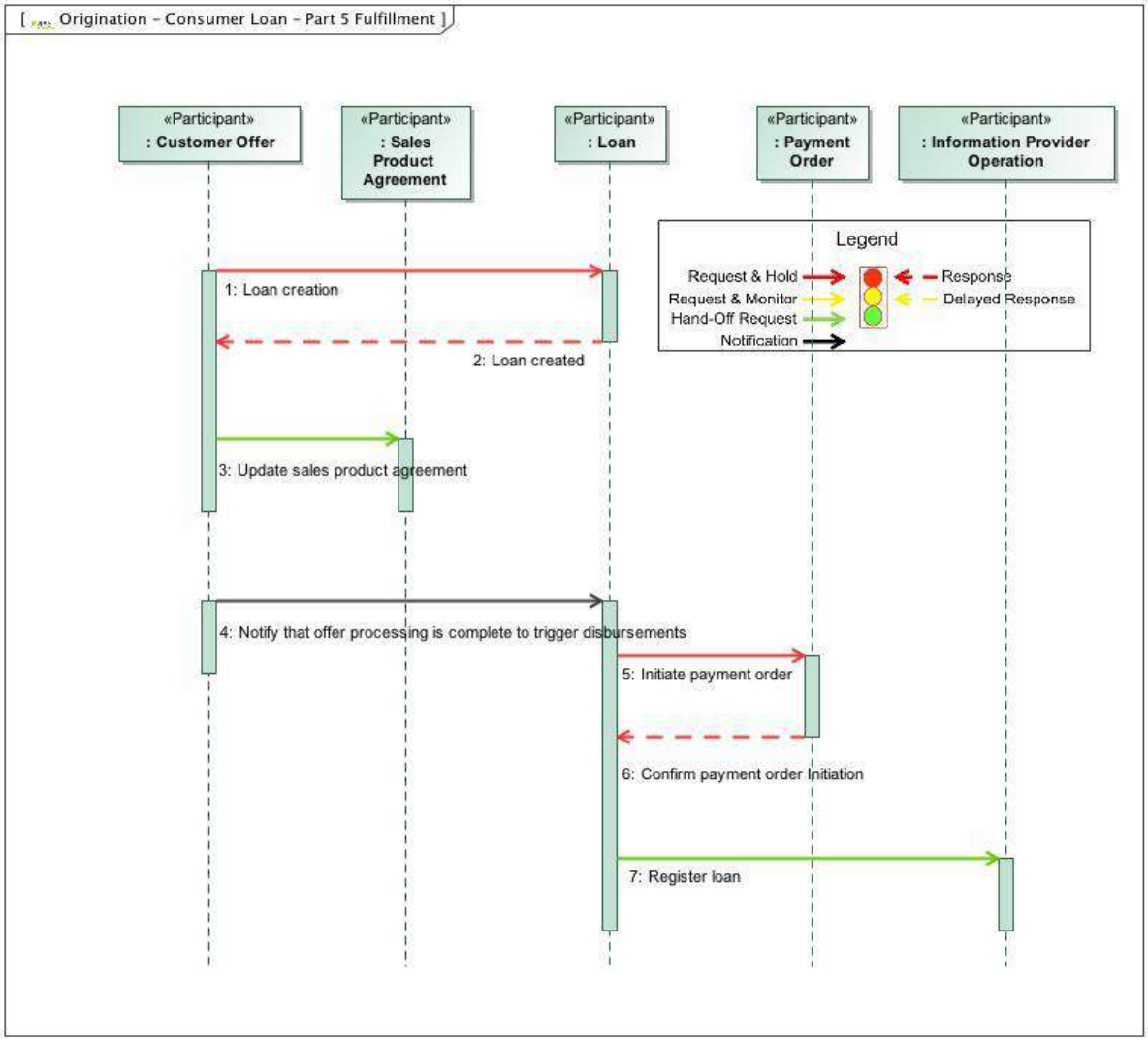
[20] BIAN Business Scenario Origination - Consumer Loan - Part 2,3,4. (2015) Retrieved from https://bian.org/servicelandscape/?refid=_17_0_4_1_13d303b9_1381418660698_488345_8965

Scenario 4: Origination - Consumer Loan - Part 5 [21]



[21] BIAN Business Scenario Origination - Consumer Loan - Part 5. (2015) Retrieved from https://bian.org/servicelandscape/?refid=_17_0_4_1_13d303b9_1381418592557_113644_8077

Scenario 5: Origination - Consumer Loan - Part 5 fulfillment [22]



[22] BIAN Business Scenario Origination - Consumer Loan - Part 5 fulfillment. (2015) Retrieved from https://bian.org/servicelandscape/?refid=_17_0_4_1_13d303b9_1381418607652_863366_8302

7.2. Business Scenario Exercise result

Result Summary

	Machine to Machine	Machine to Person / Person to Machine	Person to Person
Tier 1	activateCorporateTreasuryAnalysis activateFinancialPositionTracking/Log executePaymentExecutionTransaction requestCurrentAccountFulfillment retrieveCustomerCreditRatingMeasurement		
Tier 2	activateInformationFeedOperation	configureDocumentHandlingOperation initiateLoanFulfillment notifyLoanFulfillment retrieveCorrespondenceOperation retrieveDocumentHandlingOperation retrievePartyRegistration retrieveProduct/ServiceAgreement initiatePaymentOrderTransaction notifyPaymentOrderTransaction retrieveGuidelineComplianceAssessment	
Tier 3		recordDocumentHandlingOperation retrieveCurrentAccountFulfillment updateSalesProduct/ServiceAgreement	evaluateGuidelineComplianceAssessment evaluateRegulatoryComplianceAssessment evaluateUnderwritingAssessment; requestSyndicatedLoanFulfillment

Detail result of the following 5 Scenarios are shown in "Appendix Business Scenario Exercise.xlsx"

- Scenario 1: Internal Credit Transfer
- Scenario 2: Origination - Consumer Loan - Part 1
- Scenario 3: Origination - Consumer Loan - Part 2,3,4
- Scenario 4: Origination - Consumer Loan - Part 5
- Scenario 5: Origination - Consumer Loan - Part 5 fulfillment

7.3. References

Detailed information is indicated in footnotes.

- BIAN Website. https://bian.org/
- Heinz College Website. http://www.heinz.cmu.edu/
- BIAN API Working Group. (2015). Working Group Charter version 0.1.
- Rackham, Guy. (2015). BIAN Introduction September 2015.
- The BIAN Service Landscape 4.0.1 in poster format. (2015) Retrieved from https://bian.org/wp-content/uploads/2015/07/BIAN_landscape4.0.pdf
- Nishihara, Yasuyuki. Faraco, Felipe S. Gupta, Rohan. Etc. (2014). Implementation of Financial Message Standards to BIAN Architecture,
- Rackham, Guy. (2014). BIAN How-to Guide -Design Principles & Techniques.
- Rackham, Guy. (2015). BIAN Carnegie Mellon University Updated Business Scenario November 2015.
- BIAN Business Scenario (2015) Retrieved from https://bian.org/servicelandscape/

7.4. About CMU Team

7.4.1 Team Members

Chaitanya Eranki:
Chaitanya is a graduate student at Heinz College, Carnegie Mellon University currently pursuing Master of Information Systems Management. He has over three years of experience working for multinational financial corporations such as Wells Fargo and Barclays Capital in various capacities. He wants to take up IT strategy related positions in the financial space after graduation.

Hideaki Jim Nakajima:
A 2nd year graduate student of Master of Information Systems Management, Heinz College, CMU. Jim worked as a Business Analyst and a Project Manager for global financial market information systems at Bank of Tokyo-Mitsubishi UFJ in Tokyo and New York. He led the information system implementation projects globally.

Prajwal Niranjan:
Graduate student pursuing the Master of Information Systems Management degree in H. John Heinz III College of Carnegie Mellon University. He has over two years of experience working in IT Services as an Oracle DBA. He wants to work in the technology sector of the Financial Industries.

Tomofumi Ueno:
A 2nd year graduate student of Information Systems Management in H. John Heinz III School of Public Policy and Management at Carnegie Mellon University. He has been working for 15 years as a system design manager, a system development expert, and a network engineer at Mitsubishi UFJ Information Technology, Ltd. in Tokyo, Japan.

Wei Wang:
A graduate student pursuing the Master of Information Systems Management degree in H. John Heinz III College of Carnegie Mellon University.

7.4.2 Faculty Advisor

Michael McCarthy:
An Associate Teaching Professor of Information Systems at Carnegie Mellon University's H. John Heinz III School of Public Policy and Management. Formerly, he served as the Director of Undergraduate Programs and a Senior Lecturer for the Department of Computer Science at the University of Pittsburgh. Professor McCarthy’s degrees include a BA in Philosophy and an MS in Information Science (Systems Track), both from the University of Pittsburgh. He has over sixteen years of university-level teaching experience. Mr. McCarthy has also worked as a software consultant to several engineering firms.