VA Enterprise Design Patterns:

Overview of Enterprise Messaging Capabilities and Message Exchange Patterns (MEPs)

Office of Technology Strategies (TS)
Architecture, Strategy, and Design (ASD)
Office of Information and Technology (OI&T)

Version 1.0

Date Issued: February 23, 2015

THIS PAGE INTENTIONALLY LEFT BLANK FOR PRINTING PURPOSES

APPROVAL COORDINATION

X TIMOTHY L MCGRAIL 111224

Digitally signed by TIMOTHY L MCGRAIL 111224
DN: dc=gov, dc=va, o=internal, ou=people, 0.9.2342.19200300.100.1.1=tim.mcgrail@va.gov, cn=TIMOTHY L MCGRAIL 111224
Date: 2015.03.24 15:34:41 -04'00'

Tim McGrail
Deputy Director, Acting

Tim McGrail
Deputy Director (Acting)
ASD Technology Strategies

X Tibbits, Paul A.

Digitally signed by Tibbits, Paul A.
DN: dc=gov, dc=va, ou=Entities, ou=InternalStaff, cn=Tibbits, Paul A.
Reason: I am approving this document.
Date: 2015.03.13 13:23:42 -04'00'

Paul A. Tibbits, MD
DCIO Architecture, Strategy and Design

Paul A. Tibbits, M.D.
DCIO Architecture, Strategy, and Design

REVISION HISTORY

Version Number	Date	Organization	Notes
0.1	11/7/2014	ASD TS	Initial Draft
0.5	11/17/2014	ASD TS	Updated to include inputs from internal stakeholders and vendor representatives
0.6	12/5/2014	ASD TS	Updated to include additional inputs from internal stakeholders and vendor representatives
0.75	12/18/2014	ASD TS	Updated to address stakeholder feedback on Version 0.6
0.85	1/7/2015	ASD TS	Updated to address stakeholder feedback on Version 0.75
1.0	2/23/2015	ASD TS	Updated to address stakeholder inputs from Public Forum meeting held on 1/22/15

REVISION HISTORY APPROVALS

Version	Date	Approver	Role
0.1	11/7/2014	Joseph Brooks	ASD TS Design Pattern Government Lead
0.6	12/5/2014	Joseph Brooks	ASD TS Design Pattern Government Lead
0.75	12/18/2014	Joseph Brooks	ASD TS Design Pattern Government Lead
0.85	1/7/2015	Joseph Brooks	ASD TS Design Pattern Government Lead
1.0	2/23/2015	Joseph Brooks	ASD TS Design Pattern Government Lead

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Background

Using a common enterprise IT infrastructure for the VA, which includes messaging capabilities, will reduce the proliferation of redundant capabilities that increase development and support costs, and introduce security vulnerabilities. This infrastructure applies to all new applications that integrate with the VA's enterprise resources in a loosely coupled manner. It is intended for all applications that share data with VA and its partners, regardless of the end-user device.

1.2 Purpose

The purpose of this document is to provide a high-level overview of enterprise messaging capabilities that are provided through enterprise message transport middleware, such as the VA's Enterprise Messaging Infrastructure (eMI). This document includes references to pertinent Message Exchange Patterns (MEP) (as defined in Section 3.1) as well as references to messaging implementation guidance. These patterns apply to both COTS software (including open-source) acquisitions as well as custom applications developed internally. Additionally, the guidance applies to both internal and external service message exchanges.

This document constitutes a follow-on installment to the VA Enterprise Service-oriented Architecture (SOA) Design Pattern that was formally approved by ASD DCIO Dr. Paul Tibbits in July 2014. The Enterprise SOA Design Pattern provides a top-level overview of the enterprise SOA infrastructure capabilities, Enterprise Shared Services (ESS), and guidelines that all VA IT projects will apply to establish solution architectures in accordance with the OneVA Enterprise Technical Architecture (ETA).

1.3 Scope

This document provides a vendor-agnostic framework of functionality in terms of documented Message Exchange Patterns (MEP), and it refers to appropriate design guidelines and reference implementations. The guidance in this document applies to messaging requirements that address the business needs in each of the VA's Lines of Business (LOB). It also refers to the consensus set of standards and protocols established for interoperable message exchange in the VA, as defined by the Information Exchange Working Group (IEWG) chartered by Office of Information and Technology (OI&T) Architecture, Strategy, and Design (ASD).

The following content is beyond the scope of this document, but may be referenced in appropriate locations to guide further technical planning:

- Implementation details and design specifications for messaging capabilities (covered in pertinent implementation guides referenced in Section 4.3)

- Details regarding the consensus set of enterprise messaging standards and transport protocols (covered in the ESS Message Exchange Guide referenced in Section 4.3)
- Standards for message-level security (covered in a related Enterprise Design Pattern document regarding authentication, authorization, and audit (AA&A))

1.4 Intended Audience

This document is meant to be used by all project-level integrated product teams (IPTs) that are developing new applications that are deployed into production within the VA's IT infrastructure. These applications are device-independent, and include acquisition of COTS software (including open-source solutions) intended to meet data sharing requirements. They will make web service calls to enterprise resources utilizing messaging capabilities provided by enterprise messaging middleware platforms, such as the eMI.

1.5 Document Development and Maintenance

This document was developed collaboratively with relevant stakeholders, including those from VA's Office of Information and Technology (OI&T), Product Development (PD), Office of Information Security (OIS), Architecture, Strategy and Design (ASD), and Service Delivery and Engineering (SDE). In addition, the development effort included engagements with industry experts to review, provide input, and comment on the proposed pattern. This document contains a revision history and revision approval logs to track all changes. Updates will be coordinated with the Government lead for this document, who will also facilitate stakeholder engagements and subsequent re-approval depending on the significance of the change.

2 ENTERPRISE MESSAGING CONTEXT AND PROBLEM

2.1 Context for Enterprise Messaging

The VA provides enterprise messaging capabilities to enable efficient connections of disparate applications that support both internal and external users, in accordance with the initial VA Enterprise SOA Design Pattern. These capabilities hide many of the details of communication from application developers and provide abstracted messaging endpoints. This simplification allows developers to concentrate on the business problem instead of worrying about matters such as recovery, reliability, and operating system differences. A further feature of messaging is the decoupling of one application from another. Many mechanisms for communicating between applications require that both applications are available at the same time. While messaging may occur in a synchronous fashion, messaging may also occur asynchronously. Messages can be sent to specific applications or distributed to many different applications at the same time.

2.2 Enterprise Messaging Problem Areas

Historically, projects within the VA developed or acquired applications in a stove-piped fashion, resulting in the proliferation of duplicative functionality, such as messaging, throughout the enterprise. The duplication in functionality through different types of middleware systems, results in significant maintenance costs and system management burden. Many deployed solutions utilized proprietary components and closed standards, leading to challenges achieving agile information sharing and interoperability throughout the enterprise.

3 ENTERPRISE MESSAGING CAPABILITY ATTRIBUTES

3.1 Overview of Message Exchange Patterns (MEP)

Message transport will be accomplished through the application of a standard set of messaging exchange patterns (MEPs)[1]. The basic MEPs make use of message channels provided through the enterprise messaging capabilities, and are typically point-to-point or broadcast publish-subscribe, as explained in Reference 6 of Appendix A. Channels represent logical pathways to transport messages. A channel behaves like a collection or array of messages, but is shared across multiple computers and can be used concurrently by multiple applications.

The high-level set of standard MEPs provided by VA's enterprise messaging capabilities include:

3.1.1 One Way Delivery

This pattern refers to a message being sent from a service consumer to a service provider. It is commonly referred to as "fire and forget" (e.g., Section 4 of the OneVA Enterprise Application Architecture) while the service consumer can continue to operate without waiting for a response from the service provider. The service provider does not send a reply message, which provides opportunities to fine-tune the performance of applications and to add more flexibility after the service consumer calls the service provider. Messaging middleware solutions provide one-way delivery through point-to-point messaging channels. Appendix E describes the use case involving one-way message channels in conjunction with a commercial Enterprise Service Buses (ESB) implementation that routes the one-way messages to the appropriate destination.

The "duplex pattern" is a variation of the one-way delivery pattern that permits simultaneous one-way messages between service consumers and producers. In this pattern, both the service consumer and the service provider initiate communications utilizing point-to-point message

[1] In the context of enterprise messaging in the VA, Message Exchange Patterns (MEP) are defined as the defined sequence and cardinality of abstract messages exchanged between a consumer and a service. MEPs also define which other nodes (e.g. intermediaries) send messages to, and receive messages from, the service.

channels. When the service consumer makes a service call, the service provider can use a service consumer callback to communicate with the service consumer. This pattern is used when applications require the service provider to send a notification or alert upon completion of the service call. More information about this pattern can be found in Appendix A, Reference 7, using the Windows Communication Foundation (WCF) services as a reference implementation.

3.1.2 Request / Reply

This pattern is composed of a pair of one-way messages, and is used for synchronous messaging, Reference 2 of Appendix A. In the request/reply pattern, the service consumer sends a message to a service provider and waits for a reply. This pattern is used when an application requests data from a service provider. For example, a service consumer calls a service provider regarding an inquiry, and the consumer must wait for a response from the provider for further processing. This pattern is also used when a service consumer makes a request of a service provider to perform an action, and the service consumer requires confirmation.

The request/reply pattern is normally associated with synchronous messaging (Appendix A, Reference 6), although it may be applied to asynchronous communications. This pattern can be separated into multiple sub-patterns to address use cases that require decoupling of the service consumer from the service provider. Examples include: Request-Reaction (Appendix A, Reference 6, Section 5.2, pages 114-120) and Decoupled Invocation (Appendix A, Reference 6, Section 3.1, pages 47-51). These patterns employ a pair of separate one-way messages set up for synchronous communication, but they permit asynchronous responses on a separate message channel. This pattern decouples the request from the service consumer and the reply from the service provider, and relies on correlating the request and response messages (e.g., using a message ID or timestamp).

3.1.3 Event-Driven Messaging

This pattern is normally associated with Event-driven Architecture (EDA) and Complex Event Processing (CEP) capabilities, which are integrated with most messaging middleware solutions. This pattern is implemented with broadcast publish-subscribe ("pub/sub") channels (per Appendix A, Reference 1), and allows multiple receivers to register (subscribe) to receive notification of an event or to receive updates through a message persistence capability. This pattern applies when it is necessary to publish life event triggers (e.g., first notice of death). An example implementation is described in Appendix E. This may be implemented for other triggers such as process failures that require fault message transmissions. When an event is published into the channel, the channel delivers a message to each of the subscribing service consumers. Only the service consumers subscribed through the enterprise messaging

middleware will receive the event-driven message. This pattern allows asynchronous processing to provide a service consumer notification when previously initiated processing is complete.

Event-driven messaging may be conducted synchronously or asynchronously, but usually is conducted asynchronously to mitigate performance issues, such as locking the application while listening for an event. As described in the book *Service-Oriented Architecture: Concepts, Technology, and Architecture* by Thomas Erl, when pub/sub is implemented for event-driven messaging, the subscription is synchronous, as a callback is provided to the subscriber, but the event messages are transmitted asynchronously to the subscribers (Appendix A, Reference 3). Also, the message direction may vary depending on vendor product specifications, but typically the messaging is executed in a one-way manner. As a result, this document assumes that event-driven messaging may be constructed of multiple one-way messages and delivers messages to subscribers asynchronously.

3.2 Enterprise Messaging Persistence

Messages exchanged between a service consumer and service provider also leverage message persistence functionality included with the messaging middleware. Message persistence can be applied to each of the MEPs described in Section 3.1, and will ensure guaranteed delivery of the messages in the event of an interruption. Thomas Erl states in his book *SOA Design Patterns* that "middleware, service agents, and data stores are deployed to track message deliveries, manage the issuance of acknowledgements, and persist messages during failure conditions" (Reference 4a, Appendix A). The mechanics of message persistence vary based on the vendor products that constitute the messaging middleware (e.g., use of a cache or a message queue), and Section 4.3 provides references on how to apply message persistence using the VA's enterprise messaging implementation.

3.3 Use of ESB for Enterprise Messaging

Enterprise Service Buses (ESBs) fundamentally provide the capability to transfer messages between services. However, ESBs have no standard set of capabilities and can vary significantly depending on specific vendor implementations. ESB messaging can either be based on synchronous or asynchronous communication approaches, and use either SOAP or REST-based message packaging. Messages can either be persisted in a message queue or cache, or sent directly to a service consumer without persistence. The VA currently has deployed a national ESB as a key component of the IT infrastructure, providing messaging functions such as routing and protocol transformation, using a common set of messaging standards as described in the ESS Message Exchange Guide. More information on the VA's implementation of the ESB and how to leverage messaging capabilities is provided in Sections 4.2 and 4.3.

4 APPLICATION OF ENTERPRISE MESSAGING CAPABILITIES

4.1 Choosing MEPs

The following high-level process provides a way to guide projects to determine whether to apply a synchronous request/reply, asynchronous one-way delivery or asynchronous event-driven MEP. Included is a provision for message persistence, which may be implemented through the use of the enterprise messaging middleware. While numerous messaging methods exist, this figure is based on the assumptions that request/reply messaging (e.g., remote procedure calls or web service calls) is commonly synchronous in nature, and event-driven messaging (e.g. pub/sub) is typically asynchronous in nature and may be constructed with a set of multiple one-way messages.

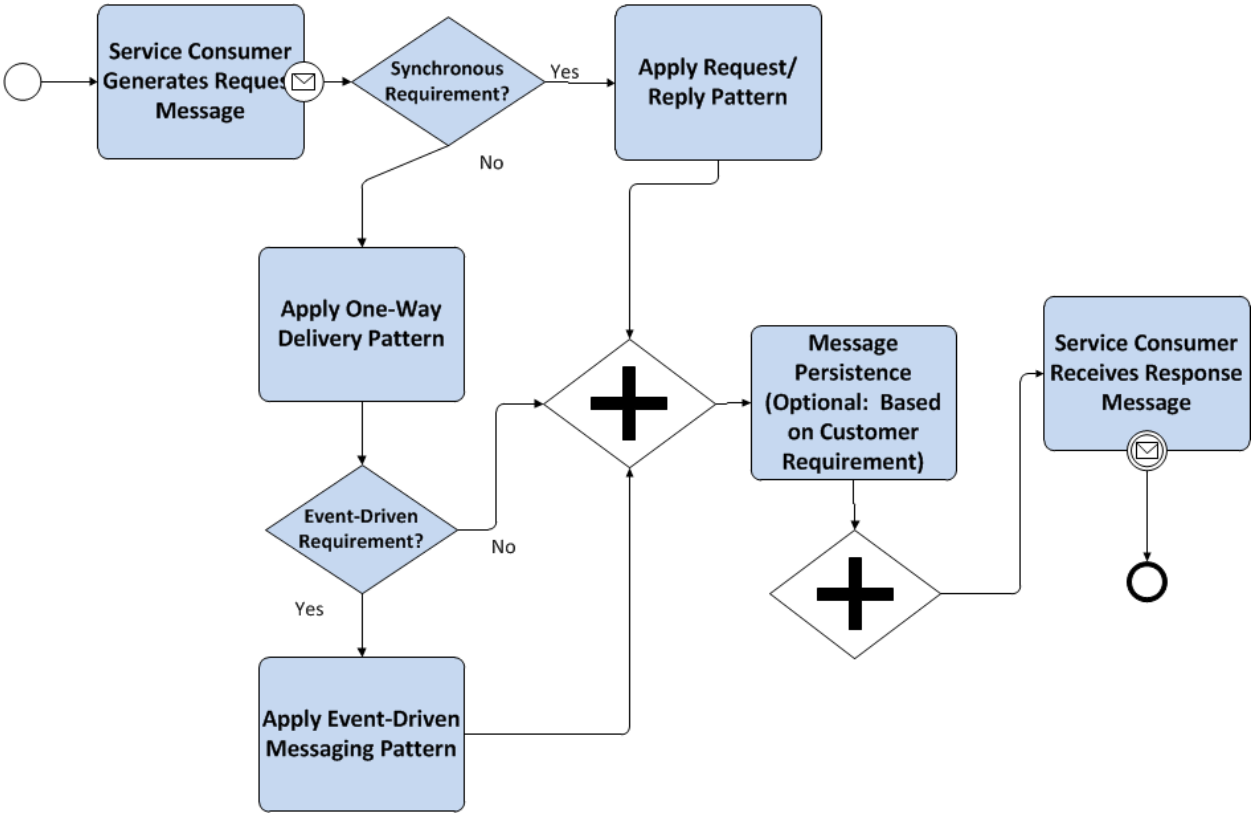

Figure 1: Process Flow for Selecting an Appropriate Set of MEPs for Service Calls

4.2 Architectural Guidance and Constraints

The "to-be" state for messaging in the VA currently involves the use of a nationally deployed ESB that provides a logically centralized enterprise messaging capability, and allows the use of open standards (REST and SOAP services; XML/JSON messages) to enhance interoperable data sharing. The ESS Message Exchange Guide developed by ASD Product Engineering provides an authoritative reference on the common set of web service messaging standards and their usage

depending on different messaging scenarios. The following are the high-level architectural guidelines for MEPs using enterprise messaging capabilities:

- All projects will use open standards for messaging (SOAP, REST) as defined in the ESS Message Exchange Guide.
- All messages used for service calls will contain sufficient context to ensure loose coupling, in accordance with the standards in the ESS Message Exchange Guide.
- All projects will apply the MEPs, including the message persistence capabilities, provided by the enterprise messaging middleware.
- All projects will apply secure message guidelines (e.g., message-level security such as WS-Security), as defined by the Authentication, Authorization, and Audit (AA&A) Enterprise Design Pattern.
- All projects will use the ESB capabilities when available to take advantage of event-driven messaging capabilities such as pub/sub or Complex Event Processing (CEP).

4.3 Implementation Guidance References

This section provides a set of implementation documents that will guide the development community is selecting the appropriate use of VA enterprise messaging capabilities and the application of MEPs described in Section 3.1. These documents provide the lower-level details for messaging design specifications to be included in the System Design Documents (SDD) prior to Milestone 1.

4.3.1 ESS Message Exchange Guide

VA has deployed the Enterprise Messaging Infrastructure (eMI) to provide the messaging capabilities described in this document for use in healthcare, benefits, and memorials data sharing needs. The eMI provides a register of the Enterprise Shared Services (ESS) to be used by all applications. , The ESS Message Exchange Guide will make Enterprise Shared Services (ESS) usable and interoperable across VA by specifying the message exchanges in the context of service interfaces to improve interoperability.

4.3.2 Enterprise Messaging Infrastructure (eMI) Guidance

The eMI team has completed detailed software development references and processes for service onboarding and have an extensive set of message pattern models and use cases based on the current IBM Integration Bus implementation. Additionally, the eMI team has completed references describing enterprise integration governance mechanisms as well as a detailed software development guide (referred to as a “software development kit”) to facilitate effective implementation of messaging capabilities. All projects developing new applications consuming ESS will coordinate with both the ASD and eMI teams to determine how to apply messaging capabilities and ESS to address their technical requirements.

Appendix A. FURTHER READING

The following links provide industry-validated design patterns related to message exchanges, and are related to the MEPs described in this document. The messaging capabilities deployed in VA align to these industry patterns. They provide the flexibility needed by projects to convey the appropriate messages to the appropriate place, and in the appropriate manner to satisfy business needs.

1. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*, Gregor Hohpe and Bobby Woolf, Addison-Wesley, 2004:
    a. http://www.eaipatterns.com/

2. *SOA Patterns,* Arnon Rotem-Gal-Oz, Manning 2012, Section 3.1 on Decoupled Invocation and Section 5 on Message Exchange Patterns (MEPs):
    a. http://arnon.me/soa-patterns/decoupled-invocation/
    b. http://arnon.me/soa-patterns/request-reply/
    c. http://arnon.me/soa-patterns/request-reaction/

3. *Service-Oriented Architecture: Concepts, Technology, and Architecture*, Thomas Erl, Prentice-Hall, 2005, Section 6.1 on Message Exchange Patterns (MEPs)

4. *SOA Design Patterns,* Thomas Erl, Prentice-Hall, 2009:
    a. Reliable Messaging: http://soapatterns.org/design_patterns/reliable_messaging
    b. Service Messaging: http://soapatterns.org/design_patterns/service_messaging
    c. Event-driven Messaging: http://soapatterns.org/design_patterns/event_driven_messaging

5. IBM Redbook on WebSphere MQ (which is included with eMI):
    a. http://www.redbooks.ibm.com/abstracts/redp0021.html?Open

6. MSDN Messaging Design Patterns:
    a. http://msdn.microsoft.com/en-us/library/aa480027.aspx

7. MEPs Overview and Tutorial for WCF:
    a. http://www.codeproject.com/Articles/566543/WCF-Message-Exchange-Patterns-MEPs
    b. https://msdn.microsoft.com/en-us/library/ff395349.aspx

8. W3C WSDL 2.0 primer on MEPS:
    a. http://www.w3.org/TR/wsdl20-extensions/#meps

Appendix B. DEFINITIONS

Key Term	Definition
Enterprise Shared Service	A SOA service that is visible across the enterprise and can be accessed by users across the enterprise, subject to appropriate security and privacy restrictions.
Message Exchange Pattern (MEP)	The defined sequence and cardinality of abstract messages exchanged between a consumer and a service. Message exchange patterns also define which other nodes (e.g. intermediaries) send messages to, and receive messages from, the service.
Service	A mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description.
Service Oriented Architecture	A paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations.

Appendix C. ACRONYMS

Acronym	Description
ASD	Architecture, Strategy and Design
CEP	Complex Event Processing
COTS	Commercial Off-the-Shelf
eMI	Enterprise Messaging Infrastructure
ESB	Enterprise Service Bus
IPT	Integrated Product Team
JSON	JavaScript Object Notation
MEP	Message Exchange Pattern
PD	Product Development
REST	Representational State Transfer
SOA	Service-oriented Architecture
SOAP	Simple Object Access Protocol
XML	Extensible Markup Language

Appendix D. APPLICABLE REFERENCES, STANDARDS, AND POLICIES

This Enterprise Design Pattern is aligned to the following VA OI&T references and standards applicable to all new applications being developed in the VA, and are aligned to the OneVA ETA:

#	Issuing Agency	Applicable Reference/Standard	Purpose
1	VA OIS	VA 6500 Handbook	Directive from the OI&T OIS for establishment of an information security program in the VA, which applies to all applications that leverage ESS.
2	VA ASD	VA Enterprise Design Patterns, Office of Technology Strategies	Provides references to the use of enterprise messaging capabilities as part of the integration with SOA support infrastructure services. These documents are intended to standardize and constrain the solution architecture of all healthcare applications in the VA.
4	VA ASD	Enterprise Application Architecture (EAA) Section 4	Provides technical underpinnings for cross-cutting development concerns for VA applications, including the use of SOA messaging.
5	VA ASD	SOA Technical Framework (SOA-TF) Section 2-4	Provides technical underpinnings for cross-cutting development concerns for VA applications, including SOA messaging and mediation. Section 2.4.3 and 3.7.3.1provides details on ESB used for messaging.
6	VA ASD	ESS Strategy Document and Directive	Provides the overarching strategy for developing, deploying, and managing ESS throughout the VA. ESS guidelines for Message Exchange provide the consensus set of standards for interoperable messaging.
7	VA ASD	VA ETA Compliance Criteria	Enterprise messaging currently maps to 2.6.1 and 2.6.2 with regard to messaging standards. Section 2.6 is aligned to the Enterprise SOA Design Pattern developed by ASD Office of Technology Strategies.
8	VA ASD	VA Technical Reference Model (TRM)	Enterprise messaging directly maps to approved tools and technologies located in the Integration Software and Data Integration categories.

Appendix E. EXAMPLE ENTERPRISE MESSAGING IMPLEMENTATIONS

There are multiple messaging solutions leveraging message exchange patterns in the VA. The following subsections provide example solutions encompassing both health and benefits domains, and represent current uses of enterprise messaging capabilities.

1. **Event-Driven Messaging through Life Event Notification Service (LENS)**

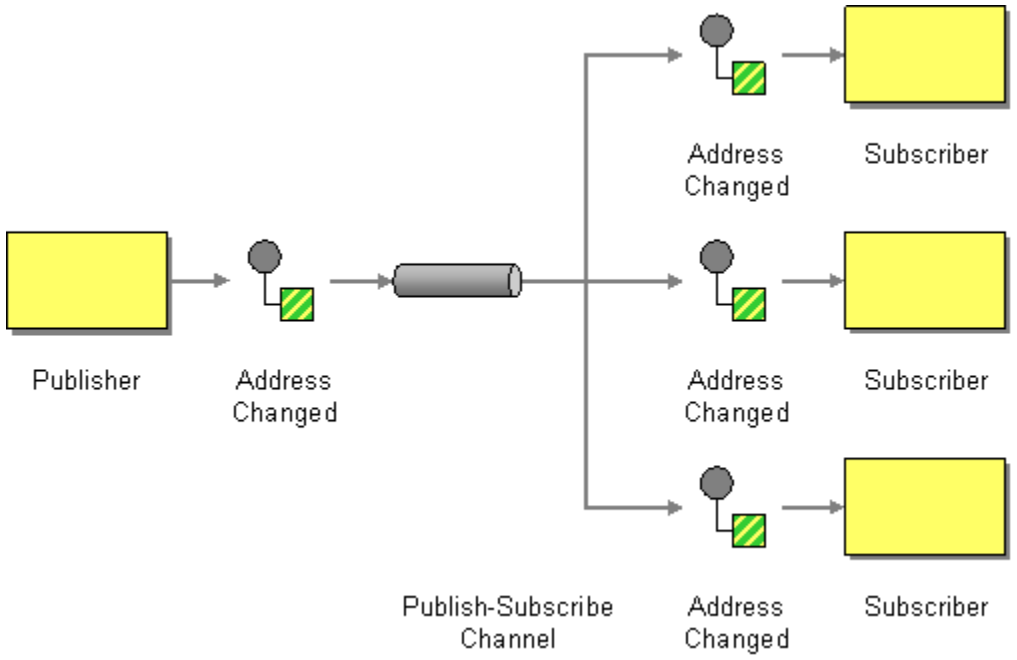

Figure 2 – LENS Publish/Subscribe Use Case

The Virtual Lifetime Electronic Record (VLER) LENS serves as a proxy between publisher and subscriber services of enterprise shared data stores, for events triggered by changes in data. Publishers post messages to topics or channels in LENS service and subscribers register subscriptions with that topic or channel in LENS service, letting LENS service perform the filtering.

The Life Event Notification Service follows the traditional publish and subscribe capability in which a publisher may publish an event to a topic and a subscriber may receive events for one or more topics. Publishers are not blocked while producing events. Subscribers can get asynchronous notifications of occurrences of events while performing concurrent tasks. Production and consumption of events does not happen in the main flow of control, so interaction does not have to be synchronized. Publishers expose their content as Atom (or RSS) feeds to VLER LENS Server Topic. An event notification (RSS feed) is broadcasted to all authorized subscribers of that topic.

The relationship between the publisher and subscriber is mediated by the LENS service that receives publication requests, broadcasts event notifications to subscribers, and enables privileged entities to manage lists of people or applications that are authorized to publish or subscribe. The focal point for publication and subscription is a "Topic" or a named logical channel to which publishers send data and from which subscribers receive event notifications.

There is expected to be a set of 80 or more topics which an application can subscribe to. These topics include “change of address”, “first notice of death” and other events which are detected by one application and processed by another. Topics can be created/ deleted by the owner/administrator of VLER LENS. Topics can also maintain a history of events for auditing and Guarantee of Service purposes to subscribers.

2. One-Way Messaging through Business Transaction Service

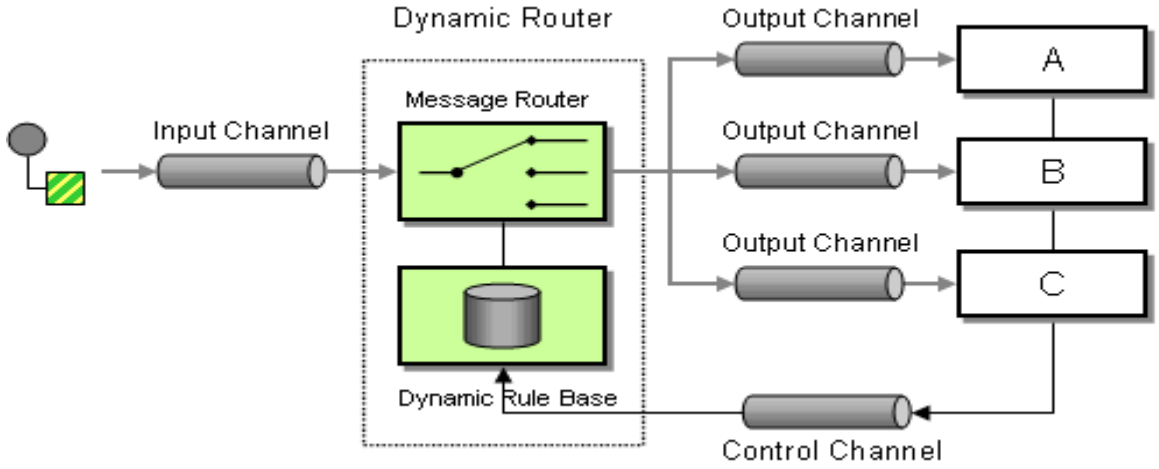

Figure 3 – BTS Dynamic Router Use Case for One-Way Messaging

The Business Transaction Service (BTS) provides a mechanism for VA Partner Organizations to request services to be performed inside and outside the VA that have a real world effect. The BTS will not perform the service directly, but rather it will forward the request to the appropriate enterprise service based on message content.

To facilitate the routing, the BTS will reference a rule base that describes the conditions under which an enterprise service can handle a request, the synchronicity of the request, as well has how to contact the service. The rule base will be automatically configured by the participating enterprise service. On startup, the enterprise service will publish its rules to the BTS on a special control channel.

The rule base will be persisted to permanent storage so that in the event the BTS should fail it would be able to recover the routing rules when it is restarted. All Messages will be persisted in a data store for auditing and guaranteed delivery purposes (Asynchronous Transactions). If a service becomes unavailable, the BTS will resend the message when the service is available again (Asynchronous Transactions). In the case of a synchronous transaction, the Consumer will receive a dead letter message in return.

When a message arrives within BTS, BTS will evaluate the message based on rules to determine which services should receive the message. This allows for efficient routing instead of each service handling its own message evaluation.

BTS will utilize the practice of "*decoupling of services.*" Decoupling of services means that services are insulated from other services. By decoupling services, adding, removing, changing or a malfunction in one part of a service will not affect other services.

BTS will implement the Enterprise Integration Pattern of Dynamic Router. A Dynamic Router is a router that is self-configured based on configuration messages from participating destinations. A dynamic router uses a special control channel to receive message routing rules. During system start-up, each potential recipient sends a special message to the Dynamic Router on this control channel, announcing its presence and listing the conditions under which it can handle a message. The Dynamic Router stores the 'preferences' for each participant in a rule base. When a message arrives, the Dynamic Router evaluates all rules and routes the message to the recipient whose rules are fulfilled. This allows for efficient, predictive routing without the maintenance dependency of the Dynamic Router on each potential recipient.