New Mexico HHS 2020

Reference Architecture

(RA)

Version: 1.0

Last Modified: Aug <TBD> 2019

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description of Changes** |
| 0.1 | 10/5/2016 | Marty | Original Version |
| 0.2 | 12/01/2017 | Marty | Latest update, adding content |
| 0.3 | 12/11/2017 | Marty | Incorporating content from TMS, SDD and other CMS documents |
| 0.4 | 4/2/2018 | Marty | Updated content, moving content to appendices |
| 0.5 | 4/5/2018 | Marty | Updated Glossary |
| 0.6 | 5/18/2018 | Marty | Updated sections on Cloud computing in the Infrastructure Architecture |
| 0.7 | 6/15/2018 | Marty | Updated to match James’ requirements for DoIT edition |
| 0.8 | 11/TBD/2018 | V. Vilensky | Gap revisions and major section realignment |
| 0.9 | 07/20/2019 | V. Vilensky | Major document structure revision to accommodate standards, governance mechanisms and work products for all 3 realms of MITA EA |
| 1.0 | 08/<TBD>/2019 | V. Vilensky | Re-write based on structure revisions approved in version 0.9 |

**TABLE OF CONTENTS**

[1. HHS 2020 Reference Architecture Overview 7](#_Toc12475142)

[1.1 Governing Standards 8](#_Toc12475143)

[1.2 Governance Processes 8](#_Toc12475144)

[1.3 Architecture Work Products 9](#_Toc12475145)

[1.4 Reference Architecture Document Purpose 11](#_Toc12475146)

[2. HHS 2020 Technology Architecture Goals 12](#_Toc12475147)

[2.1 Deploy Modular, Autonomous, Decoupled Components 12](#_Toc12475148)

[2.2 Achieve SOA-Compliant Design and Integration 12](#_Toc12475149)

[2.3 Technology Independence 14](#_Toc12475150)

[3. Infrastructure Architecture 17](#_Toc12475151)

[3.1 Lower Non-Load Balanced Environment Pattern 17](#_Toc12475152)

[3.2 Production Non-Load Balanced Environment Pattern 17](#_Toc12475153)

[3.3 Lower Load Balanced Environment Pattern 17](#_Toc12475154)

[3.4 Production Load Balanced Environment Pattern 17](#_Toc12475155)

[3.5 Disaster Recovery Cloud Environment Pattern 17](#_Toc12475156)

[4. Information Architecture 18](#_Toc12475157)

[4.1 Data Repositories 18](#_Toc12475158)

[4.1.1 Raw Data Lake 18](#_Toc12475159)

[4.1.2 SMR/ODS 18](#_Toc12475160)

[4.1.3 MDM 18](#_Toc12475161)

[4.1.4 Data Warehouse 18](#_Toc12475162)

[4.2 Information Flow 18](#_Toc12475163)

[4.3 End-to-End Data Integration 18](#_Toc12475164)

[4.4 Enterprise Metadata Management 18](#_Toc12475165)

[4.4.1 MarkLogic Metadata Management 18](#_Toc12475166)

[4.4.2 IBM InfoSphere Governance Catalogue Metadata Management 18](#_Toc12475167)

[4.5 Data Security 18](#_Toc12475168)

[4.5.1 Encryption at Rest 18](#_Toc12475169)

[4.5.2 Restricted Data Masking and Deidentification 18](#_Toc12475170)

[4.6 IA Service Enablement 19](#_Toc12475171)

[5. Software architecture 20](#_Toc12475172)

[5.1 Sub-System Overview 22](#_Toc12475173)

[5.2 Security 25](#_Toc12475174)

[5.2.1 Introduction to Multi-Factor Security 25](#_Toc12475175)

[5.2.2 User Authentication 28](#_Toc12475176)

[5.2.2.1 State Employee Authentication 28](#_Toc12475177)

[5.2.2.2 BPO/Partner User Authentication 29](#_Toc12475178)

[5.2.2.3 External User Authentication 30](#_Toc12475179)

[5.2.2.4 External System Authentication 31](#_Toc12475180)

[5.2.3 Authorization 31](#_Toc12475181)

[5.2.3.1 State Employee Authorization 31](#_Toc12475182)

[5.2.3.2 BPO/Partner User Authorization 32](#_Toc12475183)

[5.2.3.3 External User Authorization 33](#_Toc12475184)

[5.2.3.4 System Authorization 34](#_Toc12475185)

[5.2.1 Encryption 34](#_Toc12475186)

[5.2.1.1 Public Key Infrastructure 34](#_Toc12475187)

[5.2.1.2 In-Transit 34](#_Toc12475188)

[5.2.1.3 At-rest 34](#_Toc12475189)

[5.2.2 Message Integrity 34](#_Toc12475190)

[5.2.3 Auditability 34](#_Toc12475191)

[5.3 Architectural Layering 34](#_Toc12475192)

[5.4 Service Orientation 37](#_Toc12475193)

[5.4.1 Role of the ESB 39](#_Toc12475194)

[5.4.2 Role of BRE 40](#_Toc12475195)

[5.4.3 Transaction management 41](#_Toc12475196)

[5.4.4 Shared Services 45](#_Toc12475197)

[5.4.4.1 Enterprise Identity Management 46](#_Toc12475198)

[5.4.4.2 Enterprise Communications Management 46](#_Toc12475199)

[5.4.4.3 Enterprise Document Management 46](#_Toc12475200)

[5.4.4.4 Address Validation and Verification 46](#_Toc12475201)

[5.4.4.5 EDI 46](#_Toc12475202)

[5.4.5 Specialized Business Services 46](#_Toc12475203)

[5.4.6 Presentation Services 46](#_Toc12475204)

[5.4.7 Data-as-a-Service 47](#_Toc12475205)

[5.4.8 Data Value-Added Services 47](#_Toc12475206)

[5.5 Multi-channel Architecture 47](#_Toc12475207)

[5.5.1 External Portal Web UI Guidance 47](#_Toc12475208)

[5.5.1.1 Web UI for External Use 47](#_Toc12475209)

[5.5.2 Internal Portal Web UI Architecture 50](#_Toc12475210)

[5.5.2.1 Human Use Case Participant Components 51](#_Toc12475211)

[5.5.2.2 Participating Components Interaction Sequences 58](#_Toc12475212)

[5.5.2.2.1 “Home” Page Creation 59](#_Toc12475213)

[5.5.2.2.2 Non-workflow Page Invocation 62](#_Toc12475214)

[5.5.2.2.3 Workflow (Initiating/Participating) Page Invocation 65](#_Toc12475215)

[5.5.2.3 Web UI for Internal Use 67](#_Toc12475216)

[Appendix a: Solution Modeling Guidelines 70](#_Toc12475217)

[Appendix B: PADU Prescriptive Framework 72](#_Toc12475218)

[Appendix C: Glossary 75](#_Toc12475219)

**LIST OF FIGURES**

The following is a List of Figures appearing within the document along with a page reference for each identified figure.

[Figure 1: Conceptual Architecture of MMISR 21](#_Toc14686218)

[Figure 2: Architectural Layering of MMISR 35](#_Toc14686219)

[Figure 3: Details on the SOA-Based Design of the MMISR Integration Platform 38](#_Toc14686220)

[Figure 4: BPEL Transaction Boundary Illustration 42](#_Toc14686221)

[Figure 5: BPEL Compensating Transaction Illustration 43](#_Toc14686222)

[Figure 10: Web UI-related Components for External Use 48](#_Toc14686223)

[Figure 6: Human Use Case Participants 51](#_Toc14686224)

[Figure 7: Illustrative “Home” Page creation Component interaction sequence 59](#_Toc14686225)

[Figure 8: Illustrative Non-Workflow Page Invocation Component interaction sequence 62](#_Toc14686226)

[Figure 9: Illustrative Workflow (Initiating/Participating) Page Invocation Component interaction sequence 65](#_Toc14686227)

[Figure 11: Web UI-related Components for Internal Use 67](#_Toc14686228)

**LIST OF TABLES**

The following is a List of Tables appearing within the document along with a page reference for each identified table.

[Table 1: Governing Standards Locations in MITA TMS Deliverable 8](#_Toc14686229)

[Table 2: Change Management Mechanism Definitions 9](#_Toc14686230)

[Table 3: EA Work Products and Deliverables 10](#_Toc14686231)

[Table 4: Key COTS Technologies 15](#_Toc14686232)

[Table 5: SOA Industry References 39](#_Toc14686233)

[Table 6: Human Workflow Participating Component Summary 51](#_Toc14686234)

# HHS 2020 Reference Architecture Overview

The State of New Mexico Human Services Department (HSD) is sponsoring an HHS-wide investment in Enterprise Architecture (business, information and technology architectures). The key drivers are to achieve increased quality of outcomes for NM citizens and state workers, as well as to realize higher qualities of shared data and reductions in operating expenses through technology sharing both inside the State as well as with external partners and agencies.

The vision is to put into place an enterprise-wide technology ecosystem for all HHS systems and organizations to integrate more seamlessly, to reuse technology assets, to optimize the experience of end consumers (patients, members, and dependents), to optimize the relationships with vendors and partners (providers and payors), and to optimize the operation of state organizations and workers.

By adopting a modular approach that blends technology components with Business Process Outsourcing (BPO) solutions, HSD anticipates realizing the following benefits:

* Opportunity to support an outcomes-based approach to planning, delivering and assessing service delivery – focusing on the end impact and value to New Mexico citizens instead of on internal processing requirements;
* Realization of a customer-focused approach to service management and delivery that can make it easier for New Mexico citizens to understand and receive needed services and for providers to interact with the State;
* Increased ability to respond promptly and insightfully to changing program or population needs;
* Enhanced ability to operate in a dynamic environment with increasingly restricted funding without degrading service levels, by leveraging modern, standards-based and cloud-based solutions;
* Greater flexibility to take advantage of rapidly evolving technologies to achieve service improvements and/or to reduce maintenance and operation costs; and
* Reduced redundancy through sharing technology, services, data and processes enterprise-wide to deliver high-quality customer service.

In accordance with Medicaid Information Technology Architecture (MITA) v3.0 initiative framework, HHS 2020 Enterprise Architecture (EA) is organized into 3 realms:

1. Business Architecture
2. Information Architecture
3. Technology Architecture

Each of the realms is subject to governance spanning the entire program lifecycle from the earliest visions of the future state business architecture to operation of technological solutions in production environments. Design-time, run-time and change-time governance includes standards to be applied to all architecture deliverables.

## Governing Standards

Standards governing HHS 2020 EA have been defined as part of the high-level vision expressed in MITA Technical Management Strategy (TMS) deliverable. The following table lists locations of all applicable standard families within MITA TMS:

Table 1: Governing Standards Locations in MITA TMS Deliverable

|  |  |
| --- | --- |
| **Standards Category** | **MITA TMS Section** |
| Government and Regulatory | 2.1.1 |
| Visual Modeling | 2.1.2 |
| Enterprise Architecture | 2.1.3 |
| Business Architecture | 2.1.4 |
| Information Architecture | 2.1.5 |
| Technical Architecture (aggregating Infrastructure and Software Architecture standards) | 2.1.6 |

## Governance Processes

Change management mechanisms are invoked whenever decisions are needed by the program related to scope, schedule and approach either on the global level of HHS 2020 initiative or pertinent to individual architecture realms.

Table 2: Change Management Mechanism Definitions

| **Realm** | **Mechanism** | **Description** | **Defined in Deliverable** | **Section** |
| --- | --- | --- | --- | --- |
| Global | Change Control Board | Program-wide scope, schedule and cost change management | PMO 10? | ? |
| Business Architecture | Business Technology Council (BTC) | Approval of important business architecture considerations | ? | ? |
| Information Architecture | Data Governance Committee (DGC) | High-level Information Architecture decisions | ? | ? |
| Information Architecture | Data Management Workgroup | Support DGC with lower-level contributions | ? | ? |
| Technical Architecture | Architecture Review Board (ARB) | Software and hardware technology decisions, major architectural changes | ? | ? |

## Architecture Work Products

The key architecture work products and corresponding deliverables from all 3 realms are summarized in the table below.

Table 3: EA Work Products and Deliverables

| **Realm** | **Work Product** | **Description** | **Found in Deliverable** | **Section** |
| --- | --- | --- | --- | --- |
| Business Architecture | Business Architecture Vision | High-level definition of HHS 2020 enterprise business architecture | MITA TMS | 3 |
| Business Architecture | Business Architecture Details | Detailed business processes, goals, success factors, organizational models supporting the new business architecture | ? | ? |
| Information Architecture | Enterprise Information Architecture (IA) Vision | Preliminary, high-level IA definition with a few known gaps | MITA TMS | 4 |
| Information Architecture | Enterprise IA Elaboration | Closing the gaps in IA vision, making it ready for detailed design | This document | 4 |
| Information Architecture | Data Models | Detailed Data Models covering all of IA capabilities | ? | ? |
| Technical Architecture | Software Architecture Vision | High-level software architecture | MITA TMS | 5 |
| Technical Architecture | Software Architecture | Detailed architecture patterns, approaches and sample designs for all sub-systems | This document | 5 |
| Technical Architecture | Software Design | Detailed designs of HHS 2020 software sub-systems | ? | ? |
| Technical Architecture | Infrastructure Architecture Vision | High-level infrastructure architecture | MITA TMS | 6 |
| Technical Architecture | Infrastructure Patterns | Environment design patterns | This document | 3 |

## Reference Architecture Document Purpose

<Document purpose described. Includes primary role of holding IA and Technology architecture work products. Include revision to the legacy text below.>

This document provides guidance and direction on the evolution and integration of the systems which will utilize the HHS 2020 “ecosystem.” The term “ecosystem” is a term used to represent the full set of evolving technologies and systems that support HHS 2020; everything from user experiences on mobile devices to backend databases, to shared services like Single Sign-On, and enabling technologies like Enterprise Service Bus (ESB). This Reference Architecture (RA):

* Defines the types of technologies, the technology standards, design goals and objectives for all technologies in the HHS 2020 ecosystem;
* Provides guidance for making decisions associated with application services, data sharing, and seamless interoperability of applications; and
* Reduces risk to system development by decreasing the number of custom solutions and promoting interoperability and data sharing.

# HHS 2020 Technology Architecture Goals

In order to achieve the vision expressed in the MITA TMS deliverable, the following goals are paramount in defining a technology architecture:

## Deploy Modular, Autonomous, Decoupled Components

The HHS 2020 TA requires a modular, flexible approach to systems development, including the use of reusable assets. The TA is broken into separate Architectural Domains, which support modularity, evolve independently, and which will deliver business solutions that meet or exceed business needs. One major advantage of separate domains is the creation of modular components that are autonomous and decoupled from one another. This minimizes the impact of changes in any one asset and protects an asset from being hampered by a failure in another asset. Other advantages of componentization include:

* Enable improved creation of applications through the use of modern DevOps tools and processes;
* Enable the creation and evolution of composite applications through the use of Business Process Management (BPM), Workflow, Orchestration, and Business Rules Engine (BRE) tools;
* The separation of business rules from core programming; and the availability of business rules in both human and machine-readable formats allow component and process changes by changing business rules that are not codified in software;
* Improve the quality, reliability and performance of HHS 2020 systems and assets by allowing them to specialize;
* Implement systemic Performance Management.

## Achieve SOA-Compliant Design and Integration

This goal defines how to achieve modular, autonomous, decoupled components. SOA compliance embodies the following design principles (see table below for public references):

* **Abstraction via Use of Policies.** Policies hide implementation details and constraints of services from outside service clients. Details of a service’s implementation are hidden completely from all service clients. Clients operating in a .NET environment, that may be restricted to only use the HTTP protocol, should not be prevented from interacting with services operating in a JMS stack. Services that can use a particular encryption/decryption standard, or that use SOAP vs JSON messages should still be able to interact. The ESB stores, manages and enforces all service Policies.
* **Boundaries Defined by Contracts.** Contracts describe the purpose of a service interface. They also serve the critical goal of decoupling services by eliminating legacy integration methods of sharing Application Programming Interfaces (APIs), classes or memory references (remote procedure calls (RPC), remote method invocation (RMI), remote JDBC/ODBC database connections). This provides for decoupled software assets that make no assumptions about each other, but just interact based on what the services “advertise” about themselves.

Much of the current evolution of “micro services” actively use “APIs” as references to service interfaces. This actually causes confusion, because APIs tend to be tightly coupled, where service interfaces are by Contract and by Policies, not APIs per se.

* **Loose Coupling.** Although a byproduct of integration by contract, this is a goal that requires specific focus. Rigorously applied, this ensures that service integration eliminates to the highest degree possible interdependencies between services. Taken to the maximum extent possible, services can become autonomous and able to defend their integrity in the face of unforeseen technical and business events. Loose Coupling involves intentional focus on the following:
  + Asynchronous vs Synchronous Communications – asynchronous messaging between services helps ensure services are independent from one another and are stateless and idempotent where possible;
  + Location Independence – service invocations should be bound at run-time, not at compilation of software; this ensures the physical deployment, replication and scaling services is via a virtual connection instead of a specific IP address, memory address or URI;
  + Version Independence – services are envisaged to change over time as they take on more functional responsibilities; it is therefore imperative that the existing interfaces change as little as possible and the services support backward-compatibility with existing APIs;
  + Schema Adherence – services should be invoked and should reply to requests with DGC-approved shared schemas; this ensures implementation details are hidden from other client services.
* **Autonomy.** Taken as an ultimate goal of service design, Autonomy ensures that a service is resilient and available regardless of other services they may use. This design goal also ensures that the underlying mechanisms used within a service may be changed without causing any changes in its clients. Because of the run-time binding employed by SOA, as implemented through an ESB, services should be able to evolve in real time without causing any failures to clients, as long as its Policies and Contracts stay the same.
* **Message-Encapsulated, Document-Based Integration.** This kind of integration relies on passing data encapsulated in “documents” between services, usually referred to as “messages.” A document in the context of SOA is a rigorously structured human-readable file of name-value pairs encoded as Simple Object Access Protocol (SOAP) or JavaScript Object Notation (JSON) files. As opposed to sharing classes between tightly coupled APIs, messages enable data to be shared via decoupled interfaces. Messages are managed via an Enterprise Service Bus. The ESB helps ensure decoupling, as it manages contracts and policies manage message flow and help decouple services. More modern “microservices” approaches are beginning to advocate passing messages between very fine-grained services, but do not yet require the use of a mediation “service” such as an ESB.
* **Reusable.** “Good” service design ensures that the long-term evolution of services’ capabilities are architected and designed into the service from the start. This applies to both services as well as messages. Even if the future capabilities are not yet utilized, they should be designed into the service to alleviate ripple effects when those capabilities are released. This is part of loose coupling and autonomy in a SOA ecosystem.
* **Stateless** processing that supports idempotency and reduces coupling
* **Discoverable.** Services will store metadata about their Contracts and Policies in a repository where the services are discoverable by other services at design, test and run-time. Of services and messages for the purpose of reuse
* **Composability** that supports orchestrations of course-grained services from finer-grained services

## Technology Independence

To offer maximum flexibility to various HHS 2020 ecosystem participants to deploy best possible solutions to State’s needs, we are pursuing a technology-independent architectural approach to all specialized and shared sub-systems. By following standards for technological interoperability, service orientation and componentization we intend to bring together optimal solutions developed in-house and sourced from contracted vendors regardless of their respective implementation approaches. Such an enterprise architecture would allow for eventual vendor changes and additions with minimal technological and operational disruption. Additional benefit would arise from the ability to replicate our proven solution approach to other departments with NM State Government as well as to other States’ Medicaid agencies, thus furthering a consistent, nation-wide Medicaid Ecosystem.

At present time the key Commercial-off-the-shelf (COTS) technologies making-up HSS 2020 ecosystem are:

Table 4: Key COTS Technologies

|  |  |
| --- | --- |
| Technology | Purpose |
| Microsoft Active Directory v.<TBD> | 1. Public Key Infrastructure (PKI) supporting enterprise cryptographic needs 2. State employee enterprise group membership 3. State employee authentication through user ID and password combination |
| Oracle Identity and Access Management (IdM) platform v.<TBD> | 1. Multi-factor authentication for all HHS 2020 users, including State employees, BPO employees, providers, constituents and system/service accounts via <TBD Oracle IdM name> 2. Application role-based authorization for all authenticated users applied to all publicly accessible and private URLs (web pages, service endpoints, process orchestration end points) |
| Oracle Fusion Middleware suite v.<TBD> | 1. Long-running business process configuration, orchestration and monitoring via <TBD Oracle Fusion BPEL component name> 2. Service publishing, discovery and run-time invocation via <TBD Oracle API Manager name> 3. Service end-point hosting and message routing via <TBD Oracle Enterprise Service Bus name> 4. Enterprise rule management and enforcement via <TBD Oracle BRE name> 5. Service implementations using EAI adaptors and short running orchestrations via <TBD Oracle EAI name> |
| Liferay User Interface Portal v.<TBD> | 1. Unified Presentation layer web-based portals supporting all HHS 2020 stakeholders via <TBD Liferay portal name> 2. Digital content management via <TBD Liferay DCM name> |
| <TBD Enterprise KM solution> |  |
| MarkLogic Non-SQL Database v.<TBD> | 1. Bi-directional ETL file-based data supporting both legacy system migration onto HHS 2020 platform and ongoing file-based integration with external systems via <TBD> 2. Metadata management via <TBD> 3. Master Data Management via <TBD> |
| <TBD DS sub-system> |  |
| <TBD BPO systems> |  |
|  |  |

# Infrastructure Architecture

<Explain overall approach using virtualization, scaling etc. Provide Infrastructure Architecture environment pattern overview for all conceivable types of environments.>

## Lower Non-Load Balanced Environment Pattern

<Environment purpose, diagram of boundaries, nodes, tags associations etc., necessary description>

## Production Non-Load Balanced Environment Pattern

<Environment purpose, diagram of boundaries, nodes, tags associations etc., necessary description>

## Lower Load Balanced Environment Pattern

<Environment purpose, diagram of boundaries, nodes, tags associations etc., necessary description>

## Production Load Balanced Environment Pattern

<Environment purpose, diagram of boundaries, nodes, tags associations etc., necessary description>

## Disaster Recovery Cloud Environment Pattern

<Environment purpose, diagram of boundaries, nodes, tags associations etc., necessary description>

# Information Architecture

<Provide IA overview with further elaboration of concepts and principles laid out in the vision in MITA TMS.>

## Data Repositories

### Raw Data Lake

### SMR/ODS

### MDM

### Data Warehouse

## Information Flow

## End-to-End Data Integration

## Enterprise Metadata Management

### MarkLogic Metadata Management

### IBM InfoSphere Governance Catalogue Metadata Management

## Data Security

### Encryption at Rest

### Restricted Data Masking and Deidentification

## IA Service Enablement

<Provide links to sections in the Software Architecture portion covering canonical and data value-added services.>

# Software architecture

<Update/re-write section with latest model and descriptions.>

In accordance with HHS 2020 EA standards all software-related architecture views, including the Conceptual Architecture depicted in Figure 4 are modeled in Unified Modeling Language (UML) v.2.4.1. Figure 4 shows groupings of components packages comprising both the HHS 2020 Enterprise and the External Systems. SOA integrations are shown as bi-directional associations decorated with a descriptive stereotype. While significant Package and Component Package contents are included, the package-to-package and component-to-component dependencies are omitted from the conceptual architecture depiction.

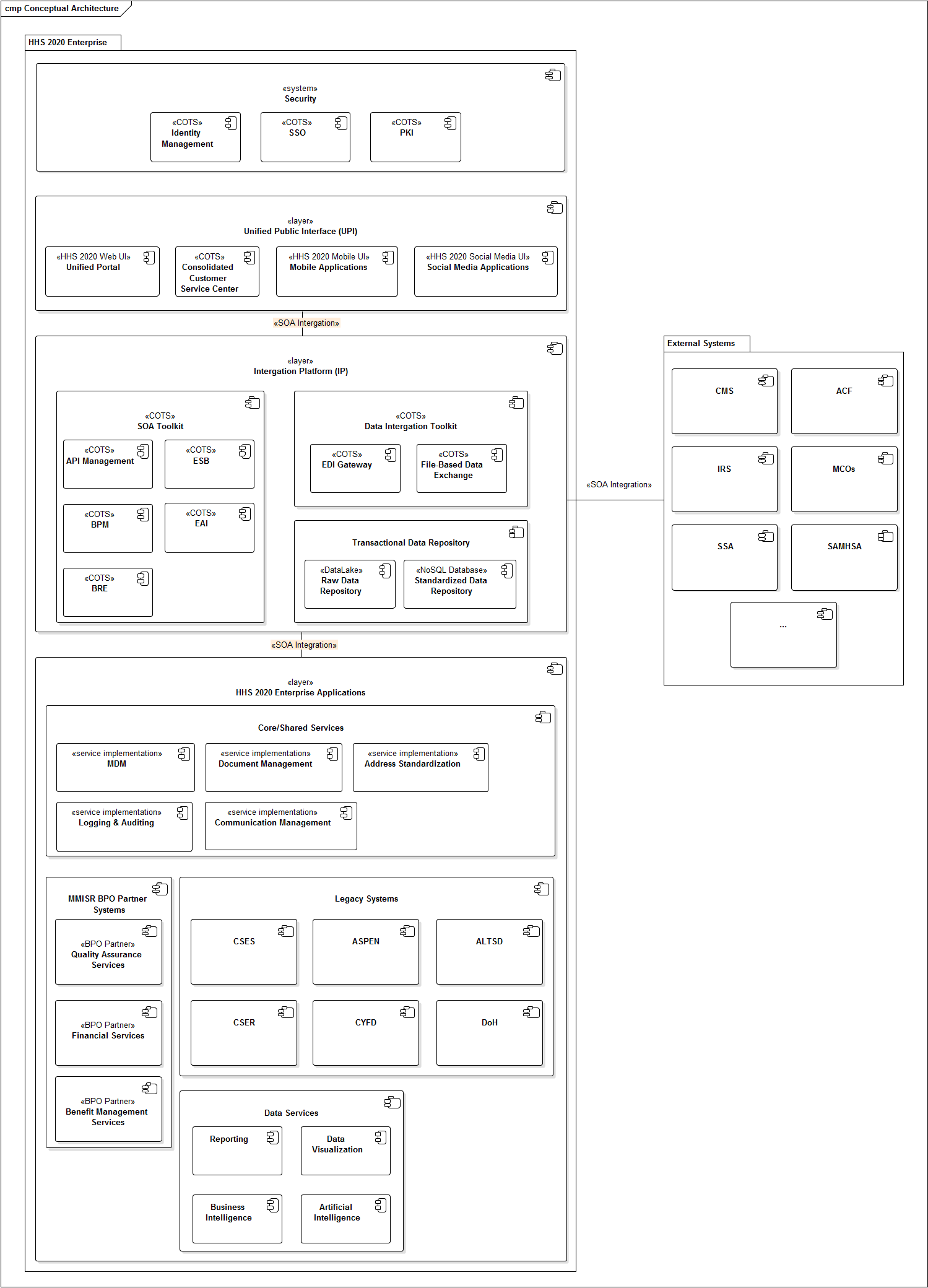


Figure 1: Conceptual Architecture of MMISR

## Sub-System Overview

**Security**

All requests for functionality and data contained within the HHS 2020 Enterprise pass through the Security System. It is comprised of purchased COTS components, configured to meet HHS 2020 requirements for user authentication, role-based access authorization, cryptographic public key infrastructure management and Single Sign-On allowing authenticated HHS 2020 users seamless access to BPO partner and external systems.

**Unified Public Interface**

The user interface layer of HHS 2020 Enterprise is executed internally as a separate project. The layer includes all human user interaction channels, including Unified Portal (for all external and internal web interfaces and applications), Mobile and Social Media Access (for incorporating mobile devices/interfaces and tying into social media outlets), and a Customer Care Service Center (CCSC)/Call Center Interface (which may be available via the Unified Portal, subject to a future decision).

**Integration Platform (IP)**

The sole responsibility of the Systems Integration vendor, this is the central mechanism for integrating all backend systems, all interfaces to external systems, and users via the Unified Public Interface. The IP layer contains three distinct sub-groupings of functionality.

***SOA Toolkit*** includes Enterprise Service Bus and other service-oriented architecture enablers like COTS EAI adaptors to be used in service implementations, Business Process Management components for process definition and execution, Business Rule Engine to assist in rule-based business processing and context-based message routing through the ESB, and API Management for service discovery and invocation by the UPI layer clients. All systems, subsystems, modules and services will integrate using SOA principles of decoupled services exchanging messages through a common integration platform, briefly listed below:

* Services will be autonomous and decoupled from other services.
* Services will be discoverable through the IP.
* Services will be composable.
* Services will interoperate via sending/receiving asynchronous messages.
* Services and messages will be built to be reusable.
* Services will be stateless and idempotent, as much as possible.
* Services will communicate via asynchronous messages.
* Services will hide their implementation details from other services.

All services will be registered in the API Manager services catalog, which will enforce any policies regarding knowledge of and access to those services. All shared schemas which comprise messages will also be registered in the IP, which will enforce data integrity and Data Quality Management where required. All message requests and responses will be managed by the Integration Platform, which will enforce common security protocols to ensure access to the IP and integrity of messages is secure.

***Data Integration Toolkit*** allows for EDI transaction handling and all other ongoing file-based data exchanges in standard (e.g. HL7) and custom formats with HHS 2020 Legacy and external systems.

***Transactional Data Repository*** serves a dual purpose. It is a continuously updated Operational Data Store (ODS) housing all data required for reporting, analytics and operational decision-making in a standardized canonical form. ODS enables uniform canonical data access by serving as a back-end to Data-as-a-Service services. In addition to the permanent ODS role this sub-system takes on the System Migration Repository (SMR) responsibilities in support of sunsetting of legacy applications and onboarding of new internal and exterenal (BPO partner-provided) applications. Data migrated though SMR will be availab le to consumers in extract files created as required to support migration and new system onboarding activities.

All data entering the transactional repository sub-system will be stored indefinitely in the Raw Data Lake (RDL) repository to prevent any information loss and to provide additional analysis opportunities that are not easily achived with ODS.

**HHS 2020 Enterprise Applications**

This is a grouping for all new BPO partner applications, services and legacy systems integrated into HHS 2020 Enterprise.

***Shared Services***  are the responsibility of the SI vendor to implement and includes all software which shared services to the rest of the MMISR ecosystem are. This logical group consists of:

* One or more Master Data Management (MDM) instances (for managing Master Indices for members/consumers, providers, and potentially others) and access to shared reference data
* Document Management solution (for all scanning, storing, imaging, creating and managing all documents)
* Communication Management solution enabling outreach to constituents via electronic and print channels
* Address Standardization and Verification service
* A set of shared auditing/logging services

***MMISR BPO Partner Systems*** are component packages procured from BPO Partners which implement distinct business functionality. This consists of all decoupled, stand-alone applications licensed/implemented through the distinct procurements of the MMISR program:

* the Quality Assurance system
* the Behavioral Health Management system
* the Financial Services system

Each of these modular systems will rely on tooling procured and implemented via the Systems Integrator procurement, which is responsible for the IP, common SOA tooling Business Process Modeling (BPM), a Business Rules Engine (BRE) and a Workflow solution.

**Legacy Systems**

This an architectural grouping of systems which exist within HHS, and which may or may not provide functionality for MMISR. This includes all existing systems that are being operated currently, and which may undergo improvements to increase compliance with the MMISR vision of SOA-enabled integration. System failing to meet desired degree of interoperability at acceptable Total Cost of Ownership (TCO) levels will eventually be deprecated and replaced with architecturally sound alternatives.

**Data Services**

A set of data-centric functionalities implemented and operated by a technology partner, IBM Watson Health (formerly known as Truven Health Analytics). The sub-system includes pre-canned and ad hoc report management, data visualization, analysis, business intelligence and artificial intelligence components.

**External Systems**

This is an architectural grouping of all systems which exist outside of NM and may be hosted by other States, by partner organizations (commercial, educational, or other) as well as Federal systems.

## Security

HHS 2020 EA is governed by a combination of security control requirements found in MARS-E 2.0 and FIPS 140-2 standards intended to prevent unauthorized access to system data and functionalities. Compliance with 18 MARS-E 2.0 security control families is distilled to four separate focus areas of HHS 2020 Enterprise:

* Application security is implemented in software components and covers run-time execution of the following capabilities:
  + Authentication
  + Authorization
  + Encryption
  + Non-repudiation
  + Auditing
* Network and System Security covers run-time operation of server nodes and networks providing the following capabilities:
  + Network partitioning
  + Minimization of potential attack surface
  + Access monitoring
  + Auditing
  + DDoS attack detection and response
  + Software intrusion detection and response
* Physical Security covers the following aspects:
  + Physical Access controls
  + Physical intrusion detection and response
* Configuration Management
  + Application Security configuration changes
  + Network and System Security configuration changes
  + Physical Security configuration changes

This Reference Architecture document covers the salient features of HHS 2020 Application Security. Please see <TBD> for overview of Network/System and <TBD> for Physical Security.

### Introduction to Multi-Factor Security

HHS 2020 enterprise will expose functionality to 4 categories of users:

* State employees using sanctioned devices to perform business functions
* BPO partner employees using their respective employer sanctioned devices to access their solutions, which invoke HHS 2020 services
* External users, such as constituents and providers, accessing Unified Web Portal screens from devices of generally unknown origin
* Other systems accessing HHS 2020 services without involving any human user interactions

Different user categories require different authentication mechanisms, based on how different their usage contexts are vis-à-vis device and network trust, geographic location, time of day etc. Modern best practices for user authentication do not consider a user ID and password combination a sufficiently strong measure to prevent unauthorized access. Authentication should be based on a combination of multiple factors and the enterprise must decide on a case-by-case basis if evidence of user’s identity presented is sufficiently strong for a given usage request.

Authentication can occur based on different factors describing the user and falling into one of the following categories:

* What you know
* Who you are
* What you have
* Where you are

**What You Know**

User IDs, passwords, PINs, secret questions and other similar techniques fall into the category of “what you know”. Because secret information can be inadvertently disclosed by the user (written down to avoid memorization), reused across applications/sites etc., even a combination of multiple known secrets does not produce sufficiently high level of confidence in user’s identity, especially in situation of heightened liability resulting from unauthorized access and data loss (e.g. PHI, PII, IRS data).

**Who You Are**

Biometric security mechanisms, like face recognition and fingerprint scanning uniquely identifying each user (“who you are”) are becoming more commonplace especially with advances in mobile device technologies. However biometric techniques may have limited applications for users in lower socio-economic segments as better, more intuitive version of devices that implement biometric capabilities are priced outside of their reach. Although bottom-of-the-market devices do expose some biometric security capabilities, the level of hardware integration, ease of configuration and use as well as other considerations slow adoption among the less affluent users.

**What You Have**

Users in possession of trusted devices (“what you have”) are themselves more trusted for authentication purposes. Device trust can be established in a variety of ways, depending on device provenance and ownership. State-sanctioned computes accessing HSS 2020 systems via private or virtual private networks represent the most trusted device category. Once the access moves to public networks, the level of trust diminishes, and stronger authentication mechanisms would be invoked.

A similarly strong level of trust would exist for state-provisioned mobile devices, managed through enterprise Mobile Device Management (MDM) solutions. With bring your own device (BYOD) usage becoming more commonplace than employer-provisioned alternatives, access to HSS 2020 systems from applications housed in secure containers and possessing state-issued device credentials equates to the level of trust similar to MDM-managed device scenarios.

Trust at lower levels of confidence can be established even for user-owned devices without involving MDM or secure application containers. Assuming that users are accessing Web-based though internet browsers, the dominant design for such trust solutions is to store a device-level token within the browser (web cookie) and to require the user to “re-enroll” his/her browser with the enterprise whenever the token is missing or expired.

**Where You Are**

Users location can play a key role in establishing trust for authentication purposes. MDM suites include geo-location and geo-fencing capabilities that keep track of device location regardless of the user’s consent. Authentication (as well as authorization) policies can be tailored based on location.

Since BYOD devices, including those with secure application containers cannot be tracked without user’s consent, tailoring security based on device location require a different approach. A typical solution involves use of geo-encoded 2D barcodes. Geo-encoded authentication requires proximity of 2 devices to each other. For example, an employee in a provider’s office needs to use a BYOD device to manipulate patient’s data though a mobile application. In order to precede the user first interacts with a trusted browser instance on an office desktop computer and requests access to specific set of data within the mobile app on the device. The web page displays an image of a geo-encoded 2D barcode, and the user takes a picture of the barcode inside of the mobile application. The fact that the user was right next to the trusted device with a less trusted BYOD device elevates the trust for the latter to a higher level.

A note-worthy feature of geo-encoded barcode approach is that it is very resilient against both man-in-the-middle and man-in-the-browser attacks. While barcode presentation occurs in the browser, its captured image on the mobile device is handled via a different channel, making attempts to correlate the security challenge and the response virtually impossible.

### User Authentication

As a general standard, HHS 2020 Enterprise will favor multi-factor authentication over single factor, even if multiple forms of a single factor are involved. The “what you know” portion of authentication for all user types will consist of a user ID and password combination at a minimum. “What you have” and “where you are components” will vary depending on user and device type.

#### State Employee Authentication

**Preferred**

State employees accessing HHS 2020 Enterprise from state-sanctioned personal computers over private network (in office or VPN) will be authenticated via MS Active Directory server with same user ID and password credentials as those used to login to state Windows Domain. Employees who have successfully logged into their networked computers would not be additionally challenged for authentication while accessing HSS 2020 functionality. This approach constitutes a multi-factor authentication with “what you know” (user ID + password) with “what you have” (a trusted computer on a trusted network).

State employees accessing HSS 2020 enterprise from employer-provisioned, MDM-managed mobile devices (most likely tablet form factor devices with mobile application tools for field work) will be authenticated via MS Active Directory server with same user ID and password credentials as those used to login to state Windows Domain. Resulting application session on the mobile device will remain active per applicable requirements and no additional login steps within an active session will be performed. This approach constitutes a multi-factor authentication with “what you know” (user ID + password) with “what you have” (a managed, state-provisioned mobile device).

**Acceptable**

State employees accessing HSS 2020 enterprise from BYOD mobile devices (phone and tablet form factors with mobile application tools for field work) will be authenticated via MS Active Directory server with same user ID and password credentials as those used to login to state Windows Domain. The client applications accessing HHS 2020 data and functionalities must run in a secure application container. Resulting application session on the mobile device will remain active per applicable requirements and no additional login steps within an active session will be performed. This approach constitutes a multi-factor authentication with “what you know” (user ID + password) with “what you have” (a trusted mobile device running a secure application container).

**Discouraged**

No D-level options have been conceived of at present time.

**Unacceptable**

State employees’ access to HSS 2020 functionality is not permitted from unsanctioned devices regardless of the type of network used (public or virtual private are the only choices since unsanctioned devices are not permitted on facility networks by a broader enterprise security policy).

#### BPO/Partner User Authentication

**Preferred**

BPO/Partner users will primarily access their employer-provided solutions on employer provided/sanctioned host devices. Since conformance with applicable security standards is one of key qualification criteria for contracting with the State, BPO and Partner organizations will be trusted to enforce sufficiently high levels of multi-factor authentication within their systems (subject to Sate audits). Whenever 3rd party employees would require access to HSS 2020 functionality a single sign-on with their normal work user ID and password credentials would occur into the HHS 2020 authentication sub-systems using Oracle IdM COTS capabilities to establish the identity and application role membership for authorization purposes.

**Acceptable, Discouraged, Unacceptable**

No A, D or U-level options have been conceived of at present time.

#### External User Authentication

**Preferred**

Constituents, providers, partners and any other external users of HSS 2020 web UI layer will be subject to a minimum 2-factor authentication. The “what you know” component will consist of user ID and password combination, maintained in Oracle IdM solution. Storing an authentication token within the browser would establish the trust component with the host device fulfilling the “what you have” requirement.

The same user ID and password combinations as those established for web-based access will authenticate external users accessing state-sanctioned mobile applications. In addition to the “what you know” component, each application instance will undergo trust registration with the HSS 2020 security layer resulting in a “what you have” component of multifactor authentication.

**Acceptable**

As an added measure of security, authentication from mobile applications may be strengthened with suitable biometric mechanisms, depending on capabilities exposed by various supported device manufacturers.

**Discouraged**

No D-level options have been conceived of at present time.

**Unacceptable**

No single factor authentication for external users is permitted in order to access HSS 2020 UI layer functionality regardless of channel.

#### External System Authentication

<TBD discussion on how external systems interfacing into the HHS 2020 services layer will be authenticated. System accounts? Flowing end-user credentials into HHS 2020?>

### Authorization

A successful login by either a human or a system user into HHS 2020 applications will result in creation of a programmatically accessible security Principal complete with application roles assigned to the user. A combination or role membership, level of authentication and user’s identity (who the individual is as opposed to roles assigned to the individual) will be used to establish which functionalities are permitted to access[[1]](#footnote-2).

#### State Employee Authorization

**Preferred**

A programmatically accessible Principal object, complete with application role membership, will represent authenticated State employee identity. The Principal will be created by a combination of COTS technologies from MS Active Directory and Oracle IdM <TBD tech mix involved> and made available to all UI, service, API, database etc. end-point participating in handling of each user request.

The first level of access authorization will occur considering individual’s application role membership. This is referred to as “invariant” (or “declarative”) authorization to access functionality based entirely on role membership. The principle of PESSIMISTIC ACCESS DETERMINATION will be applied in situations where permission conflicts exist among assigned roles the most restrictive set of permissions will be assigned.

Assuming the user is allowed invariant access to an operation, the second level of “variant” (or “imperative”) checking beyond role membership will take place. <TBD discussion of how we will do this consistently across layers. Is there a blacklist for state employees precluded from accessing their own or immediate/extended family data? Blacklist seems like a suitable approach since chances of an employee related to covered individuals are fairly small and otherwise role-based access is sufficient.>.

An additional aspect of imperative identity-based authorization would include considerations of usage context, like device trust, geographic location etc. in order to potentially restrict access to information that would otherwise be available.

<TBD discussion of how access restrictions would be manifested. Do we exclude disallowed portions from messages? Do we disallow service invocation altogether if any of the data returned is restricted based on user’s identity? Architecturally, whose responsibilities are these? Can we us the BRE to handle the enforcements?>

**Acceptable**

It is possible that certain HHS 2020 systems would consider invariant permission grants to individual users regardless of role membership. It is also possible that individuals may be blacklisted form access to certain functionalities regardless of role memberships. In such cases considering individual level permissions instead of the role-based ones will validate the first level of access. Like in the case of the role-based access, we will use PESSIMISTIC ACCESS DETERMINATION approach defaulting to the most restrictive permission set. All other variant (imperative) steps outlined in the Preferred section still apply under this approach.

**Discouraged**

No D-level options have been conceived of at present time.

**Unacceptable**

No exclusively role-based (invariant or declarative) access authorization for State employee users will be permitted.

#### BPO/Partner User Authorization

**Preferred**

BPO employees contracted to the State perform their duties on behalf of the State, making them functionally similar to State employees. A programmatically accessible Principal object, complete with application role membership, will represent authenticated partner’s employee identity. The Principal will be created by Oracle IdM solution and will be suitable for invariant permission checks across all application layers.

Because of the difficulty and impracticality of tracking partner organization personnel changes we are not in a position to perform any additional invariant checking based on user’s inclusion into white/blacklists, or imperative identity-based access exclusions. Since conformance with applicable security standards is one of key qualification criteria for contracting with the State, BPO and Partner organizations will be trusted to enforce sufficiently stringent authorization practices precluding unauthorized identity-drivel access to HSS 2020 functionality exposed as part of their systems through their own means (subject to State audit).

**Acceptable, Discouraged, Unacceptable**

No A, D or U-level options have been conceived of at present time.

#### External User Authorization

**Preferred**

It is absolutely necessary to limit access to HHS 2020 data and functionality using identity-driven approach combining invariant (role based) and variant (identity and usage context based) techniques.

A programmatically accessible Principal object, complete with application role membership, will represent authenticated external user’s identity. The Principal will be created by Oracle IdM solution and will be suitable for invariant permission checks across all application layers.

Since it is likely that an external user will have access to fairly limited amounts of data related to self (member) or to a few entities (provider) the system will implement a white list concept, allowing the authorization components to programmatically enquire about permissibility of access by a particular user identity to a particular set of data. <TBD discuss on how this will happen both form the establishing of linkages (e.g. via Preferences Management functionality) to applying of restrictions and the resulting system behaviors.

**Acceptable**

It is foreseeable that certain business requirements may call for creation of end-user-managed white and blacklists. For example, a constituent may expand the default, state-provided list of permitted individual or organizational entities, or limit those with whom data sharing is permitted and to what extent. Such capabilities coupled with the resulting access grants or restriction may coexist alongside the Preferred approach.

**Discouraged**

No D-level options have been conceived of at present time.

**Unacceptable**

Foregoing identity-based, imperative authentication checks and only performing role-based invariant checks for external users is not permitted in HSS 2020 solutions.

#### System Authorization

<TBD discussion on how external systems interfacing into the HHS 2020 services layer will be authorized. System accounts? Flowing end-user credentials into HHS 2020?>

### Encryption

#### Public Key Infrastructure

#### In-Transit

#### At-rest

### Message Integrity

### Auditability

## Architectural Layering

HHS 2020 enterprise will rely on a proven pattern of separation of fundamental from derived functionality into architectural layers.

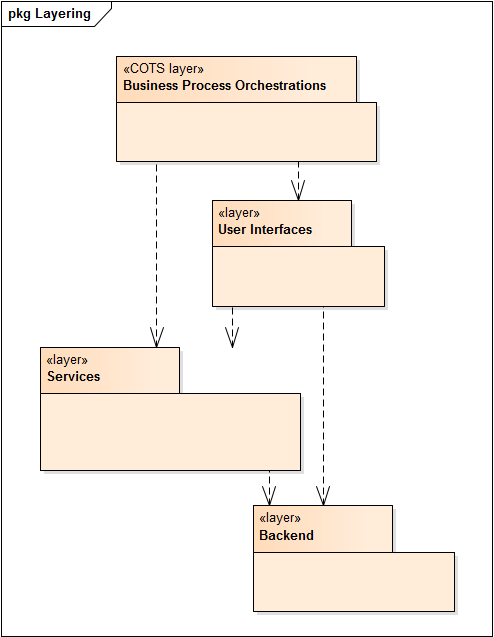


Figure 2: Architectural Layering of MMISR

At the top of the hierarchy one finds the layer of Business Process Orchestrations. It is the “home” of long-running business processes that orchestrate interactions among humans (web UI, email, mobile apps etc.) and systems (business and shared services invocations). The top layer is stereotyped as COTS to denote the implementation approach based on configuration of processes into a purchased Business Process Execution (BPEL) engine assisted by a Business Rules Engine (BRE).

Human-accessible system boundaries are grouped into the layer of User Interfaces. Further examination will reveal that the UI layer is comprised of Web, Mobile and, even (if necessary) Windows Desktop applications. The layer includes all 3rd-party UIs supplied by BPO partners as well as any legacy application UIs that would remain in use post HHS 2020 go-live.

Services layer is intended to house both service interface contracts and service implantations for the specialized and the shared categories of services.

The back-end layer is comprised of all new and legacy data sources, including BPO partner systems accessed by services to fulfill all data-centric system requirements.

Cross-layer dependencies can be described as follows:

* Business Process Orchestration layer depends on the UI and the Services layers since the contents of both are used during process orchestration.
* User Interface layer depends on the Services layer for all data acquisition, transformation and processing needs. Additionally, legacy and 3rd-party UI layer sub-systems in use post HSS 2020 go-live will continue direct connections to their respective back-ends, thus necessitating the overall dependency between the UI and the Back-end layers.
* Services layer depends on the Back-end layer for fulfillment of all data centric and certain processing-centric requirements.

Because of the consistent downward direction of cross-layer dependencies, HHS 2020 reference technical architecture can be fairly described as strictly layered. The absence of bi-directional dependencies serves as the highest-level indicator of the loose coupling, which is a much-desired architectural trait. Layering and loose coupling make HHS 2020 architecture stable in the face of future changes, enhancements and revisions. For example, future changes to the Business Process Orchestration layer functionality will have no impact on any other layers. Changes to the UI has potential to impact only the Orchestrations. Changes to Services would have to be evaluated against both the UI and the Orchestrations, while Back-end Layer would be unaffected. And finally, changes to the Back-end systems are likely to have widespread impact throughout the enterprise and must be considered thoroughly.

## Service Orientation

<Re-write in accordance with up-to-date models and descriptions of SOA ecosystem architecture.>

Figure 10 shows a component static structure depiction of HHS 2020 SOA approach allowing for loose coupling of service consumers form service providers and covering 3 service categories:

1. HHS 2020 Shared Services enabled by newly developed components and data repositories within the Integration Layer (e.g. Address Standardization, Entity Management, Logging and Auditing etc.)
2. HHS 2020 Specialized Business Services enabled by a combination BPO partner and Legacy systems (e.g. Claim, Prior Authorization, Reporting etc.)
3. BPO Partner UI Layer Presentation Services proxied for integration into HHS 2020 Unified Portal

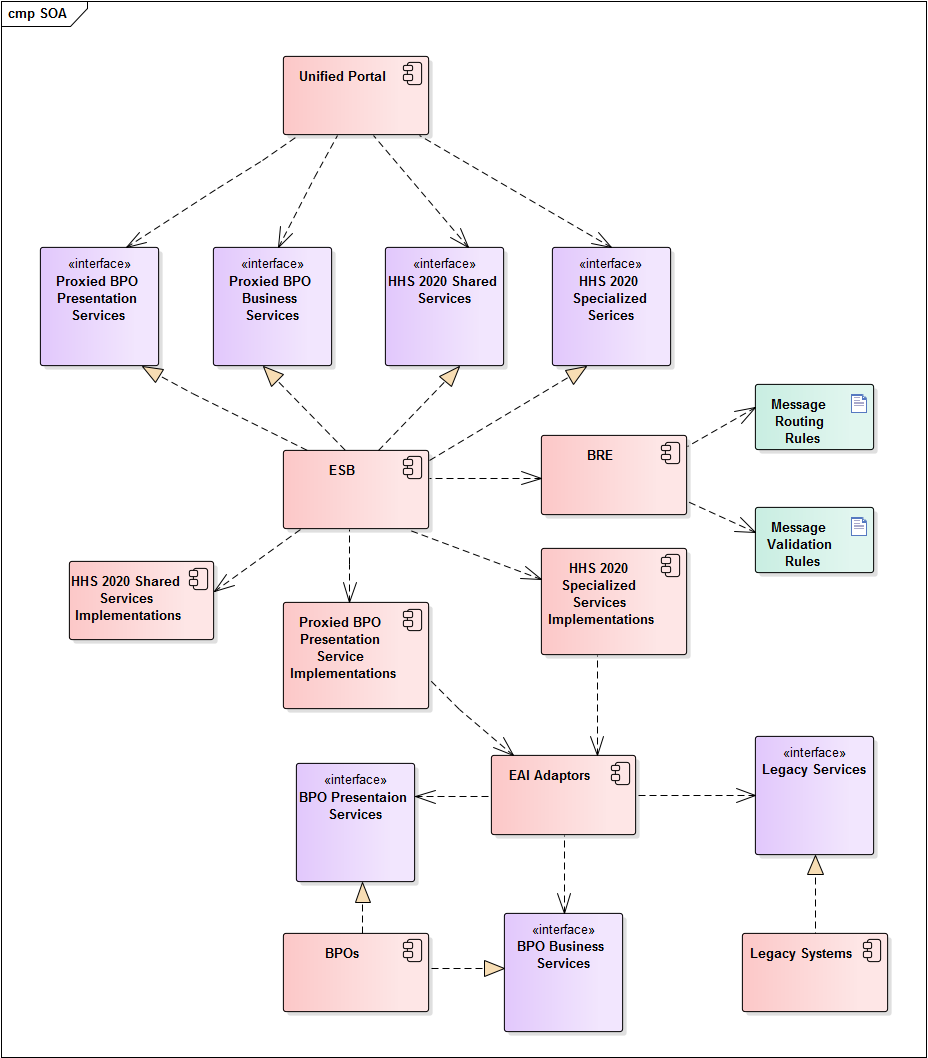


Figure 3: Details on the SOA-Based Design of the MMISR Integration Platform

Table 3 below lists SOA-related guidance and best practices adopted by HHS 2020 EA.

Table 5: SOA Industry References

|  |  |  |
| --- | --- | --- |
| **Name** | **URL** | **Description** |
| Thomas Erl – SOA Principles of Service Design |  | This is a book, not an online reference, but it still is one of the best references available on SOA |
| 10 Principles of SOA | <https://www.innoq.com/blog/st/2006/12/10-principles-of-soa/> |  |
| 10 Principles of SOA | <https://www.infoq.com/articles/tilkov-10-soa-principles> |  |
| The 9 Principles of SOA Design | <https://arch.simplicable.com/arch/new/the-9-principles-of-soa-design> |  |
| Characteristics of SOA Services | [https://www.ehealthblueprint.com/en/ documentation/chapter/characteristics-of-service-oriented-architecture-services](https://www.ehealthblueprint.com/en/%20documentation/chapter/characteristics-of-service-oriented-architecture-services) |  |
| Guiding Principles for SOA Governance | <https://www.cio.com/article/2445024/service-oriented-architecture/guiding-principles-for-soa-governance.html> |  |
| SOA Reference Architecture – Key Principles | <http://www.opengroup.org/soa/source-book/soa_refarch/p3.htm> |  |
| SOA Design Principles | <https://blogs.sap.com/2010/12/20/soa-design-principles/> |  |
| SOA Design Principles for Dummies | <https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=WSM14016USEN> |  |
| SOA Manifesto | <http://www.soa-manifesto.org/> |  |
| Web Services for Remote Portlets (WSRP) | <http://docs.oasis-open.org/wsrp/v2/wsrp-2.0-spec.html> |  |

### Role of the ESB

<Expand this sub-section with core ESB capability definitions like message routing, transformation etc.)>

ESB-mediated service invocation is at the core of the HHS 2020 EA’s vision for SOA. ESB is shown in the Implements relationship with respect to all service interfaces. The relationships are justified since all service consumers that are aware of service interfaces (e.g. the HHS 2020 UP supporting human stakeholders or outside systems requesting data from HHS 2020 Enterprise) will request the ESB to invoke the appropriate service end-points rather than having any direct end-point knowledge.

Consumers will communicate with services via messages routed to the appropriate end points by the ESB. Message will adhere to Shared/Canonical Schemas and validation for both schema and content as well as context-based routing will occur with assistance from a BRE enforcing appropriate sets of message validation and routing business rules. ESB will provide encoding (XML, JSON) and protocol (HTTP, JMS) translations to handle messages serving client of varying technological capabilities and needs. ESB will enforce role-based authorization for service access and will carry out necessary logging of service interactions for auditing purposes.

### Role of BRE

To externalize business logic controlling various long running (process orchestrations), short running (service compositions) and system-level (ESB message validation and routing) functionalities HHS 2020 enterprise makes heavy use of commercial BRE technology. The same BRE sub-system and its constituent components[[2]](#footnote-3) are intended to serve all the consumers listed above. Each BRE integration will come with its own set of rules and will allow for rule reuse across rule sets. The primary form of fact targets for rule enforcement will be messages exchanged among service consumers and providers. Additional fact types may include in-memory objects within complied solution source code, data in relational databases and contents of files.

### Transaction management

One of the cornerstone capabilities within the business process orchestrations is handling of distributed transactions. Numerous business requirements call for simultaneous changes to multiple underlying data sources resulting from application users’ manipulation data. For example, an update to member’s employment and income information has to be automatically correlated with changes to benefit eligibility. Updates to 2 or more systems or “subject areas” within a system, which must take place either successfully in every destination or nowhere at all, constitute a distributed transaction.

It is a common misconception to equate transaction with databases. Although all relational databases support the concept of transactions, so do transactional message queues and even file-based data stores.

In theory transactions can be handled either imperatively within the code implementing business functionality (e.g. within stored procedure SQL instructions, within service implementation in programing language instructions) or declaratively within the run-time container housing business logic implementation components (e.g. EAI engine hosting service orchestrations). HHS 2020 EA standards forbid any new uses of imperative transaction implementations and require adherence to declarative, container-based transaction management.

**Preferred**

The standard method for distributed transaction management is use of Open XA[[3]](#footnote-4)-compliant data sources in service-implementing EAI orchestrations. The transaction boundaries are to be defined explicitly in BPML flows so that the Distributed Transaction Coordinator available as part of the EAI engine takes care of all commits and rollback logic based on the collective outcome of all functionality invocations within transaction’s scope. Orchestrated XA transactions follow ACID[[4]](#footnote-5) properties and are 2PC compliant without any additional custom development efforts.

The following sample diagram[[5]](#footnote-6) illustrates placement of a transaction boundary into BPEL orchestration using Oracle Fusion EAI suite.

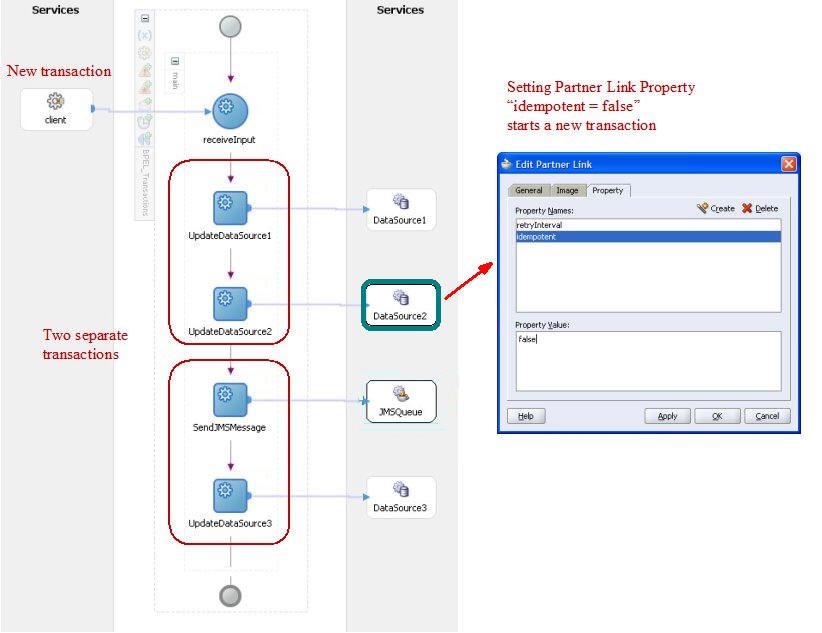


Figure 4: BPEL Transaction Boundary Illustration

**Acceptable**

Should there arise a need to integrate into orchestrations access to legacy and 3rd party data sources that explicitly manage their transactions regardless of any instructions from the process containers, the solution designer will configure compensation logic into the BPEL process to be invoked in case of exceptions. The following diagram[[6]](#footnote-7) provides a sample process with global error handler to invoke compensating transaction.

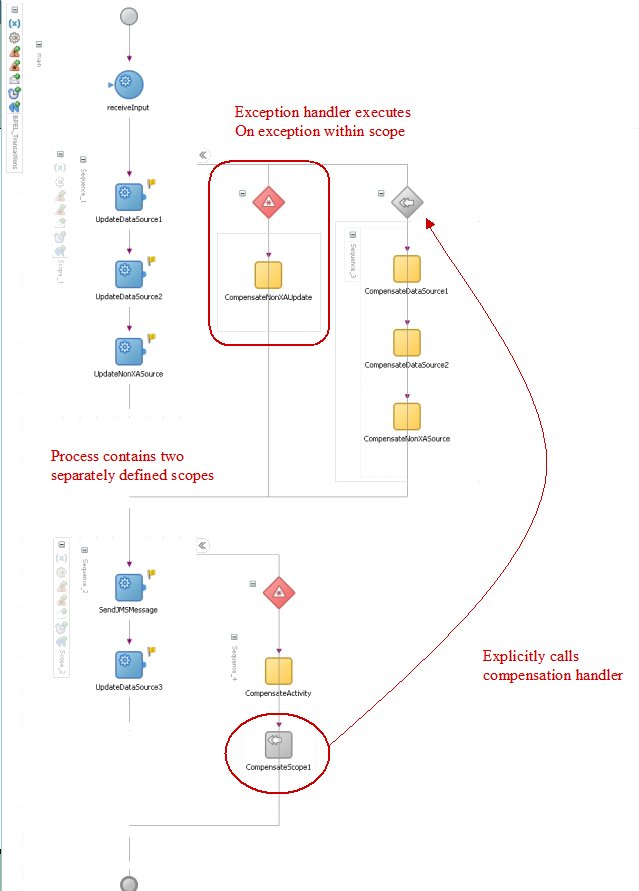


Figure 5: BPEL Compensating Transaction Illustration

Unlike immediate consistency across all data sources in a 2PC transaction carried out by a Distributed Transaction Coordinator, compensation logic will result in eventual consistency. The delay in data source consistency “synchronization” must be considered against the business requirements for the process being orchestrated.

**Discouraged**

No D-level options have been conceived of at present time.

**Unacceptable**

Any implicit transaction management carried out in the scope of any service implementation as part of designs supporting HSS 2020 requirements is unacceptable. This includes any explicit transaction management statements in SQL or application source code as well as any declarative attributes affecting transactional behavior placed onto implementation classes or methods in executable source code.

### Shared Services

Shared services implement shared behavioral (e.g. Reporting, Document Management) and non-behavioral (e.g. Logging, Address Standardization) functional requirements and are intended for broad re-use across business processes. Development of shared services is the responsibility of the SI vendor and all necessary components will reside outside of BPO and Legacy backend systems.

Some of the shared services will rely on the MDM repository of reference data about core business concepts (e.g. Entity Management service will establish relationships among entities and preclude access to information about one entity, e.g. Member by another Entity e.g. State Employee if both entities are part of the same family unit). Others (e.g. Address Standardization) will make calls to external business partners (e.g. USPS, Fair Isaac Corporation etc.) that perform data collection and validation tasks impossible/impractical to perform with HHS 2020 Enterprise. Finally, some impetrations (e.g. Document Management, Communication Management) will enable access to purchased general-purpose COTS systems in a consistent, centralized and mediated fashion.

#### Enterprise Identity Management

#### Enterprise Communications Management

#### Enterprise Document Management

#### Address Validation and Verification

#### EDI

### Specialized Business Services

Data and functionality contained in the HHS 2020 back-end (comprised of externally-hosted BPO partner and integrally-hosted legacy systems) will be exposed via Specialized Business Services. Since the back-end systems integrated via Business services are based on industry-standard technologies (services, web sites, relational databases, file stores etc.) HHS 2020 service implementations will rely primarily on short-running orchestrations of calls though COTS EAI adaptors (e.g. SOAP, REST, HTTP, ODBC etc.) and assembly of obtained results into canonical schema-based responses.

### Presentation Services

Although most HHS 2020 UPI presentation layer functionality will be custom-built as portlets, it is entirely foreseeable that UPI portal may want to consume data exposed in complete or partial UI pages within BPO partner systems rather than developing a custom version.

To enable such UI layer integration, HHS 2020 has mandated adherence to Web Services for Remote Portlets or WSRP 2.0 standard for its BPO partner systems. Under this standard, an external portlet acts as a presentation layer service returning both business data in a desired format (XML or JSON) and presentation markup (e.g. lists, pagination etc.) making it easy for the consuming portlet to render the response with minimal additional development effort.

Since HHS 2020 EA does not allow its components to access any service end points (including presentation services) directly, the BPO partner’s services will be proxied by HHS 2020 counterparts hosted on the ESB. Each proxying service will at a minimum make calls to the external portlet via appropriate COTS EAI adaptor and pass the unchanged results to the HHS 2020 clients. We may want to embed additional value-add functionality into the proxy implementations to make results display in the UI pages even easier (e.g. HHS 2020 styling additions).

### Data-as-a-Service

<Canonical entity access via services>

### Data Value-Added Services

<BI, reporting and other services from IBM>

## Multi-channel Architecture

Ultimately HHS 2020 Enterprise will serve user needs through a variety of access channels including:

* Adaptive Web Applications
* Mobile Applications
* SMS Applications
* Desktop Applications

However, Adaptive Web Applications exposed via UPI Portal will be the initial and the ongoing primary means of self-service interactions of HHS Enterprise and its various human stakeholders.

### External Portal Web UI Guidance

<Provide a listing of considerations applicable to external portal channel, given that it’s development will like to be outsourced. Treat guidance the same way as for all other vendor-delivered solutions.>

#### Web UI for External Use

All web user interfaces provided for the external users (constituents, providers, MCOs etc.) will be custom implemented in accordance with specific requirements for each applicable use case. HHS 2020 EA strives to achieve a consistent, productive and high-quality user experience for all our constituents. Because of this goal, we will always provide a custom-built UI component no matter how similar it may be to other off-the-shelf alternatives that may be available from our legacy systems or the new BPO partners.

**Preferred**

The non-workflow, workflow-initiating and workflow-participating UI components for external consumption will be custom built on the Liferay portal platform using the following approach:

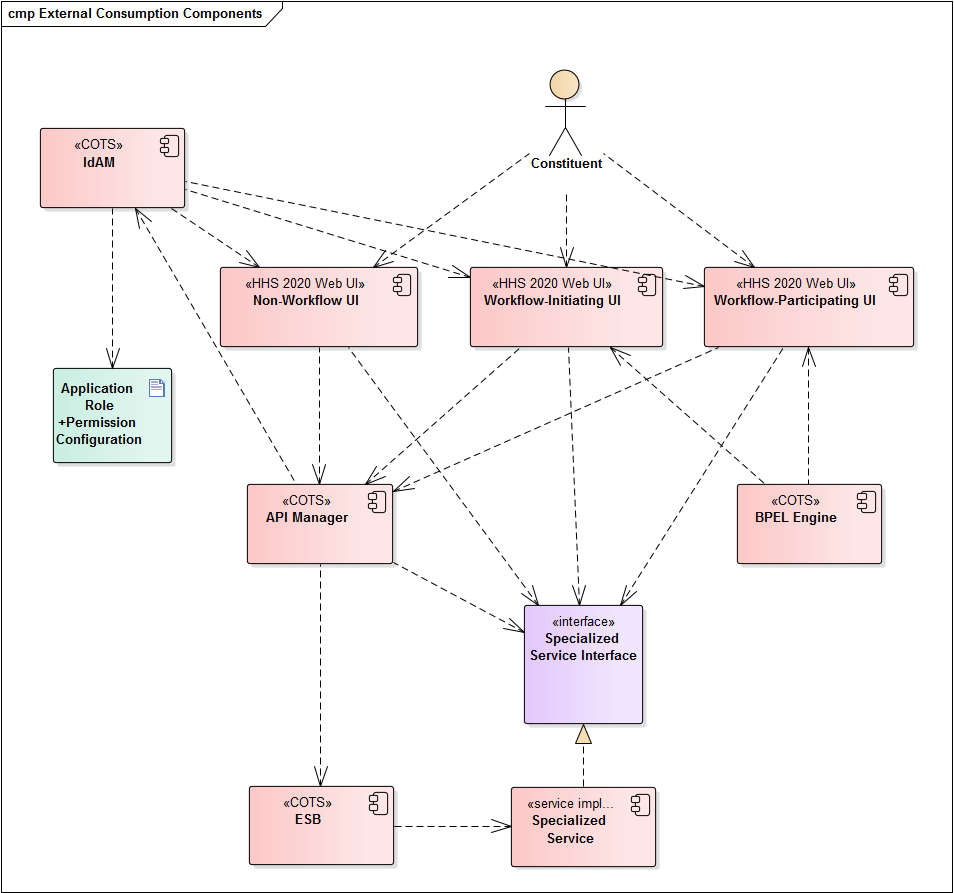


Figure 10: Web UI-related Components for External Use

All human-accessible UI layer components will be custom built on Liferay portal platform as a combination of pages and portlets as necessitated by the specific functional requirements of implemented use cases (we are purposely omitting the UI navigation components from this discussion to remain focused on specialized functionality implementations).

The components that initiate or participate in business processes will be configured as human workflow steps into the BPEL engine so that outcomes of their execution can be automatically captured by the engine to drive the corresponding workflow to the next steps.

Regardless of workflow enablement, all UI components will consume business services from the API Manager, that will, in turn, route service requests to the ESB. The latter will be the only architectural element with knowledge of service implementation end-point locations.

IdAM will prevent unauthorized access to all URLs within HHS 2020 enterprise including both the UI layer screens and the service invocation locations. Access requests failing to meet the role-based credentials will be denied and unauthorized access attempts will be logged for investigative purposes.

**Acceptable, Discouraged, Unacceptable**

No A, D or U-level options have been conceived of at present time.

### Internal Portal Web UI Architecture

Unified Web Portal UI will be the primary means of user access to HHS 2020 functionality. The web User interface will be based on a COTS portal framework (Liferay) rendering not only responsive, but also adaptive user experience. Responsive experience refers to the Web application’s ability to scale presentation area and UI mechanisms with available screen resolution on the browser host device. Adaptive applications go beyond resolution-based display scaling and include customizations of content and functionality depending on the usage context. The more the system “knows” about the current user and his/her needs, the more tailored an experience will be presented. The continuum of context is broad, starting with unauthenticated users browsing sites with content tailored based on the user’s geographic location (assuming access to location has been granted), to authenticated user during program enrolment with limited, goal-oriented functionalities, to authenticated, fully enrolled user interacting with most complete sets of UI features to receive best possible service.

The preferred mechanism for State Employees to interact with functions exposed by both the HHS 2020 services and 3rd-party applications is via State-provided, tailored UI components in the Unified Web Portal. These components would have consistent look and feel and will participate directly in UI process orchestrations, including immediate posting of workflow step outcomes to the process orchestration engine. Such custom-built UIs would rely on the underlying services (HHS 2020 and 3rd-party) for data acquisition and manipulation. Same UI components could be reused in multiple workflows.

While it is highly desirable for State employees to access all UI functionalities through custom-build HHS 2020 Web Portal screens, it may not be immediately feasible due to both implementation costs and limitations on part of legacy and 3rd-party systems which cannot expose services for direct consumption by HHS 2020 UI layer in an economical and practical way. HHS 2020 EA is prepared to support a heterogenous UI model, combining both the newly-build UIs and the legacy UIs as part of a single Web Portal experience. The architectural mechanisms used to accomplish this goal are:

* Business process execution engine running process configurations containing Web UI workflows…
* …implemented in the portal workflow engine, which orchestrates…
* …portal pages displaying output…
* …from legacy and 3rd-party UI screens and portlets acting as presentation layer services…
* …brokered through HHS 2020 ESB to avoid direct connections from the HHS 202 UI layer to service end points

Under the above setup, a State worker would always interact with UI portal pages, orchestrated into UI workflows, which, in turn are orchestrated into business processes, and never have to access legacy and 3rd-party systems directly.

HHS 2020 ecosystem will be comprised of reusable, loosely-coupled components that can be brought together through an orchestration to achieve both the long and the short-running business process capabilities.

#### Human Use Case Participant Components

Typical human user-facing functionality accessed via Web-based UI would be accomplished through the following component combination:

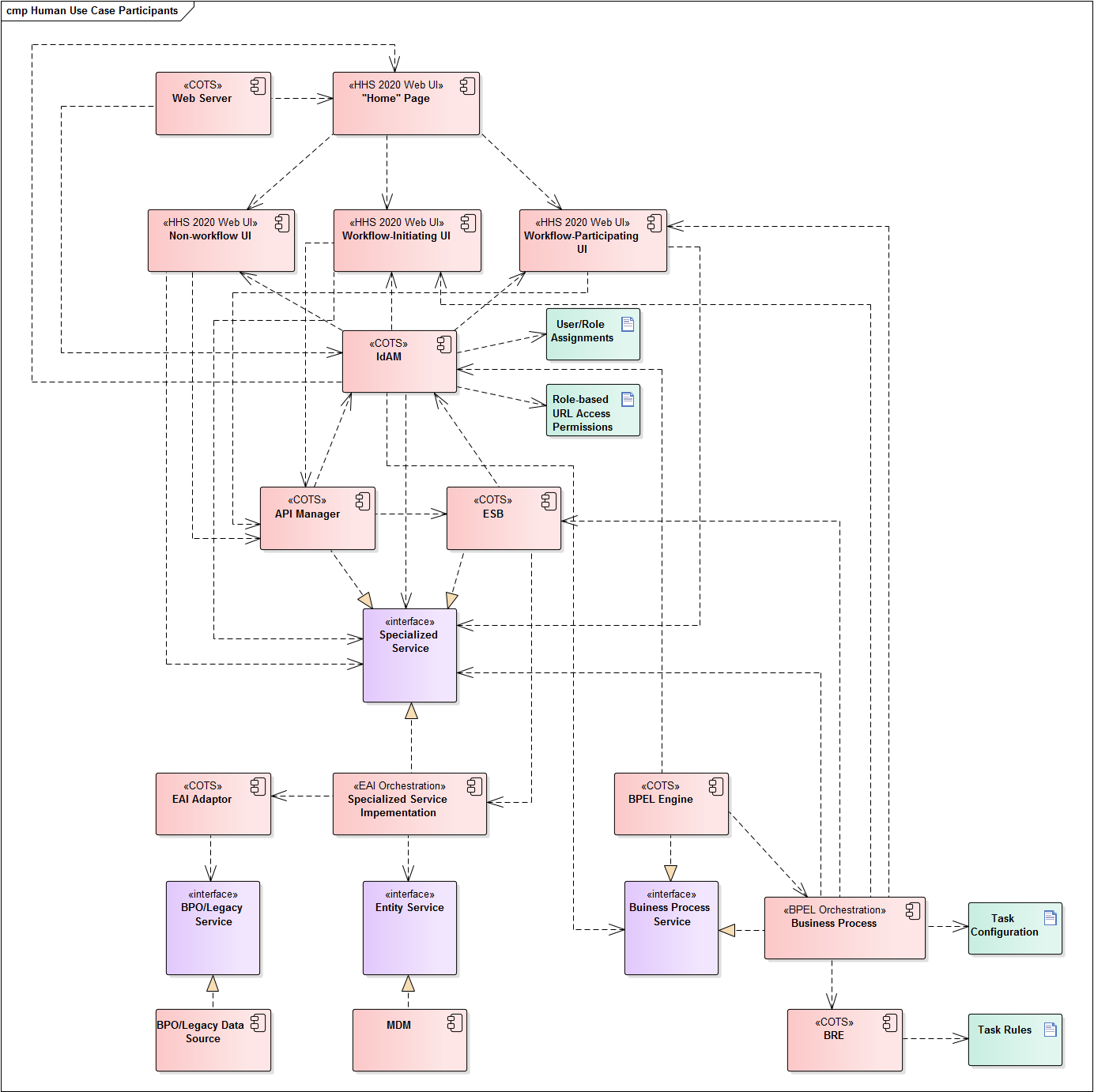


Figure 6: Human Use Case Participants

Before discussing how the above components collaborate with each other to fulfill use case requirements, we will first introduce each component in terms of its provenance (e.g. COTS, Custom built) and architectural functions.

Table 6: Human Workflow Participating Component Summary

|  |  |  |
| --- | --- | --- |
| **Component** | **Provenance** | **Functions** |
| Web Server | COTS | Web Server hosts Unified Portal and all web pages and portlets. Depends on IdAM to establish user’s ability to access page URLs. Directs user browser requests to appropriate page URLs, starting with the “Home” page. |
| “Home” Page | Custom built | This is the starting point for authenticated user’s interactions with all other portal pages. (Authentication mechanism, including authentication page is omitted from this discussion since we are focused on interactions of already authenticated users only who are represented by a programmatic identity inclusive of application role membership as assigned by IdAM). “Home page” includes mechanisms to access non-workflow enabled, workflow-initiating and workflow participating page URLs permitted to the user based on his/her application role membership. |
| Non-workflow UI | Custom built | Web pages and portlets to perform non-workflow enabled human tasks at user’s discretion (e.g. content searches, Claim status lookup etc.) |
| Workflow-Initiating UI | Custom built | Web pages and portlets that initiate workflows configured in the BPEL Engine by performing a human task. Upon task completion the UI informs the initiated business process instance about the outcome (e.g. success vs. failure). |
| Workflow-Participating UI | Custom built | Web pages and portlets that participate in workflows already running in the BPEL Engine by performing a human task. Upon task completion the UI informs the corresponding business process instance about the outcome (e.g. success vs. failure). |
| IdAM | COTS | Identity and Access Management system performs multiple security-related functions:   * Defines application roles * Maintains user assignment to application roles * Authenticates users and assigns application roles to the identity established during successful authentication * Monitors access to all URLs (UI pages, business services, business process services) and prevents URL access by users with insufficient application role membership * Exposes APIs to query for security-related settings (e.g. user’s ability to access a URL)   Being the hub of the security ecosystem, IdAM is dependent upon by numerous components to prevent unauthorized access to URLs:   * Web Server for web pages * Both API Manager and the ESB for Specialized Services * BPEL Engine for Business Process Services |
| API Manager | COTS | API Manager performs functions like the ESB but is intended for clients that reside in front of the firewall which protects the ESB and the rest of middleware and back-end systems from the outside access. API Manager forwards properly authorized requests for services to the ESB and returns responses back to callers. Since API Manager performs ESB-like function it is appropriate to model it as implementing interfaces of all exposed business services. UIs running in the UP layer depend on the API Manager for all service access. |
| ESB | COTS | ESB provides consistent, role-based access to all HHS 2020 services while providing service end-point location transparency and any necessary message validation, routing, protocol translation and usage/error logging functions. |
| Specialized Services | Custom built | For current discussion purposes the category of Specialized Services includes all Business and Presentation Services used to implement UI screen and Business Process Service tasks functionalities. All specialized services expose programmatic contracts consumed by their clients while all implementations are hosted in the ESB. Specialized services are constructed as short-running orchestrations making EAI Adaptor-based calls to the programmatic interfaces exposed by the BPO/Legacy Data Sources. |
| EAI Adaptor | COTS | EAI Adaptors are used by Specialized Service Implementations to make calls to programmatic interfaces exposed by BPO/Legacy Data Sources. |
| BPO/Legacy Data Sources | Multi-sourced | All back-end systems that serve data of either specialized or presentation service granularities are modelled for illustration purposes as implementing a BPO/Legacy Service interface. The nature of the interface (SOA, language-specific API, WSRP etc.) varies by system and is accounted for during HHS 2020 specialized service implementation through use of the appropriate EAI Adaptor as part of Specialized Service Implementation. |
| Entity Service | Custom built | Entity service is one of the functions implemented on top of the HHS 2020 MDM solution. Entity Management will establish relationships among entities and preclude access to information about one entity, e.g. Member by another Entity e.g. State Employee if both entities are part of the same household or family unit. Entity Service will be invoked by Specialized Business services as part of additional entitlement checks before returning data to users who would generally be authorized based on the set of held application roles but may be precluded in a specific instance because of the nature of Entities involved. |
| BPEL Engine | COTS | Business processes subject to automation via HHS 2020 ecosystem are defined and executed via BPEL Engine. Each configured process orchestration is treated by the system as a service with an end-point URL. All process service URLs are registered with IdAM enabling the UI layer functionality of proactively building lists of UI pages that either start services allowed for the current user or require user’s participation in running process instances. |
| Business Process | Custom built BPEL Orchestration | Individual business processes are implemented as BPEL orchestrations combining the following three types of tasks:   * Human tasks * Service tasks * Rule-based tasks   Human tasks are implemented as UIs, including web pages and portlets in the UPI layer. Service tasks make calls to HHS 2020 shared and specialized services via the ESB. Rule-based tasks invoke the BRE to declaratively carry out task-related rule set execution and then proceed according to rule execution outcomes. |
| BRE | COTS | BRE is used as part of BPEL orchestrations to invoke rule-based task rule sets and to return rule execution outcomes to business process orchestrations. |

Human actors would come to interact with UI components after navigating to portal pages at own volition or being prompted to do so via business process orchestration steps (e.g. receiving an email with a call to action link). UI pages can be placed into business workflows, so that the enterprise can keep track of UI interaction progress and prompt the users to perform necessary functions at optimal times. Portal workflows will be rule-driven, with the UI framework taking advantage of the COTS BRE component containing Portal Workflow Rules.

Portal workflows may combine screens built entirely using HHS 2020 capabilities (aka the “clean” implementations) with those that rely on legacy and 3rd-party screens as presentation layer services (aka the “hybrid” implementations). Regardless of the type of the UI component, screens will serve as web application building blocks that can be re-used to fulfill common requirements across multiple workflows.

UI layer components will achieve desired functionalities by accessing various services made available via API Manager COTS component. Services would fall into 2 categories, depending on their extent of functional specialization and reusability. There will be general-purpose services performing shared enterprise functions. An example of such a service is an Entity Service. HHS 2020 enterprise will treat its numerous human and organizational constituents as “entities”. Entities may be combined to reflect their relationships, for example a Provider Organization entity may have numerous Provider entities in an employment relationship. A constituent entity may be linked to other constituents as family members etc. Other envision shared services include:

* Address Validation
* Document Management
* Master and Reference Data Management
* Auditing and Logging

Specialized services will obtain, manipulate and persist into the various back-end repositories information pertinent to the <TBD number> business processes supported by HHS 2020 technologies. Examples of such services include Get Claim Status, Get Prior Authorizations, Get Eligibility, Enroll Member etc.

Regardless of service’s purpose, the default mechanism for service implementation will be short-running composition orchestrations of COTS Enterprise Application Integration (EAI) adaptors making calls to the underlying system end-points. Given the COTS preference of HHS 2020 enterprise, the need for custom service implementation will be greatly diminished and used only as a last resort when no COTS adapter and/or orchestration is available of feasible. Service compositions will rely on BRE technology to house and enforce any applicable composition rules, further extending the rule-driven approach of the HHS 2020 enterprise.

Service implementations will be hosted on the ESB so that service consumers would only communicate to services through their programmatic interfaces and without any regard for how services are implemented or for the location of implementing components. This approach for implementation independence and location transparency is a key enabler of loose coupling across HHS 2020 architecture. The ESB will rely on the BRE to dynamically enforce message validation, transformation and routing rules, ensuring that each client’s requests is handled by the best suited service provider every time.

To simplify the task of both developing externally-accessible functional components (e.g. UI screens) that depend on services and to make the runtime service consumption adhere to the published service contract rules, the architecture will rely on the COTS API Manager component as the single point of design-time service discovery and run-time invocation. API manager will prevent service access by unauthorized consumers, as well as ensure that only properly-constructed and authorized requests enter the ESB on the onward journey towards service implementation end-points.

Information exchanged via all services will follow the HHS 2020 canonical message model, comprised of well-defined data contracts that represent business entities, attributes and relationships. Full-fledged canonical entities will have numerous data attributes and it is very unlikely that any consumer would ever require a complete entity instance to perform a business task at hand. To minimize the amount of data traversing the system and to improve application performance all services would follow a pattern of injecting response schemas into requests. Under this pattern, the service consumer will specify as part of service invocation parameters which elements of the returned entities and their attributes are of interest during the current interaction. The service implementation would shape the data obtained from the back-end components in accordance with caller’s wishes and return entity instances containing only the requested attributes.

#### Participating Components Interaction Sequences

The following sub-sections describe interactions among the components listed above necessary to achieve five demonstrated capabilities:

* Build-up of “Home” page with links to permitted pages
* Invocation of Non-workflow functionalities
* Invocation of Workflow-initiating functionalities
* Invocation of Workflow-participating functionalities
* Performance of Entity-level authorization checks during functionality invocation

PLEASE NOTE: The diagrams depict conceptual component-to-component interactions. Actual sequences of method invocations across technology-specific implementations (e.g. Liferay Portal, Oracle IdM etc.) have much greater detail reflecting each participating object’s programmatic interfaces.

##### “Home” Page Creation

The following sequence assumes that the user is authenticated and has role membership allowing for access to “Home” page.

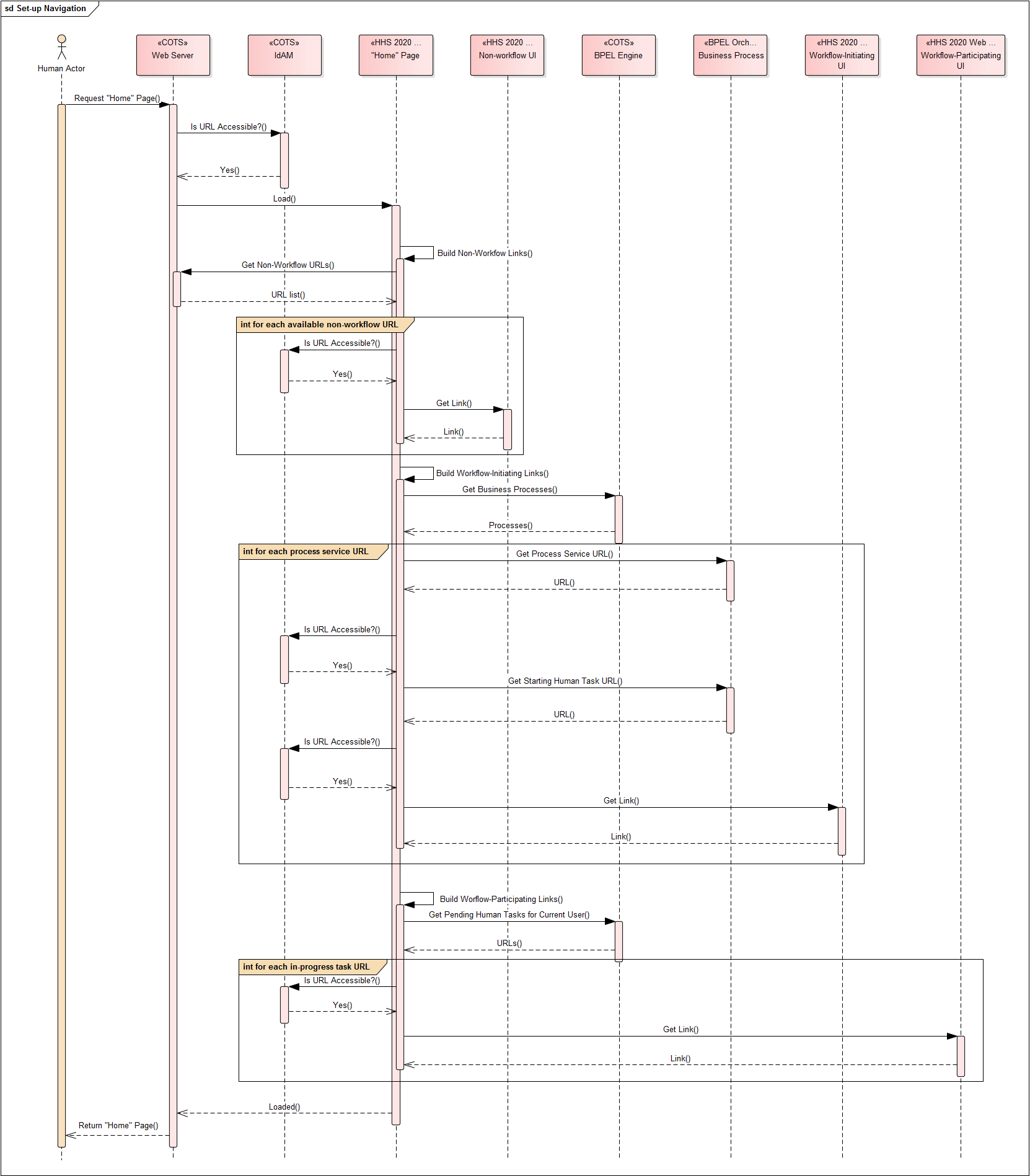


Figure 7: Illustrative “Home” Page creation Component interaction sequence

1. The Actor requests the “Home” page from the Web Server.
2. Web server queries the IdAM and determines that requested URL is accessible to the Actor.
3. Web Server loads the “Home” page.

While loading the “Home” page first builds non-workflow page links.

1. “Home” page requests from Web Server the list of non-workflow page URLs.

Steps 6-7 are repeated for each returned non-workflow URL

1. “Home” page queries IdAM and determines that non-workflow URL is accessible to the Actor.
2. “Home” page creates a navigation link for the non-workflow URL.

“Home” page proceeds to render workflow-initiating page links.

1. “Home” page requests the list of business processes from the BPEL Engine.

Steps 8-10 are repeated for each returned business process

1. “Home” page queries the Process for the URL of the initiating human task.
2. “Home” page queries IdAM and determines that initiating task URL is accessible to the Actor.
3. “Home” page creates a navigation link for the task-initiating URL.

“Home” page proceeds to render workflow-participating page links.

1. “Home” page requests the list of pending human task URLs from the BPEL Engine.

Steps 12-13 are repeated for each returned pending task URL

1. “Home” page queries IdAM and determines that initiating task URL is accessible to the Actor.
2. “Home” page creates a navigation link for the task-participating URL.

The completed “Home” page is returned from the Web Server to the Actor and ready for interaction.

##### Non-workflow Page Invocation

The following assumptions are made in the sequence below:

* User is authenticated and has role membership allowing for access to “Home” and non-workflow page.
* “Home” page has been initialized and includes a link to non-workflow page.

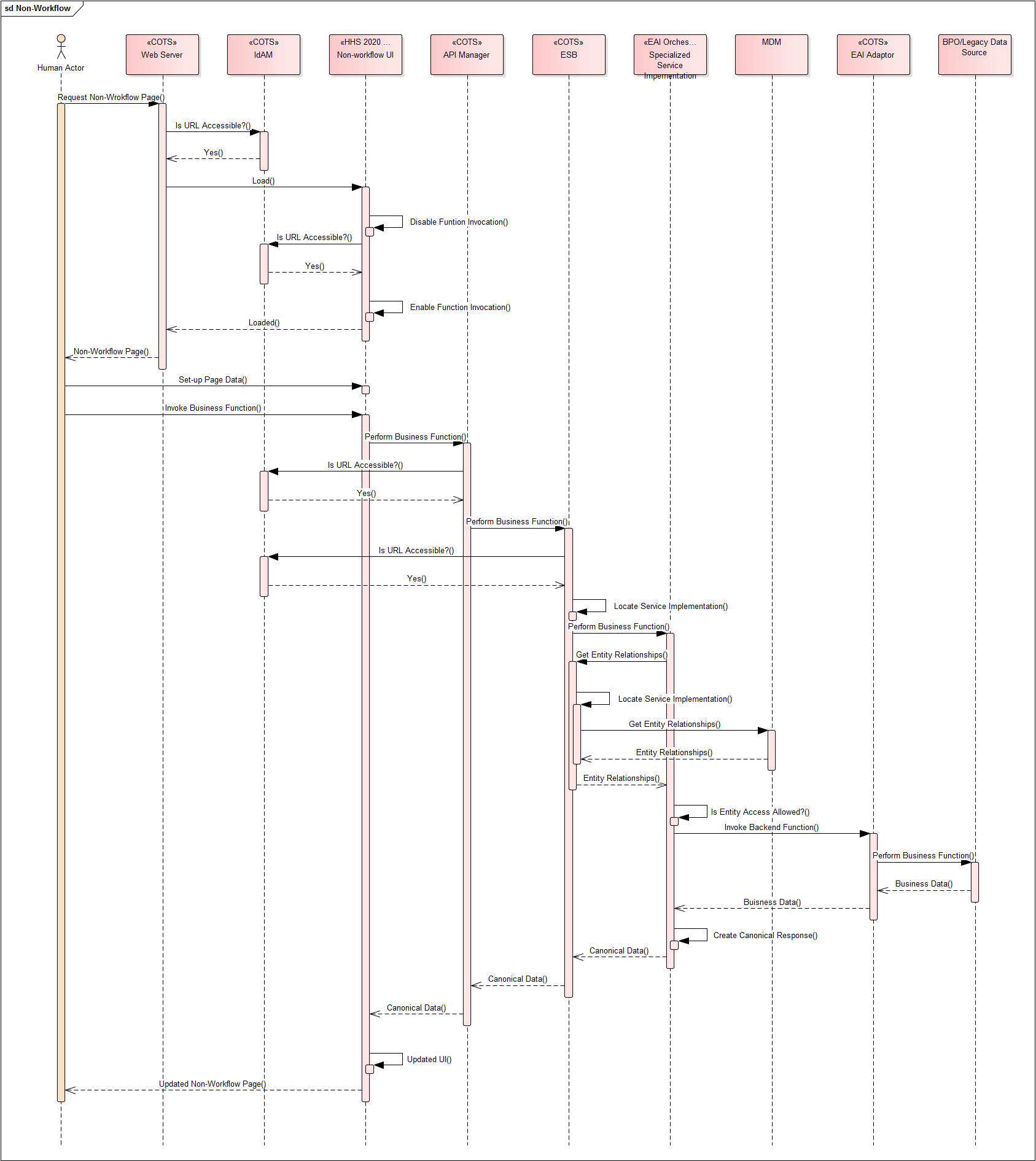


Figure 8: Illustrative Non-Workflow Page Invocation Component interaction sequence

1. The Actor requests a non-workflow page from the Web Server.
2. Web server queries the IdAM and determines that requested URL is accessible to the Actor.
3. Web Server loads the non-workflow page.
4. The non-workflow page contains a mechanism to perform a business function (e.g. a button to retrieve a claim based on entered criteria). The implementation of the page assumes that the current actor is not permitted to perform the business function and disables the invocation mechanism (e.g. a button is disabled and grayed out) by default.
5. Page implementation queries the IdAM for actor’s ability to access the service URL of the business service that would be invoked from the page if the invocation mechanism were enabled (e.g. Get Claim Service) and determines that access to the service is permitted.
6. Page implementation enables the invocation mechanism since the user is entitled to access the service based on the combination of application roles and permissions assigned in IdAM.
7. The initially-rendered page is returned to the Actor.
8. The Actor interacts with the page UI and sets-up necessary data to invoke the business function (e.g. specifies which Claim is to be shown by providing Claim ID).
9. The Actor invokes the business function on the page (e.g. clicks the button).
10. Page implementation makes the call to the API manager to invoke the appropriate service to perform the business function.
11. API Manager determines that current user is entitled to invoke the service and forwards the request to the ESB.
12. ESB queries the IdAM for actor’s ability to access the service URL of the business service and determines that access to the service is permitted.
13. ESB locates service implementation end-point and makes the call to the specialized service implementation component (e.g. Get Claim Service).
14. Specialized service implementation determines that the service can only be invoked if there is no conflict of interest between the Entity representing the Actor (e.g. State Employee) and the Entity to whom the requested Claim applies (e.g. Medicaid Member). Should both Entities be part of the same family unit or reside in the same address, the conflict of interest should prevent the Actor from seeing data for the requested claim.
15. Specialized Service Implementation calls the ESB to invoke the Entity Management Service (implemented by the MDM solution) to get the relationships among entities involved in the requested business function.
16. ESB determines the end-point location for the Entity Service and forwards the request.
17. MDM solution generates a representation of relationships among requested entities and returns the data to the ESB, which returns relationship data back to the specialized service implementation.
18. Specialized service implementation determines that the Actor is permitted to access the requested Entity data as no conflict of interest exists.
19. Specialized service implementation uses an appropriate EDI Adaptor (e.g. ODBC, JMS, CICS, SOA etc.) to make the call to the Backend/Legacy system that hold requested data and/or performs requested business function.
20. The backend system performs the requested function and returns data to the EAI Adaptor, which, in turn, return data back to the specialized service implementation.
21. Backend data is transformed into canonical message model and returned all the way back to the non-workflow UI page.
22. The page renders necessary updates to the UI and returns to the Actor for subsequent interactions.

##### Workflow (Initiating/Participating) Page Invocation

The following assumptions are made in the sequence below:

* User is authenticated and has role membership allowing for access to “Home” and the workflow page.
* “Home” page has been initialized and includes a link to workflow page.

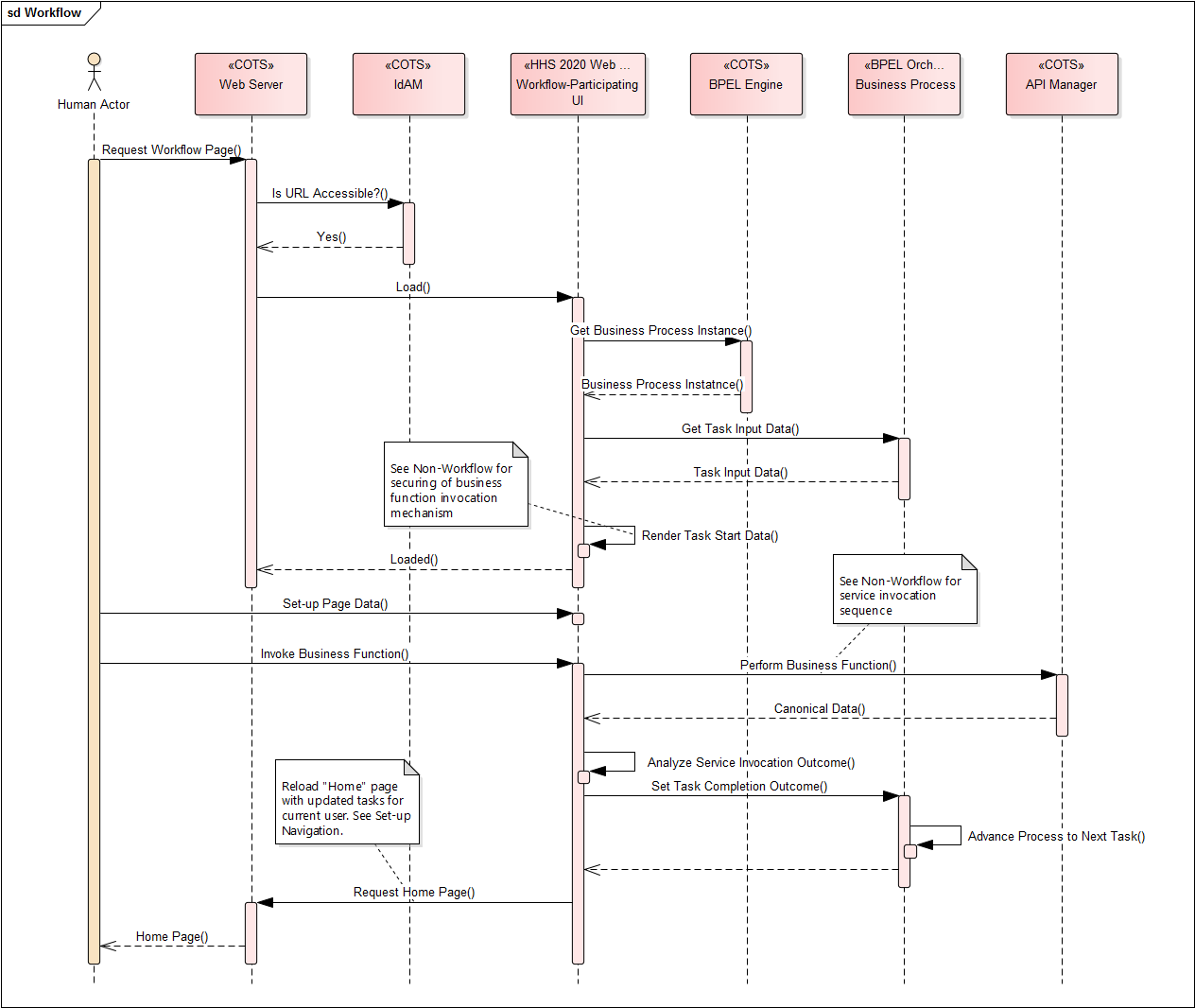


Figure 9: Illustrative Workflow (Initiating/Participating) Page Invocation Component interaction sequence

1. The Actor requests a workflow page from the Web Server.
2. Web server queries the IdAM and determines that requested URL is accessible to the Actor.
3. Web Server loads the workflow page.
4. Workflow page implementation contacts BPEL Engine for the corresponding Business Process instance and queries the Process instance for the data to be used as an input to the current human task.
5. Workflow page renders start-up data for the task and secures the business function invocation mechanism based on user’s application role membership (see Non-Workflow Page Invocation for description).
6. A properly initialized workflow page is returned to the Actor.
7. The Actor manipulates the data in the page and requests invocation of a business function that would cause data to process (see Non-Workflow Page Invocation for details on business function invocation via a specialized service).
8. Once canonical data is returned from the business service a decision is made in the implementation of the page about the task completion status (e.g. success vs. failure) and the status is communicated to the Business Process instance causing the process to advance to the next step.
9. Since the current page’s work is finished the system navigates the user back to the “Home” page which is rebuilt with updated workflow page links reflective of the next steps in various workflows applicable to the current user (see Set Up Navigation for workflow page link generation details).

#### Web UI for Internal Use

Although it is desirable to extend a consistent, custom built UI experience offered to the external stakeholders to the State employee users, the approach may not be optimal in terms of implementation complexity, cost, effort and schedule. To minimize the Total Cost of Ownership in the HHS 2020 enterprise, EA is considering accommodating of 3rd-party user interface functionalities integrated into the UI layer components though use of external portlets as presentation layer services combining business data with elements of presentation markup.

**Preferred**

The preferred mechanism for the non-workflow, workflow-initiating and workflow-participating UI components for internal consumption will be custom built on the Liferay portal platform using the approach like the one outlined in Web UI for External Use.

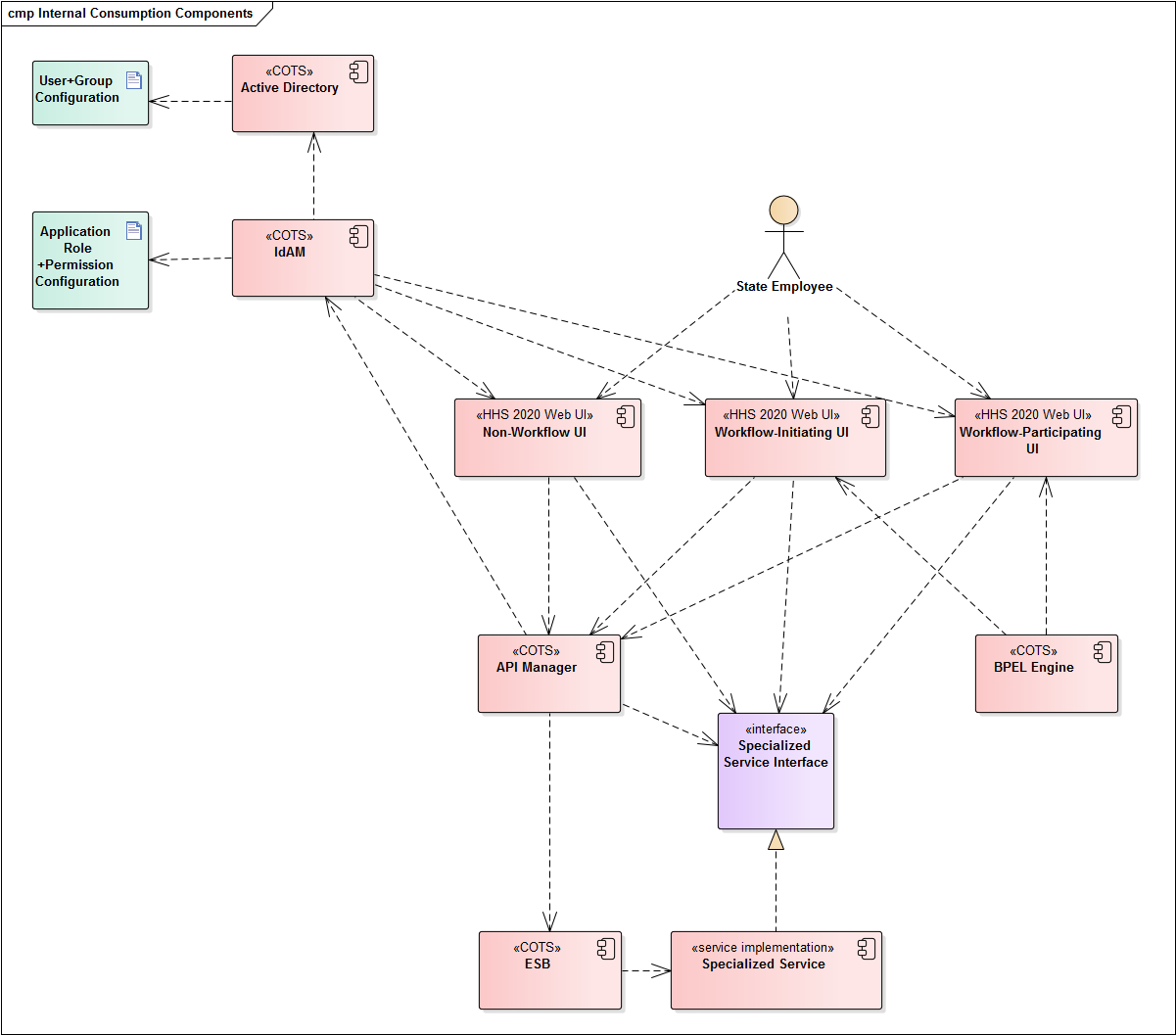


Figure 11: Web UI-related Components for Internal Use

The only slight difference is that user’s ID and password credentials will be authenticated in the HHS 2020 Active Directory before application role membership is established in the IdAM solution.

The components that initiate or participate in business processes will be configured as human workflow steps into the BPEL engine so that outcomes of their execution can be automatically captured by the engine to drive the corresponding workflow to the next steps.

Regardless of workflow enablement, all UI components will consume business services from the API Manager, that will, in turn, route service requests to the ESB. The latter will be the only architectural element with knowledge of service implementation end-point locations.

**Acceptable**

When it may not be feasible to provide a custom-built UI to exercise functionalities found in the legacy, 3rd-party and BPO partner systems, HHS 2020 EA allows for the option of consuming of such external functionalities as presentation layer services implemented using Web Services for Remote Portlets (WSRP) 2.0 specification. See Diagram <TBD> in the section describing Service Orientation.

This approach essentially amounts to the State Employee user having to access functionality exposed by a 3rd party web UI. For the user to be granted permission to access the 3rd party two preliminary security configuration steps are required.

* Application roles defined in the 3rd-party system must be replicated into HHS 2020 IdAM repository. The process for keeping role definitions consistent across the 2 repositories is of upmost importance and it is discussed in section <TBD>.
* Single Sign-On (SSO) channel allowing the user to automatically login into the 3rd-party system with State-issued user ID and password credentials must be established.

Access to the 3rd-party UI screens will occur using SSO credentials on behalf of an authenticated (via HHS 2020 Active Directory and IdAM SSO) user who has been given a set of application roles, including 3rd-party roles (via HHS 2020 IdAM).

HHS 2020 UI will rely on WSRP-compliant external portlets to send data and presentation markup from the 3rd-party screens and present a version formatted according to the stylistic rules of HHS 2020 UPI.

**Discouraged**

No D-level options have been conceived of at present time.

**Unacceptable**

No other options for presentation of Web UI functionality to State Employee users are permitted.

Appendix a: Solution Modeling Guidelines

HSS 2020 reference technical architecture will be visualized using industry-standard tools and notations. The software and the deployment aspects will be modelled in Unified Modeling Language (UML) using and Sparx Systems Enterprise Architect (Sparx EA) v.14.1. The hardware and network designs will be modelled in MS Visio Professional 2016 with Infrastructure stencil.

Software solution modeling will consist of the following elements, views and diagrams:

* Layering and sub-system UML static structure diagrams showing packages (stereotyped to convey architectural meaning as necessary, e.g. “layer”, “sub-system”, “COTS” etc. as well as package-to-package dependencies. The dependency relationships among packages are intended to both demonstrate the extent of coupling (unidirectionality of dependencies being indicative of loose coupling) as well as to convey strictness of layering, which in turn translates into greater architectural stability in the face of future changes.
* Component static structure diagrams showing conceptual “building blocks” of HHS 2020 ecosystem. Whenever necessary to convey architectural meaning, components will be stereotyped (e.g. “COTS”, “RDBMS” etc.) Interfaces exposed by components will be modelled using the classifier notation since Sparx EA modeling tool does not have ability to show a “connectable” version of implemented interface on lollipop shape (preferred for space conservation purposes in diagrams). Component interface implementations will be shown via UML implements relationship. Components will be shown in dependency relationships to consumed interfaces and among themselves. Whenever necessary to illustrate the configuration-driven or rule-based nature of the solution UML artifact elements containing such configurations and rules will be introduced. Components will be shown in dependency relationships with artifacts.
* Component placement into computing nodes will be illustrated through deployment diagrams. Such diagrams will not show design specifications for the infrastructure nodes. All system design information will be found in Infrastructure Design diagrams created using MS Visio Professional 2016 with Infrastructure stencil.
* Component-level requirement realizations will be shown as a combination of UML sequence and collaboration diagrams. Both types of diagrams will be used to illustrate conceptual level of detail in component-to-component communications in a course of implementing a given set of system requirements. Sequence diagrams will convey the end-to-end communication paths across participating components, while collaboration diagrams will present more of a view focused on a given sub-set of components of interest with invocations among all participating components having lesser significance.

The above-mentioned models and diagrams will minimize technology-specific implementation details tied to COTS platforms. All model element names will be illustrative of their architectural purpose, not design-level detail. As such, plain English naming with spaces between words and capitalization as required will be employed. Individual solution designers tasked with actual requirement implementations will be guided by the reference technical architecture models to produce fine-grained, detailed and technology-specific documentation as part of the solution delivery workstreams.

Appendix B: PADU Prescriptive Framework

To assist designers in putting together optimal solution specs, which will eventually become functional components, system and application implementations, HSD ARB and the Enterprise Architecture team are striving to make application of Technical Governance as prescriptive as possible. We are adopting a PADU framework for assessment of each proposed design’s adherence to the stipulated standards and best practices as well as to assist solution designers with their tasks.

PADU abbreviation stands for **P**referred-**A**cceptable-**D**iscouraged-**U**nacceptable, a scale of solution’s compliance with adopted EA standards, principles and best practices.

**Preferred**

Preferred approaches fully adhere to the specified standards. An example of such an approach is use of COTS, EAI vendor-provided adaptors and short-running orchestrations to implement a business service. Solutions designed at P-levels of EA standards adherence are seen as permanent until enhancements or replacements are necessitated by factors extraneous to the shortcomings of the solution design itself, such as future requirement changes, technology obsolescence and the like.

**Acceptable**

An acceptable approach is sub-optimal to the preferred when compared on total cost of ownership of the resulting technical solution. Like preferred, acceptable solutions are permanent in the absence of external change drivers. Acceptable approaches should adhere to the optimal set of standards, but generally result in more effort to implement that ideally desired.

An example of A-level variation of EAI-based service implementation would involve creation of a custom adaptor, otherwise unavailable on COTS basis to connect service-implementing orchestration to a non-standard data source. The rational for deeming such an approach as acceptable is fairly straight forward. The extra custom development efforts spent on the missing adaptor functionality should constitute an architectural component implementing shared as opposed to specific functional requirements. Such an architectural component will be reused as part of numerous specialized functional implementations, each following the preferred EAI adaptor/orchestration approach.

**Discouraged**

Certain shortcuts in solution designs may be necessitated by a variety of factors like availability of skilled resources, prohibitive immediate implementation costs, schedule constraints etc. It is therefore necessary to provide reasonable Governance accommodations for scenarios when sub-optimal decisions must be allowed. The key point to understand discouraged practices is that they will only be allowed if a corrective plan of action to alleviate the problem to at least A-level are presented. The business case outlining TCO for discouraged solutions must have the cost analysis to eventually bring the proposed designed to a higher level of compliance.

To continue the line of EAI-based service implementation examples, a D-level approach would involve custom developing a service implementation outside of EAI adaptors and orchestrations. The resulting service implementation would still be hosted by the ESB and accessed in a consistent, ARB-approved fashion. Design for such a service would only be approved if accompanied by a corrective action plan with a clear analysis of future operating consequences and eventual remediation costs.

**Unacceptable**

As the name implies, certain approaches will not be permitted under any circumstances. Reasons for U-level of compliance may include:

* Constraint[[7]](#footnote-8) (technological, legal, regulatory etc.) violations
* Fundamental deviations from approved standards and best practices with no justifiable future correctional remedies

An example of a U-level design decision is creation a service implementation completely outside of the EAI/ESB framework.

Appendix C: Glossary

<Insert instruction to use the Program-Wide Glossary.>

1. Identity-based as opposed to role-based authorization is the best practice postulated by ONC-SIM Health IT Resource Center in STATE HEALTH IT MODULAR FUNCTIONS FOR VALUE-BASED PAYMENT STRATEGIC IMPLEMENTATION GUIDE V. 1.1, published December 6, 2017 [↑](#footnote-ref-2)
2. The component diagram above shows multiple instances of COTS BRE, which is not the intended modeling outcome. Component instance modeling elements had to be used due to a limitation of Sparx EA preventing placement of the same model element (in this case BRE component) more than once into the same diagram. Having the actual component on the diagram only once (instead of the multiple instances), while technically more correct, would have resulted in all dependency lines form consumers of BRE functionality pointing to the same component, thus increasing clutter and reducing diagram’s readability. [↑](#footnote-ref-3)
3. X/Open XA – Open Group standard for Two-Phase Commit (2PC) protocol compliant data sources (http://pubs.opengroup.org/onlinepubs/009680699/toc.pdf) [↑](#footnote-ref-4)
4. ACID – Atomic, Consistent, Independent, Durable properties of a 2PC-compliant transaction. [↑](#footnote-ref-5)
5. Diagram sourced from http://www.oracle.com/technetwork/topics/soa/bpel-explicit-115054.jpg [↑](#footnote-ref-6)
6. Diagram sourced from http://www.oracle.com/technetwork/topics/soa/bpelwithcompensation-115079.jpg [↑](#footnote-ref-7)
7. A Constraint is a type of requirement that dictates the only acceptable implementation approach and is not subject to architects or solution designer’s discretion. Constraints may arise from a variety of sources including, but not limited to Technological (approved COTS platform usage), Legal (limitation of liability) and Regulatory (mandated ways of serving constituents). [↑](#footnote-ref-8)