Instant OpenHIE

Exchanging health information is difficult. Security and privacy are paramount. There are many potential components and use cases. Many implementers need or start with HIEs for one project or a narrow national use case, then face the challenge of scaling up and out into new areas.

The Instant OpenHIE project aims to reduce the costs and skills required for software developers to deploy an OpenHIE architecture for quicker initial solution testing and as a starting point for faster production implementation and customization. Instant OpenHIE will be a simple way for technical persons to install and see a complex system working against a real-world use case. It will allow technical persons to illustrate how interoperability can work to solve health challenges and show how a national interoperability architecture could be created with open-source software and standards.

Partnerships especially with regard to security, privacy, and standards-testing are critical, including future coordination with IHE and other entities to ensure alignment and to follow best practices of the leading institutions and prevent duplication.

Vision

At maturity, Instant OpenHIE activities will provide portable, launchable versions of multiple OpenHIE components to facilitate:

1. Demonstrable reference products -- those that align with the OpenHIE Community's vision for low resource contexts
2. Rapid software development of mediators and point-of-service systems by making it possible to launch several applications easily so the developer can focus on their task
3. Reproducible, version-controlled infrastructure for user-contributed tests of the OpenHIE Architecture profiles, workflows, and use cases.
4. Production-ready containers and orchestratable components that are deployable in any context.

The first phase of activities addresses items 1-2 while production-ready deployments and infrastructure for testing will be built incrementally upon the innovations and lessons learnt from the efforts of the earlier phase.

Deploying and managing private health information on patients and providers is among the most sensitive of any data. It is critical to ensure security and privacy, backups and data recovery, authentication, authorization and other enterprise standards. At maturity, Instant OpenHIE would provide production-ready containers and some orchestration assets, such as Kubernetes manifests, but these would be borrowed from Instant OpenHIE and still managed by implementers, who are responsible for databases, upgrades, privacy, security, backups and recovery, authentication, authorization, and other production-ready concerns. *Instant OpenHIE would be a way for implementers to develop their tooling around the OpenHIE Architecture and the versions of it, rather than as a substitute for enterprise HIS implementation, support and management.*

Roadmap

At maturity, Instant OpenHIE activities will provide portable, launchable versions of multiple OpenHIE components to facilitate:

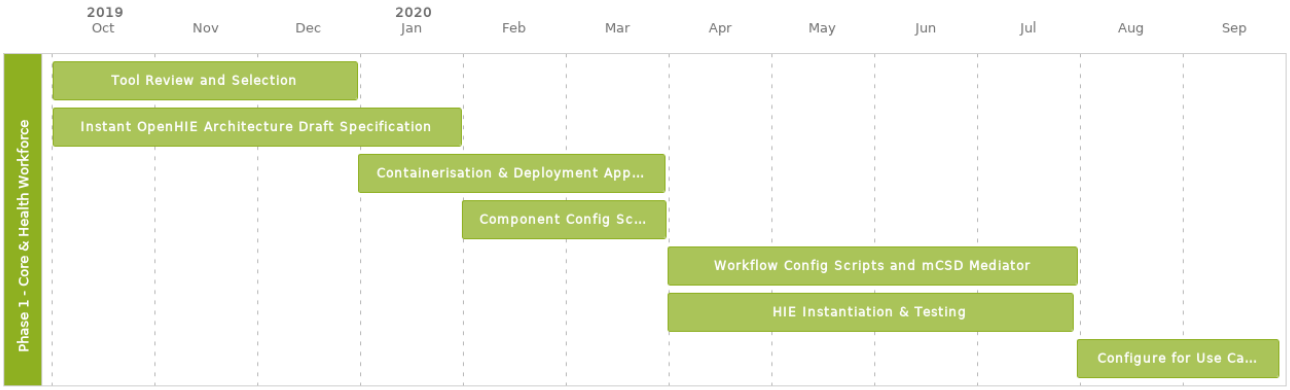
- Demonstrable reference products -- those that align with the OpenHIE Community's vision for low resource contexts
- Rapid software development of mediators and point-of-service systems by making it possible to launch several applications easily so the developer can focus on their task
- Reproducible, version-controlled infrastructure for user-contributed tests of the OpenHIE Architecture profiles, workflows, and use cases.
- Production-ready containers and orchestratable components that are deployable in any context.

The first phase of activities addresses items 1-2 with a focus on a particular use case and set of packages, while production-ready deployments and infrastructure for testing will be built incrementally upon the innovations and lessons learnt from the efforts of the earlier phase.

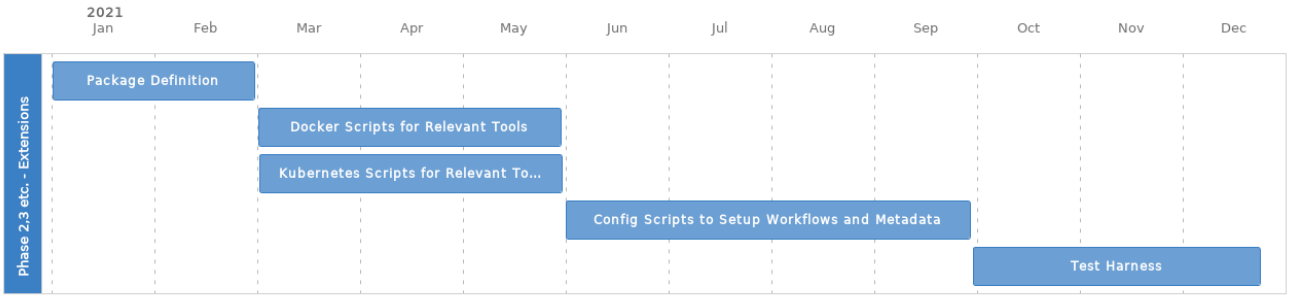
The first phase focuses on:

- Consultations with the relevant communities on profiles, use cases, and workflows, including with the OpenHIE Architecture Community and DevOps Community.
- Capturing the community feedback into an Instant OpenHIE Technical Design document for sharing. The Technical Design document is the entry point for understanding how to get started and make use of the stack as well as contribute to it and its future.
- The initial efforts at creating a **core** prototypical health information exchange using open standards and open source software to help developers add interoperability to your own products.
    - Two packages will be produced in the first phase, the **core package** and the **health workforce package** that extends from core and adds health workforce related functions and metadata.

Phase 1 Roadmap



Illustrative Roadmap for Future Use Cases



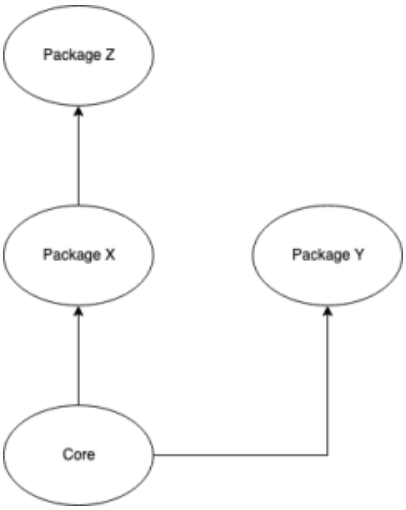
Architecture

The fundamental concept of Instant OpenHIE is that it can be extended to support additional use cases and workflows. This will be achieved through packages. A **core package** will be produced in this first phase which other packages will all derive from. A package will either extend directly from the core package or from another package.

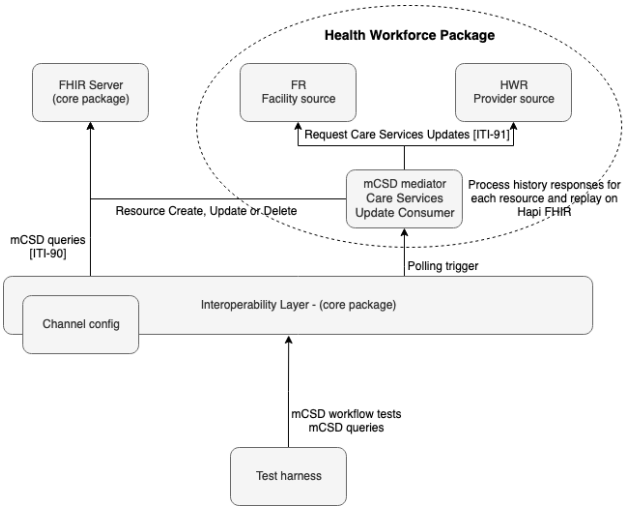
Each package will contain the following sorts of technical artefacts:

- Docker compose scripts for setting up the applications required for this package's use cases and workflows
- Kubernetes scripts for setting up the applications required for this package's use cases and workflows
- Configuration scripts to setup required configuration metadata
- Extensions to the test harness to test the added use cases with test data

The diagram below shows how packages will extend off each other to add use cases of increasing complexity:



The following diagram shows a generic logical architecture showing the involved OpenHIE components for the first phase, covering the core and health workforce packages. Each of these components' roles could be played by any application that supports the necessary OpenHIE workflows and component requirements. It would be possible to swap out applications for others as long as they conform to these specifications. In the future Instant OpenHIE could support multiple application options for each role.



For more details on the Instant OpenHIE architecture, see Instant OpenHIE Technical Architecture.

Contributing

Instant OpenHIE is designed to be extensible. There are two broad areas of contribution to the Instant OpenHIE stack:

- Contributing new components/apps to new and existing packages,
- Contributing new sets of workflows, use cases and tests to new and existing packages.

When contributing new components/apps, the following artefacts are expected to be produced to support this:

App owner responsibilities	**Description**
Tagged releases	Releases should be tagged in git or other version control system and in a public repository.
Environment variables	Configurations must be stored in or be able to be overridden by environment variables. See the Twelve Factor App: https://12factor.net/config

Dockerfile	Create a publicly available Dockerfile used to build the image and a link to it.
Container image	Make available a link to a public image of the application. A tagged release image should be available.
Docker Compose	Provide a link to a versioned Docker Compose script. A Docker Compose file should exist for running the application stack, including databases or web servers or other needs. Where possible use existing containers for things like databases or web servers. Slim images (e.g. Alpine) are recommended as many images will be run concurrently.
Automated configuration	Provide detailed information or scripts that can run in a non-GUI environment for automated configuration.

When contributing new workflows or implementation use cases, the following artefacts are expected to be produced to support this:

Workflow owner responsibilities	**Description**
Test data	Generate fake but realistic data for E2E tests and general functionality.
Package test data	Make test data available or reproducible online.
Tests	Write tests for the expected workflows supported using the containers, configuration, and fake data.
Dockerfile	Tests can be written in any language. Provide a Docker container for tests so that they can be run easily (with environment variables) against any stack (not just Instant OpenHIE)
Container image	Make available a link to a public image of the tests. A tagged release image should be available.

Project Team

Digital Square

Jembi Health Systems

IntraHealth International

Engaging with Us

Please join us in the OpenHIE DevOps Community Call OpenHIE DevOps Community Call

or, in the OpenHIE Architecture Community https://groups.google.com/forum/#!forum/ohie-architecture