How Smart4Health uses FHIR

A short introduction to FHIR

The Smart4Health project uses the FHIR standard (release R4) for exchanging healthcare data, the most modern, globally used healthcare exchange standard. Here, we provide a very brief introduction to the FHIR standard and explain how we use it in the Smart4Health (S4H) project, as documented in the S4H FHIR Implementation Guide. Extensive technical documentation of the FHIR R4 standard itself can be found online at https://hl7.org/fhir/R4.

FHIR resources

The abbreviation "FHIR" stands for Fast Healthcare Interoperability Resources. In the context of Smart4Health, it is used to exchange data coming from citizens, physiotherapists, and hospitals with the Citizen Health Data Platform (CHDP). The FHIR standard contains multiple components, among others

1. A set of predefined data units (called *resources*) for different kinds of health data
2. A mechanism for specifying further constraints on resources for a particular use case
3. Pre-defined methods for packaging and sending FHIR data to another system or fetching FHIR data from a system

We make extensive use of the two first components, as discussed below. However, since CHDP only stores and exchanges fully encrypted information, we use a custom interface when moving data to and from the platform.

FHIR resources

In FHIR, a resource is the basic unit of data. There are different resource types to represent different types of data. The most intuitive resource types are those that carry specific pieces of health information about a person. As an example, a resource of the type Patient contains core personal information like name and birth date, an `AllergyIntolerance` resource carries information about an allergy, and an Observation resource carries information about a specific measurement performed (e.g. the blood pressure or the outcome of a laboratory test). Some resource types are for information that is not tied to a specific person, e.g. `Medication` resources capturing information about a specific medication. There are also resource types for transmitting information about, e.g. a list of codes for a particular use case (`ValueSet` resources) or even the full definition of a resource type (`StructureDefinition` resources).

For each resource type, there is a defined list of data elements they can contain. Some data elements can contain sub-elements, so that the final data structure takes the form of a tree (this is also how it is usually represented graphically). FHIR resources can be written down

(serialized) as JSON or XML objects; this is how they are exchanged and the typical format for resource examples. Most data elements are optional, but a few will be required for every valid instance of a given resource type. For instance, every valid Observation resource must contain the `status` data element, which indicates whether the measurement is considered final, preliminary etc. Some standard data element types occur in many places, e.g. the `CodeableConcept` element type which allows capturing one or more codes for a specific concept (for instance the ICD-10 code for a disease or a code designating a specific medication).

FHIR resources can refer to each other using `Reference` data elements. For instance, an `Observation` resource can point to a `Patient` resource to say who this measurement was performed on, or to a `Device` resource carrying information about the instrument used to make the measurement. The referenced resource can be identified in two ways:

1. The URL under which it can be found on a particular server (often relative to a base URL), composed of a path indicating the resource type and the ID of the resource on the server (e.g. the relative URL of an Observation could be `Observation/abc123`). The server-specific ID is found in the `id` data element of the referenced resource (referred to in FHIR as the *logical ID* of the resource).
2. A globally unique identifier from a specific system of coidentifierdes. For instance, a unique patient number assigned by a specific hospital. This is often called a *business ID* and is given as a combination of the name of the system assigning the identifier and the identifier itself. In the referenced resource, this ID is found in the `identifier` data element (present in most, but not all, resources).

Below is an example (in JSON) of an `Observation` resource that captures that the body temperature of a patient was found to be 36.5 degree Celsius on June 2nd, 1999:

```
{
      "resourceType": "Observation",
      "id": "abcd123",
      "status": "final",
      "code": {
            "coding": [{
                  "system": "http://loinc.org",
                  "code": "8310-5",
                  "display": "Body temperature"
            }],
            "text": "Body temperature"
      },
      "subject": {
            "reference": "Patient/abc123"
      },
      "device": {
```

```
            "identifier": {
                  "system":
      "http://www.superduperhospital.example.com/our-device-id-system
      ",
                  "value": "super-duper-thermometer-id"
            }
      },
      "effectiveDateTime": "1999-07-02",
      "valueQuantity": {
            "value": 36.5,
            "unit": "C",
            "system": "http://unitsofmeasure.org",
            "code": "Cel"
      }
}
```

This example illustrates both ways of referencing other resources (in the `subject` and `device` elements, respectively). The `code` data element (a `CodeableConcept` element) is used to specify a code from the standard LOINC code system that uniquely identifies the measurement of body temperature. The use of codes from standard, globally known code systems is a key method for ensuring interoperability between different systems.

FHIR profiles, extensions, and implementation guides - how to use FHIR for specific use cases

FHIR is what is sometimes referred to as a *platform standard*. That is, rather than being intended for use out-of-the-box without modifications, the predefined resources are typically taken as a basis that is then adapted to a specific use case using the mechanism built into the FHIR standard. These adaptations reflect the particular requirements of the use case at hand. For instance

- Requiring that particular data elements always be filled (even if the base resource definition allows them to be left out) - e.g. requiring that all Patient resources include an address
- Requiring that codes from a specific terminology is used in a certain element - e.g. requiring that all all diagnosis come with an ICD-10 code
- Indicating alternative or preferred ways of encoding certain information in a resource - e.g. showing which subset of the SNOMED CT codes can be used to indicate a diagnosis.

FHIR uses so-called *profiles* to capture such requirements and check whether a given resource instance conforms to them. A profile is always for a specific resource type. It defines the requirement that a resource instance of this type should conform to for this specific use case (in addition to the requirement from the base resource type). That allows clearly specifying the

expectation for a given type of data. For instance, the FHIR standard itself defines a profile on the `Observation` resource for capturing the height of a patient (height profile documentation page). It specifies that resources conforming to this profile must e.g. carry a specific code from the LOINC code system and that the recorded value must be given in either centimeters or inches. When rendering profiles on web-pages, they are typically displayed either as *snapshots* which show *all* requirements (whether from the base resource or added by the profile) or *differentials* (which show only the additional requirement added specifically by the profile). One can also define a profile on top of an already existing profile (this is called *reprofiling*). In that case, a resource must fulfill all the requirements of the base resource type, the underlying profile, *and* the new profile in order to be conformant. The profile snapshot will now only contain the differences relative to the underlying profile.

Each profile is uniquely identified by a so-called canonical URL (and, optionally, a version). The canonical URL is assigned by the organization defining the profile and provides a globally unique way to refer to a given profile. For instance, the heigh profile mentioned above has the canonical URL `http://hl7.org/fhir/StructureDefinition/bodyheight`. A canonical URL need not point to an actually existing webpage (though ideally it should). However, since it has to be globally unique, it is common to make it a URL inside a web domain controlled by the creator of the profile.

An important FHIR principle is that resource instances do not have to declare the profile(s) they conform to (though they can claim conformance to profiles by listing their canonical URLs). Indeed, they could not possibly do so since a given instance may "by accident" conform to a whole range of profiles that the data owner is not even aware of. Instead, a resource instance has to be *validated against the profile* to check if it conforms to it or not. There is standard tooling for doing such validation given the resource instance and a profile (see e.g. the documentation for the standard command-line Java validator).

Profiles can only add constraints, never remove them. That is, profiles can only narrow down what is allowed. Thus, since the base definition of the Observation resource requires that the `status` element occurs exactly once, an Observation profile can never specify that the `status` should occur twice or that it can be left out. As a result, any resource instance that conforms to a given profile is automatically also a valid instance of the underlying resource type (since the profile must subsume all the base requirements).

FHIR also provides a mechanism - *extensions* - for specifying new data elements that are not in the base specification of the relevant resource type, but can be used in profiles. This is consistent with the "narrower-only" nature of profiles because the base definitions of all resources already allow the addition of an arbitrary number of extensions, and a new extension simply fills one of these predefined slots. Some commonly used extensions are defined as part of the FHIR specification itself, but it is also possible to add custom extensions. Extensions, especially custom extensions, make interoperability more difficult because they require the systems handling data to understand data elements that are not part of base resource definition. Hence, extensions should only be used when there is no viable alternative available - one

example is to indicate why an expected data element could not be provided (documentation of the data absent reason extension).

While profiles usually capture most of the details about the use case specific requirement on the data to be exchanged, they do not cover everything. For a given use case, there may also be specific requirements for how to access the data, custom code systems (themselves FHIR resources), general rules for the data exchange etc. These can be combined with the profiles to form a *FHIR Implementation Guide*, a set of generated web pages listing all the involved artifacts and providing written explanations. An implementation guide (often abbreviated IG) is, as it were, a complete manual for using FHIR to exchange data for a specific use case. An example is the International Patient Summary(IPS) FHIR IG, which can be found at http://www.hl7.org/fhir/uv/ips/STU1/. It provides a complete set of FHIR profiles, extra code systems etc. for the international exchange of minimal summaries of a person's healthcare record.

The Smart4Health FHIR implementation guide

The exchange of data within Smart4Health is one particular use case for FHIR. Accordingly, a Smart4Health implementation guide has been written to capture and communicate how FHIR is used within this project. To provide the best possible basis for international data exchange, the Smart4health implementation guide is based very directly on the existing International Patient Summary (IPS) implementation guide (see above). This is the most mature standard for the global exchange of data using FHIR and has been developed by an international team of experts over several years. In some cases, IPS profiles could be used directly without modifications. In other cases, small modifications had to be made. For instance, many IPS profiles require that the person to whom a piece of data applies is referenced using a URL-based reference (see above). Due to the encryption used in the CHDP platform and the way data is ingested, this is not possible in Smart4Health, so the project-specific profiles do not include this constraint. In yet other cases, entirely new profiles of resource types not covered by IPS were created (e.g. for `DocumentReference` or `Encounter` resources).

The current version of the Smart4Health implementation guide can be found at https://simplifier.net/guide/smart4health. Each resource type that has been profiled (i.e. for which a profile has been created) has its own page with a description, renderings of the profile snapshot and differential, and links to example resources (see for instance the page for the AllergyIntolerance profile). There is also a page listing all custom `CodeSystem`s defined for the Smart4Health use case (code system page), as well as pages describing the various questionnaires used for the Smart4Health back pain treatment and prevention citizen use cases (e.g. the page for the back pain prevention questionnaire). The implementation guide also contains general guidance on data exchange, e.g. which resource types are supported. The canonical URLs for FHIR artifacts defined as part of Smart4Health all have the base URL `http://fhir.smart4health.eu` . Since this domain is controlled by the project, it ensures that these URLs are globally unique.

All formal FHIR artifacts for the Smart4Health project are also published as a so-called *FHIR package*. This makes it easy to download a given version of the artifacts. FHIR packages can also be used by standard FHIR tooling, e.g. publishing the profiles as a FHIR package makes it easy to validate a resource against them (see the documentation of the standard FHIR validator). The Smart4Health package has the unique ID `smart4health.eu.core` and can be accessed via the project's package page: https://simplifier.net/Smart4Health/~packages.