FHIR in Action

wanghaisheng

Published
with GitBook

目錄

fhir-in-action

example and tutorial for fhir spec

FHIR ,a new emerging and appealing healthcare standards, with the all difficulty embedded,through the off the shelf Web standards,you can flood health information anywhere on the web 2.0.

电子书在线阅读地址

一些是博客的翻译和实验的例子
一个是根据2论文中的思路编写的基于FHIR的APP

参考资料 Reference:

1. David Hay`s blog
2. 硕士论文
3. Ewout Kramer`s blog

FHIR 简介

主要是介绍一些FHIR 缘起的大背景，以及产业界对其唱衰唱好的各种论调。

- 我眼中的FHIR
- awesome FHIR
- FHIR 概述
- FHIR 开发者指南
- 开放数据专题

对FHIR的一些浅显认识

互联网医疗在过去的一年里如火如荼，希望大家能够一起来汉化和开发FHIR相关的产品 FHIR – Fast Health Interoperable Resources (hl7.org/fhir) – 是由HL7创建的新一代标准框架.FHIR 整合了 HL7 V2,V3 和 CDA 的优点,同时利用了最新的Web标准,紧紧围绕着 implementability 可实现性.

FHIR 解决方案是基于一些称之为“资源”的模块化组件的. 这些资源可以很容易的组装进生产系统中,以已有方案的一小部分成本来解决实际的临床和管理上存在的问题. FHIR适用于多种场景– 智能手机APP、云平台上的通信、基于EHR的数据共享、大型医疗机构内服务器通信和其他。

大多数在HIT这个行业浸淫略久的人都听到过HL7的字眼，HIT 行业的标准不外乎有2个目的，交互共享数据(HL7 V2消息，V3消息，CDA，X12,共享文档规范诸如此类)，表达医疗行业的知识(各类术语字典，数据集数据元标准，Arden syntax，CDSC，GELLO诸如此类)，而FHIR应该归属于第一类，与它的前辈不同的是，它抛弃了既往顺着发展了10多年，乃至于20年的那块田(封闭又自恃过高，怎么说这点呢，所有的标准文件都有自己独特的脱离了整个软件行业的编辑器生成，这些模型也不能为其他通用型软件所读取，最可恨的是没有配套的各种开源库开放给大家试错，降低学习成本，恶心的是居然个破标准还要收费)，也就是在上世纪90年代末XML在整个软件领域刮起一阵风，到处都在说系统集成时应运而生的V3消息，CDA等，毫无疑问它们都是为不同厂家的不同系统之间交换数据而生的，而它们的下场都很惨，用现在的话说，都不够敏捷，学习成本过高，迭代也不够快。

在FHIR卷土重来的时候就把基调定好了，思想上这次要全面拥抱互联网技术，用通用的互联网技术来做原来没有成功的事情，仍然还是想实现医疗健康领域数据的无缝流动，打通整个数据闭环。 它的诱人之处我认为有如下三点：

1、过去的很多年，美国人造了各种各样的标准，国际友人也造了各式各样的轮子来解决上面的问题，最近的2-3年里很多人抛出了这样一个问题，能不能用同一种模型，经过适当的演化就能表达医疗数据(电子病历、健康档案、个人健康记录)，又能表达医学知识(临床指南类决策支持用的知识，质控指标类的知识)，就目前FHIR发展的现状来看，这是一个还不错的选择。有很多这方面的尝试，美国的ONC最近也在这件事上砸了些钱希望能推动的更快一些。

2、拥抱互联网技术。卫生部这几年在推的区域平台、医院平台的技术点在我看来离现如今的互联网技术太远了，作为已经被抛弃的SOAP流的SOA架构的残留物，着实没听到在BAT等企业有何应用，最近几年在人人学Amazon的同事，新浪、京东等一批国内企业都在推一个叫open API 或者是restful api或者叫HTTP　API的东东，以此来解决各自内部千千万产品间、产品内部的数据流动的问题，简而言之就是以一种方式圈定某个领域的业务对象，每种对象都使用同样的方法来实现一些功能，类比到厨艺的话，就是说约定好如何区分食材，每种食材都有哪些烹饪方法，这样子整个医疗领域就大约有100-200种资源(最小的信息单元，当然这里面的粒度的拿捏很是讲究)，用到的“烹饪方法”就是HTTP协议定义好的(诸如put/post/get等)。数据本身的表达格式也从原来V2　V3单纯的EDI格式、XML格式演化到了目前比较流行的JSON，以后或许还会演化出其他更为适合的方式。这件事情一方面降低了学习认知整个标准的门槛，另一方面即使不使用它所规定的格式，顺着这样的思路，你也能解决一些问题，这里的问题不单单是之前它的老前辈(v2\v3\cda)所更care的产品与产品间的数据交换数据共享，你也可以借此实现产品内部的功能(京东的李大学总裁就介绍过它们在这方面的一些探索和实践)，比如面临的移动端和PC端开发时功能复用的问题。当然也有人会问，这东西能作为数据的存储模型来用么，这个问题是HL7 V3 RIM CDA所没能很好的解决的问题，它们的抽象程度太高了，但时代变了，Nosql数据库现在已经有很多可供选择，如果你要使用关系型数据库的话，也有一些这方面的探索可供参考。

3、尽管国内目前关注度不够，但在著名的代码托管平台Github上已经有大量的各种编程语言(java c# dephi javascipt swift等)、各种平台可用的一些开源代码，有适合PC端的，也有适合移动端的，这是很喜人的。

欢迎交流，联系方式：

webChat：搜索“edwin_whs”

qq：512139097

qq群：194806088

讨论组：https://fhircn.bearychat.com/messages/%E6%89%80%E6%9C%89%E4%BA%BA

奉上一些链接供大家把玩

1、汉化版https://github.com/wanghaisheng/fhir-cn

英文版http://hl7.org/implement/standards/fhir/

2、开源库

https://github.com/jamesagnew/hapi-fhir https://github.com/smart-on-fhir/smart-on-fhir.github.io https://github.com/fhirbase

FHIR Implementation

- The current specification: http://www.HL7.org/fhir/ (or the development version)
  - FHIR Profiles from other Organizations
- Contact Information
  - Implementation help: [ask questions about FHIR]
  - Formal Contact point for the project: [fmgcontact@hl7.org]
  - Skype Group Chats
  - FHIR gForge Tracker for change requests/corrections
  - FHIR Project Team Leads (FHIR Core Team): [Grahame Grieve], [Ewout Kramer], [Lloyd Mckenzie]
  - List server - project email list
- Help / Getting Started
  - FHIR Starter - tutorial for FHIR newbies
  - FHIR Cheat Sheet (DSTU 1)
  - Help desk FAQs & knowledge-base articles (HL7 members only)
  - FHIR Tools Registry - a list of useful tools for FHIR implementers
  - FHIR for Clinical Users - an introduction to FHIR for non-technical people that will migrate to the specification in the future
  - FHIR User Group
- Social Media on FHIR
  - FHIR blogs David Hay, Ewout Kramer, Grahame Grieve
  - FHIR News on Twitter FHIR News
  - FHIR Videos HIMSS FHIR Session, HL7.tv and Ringholm.
- Testing
  - Publicly Available FHIR Servers for testing
  - Open Source FHIR implementations
  - FHIR Connectathon 8 (January, San Antonio)
  - Organizations interested in FHIR
  - Profile Tooling
- FHIR Implementations
- Connectathons
  - Connectathon 9 (May 9-10, Paris, France)
- Previous Connectathons and other events
  - Historical Connectathons (list)
  - Clinical Connectathon 1 (September 2014, Chicago) (+ Clinical Connectatathon 1 Tooling)
  - 2 day seminar and Connectathon (Nov 6-7, Melbourne Australia)
  - International FHIR Development Days (Nov 24-26, Amsterdam)

FHIR Development

- How to
  - FHIR DSTU monitoring - how to monitor DSTU feedback
  - FHIR Ballot Prep - tasks for the next ballot and milestone dates
  - FHIR Build Process - Setting up and running the FHIR build process

    - Technical Guide - How to create resources
    - Materials: gForge, SVN Trunk
        - For read-only SVN access, use "anonymous" and your email as a password.
        - For Commit privileges, send a request to lloyd@lmckenzie.com
    - FHIR resource and profile proposals - proposals for new resources & profiles
    - FHIR Profile authoring - Creating and maintaining FHIR profiles (see also Profile Tooling)
    - FHIR Change requests - Process for managing and resolving
- Guidelines
    - Fundamental Principles of FHIR
    - FHIR Methodology Process - how the methodology is developed and maintained
    - Methodology Guidelines - Content and quality guidelines
    - Design Patterns
    - FHIR Comparison to other RESTful API specifications
- Resources
    - List server - discussions
    - Discussion pages: Active Discussions, All
    - FHIR Design Requirements Sources
    - FHIR Resource Types
    - FHIR Resource Considerations
    - (old) - governance discussion with bits of methodology mixed in (to migrate to other pages)
    - FHIR Terminology Service
    - FHIR Digital Signature Working Page
- WG FHIR pages
    - Patient Administration Resource development
    - CDA to FHIR Samples Group
    - FHIR_Patient_Care_Resources

Organizational

- Governance
    - FHIR Governance Process
    - FHIR Governance Board (FGB)
    - FHIR Management Group (FMG)
    - Modeling and Methodology (MnM)
    - Work Groups
    - FHIR Escalation Processes
    - FHIR Ballot Process
    - FHIR Web Server Hosting Record
- Agendas
    - **Paris WGM** (next meeting, May 2015)
    - Past Working Group Meetings (list of agendas/notes)
    - MnM agendas
    - FGB Agendas & Minutes
    - FMG Agendas & Minutes

opensource project

- Testing
  - FHIR 公开测试服务器的测试报告
- Lib
  - Ruby语言的FHIR模型 工具等

tips: 详情请阅读 中文版网站FHIR概述

1.7.0 FHIR概述

欢迎使用 FHIR 标准，它是卫生保健信息电子化交换的一种标准。这部分是对标准的概述，作为一个路线图，希望能帮助初次接触的读者更快上手。

1.7.0.1 背景

医疗保健记录越来越多的被数字化。当病人在整个医疗系统中转诊的时候，要求他们的病历能够获得、能够找到和能够理解。更深层次的，能够支撑自动化地临床决策支持和其他机器处理，要求数据是结构化的 并且是标准化的。(参考卫生保健所面临的数字化挑战)

HL7在过去20多年里，一直致力于构建卫生保健数据交换和信息模型标准来解决这些难题。FHIR是一种新标准，采用了其他行业的通用方法，同时借鉴了在定义和实现 HL7V2，V3,RIM 和 CDA 标准过程中所获得成功、失败的教训。FHIR可以单独作为数据交换标准来使用，也可以和其他广泛应用的标准一起来使用(参考FHIR与其他HL7标准的比较)

FHIR旨在不牺牲信息完整性的前提下简化开发和实现。它利用了现有的逻辑和理论模型，为不同的应用程序间交换数据提供了一种一致的、易于实现的、健壮的机制。FHIR的内在机制使得其能够追溯到 HL7 RIM 和其他内容模型，这就保证了与HL7之前定义的模式，最佳实践间的保持一致，毋须开发人员充分了解 RIM 和 HL7 V3 的其他衍生制品。(参考FHIR与其他HL7标准的比较)

1.7.0.2 组件

FHIR 中最基本的组件叫做资源.所有可交换的内容都被定义成一个个资源。所有资源都拥有如下特征：

- 同一种定义、表达它们以及从数据类型(最基本的可重用元素)构建它们的方式
- 同样的元数据集合
- 供人可读的部分

1.7.0.3 方法论

1.7.0.3.1 信息建模的方法

FHIR 的理念在于定义一个资源的基础集合，要么利用它们，要么相互结合来满足大多数常见的应用场景的需求。FHIR 资源旨在定义绝大多数开发实现中通用的核心信息集合的信息内容和结构。如果需要的话，有内在的扩展机制来满足剩下的需求。 FHIR 的建模采用了一种组合式的方法论。相比而言，HL7 V3 建模是基于“model by constranit”(参考FHIR与其他HL7标准的比较) 。对于 FHIR，特殊的应用场景通常是通过利用资源引用整合资源来实现的。尽管对于一个特定场景，单独一个资源可能是存在价值的，更多的是对资源互相整合和裁剪来满足特定的需求。用来描述资源如何整合使用的两类特殊资源：

- 一致性声明——描述一种实现中所暴露的交换数据的接口
- 规范profile——描述该实现中所使用的资源中定义的用以约束基数、可选性、术语、数据泪下和扩展的其他规则。

1.7.0.4 标准

基本上，FHIR标准分为三大块：

- 基础文档部分——描述了资源是如何定义的，数据类型、编码的定义和XML 、JSON格式的相关背景信息。
- 开发实现部分——描述在REST 、消息、文档和SOA中使用资源。
- 资源列表——resourcelist.htmlFHIR中定义的所有资源的列表。其中又分为临床类、 行政管理类 和基础架构类三大类。

资源有多种用途，从最基本的护理计划和诊断报告等临床内容到如消息头、一致性声明等基础架构。虽然它们具有共同的技

术特性，但却以完全不同的方式来使用。注意毋须为了使用资源而必须使用REST。

1.7.0.5 如何入门

最好是快速阅读一下 资源列表，对已经有了哪些资源有个感性认识，然后看一下患者的定义来看看资源的定义是什么样的，接着读一下以下介绍背景信息的章节：

- 资源——资源是如何定义的
- 所有资源都有的叙述性文本，以及资源之间如何互相引用
- 格式：XML 、JSON
- 扩展相关——标准能够保持简单的关键
- 如果你之前了解 HL7 标准(V2 V3 CDA)的话，FHIR 与其他 HL7 标准的比较也值得一看。

1.7.0.5.1 顶部标签

整个标准中都会看到这些标签，很多读者可能会遗漏：

资源和数据类型都是以一种类似XML的 易于阅读的方式来呈现的，它们也有详细描述内容的正规定义。另外，大多数资源映射到很多不同的格式，如HL7V2，HL7V3 RIM，CDA，DICOM等。同时，所有资源至少包含一个实例(有时候会有更多)，适当的时候，也会有描述它们如何在特殊情况下使用的profile规范。最后，一些资源包含了 帮助开发人员理解它们背后的设计原理的小贴士/备注。

1.7.0.6 寻求额外信息和提供反馈

为了能够让更多读者看懂的同时，FHIR标准是面向开发人员社区的——这些真正利用标准编写程序的人。为了满足开发人员社区的需求，编辑人员力求标准行文精确，减少在编写程序之前的阅读时间(然而这份标准并不如我们所想的那么简明扼要，有时候是医疗保健和现实世界的复杂度所致)。鉴于此，在开发过程中并不必要的信息，诸如原理、备选方案、一些争论点和将来的计划等并没有包含在标准里面。同样，开发人员时不时会发遇到标准不明晰或者不完整的清空。最终，会有一些情况，标准可能是错的，或者是对其进行修订以更好的满足开发人员的需求。 因此HL7提供了多种方法，通过这些方法可以维护和获取一些额外的FHIR相关信息，能够提供一些帮助，响应一些变更请求。

1.7.0.6.1 评论

在每页底部，都有一个评论的部分，可以就特定章节进行提问和讨论。评论由 FHIR 编辑人员和HL7工作组来监管，每个问题都会在一定的时间内给予答复。This content will occasionally be currated to ensure ongoing relevance, particularly if the specification is subsequently updated to eliminate confusion that may have spawned an initial comment.

1.7.0.6.2 FHIR Wiki

FHIR 项目团队维护着一个wiki，记录了开发过程、方法学和设计决策。开发人员和其他人员也可以参与到wiki中来，提供一些暂未出现在标准中的指导和补充信息。注意 FHIR wiki上的内容不具有权威性，与FHIR标准的一致性无关。同样，一些内容可能也没有和最近的FHIR版本保持一致。 FHIR标准中的每个页面在wiki中都有一页内容。用以记录原理、决策点和其他与开发人员无关的信息。额外的页面包括FHIR 方法论、FHIR 设计工具的使用等。要研究wiki的话，建议从首页开始.

1.7.0.6.3 正式变更请求

正式请求可以在这里提交。对应的工作组会审核这些请求，并作出是否将其纳入到标准中的决策，其中包括了纳入到那个版本。

1.7.0.6.3 源码/参与机制的额外信息

除了上述机制，HL7提供了一个 Stack Overflow 的标签，邮件列表和skype聊天频道来提供对开发人员各个层面的全方位的支持。如何使用这些请参考指令

1.7.1 开发者指南

FHIR (Fast Health Interoperability Resources)旨在数据交换，能够支撑医疗领域的多种流程。该标准基于Restful的最佳实践，能够实现跨团队的医疗系统的集成。

FHIR 所支持的范围很广泛，包括人、兽医、临床、公共卫生、临床试验、管理和财务等方面。全球通用且支持多种架构和场景。

1.7.1.1 框架

FHIR 是基于 `资源` 这一通用组件. 每个资源都有如下 通用特征:

- 用URL来标识
- 通用的元数据
- 供人可读的XHTML概述
- 通用的数据元集合
- 扩展的框架以支持医疗中的多样性

资源要么是 XML ， 要么是 JSON格式的. 目前已经定义了99种资源类型

1.7.1.2 Patient实例

如何用JSON来表示patient。 标准中也定义了XML的表达方式。

```
{
  "resourceType": "Patient",
  "id" : "23434",
  "meta" : {
    "versionId" : "12",
    "lastUpdated" : "2014-08-18T01:43:30Z"
  }
  "text": {
    "status": "generated",
    "div": "<!-- Snipped for Brevity -->"
  },
  "extension": [
    {
      "url": "http://example.org/consent#trials",
      "valueCode": "renal"
    }
  ],
  "identifier": [
    {
      "use": "usual",
      "label": "MRN",
      "system": "http://www.goodhealth.org/identifiers/mrn",
      "value": "123456"
    }
  ],
  "name": [
    {
      "family": [
        "Levin"
      ],
      "given": [
        "Henry"
      ],
      "suffix": [
        "The 7th"
      ]
    }
  ],
```

```
  "gender": {
    "text": "Male"
  },
  "birthDate": "1932-09-24",
  "active": true
}


`
```

每个资源包括如下内容:

- **resourceType** (line 2) - 必须要有: FHIR 中定义了多种资源类型，详细列表请查看the full index
- **id** (line 3) - 资源自身的id(而非资源中数据的ID 相当于资源在数据库中的主键). 一般而言都是要有的，除了在新建时之外。
- **meta** (lines 4 - 7) - 通常要由 : 所有资源都会有的属性(这里和其他地方对元数据的定义略有偏差，参考 https://github.com/memect/hao/issues/296)受基础架构控制. 如果没有元数据可以为空
- **text** (lines 12 - 17) - 推荐使用: XHTML 包含了资源中 供人可读的部分
- **extension** (lines 12 - 17) - 可选: Extensions由扩展框架所定义
- **data** (lines 18 - 43) - 可选: 每种资源所定义的数据项。

备注 尽管标准中总是以所定义的顺序来显示JSON中数据的顺序，但很多JSON库有其他排序标准。

1.7.1.3 交互

为了操作数据，FHIR 定义了REST API:

- Create = POST https://example.com/path/{resourceType}
- Read = GET https://example.com/path/{resourceType}/{id}
- Update = PUT https://example.com/path/{resourceType}/{id}
- Delete = DELETE https://example.com/path/{resourceType}/{id}
- Search = GET https://example.com/path/{resourceType}?search parameters...
- History = GET https://example.com/path/{resourceType}/{id}/_history
- Transaction = POST https://example.com/path/
- Operation = GET https://example.com/path/{resourceType}/{id}/${opname}

除了RESTful API之外,FHIR 中还定义了其他的数据交换方式，包括 文档, 消息和其他类型的服务.

1.7.1.4 对多样性的管理

医疗行业的一大特点就是不同地区和细分行业都存在很大的差异性,并不存在一个集中式的权威机构来定义通用的行业规范。鉴于此, FHIR 中定义了通用扩展框架和 管理多样性的框架.

1.7.1.5 新增资源

为了新增资源, 需要发送一个 HTTP 的 POST 请求到某个资源节点(也就是某个URL).如下所示

```
  POST https://example.com/path/{resourceType}
`
```

```
POST {some base path}/Patient HTTP/1.1
Authorization: Bearer 37CC0B0E-C15B-4578-9AC1-D83DCED2B2F9
Accept: application/json+fhir
Content-Type: application/json+fhir
Content-Length: 1198
```

```
{
  "resourceType": "Patient",
  ...
}
`
```

向服务器提交一条患者记录, 服务器可以根据自己的情况分配ID来存储该患者记录。备注：

- **/Patient** (line 1) - 处理所有患者的节点- 这里使用资源类型的名称
- **Authorization** (line 2) - 参考 Security for FHIR
- **Accept, Content-Type** (lines 3-4) - 如果资源的数据是JSON格式，content type需要设置成这样application/json+fhir (XML的话设置成 application/xml+fhir). 数据的编码始终是UTF-8
- **id** (line 9) - 待新建的记录中并没有id，由服务器来分配
- Resource Content, lines 8+ - 这时候也没有任何元数据。资源的其他部分同上述示例

1.7.1.6 新增资源的响应

响应中包含HTTP 201，表示服务器已经成功新建该条记录。location header 属性中包含了访问该资源的URL。响应中亦可包含OperationOutcome 资源来表达处理的一些细节,并不做硬性要求。

```
HTTP/1.1 201 Created
Content-Length: 161
Content-Type: application/json+fhir
Date: Mon, 18 Aug 2014 01:43:30 GMT
ETag: "1"
Location: http://example.com/Patient/347

{
  "resourceType": "OperationOutcome",
  "text": {
    "status": "generated",
    "div": "<div xmlns=\"http://www.w3.org/1999/xhtml\">The operation was successful</div>"
  }
}
`
```

Notes:

- **HTTP/1.1 201** (line 1) - 操作成功. Note that HTTP/1.1 is strongly recommended but not required
- **ETag** (line 5) - used in the version aware update pattern
- **Location** (line 6) - the id the server assigned to the resource. The id in the url must match the id in the resource when it is subsequently returned
- **operationOutcome** (line 9) - OperationOutcome resources in this context have no id or meta element (they have no managed identity)

1.7.1.6.1 Error response

出于多种原因，服务器会返回一个错误信息，FHIR 内容相关的一些错误信息以HTTP 状态码加一个OperationOutcome来表达. 如下是一个不满足服务器端业务规则时的返回信息:

```
HTTP/1.1 422 Unprocessable Entity
Content-Length: 161
Content-Type: application/json+fhir
Date: Mon, 18 Aug 2014 01:43:30 GMT

{
  "resourceType": "OperationOutcome",
  "text": {
    "status": "generated",
    "div": "<div xmlns=\"http://www.w3.org/1999/xhtml\">MRN conflict
```

```
    - the MRN 123456 is already assigned to a different patient</div>"
  },
}
`
```

Notes:

- 服务器可通过OperationOutcome来表达更为详细的错误信息

1.7.1.7 Read Request

读取资源内容是通过HTTP GET请求来实现的.

```
  GET https://example.com/path/{resourceType}/{id}
`
```

```
GET /Patient/347?_format=xml HTTP/1.1
Host: example.com
Accept: application/xml+fhir
Cache-Control: no-cache
`
```

Notes:

- **347** (line 1) - 要访问资源的id
- **_format=xml** (line 1) - 希望返回的数据格式，这种方式适合于客户端无法访问HTTP 头信息的情况，例如XSLT转换时，也可以通过HTTP 头中的accept字段来指定(see Mime Types
- **cache control** (line 4) - 如何控制并发是很重要的，但FHIR中并未做出规定,更多信息请参考http://www.w3.org/Protocols/rfc2616/rfc2616-sec13.html 或者 https://www.mnot.net/cache_docs/

1.7.1.8 Read Response

读取单个资源内容GET请求的响应是单独的一个资源.

```
HTTP/1.1 200 OK
Content-Length: 729
Content-Type: application/xml+fhir
Last-Modified: Sun, 17 Aug 2014 15:43:30 GMT
ETag: "1"

<?xml version="1.0" encoding="UTF-8"?>
<Patient xmlns="http://hl7.org/fhir">
  <id value="347"/>
  <meta>
    <versionId value="1"/>
    <lastUpdated value="2014-08-18T01:43:30Z"/>
  </meta>
  <!-- content as shown above for patient -->
</Patient>

`
```

Notes:

- **id** (line 8) - 资源的id，与请求中的id一致
- **versionId** (line 11) - 该资源的最新版本. 最佳实践中要求该值与 ETag值匹配 (see version aware update), 对于客户端而言，不能认为二者总是匹配的. 一部分服务器并不记录资源的版本信息。

- 尽管建议服务器能够保留版本信息，但不做强制性要求
- **lastUpdated** (line 12) - 如果存在该字段，字段值应与HTTP header中的值保持一致

1.7.1.9 Search Request

除了读取单个资源内容之外，也可以通过查询参数和变量 查询资源内容，形式一般如下:

```
GET https://example.com/path/{resourceType}?criteria
```

The criteria is a set of http parameters that specify which resources to return. The search operation

```
https://example.com/base/MedicationPrescription?patient=347
```

会返回该患者的所有处方信息.

1.7.1.10 Search Response

查询请求返回的对象是一个bundle: 如未明确要求，只返回满足查询参数要求的资源元数据:

```
{
  "resourceType": "Bundle",
  "id" : "eceb4882-5c7e-4ca4-af62-995dfb8cef01"
  "meta" : {
    "lastUpdated" : "2014-08-19T15:43:30Z"
  },
  "base": "http://example.com/base",
  "total": "3",
  "link": [
    {
      "relation" : "next",
      "url" : "https://example.com/base/MedicationPrescription?patient=347&searchId=ff15fd40-ff71-4b48-b366-09c706bed9c
    }, {
      "relation" : "self",
      "url" : "https://example.com/base/MedicationPrescription?patient=347"
    }
  ],
  "entry": [
    {
      "resource" : {
        "resourceType": "MedicationPrescription",
        "id" : "3123",
        "meta" : {
          "versionId" : "1",
          "lastUpdated" : "2014-08-16T05:31:17Z"
        },
        ... content of resource ...
      },
    },
    ... 2 additional resources ....
  ]
}
`
```

Notes:

- **resourceType** (line 7) - "SearchResults" is the name for a bundle returned from a search
- **id** (line 3) -服务器为该次查询响应bundle的唯一标识. 有些情况下要求该id满足 全球唯一
- **meta.lastUpdated** (line 10) - This should match the HTTP header, and should be the date the search was executed, or more recent, depending on how the server handles ongoing updates. The lastUpdated data SHALL be the same or

more recent than the most recent resource in the results

- **base** (line 12) - 返回的内容中所有资源引用 相对地址的根地址。
- **total** (line 13) - 满足查询条件的记录数量. 这里的数量指的是总数，而非仅该bundle中所包含的数量，详情查看 可以对结果进行分页查询
- **link** (line 14) - A set of named links that give related contexts to this bundle. Names defined in this specification: first, prev, next, last, self
- **item** (line 23) - 用来表达满足查询条件的实际资源的元数据信息
- 如果加上include标签，可以强制要求服务器在返回结果中包含资源的内容，详情请参阅return additional related resources

1.7.1.11 Update Request

客户端用新版本的资源记录替换服务器中的老版本.

```
  PUT https://example.com/path/{resourceType}/{id}
`
```

Note that there does not need to be a resource already existing at {id} - the server may elect to automatically create the resource at the specified address. Here is an example of updating a patient:

```
PUT /Patient/347 HTTP/1.1
Host: example.com
Content-Type: application/json+fhir
Content-Length: 1435
Accept: application/json+fhir
If-Match: 1

{
  "resourceType": "Patient",
  "id" : "347",
  "meta" : {
    "versionId" : "1",
    "lastUpdated" : "2014-08-18T01:43:30Z"
  },
  ...
}
`
```

Notes:

- **resourceType** (line 1) - "Patient" URL请求中的资源类型必须与提交的数据中的资源类型保持一致 (line 9)
- **resource id** (line 1, "347") - URL中的资源id必须与提交的数据中id值保持一致(line 9)
- **If-Match** (line 6) - 如果存在该字段，必须与资源内容中的meta.versionId值保持一致 (line 12), 服务器必须核实版本的完整性，如果不支持版本则返回412状态码
- **meta.lastUpdated** (line 10) - This value is ignored, and will be updated by the server
- **resource content** (line 14) - 这里省略了资源内容

1.7.1.12 Update Response

更新请求的响应包括了元数据、状态和OperationOutcome(可选):

```
HTTP/1.1 200 OK
Content-Length: 161
Content-Type: application/json+fhir
Date: Mon, 18 Aug 2014 01:43:30 GMT
ETag: "2"

{
```

```
  "resourceType": "OperationOutcome",
  "text": {
    "status": "generated",
    "div": "<div xmlns=\"http://www.w3.org/1999/xhtml\">The operation was successful</div>"
  }
}
```

Notes:

- **ETag** (line 5) - This is the versionId of the new version

1.7.1.13 Base Resource Content

所有资源都会包含的基础信息:

```
{
  "resourceType" : "X",
  "id" : "12",
  "meta" : {
    "versionId" : "12",
    "lastUpdated" : "2014-08-18T01:43:30Z",
    "profile" : ["http://example-consortium.org/fhir/profile/patient"],
    "security" : [{
      "system" : "http://hl7.org/fhir/v3/ActCode",
      "code" : "EMP"
    }],
    "tag" : [{
      "system" : "http://example.com/codes/workflow",
      "code" : "needs-review"
    }]
  },
  "implicitRules" : "http://example-consortium.org/fhir/ehr-plugins",
  "language" : "X"
}
```

Implementers notes:

- **resourceType** (line 2) - 每个资源都会资源类型的字段. XML的话也就是根节点
- **id** (line 3) - 在资源新建之时分配后不再变化.只有在初次创建该资源时才没有该字段
- **meta.versionId** (line 5) - 当资源内容(除了meta.security、meta.profile、meta.tag三个之外)发生变更时该值随之变化
- **meta.lastUpdated** (line 6) - 随versionId的变化而变化. 如果服务器不维护版本信息，则不用记录该字段
- **meta.profile** (line 7) - 表示资源的内容是否遵循某个规范(比方说满足阿里健康的开放API的要求或者说满足卫计委共享文档中的要求). 更多信息请参考 Extending and Restricting Resources. 当规范、值集本身发生变动时可以更改该字段的值
- **meta.security** (lines 8 - 11) - 安全类标签. 该标签将资源与某些安全策略、基础架构策略联系起来。该字段的值可随资源内容的变动而变动,或者随安全体系的控制。
- **meta.tag** (lines 12 - 16) - 其他类型的标签. 如需将资源与特定的工作流程关联起来可以使用此类标签.在解读资源内容时无需考虑此类标签的值 。对此类标签值的更新不会影响资源内容版本的变化 updated (这里好像是说 可以不变更资源的版本就修改tag标签的值 还是说tag值的修改压根就不影响资源版本 其他的security和profile tag三个字段是否都适用呢？待考证)
- **implicitRules** (lines 17) - 如何准确安全的处理资源内容而在发送接收双方达成的协议. 由于使用了该字段就意味着其他系统要使用其中的数据可能会出现解读错误的情况，限制了数据的重复利用，故不推荐适用该字段
- **language** (lines 18) - 资源内容所采用的表达语言. 当前， 资源内容中亦可包含其他语言的内容; 该字段表示的是资源的主要语言。

开放数据专题
1、立法推动usa 政府放开更多数据
2、epic和commonwealth之争
3、API能否拯救HIT

1、立法进一步推动USA政府开放更多数据—SGR repeal opens doors to big data

在2015年4月16号，一个因废止了Medicare中过时的根据可持续增长率来计算医生收入的公式而出名的Medicare Access and CHIP Reauthorization Act (MACRA)东东正式成为了法律，有人称此举给Affordable Care Act法案的Medicare Data Sharing Program注入了一剂强心剂，更有人称其中的105章节要求政府将更多的 claim(医保索赔)数据开放给准入机构，更是给中小型医疗机构带来了春天。

```
1、由于平价医疗法的存在，HHS也就是美国卫生部公开了一部分Medicare claim数据给准入机构，这些机构利用医保数据和商业化数据进行数据分析，来获取
如何更好的达到要求的医疗质量和效率等指标来指导实践。由于此举风险甚高，议会持保守态度，但很快大家都意识到这样做远远不够，故在MACRA的105章节中

2、其中要求HHS Secretary不仅开放Medicare数据，也开放Medicaid和CHIP数据。这样子系统中的医保数据将会翻倍，能够支持更加复杂的数据分析。

3、尽管开放多少数据，开放不开放的决定权握在了HHS Secretary手里，但还是有很大希望的，毕竟Medicare的医保数据已经开放了2年，如果够快的话，Me
```

背景资料

ACA平价法案

```
美国总统奥巴马上任后积极推动医保改革，去年立法的《平价医疗法》当地时间12日却被上诉法院裁定违宪，令医保改革前路茫茫。分析指，表面上，共和党明显

据报道，目前美国约有5000万人没有基本医疗保险，医院及纳税人被迫每年代为支付高达430亿美元(约合人民币2749亿元)。民主党控制的国会去年3月通过立法

据介绍，强制参保条款是奥巴马医改法案的重点，能使3200万未投保的民众被纳入医保“保护伞”下，令美国全国医保覆盖率提升至95%，并要求保险公司接受已患

分析认为，美国民主党其实并非很支持强制投保，最支持的是可赚大钱的保险公司。保险公司要求全民强制参保，才愿意接受早已患病的投保人。民主党当初为平
```

需求驱动的数据开放计划

```
https://github.com/demand-driven-open-data
Tools and methods that provide a systematic, ongoing and transparent mechanism to tell data owners what's most valuable
```

美国政府的开放数据集

```
自从Data.gov开放以来，美国已经公开了超过八万五千个政府数据集并提供免费下载。现在杀出了一个公司叫做enigma，把多个国家的开放数据整合索引，
提供开放数据的搜索服务。
enigma: https://app.enigma.io 纽约时报:http://www.nytimes.com/2014/03/16/technology/a-harvest-of-company-details-all-in-
```

开放数据计划

```
https://project-open-data.cio.gov/
https://github.com/project-open-data
```

2、epic和commonwealth之争

背景介绍

主角： Epic

```
1、参考dr2的[新文章](http://ww2.sinaimg.cn/bmiddle/be7d1ecagw1erwsfgwnlqj20c84mob29.jpg)
2、[美国的The top 100 EHR companies (Part 1 of 4)前一百](http://medicaleconomics.modernmedicine.com/medical-economics/con
```

主角： CommonWell联盟(Cerner, McKesson, athenahealth and Greenway四大公司，亦可参考上面的top100了解其大概的规模)

```
1、参考http://www.commonwellalliance.org/news/commonwell-health-alliance-announces-member-expansion-plans-for-2015/
2、参考http://www.commonwellalliance.org/
CommonWell Health Alliance today announced the commitment of five of its health IT vendor members—athenahealth, Cerner,

Already, more than 60 provider sites are live on CommonWell services across 15 states including: Alabama, Delaware, Flo
```

缘起

2014年7月17号，在众议院能源和商务委员会的小组委员会通讯技术和健康 House Energy and Commerce Committee's subcommittee on Communications and Technology and Health的听证会上 ,从医生成为议员的Rep. Phil Gingrey 指责epic道：

```
一些电子病历的供应商的系统本身就在阻碍数据的共享和交换，就不应该拿到Meaningful Use incentive计划的钱。但根据RAND的一份报告，刺激计划的24
Epic公司拿走了，可Epic玩的是自己的封闭平台，这是HITECH的初衷么，纳税人的钱是这么花的么。

无独有偶，在RAND的报告中建议国会“取消那些需要额外组件、费用和定制才能够实现数据共享的医疗信息化产品的认证资格”。同期的一份研究中也指出Meanin
第二阶段的数据共享的架构是不够健壮的，不足以支撑数据的共享。
```

原文如下

```
Electronic health record vendors--particularly Epic--may not deserve Meaningful Use incentive money because their syste

In a July 17 hearing of the House Energy and Commerce Committee's subcommittee on Communications and Technology and Hea

Pointing out that the committee has jurisdiction over the Office of the National Coordinator for Health IT and the HITE

"It may be time for this committee to take a closer look at the practices of vendor companies in this space given the p

The hearing was focused on how healthcare and technology can accelerate the pace of cures in the U.S., which is an init

Gingrey is not alone in his concern about EHRs' lack of interoperability. The coalition Health IT Now, buoyed by the RA

Another recent study revealed that the current architecture for data exchange required by Stage 2 of the Meaningful Use
```

ONC发布互操作性路线图interoperability roadmap

关于这个10年的互操作性路线图,更多中文信息可参考

1、 【OMAHA】美国联邦政府医疗信息化战略规划2015 - 2020

2、一张图看美国互操作性路线图

3、【国外】ONC发布HIT互操作路线图 助推精准医学计划

某个组的老大如是说"如果我们能够让不同的系统能够互相'说话'，那已经是相当牛逼了" 原文如下

```
"If we just get the systems themselves to talk to each other, that would be a huge accomplishment," Health IT Policy Co

尽管局部范围已经实现了一定程度的互操作性，但大规模的应用还是很大的问题。
while interoperability has increased, widespread interoperability remains a challenge. The report identifies several ba
```

Omnibus bill keeps ONC funding at same level as 2014

Omnibus法案成为导火索

```
The Omnibus Appropriations bill that passed Congress over the weekend will fund the Office of the National Coordinator

Additionally, the bill grants $14.9 million to the Federal Office of Rural Health Policy within the Health Resources ar

Of the total IT budget, $548.34 million is earmarked for IT development, modernization and enhancement.
```

国会敦促ONC ：要专注在互操作性上:

在国会通过2015 omnibus appropriation bill法案的同时提交的报告中，国会很生气，敦促ONC： “”要合理使用你们手中的认证的权利来保障合格的产品能够医疗机构和人民群众带来价值。只给那些满足我们的认证标准的且没有阻碍区域医疗(医疗信息交换)的产品给予授权，对于阻碍信息共享的产品要收回执照。

```
"[U]se its certification program judiciously in order to ensure certified electronic health record technology provides
```

同时在报告里要求ONC90天内提交一份报告，分析一下目前信息阻塞问题的严重程度，要有个大概的厂商、医疗机构的数目以及解决问题的方案。

信息化厂商和医院都是信息阻塞的罪魁祸首

这里就是上面ONC给国会交的作业 作业中指出： 1、医院或厂商通过收费来控制转诊和加强自己的市场占有率 A common charge is that some hospitals or health systems engage in information blocking to control referrals and enhance their market dominance 2、厂商反馈 原因很多啦 比如说HIPAA 但作业中指出 都是借口 和HIPAA压根没多大关联 It has been reported to ONC that privacy and security laws are cited in circumstances in which they do not in fact impose restrictions

作业中给出的解决方案：

1、Strengthen in-the-field surveillance of ONC certified health IT tools, which was proposed in the certification requirements accompanying the Stage 3 Meaningful Use rule 2、Constrain standards and implementation specifications 3、Promote better transparency in certified health IT products and services that would "make developers more responsive to customer demands and help ameliorate market distortions" that lead to vendor information blocking 4、Establish governance rules deterring information blocking, which the report notes is addressed in ONC's interoperability roadmap 5、Work with the U.S. Department of Health and Human Services Office for Civil Rights to ensure healthcare stakeholders understand HIPAA privacy and security standards, particularly those related to information sharing 6、Coordinate with the HHS Office of Inspector General and the Centers for Medicare & Medicaid Services on information blocking as it relates to physician self-referral and the federal anti-kickback statute 7、Refer illegal business practices to law enforcement agencies

8、Work with CMS to provide incentives for interoperability while simultaneously discouraging information blocking 9、Promote competition and innovation in health IT, which the Federal Trade Commission recommended in its comments on the interoperability roadmap

但看了总结就知道 美帝也是各种部门互相扯皮呀，这种东西往后10年看有起色不

```
"While important, these actions alone will not provide a complete solution to the information blocking problem," the re
```

口水仗正式开始

Senate HELP Committee Tuesday 2015年3月17日的组委会例行会议上， 在听证会的问答环节，参议员Senator Tammy Baldwin, D-Wisconsin问 为什么Epic 没有参加CommonWell Health Alliance Epic 的主管互操作性的头Peter DeVault说了如下一段话，引发了CommonWell Health Alliance的反击：

```
"When we were approached by them and asked to join, we were told that it would be multiple millions of dollars for us t
```

同时还不饶人的嘲讽 你丫们只是嘴上说说 雷声大雨点小 CommonWell至今成立2年，目前只涵盖了1000个医生 和Epic的care Everywhere的10万医生简直高下立判

```
DeVault also underscored CommonWell's low participant numbers in its interoperability project so far – which he called
```

CommonWell联盟随后在Healthcare IT News发表了联合声明：

```
We are committed to openness and transparency," the statement read. "Accordingly we publish our services and use case s
```

那么Epic要入会 大概要每年缴纳多少钱呢 根据目前CommonWell会员费和服务费用标准，结合Epic2014年的年收入18亿美元，要掏 125万美元annual subscription fee和 $50,000 to $90,000 in annual membership dues 。

周三晚上 就是3.18日 athenahealth CEO Jonathan Bush在twitter上对Epic CEO Judy Faulkner 进行了高调嘲讽

Cerner. 行业老二也发话了

```
His "rhetoric is a slap in the face to many parties working to advance interoperability," according to a statement rele
```

梗在这里 2013年HIMSS上Epic and CommonWell就有口角

```
At that HIMSS13 announcement, athenahealth CEO Jonathan Bush 说 大家都可以加入哟
emphasized that anyone was invited to CommonWell – even a vendor of "epic proportions."

In a subsequent interview, Healthcare IT News asked McKesson CEO John Hammergren whether that invitation was dangled or

"We'd like them to bite! We want them to bite," said Hammergren. "I'm hopeful that they will see it the same way we see
```

但 Epic CEO说 完全不知情,这帮sb居然想背后地里阴我们，手够黑的啊

```
But Epic CEO Judy Faulkner noted that her company wasn't asked to join before the announcement. "We did not know about
```

对于行业内愈演愈烈的竞争所引发这场闹剧，围观群众也有话说：
1、各位看官和Epic签合同要慎重啊 且要是Epic-to-non-Epic能达到Epic-to-Epic 的90%的功能，"哥就表演吃翔"
2、路人乙嘲讽到Epic到底是如何腆着脸皮说all of our systems can talk to each other
3、入会费是不是有点多
4、Epic does not need to join CommonWell. CommonWell needs to join Epic.
5、CommonWell sounds like a group of union thugs who want to strong-arm their way into other vendors' businesses
6、CommonWell or is it CommonHell? Later seems to be more appropriate.此人称Epic很好合作 方案也性价比高Stop creating mindless standards organizations with a scheme to collect additional revenues from licensing. Healthcare organizations dole out enough on their EMRs to these vendors and they don't need these vendors charging nickel and dime on every petty thing that gives patients control of their health information "我深深的表示赞同"
7、It just doesn't seem right that a private company pulling down $1.8B in revenue should be able to dictate interoperability. The government needs to step in and make sure all of these companies play well in the sandbox. Greed stands in the way of a national clinical data repository that researchers could use to find treatments and cures for diseases that have plagued us for decades. 问题不存在一个厂商上 此人观点有理
8、Joining CommonWell doesn't seem like a great business opportunity if after two years they only have 1000 doctors live on it. Epic seems to understand the fundamentals of sharing patient data if they have 100,000 doctors on their version. My guess is that CommonWell is floundering and they want Epic in there to help them out. If they can also take some digs at Epic for not joining in the meantime - all the better in their mind.

The healthcare market will drive the need for interoperability. I don't think getting the government involved will help anything.

9、What would be interesting to know is how many of Epic's 100,000 physicians are using a product other than something in the Epic family. Epic excels at interoperability between their own products, but what is their track record when connecting to someone outside the family, so to speak? Their numbers may be impressive, I don't know. But I do know that the huge challenge CommonWell has is interoperability across disparate vendors and products.
10、It seems to me that CommonWell is primarily a marketing "gimmick". Aids and standards for inter-operability e.g. LOINC, CLSI standards, and other accepted nomenclatures have existed in the marketplace for some time. We probably have enough standards but many vendors simply pay "lip service" to them. Such a "collaboration" among competitors may be a "surface phenomenon" with relatively little substance.

11、对第10条的回复 CommonWell is a response to Epic's market share and market power by vendors that heretofore did not feel compelled to cooperate with each other on data exchange. Epic dramatically dominates the large HCDO/ Academic Medical Center market and is viewed as almost unstoppable. As Epic conquers the large market, it will target the mid-size market - probably taking advantage of a cloud service at some point in time. This probably best explains why Epic doesn't want to join CommonWell - they understand the strategic alignment and competitive response that CommonWell represents.
12、It does seem a bit childish to see these types of exchanges between behemoths of the industry. However, I do wonder

why Epic would have to sign an NDA to participate in a joint effort to share and propagate openess.

后续报道 Epic 投降了

Epic CEO Judy Faulkner 在每年度最盛大HIT行业会展，也就是2015年的HIMSS15大会上宣布，直到2020年之前 咱大EPIC是不会再收大伙钱了。其实athenahealth和Cerner这俩货也是不久之前宣布他们也不收费

群众观点：
1、up-front predictable cost and a pay-per-use model两种付费模式的区别而已，钱总是会出的
2、市场的不成熟导致没有有效的激励方式来促使信息的无缝免费流动。而能因为厂商收取服务费用就认为是某个厂商作恶。要解决的是到底谁来掏钱的问题
3、不额外收钱的话，，软件的成本、维护费咨询费必然要上涨，羊毛出在羊身上
http://www.healthcareitnews.com/news/epic-latest-drop-fees-data-exchange

思考

国内最早喊互操作性的应该是李包罗老爷子了，从最初引入这样的概念至今大概也有10来个年头了，不管是学术界还是工业界，这些年来沿着的一条路是说想要给复杂的医疗领域定义出一种通用的one-fit-all的能够描述整个医疗领域的模型，进而演化出一种可以用于不同系统间交互数据的通用格式和统一的医疗术语，也就是所谓的互操作性标准。这些年也有人在提学习型的医疗信息化系统的建设，提精准医疗，大喊互联网医疗的口号，但这些目标的实现都离不开数据，离不开鲜活的在不同系统间流动的数据，就像阿里、京东、腾讯、amazon一样，没有各部门间数据的无缝整合，它们是不能提供大家每天都在享受的这些服务的。近几年不管是google还是豌豆荚，都在说一个“应用内搜索”的东西，说白了，就是想把接入各自平台的系统的数据格式标准化，这样子就可以点开具体的应用来搜索查询信息，但似乎也没有什么进展，对于如日中天的互联网应用来讲，对于宇宙第一的google来讲尚且很难突破的东西，反过来看医疗行业，业务只会更复杂，业务系统的成熟度只会更低，似乎互操作性的口号，我们喊得太早了，能从美帝、互联网厂商身上学到的是，未来势必在国内的HIT圈要成长出一两家如Epic这样的公司，在自己封闭的平台中，实现各级医疗机构的信息无缝流通，只有这样的巨无霸，通过自己所收集的患者全生命周期的数据链，利用医疗大数据的应用打造出优质的医疗服务来服务大众，但有没有其他实现互操作性的路径呢?

Epic和IBM的合作不加入是另有所谋

Epic与IBM watson的合作 是否预示着实现互操作性的另一条路 智能的解析各种异构的数据源中的数据。

3、API能否拯救HIT

Jason and the Argonauts http://geekdoctor.blogspot.nl/2014/12/kindling-fhir.html http://xmlmodeling.com/2014/12/jason-argonauts/ http://xmlmodeling.com/2014/10/jason-task-force-final-report/ http://argonautwiki.hl7.org/index.php?title=Main_Page

Apps and APIs: A Positive Step for Patients

OMAHA联盟 后感

http://getmyhealthdata.org/ 请愿书

FHIR标准解读

主要是对FHIR 标准本身中涉及的Restful API的理念、所定义的数据类型、资源数据模型、数据模型扩展机制以及接口定义扩展机制进行探讨

数据类型

数据类型的介绍和示例 数据类型的简化

RestfulAPI

state of art Restful API HTTP API定义的语言和文档生成工具 尝试使用RAML替换profile structureDefinition来定义FHIR 接口生成文档

资源的数据模型

FHIR资源模型与OpenMRS信息模型的比较 FHIR资源模型与OpenMRS信息模型的比较

数据序列化格式

XML中遇到的问题 JSON中遇到的问题 尝试使用Avro作为序列化格式

扩展机制

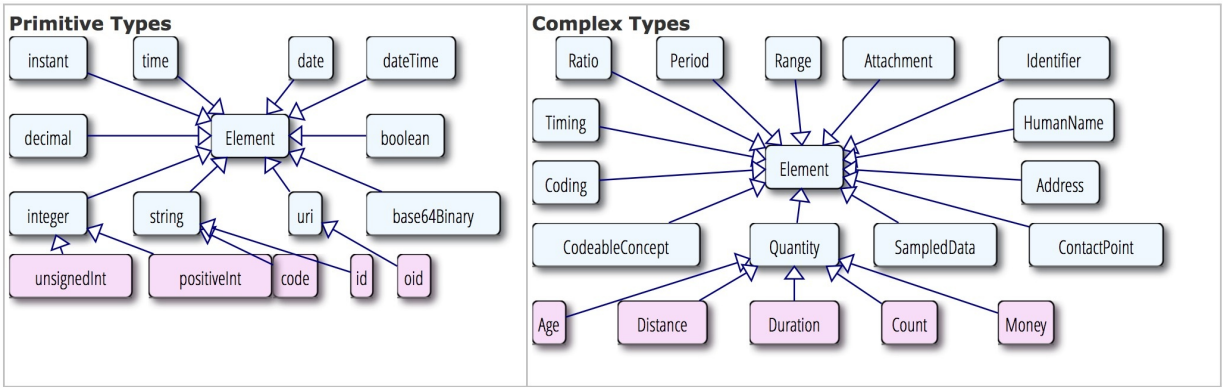
使用扩展来自定义接口

DSTU2中的 operation 框架介绍

FHIR 客户端的职责

数据类型说明和示例

FHIR 标准中规定了两大类数据类型：简单/基本数据类型和复杂数据类型。如下图所示：



White = abstract type. Light blue - types. Light Pink - Profile on Type. The data types are also available as a W3C Schema.

1 基本数据类型

下表总结了在整个规范中基本类型及其简单的限制条件，基本类型指那些没有子属性的类型，尽管它们的原语，像所有类型，都有扩展属性。基本类型的值与W3C架构（1.0）规范的第二部分定义的值具有相同的值域，大多数情况下，本规范附加约束使用粗体标记来。 基本数据类型 名称 类型 描述 boolean xs:boolean 值可以为true或者false（0和1是无效的值） integer xs:int 32位整数（对于更大的值，使用decimal） decimal xs:decimal 一个有理数。注意：在实现中，不要使用IEEE浮点类型。而是使用类似于内置精度的小数的形式 （例如Java BigDecimal）。Decimal不可以使用指数。 base64Binary xs:base64Binary base64以(RFC 4648)编码的字节流 instant xs:dateTime 一瞬间的时间-精确至少到秒，常常包含时区。注意：这种类型是系统时间，不是人类时间。 （如date 和 dateTime） string xs:string 1. 一个Unicode字符序列。注意，字符串大小不能超过1MB uri xs:anyURI 统一资源标识符的参考。它可以是绝对的或相对的，也可能是一个可选的片段标识符（RFC 3986） date union of xs:date, xs:gYearMonth, xs:gYear 在人们交流时时使用的日期或部分日期（比如年或年+月）。没有时区。日期应该是有效日期， 并且是W3C模式类型日期，如gYearMonth，和gYear。 正则表达式：-?[0-9]{4}(-(0[1-9]|1[0-2])(-(0[0-9]|[1-2][0-9]|3[0-1]))?)? dateTime union of xs:dateTime, xs:date, xs:gYearMonth, xs:gYear 在人们交流时时使用的日期，日期时间或部分日期（比如年或年+月）。如果指定小时和分钟，通常 应该加上时区。可以加上秒，也可以不加。日期应该是有效日期，如“24:00”是可以的。 日期是W3C模式类型日期，如gYearMonth，和gYear。 正则表达式：-?[0-9]{4}(-(0[1-9]|1[0-2])(-(0[0-9]|[1-2][0-9]|3[0-1])(T([01][0-9]|2[0-3]) :[0-5][0-9]:[0-5][0-9?(Z|(+|-)((0[0-9]|1[0-3]):[0-5][0-9]|14:00))?)?)?)? time xs:time 一天内的某个时间，没有指定日期（可以转化成自午夜算起的Duration数据类型）。秒可以加上也可以不加上。如“24:00” 是可以的 正则表达式：([01][0-9]|2[0-3]):[0-5][0-9]:[0-5][0-9? 注意：不是任意规定格式的日期类型都可以用正则表达式来表示，这些正则表达式要求这些日期必须是有效格式的日期

限制条件 名称 基本数据类型 描述 code string I表示该值是从其他地方定义的一组控制串（请参阅使用的代码进行进一步讨论）。从技术上讲， 一个code类型被限制为字符串，它至少有一个字符，开头或结尾都没有空格，内容中不能出现连续的空格符 正则表达式：\s+([\s]+\s+)* oid uri 用URI（RFC3001）表示的OID，如: urn:oid:1.2.3.4.5 uuid uri 用URI（RFC4122）表示的UUID：urn:uuid:a5afddf4-e880-459b-876e-e45 91b0acc11。RFC的注解：UUID值应使用小写字母，但系统应不区分大小写。 id string 任意大写或小写ASCII字母的组合（'A'..'Z', 和'a'..'z',数字('0'..'9'), '-' 和 '.',限制在64位字符长度 以内。这些可能是整型，一个无前缀的OID, UUID或者满足这些约束条件的任何其他标识模式。） Ids区分大小写。UUIDs必须用小写字母。 注意：这个版本格式通常使用ISO18232标识符格式，也可以使用除ISO18232以外不区分大小写的 标识符格式

除了具有如上所述的值外，这些基本数据类型也可以具有一个身份（例如XML：ID），并且它们可以具有像其他资源资源的任何元素一样的扩展性。需要注意的是，根据以下所有元素的标准规则，该值是可选的，并且可能不存在。例如，一个基本元素可能是没有价值的，包括数据缺失等 在XML中，这些数据类型被表示为XML元素的值类型和值属性。元素的名称和类型同时被定义。 XML元素可以有一个id属性，子元素命名为“extension”。根据XML模型，这些类型的值属性的前后空格都可以省去，包括boolean, integer, decimal, base64Binary, instant, uri, date, dateTime, oid, and uri。这意味着，当读取XML实例时，使用XML模型和不使用XML模型最后得到的值的属性是不同的。因此，这些类型的值属性前缀和后缀不应该有空格。空

格只能出现在字符串中。 在JSON中，这些数据类型代表包含它们的对象的简单属性。该属性名称和类型需要同时定义。该类型仍然有一个id属性和扩展。怎样通过JSON格式来描述这些属性。JSON中空格是很有意义的。基本数据类型的前缀后后缀不能有空格。 7.2 Attachment 附件 该数据类型适用于包含或引用各种附件，也就是用其他格式所定义的外部的数据内容。常见的使用这种类型的，在一些报表格式，如PDF包含图像或报告。然而，它可以用于具有mime类型的任何数据。

Structure Name Flags Card. Type Description & Constraints 描述和限制条件 Attachment I 1..1 Element 按照一定格式定义的内容，若存在data字段，它应具有一个contentType类型 contentType 0..1 code Mnme类型的内容，字符集等，编码取自MimeType (Required) language 0..1 code 人类语言，编码取自Language (Required) data 0..1 base64Binary 内嵌的base64编码的数据 url 0..1 uri 通过uri可找到数据 size 0..1 integer 内容的字节数 hash 0..1 base64Binary 数据的hash值(sha-1, base64ed) title 0..1 string 数据的标签 ContentType元素应始终填充。根据情况，它可以包含字符集信息和其他MIME类型扩展。在contentType中如果没有设置字符集，那么操作中会出现问题，但有些媒体类型可以使用默认的字符集，字符集可以通过内容的检查来确定。 附件的实际内容可以直接使用数据元素或者可以提供一个URL引用来表示。如果两者都设置，则引用指向的内容应该和数据的内容一致。引用不能被重用，即同一个引用指向一些不同的数据（即参考特定版本）。该URL引用应指向解析为实际数据的位置;一些URI如CID：满足这一要求。如果URL是相对引用，对于同一个资源引用，它可以理解为指向同样的路径。 应可以根据hash值来验证URL返回的内容是否被改变。 在很多使用附件的情况下，该附件基数大于1。有效重复利用可以传达使用不同MIME类型和语言的相同内容。关于重复利用元素的意义说明应该在该重复资源元素定义或指定该类型扩展引用的时候提供。对附件的语言描述使用的是BCP47编码。 约束条件 Inv-1:如果附件存在data字段，它必须有一个contentType类型 (xpath: not(exists(f:data)) or exists(f:contentType)) 若data 和url字段都不存在，value字段中应表明不存在规定的mimeType或语言的数据 使用时可以规定附件的格式和类型，也就是可以使用的mime types. Attachment 常用于: Communication, Practitioner, QuestionnaireAnswers,CommunicationRequest, Observation, SupportingDocumentation, RelatedPerson, Person, Media, DiagnosticReport,Contract and Patient 7.4 Coding 编码 编码是使用“编码系统”里面定义好的一些符号来表示和定义相关概念 Structure Name Flags Card. Type Description & Constraints 描述和限制条件 Coding I 1..1 Element 对定义编码的术语的引用，如果有提供ValueSet，Coding.system字段必须赋值。system 0..1 uri 术语系统的标识符 version 0..1 string 术语系统的版本号 code 0..1 code 由该编码系统所定义的语法符号 display 0..1 string 由该编码系统所定义的文本表示方法 primary 0..1 boolean 如果该编码是否是用户优选项的标记 valueSet 0..1 ValueSet 该编码取自哪个值集

编码的含义是由代码来定义。系统提供代码定义的源，以及一个可选的参考版本。display是human display，用于由系统定义的文本 – 它没有加入其他的值。该值对内容中选择的代码信息提供参考。 System字段的值是所引用的字典/代码系统的URI。该URI可能： • 一个已经存在的系统列表中定义的URI • 一个已经在HL7 OID注册过的OID (urn:oid:) 或UUID (urn:uuid:)。。 • 一个直接引用该系统定义的URL，它可以参考部分ValueSet资源的代码系统（即在ValueSet.define.system的值） • 或者唯一标识代码系统定义的任何其他URI 对于某个字典/编码体系而言，System的正确取值可以结合按照下面的顺序： • 标准中已经罗列出来的字典列表 • HL7 OID注册库 • 和字典/编码系统有关的文档 • 咨询代码系统的所有者 • 在HL7词汇邮件列表中提问

有时候可能会提供代码系统的版本。如果编码的含义在代码系统中不同的版本一致，则可不提供系统版本。当编码系统的不同版本不能保持语义的一致性时，必须提供version字段。如果引用的是一个值集合，并且该值集合不是由其他编码系统中的编码所形成的，而是它自身定义了一些编码，一定程度上，这个值集也就是一个简单的编码系统，如果在该值集定义编码时指定了version字段的取值，则两个version字段(ValueSet.define.version和Coding.version)的值应保持一致。请注意使用以下系统必须给version字段赋值： • LOINC编码系统 • 不同版本的ICD • 某些国家发布的SNOMED CT（在不同行政辖区之间，定义的一致性会不同，甚至一些行政辖区会指定他们自己的规则）

如果使用了code字段，编码必须是原始系统所定义的符号。在某些编码系统如SNOMED CT，可能是其他一些预定义的符号（如post-coordination）组成的表达式。需要注意的是，除非代码系统指定，否则区分大小写。display 是由系统定义代码的文本表示，在不了解系统的情况下通过一个应用来展示代码的含义。 如果编码系统定义了多个展示用的字符串，至少在display字段中应该使用其中一个。如果某个被标记为优选项，它应被优先使用。如果编码系统中没有定义文本表示（例如SNOMED CT的表达式），那么display字段不能有值，不能理解编码的表达式的系统也无法获取编码的含义。 在一些情况下，可能不知道出自哪个字典system - 只知道code编码。在这种情况下，除非该系统可以由上下文安全地判断，否则不能对编码进行处理。信息在更大范围内共享是可预见的，不能在不知道字典的情况下使用code编码，应尽可能避免出现这种做法。 如果知道是哪个字典，但没有编码，可以理解为，在系统中没有合适的代码来表达这个含义。 如果两个编码system、version 和code 字段值都相同，则两个编码表达的是相同的含义。如果version不存在，或者system, version和code取值不同，这两个编码的关系需要在编码系统中和已有的映射关系中查找。 如果用户在界面中选择了某个特殊的编码值，可以将其标记为主编码，在进行翻译等操作的时候，主编码是优先考虑的。 有时候会在valueset字段中提供对某个valueset的引用值，以帮助用户或系统来理解编码的上下文。在某些情况下，编码取自哪里会影响着编码的含义。这里需要注意的是

Coding.system字段的值并不能替代Coding.valueSet。Coding.system字段中URI的值不能指向一个valueset的引用。（如果value set中的编码是它自己所定义的，那么该使用ValueSet.define.system而不是Coding.system字段，并且这个value set包含对这个valueset资源的直接引用。 约束条件 如果使用了Coding.valueSet字段,则Coding.system字段必须赋值，且其值与ValueSet.define.system 或ValueSet.compose.include.system一致。

在以下内容中使用Coding类型 CodeableConcept, ValueSet, OralHealthClaim, Coverage, Composition,Conformance, Profile, StatusResponse, PharmacyClaim, Reversal, EligibilityRequest, QuestionnaireAnswers,PaymentReconciliation, ProfessionalClaim, OperationDefinition, ClaimResponse, ExplanationOfBenefit,SupportingDocumentation, InstitutionalClaim, EligibilityResponse, StatusRequest, VisionClaim, Readjudicate,PaymentNotice, Questionnaire, OperationOutcome, VisionPrescription, ExtensionDefinition, ImagingStudy,Provenance, MessageHeader, DataElement, SecurityEvent, PendedRequest, EnrollmentRequest, Contract andEnrollmentResponse 设计说明：本规范定义了两种数据类型来表示编码值： • Coding:简单的引用其他编码系统中所定义的编码值 • CodeableConcept: 文本描述和或一组编码列表（如系统定义的一组编码引用） Coding 针对的是最简单的情况，但在FHIR中并不常用。经验告诉我们，在一般的情况下，系统需要的编码中包括了多种翻译值和或原始文本。 如果确定值必须取自某些已经选定的编码，Coding数据类型可以直接使用。 7.5 CodeableConcept CodeableConcept所表示的值，常常可以通过所提供对一个或多个术语或本体的引用来得到(也就是这个值含义是由该术语、本体所定义的)，也可以通过所提供的文本来定义。这是在医疗数据是很常见的。 Structure Name Flags Card. Type Description & Constraints CodeableConcept I 1..1 Element Concept——所引用的术语或仅仅只是文本 用户可直接选择一组编码中的一个 coding 0..* Coding 通过术语系统所定义的编码 text 0..1 string 这一概念的纯文本表示 每个coding都是concept的一种表达方式。同样的concept可以在不同的系统中编码多次（甚至在相同的代码系统中多次编码，其中多个形式是可能的，例如在SNOMED CT中）。由于编码规则的差异，不同coding可以具有轻微不同的粒度。在CodeableConcept中Coding的排序是没有意义的。一个典型的使用CodeableConcept场景是当发送一个概念的非标化编码，与此同时，该概念可以转换到多个标化字典例如LOINC或SNOMED CT。发送本地化编码在调试和完整性审计中是非常有用和重要的。 不管coding 字段是否存在，text都是用户输入或选择的概念的表达方式，而且最能代表用户或概念的本意。很多时候，text和coding.display两个字段值是一样的。其中一个Codings可能被标记为主编码-也就是用户直接选择的那个code或concept。当没有coding元素被标记为主编码，优先选择text（如果存在）的含义。 约束条件 • Inv-2: On 一组编码中，只能有一个编码是由用户直接选择的 • (xpath:count(f:coding[f:primary/@value='true'])<=1) CodeableConcept在以下内容中使用: Condition, Supply, DeviceComponent, Communication, Group,Appointment, Slot, Contraindication, EpisodeOfCare, Composition, Conformance, NamingSystem, HealthcareService,OrderResponse, Practitioner, CarePlan, ClinicalAssessment, Substance, DeviceUseRequest, Schedule,ImagingObjectSelection, CommunicationRequest, RiskAssessment, Observation, AllergyIntolerance, RelatedPerson,Alert, ProcedureRequest, DeviceMetric, Organization, ImmunizationRecommendation, MedicationDispense,MedicationPrescription, MedicationStatement, AppointmentResponse, Media, Other, VisionPrescription,DocumentReference, Immunization, Provenance, Device, Order, Procedure, DiagnosticReport, Medication,MessageHeader, DataElement, DocumentManifest, MedicationAdministration, Encounter, SecurityEvent, List,DeviceUseStatement, NutritionOrder, ReferralRequest, FamilyHistory, Location, Contract, Basic, Specimen, Patient,CarePlan2 和DiagnosticOrder 7.6 Quantity 测定量（或可以潜在的被测量的量）。 Structure Name Flags Card. Type Description & Constraints Quantity I 1..1 Element 一个测量过的或可测量的量。 如果使用了code字段，也应该使用system字段 value 0..1 decimal 数值（隐式精度） comparator M 0..1 code < | <= | >= | > -如何理解这个value，值只能取自字典QuantityComparator（必需） units 0..1 string 单位 system I 0..1 uri 定义可编码的单位形式的字典 code 0..1 code 单位的可编码值 value 字段值包含了物理量的数值，包括一个隐含的精度。如果没有指定comparator的值，该值是一个点（即“=”）。不能忽略comparator 字段的值 units字段值包含你所测量内容的单位对应的文本表示,。后续也可以使用code和system字段来进一步对其进行编码 如果该units字段值能够使用UCUM进行编码，且存在code字段的话，code字段值必须是一个UCUM 编码。如果code字段值必须是一个UCUM 编码，我们可以得到一个可以在不同物理量之间进行比较的标准值。需要注意的是，通常情况下 units 字段值包含了UCUM单位的文字表达方式(如US$ 、US$)，但units字段值到底是不是一个有效的UCUM单位则需要进一步验证。 约束条件 • Inv-3: 如果code字段存在，则system字段也必须存在 (xpath: not(exists(f:code)) or exists(f:system)) 此类型的使用常常定义数量是多少，单位是什么。进一步也可以对单位进行编码 Quantity在以下内容中用到: Range, Ratio, SampledData, Supply, Group, OralHealthClaim, PharmacyClaim,CarePlan, Substance, QuestionnaireAnswers, ProfessionalClaim, Observation, InstitutionalClaim, VisionClaim,MedicationDispense, MedicationPrescription, MedicationStatement, VisionPrescription, Immunization, Medication,MedicationAdministration, NutritionOrder, Contract and Specimen 7.6.1 Quantity数据类型的变种 在这些资源内容模型中用到了Quantity数据类型，并对其做了一些限制： Age 时间长度（时间长度），用UCUM编码表示 Profile (XML, JSON) Count 离散元素的数量量（没有单位） Profile (XML, JSON) Money 钱的数量 Profile (XML, JSON) Distance 距离的长短 Profile (XML, JSON) Duration 时间长度 7.7 Range 范围 一组有序的数量值，有上限和下限 Range规定了一组允许值；通常，可以取范围内的一个值(例如：患者可以服2-4片药)。Range通常用于指示。 Structure Name Flags Card. Type Description & Constraints Range I 1..1

Element 伴有上下限的值的集合。下限的值不能大于上限，Quantity中不能有comparator字段 low I 0..1 Quantity 下限 high I 0..1 Quantity 上限

low和 high字段中units 和code/system元素值应一致。如果low或 high字段不存在，意思是上限或下限是未知的，因此都不是完整的范围。

在low和 high字段中不能存在comparator字段。注意，对于超出某个范围的度量值，应该使用带comparator的quantity类型来表示，而不是使用Range类型

low和 high字段值是闭区间，并且被假定为具有任意高的精度。例如范围1.5至2.5，包括1.50，2.50，但不包括1.49或2.51。约束条件 • Inv-2: 如果存在，low的值应比low小（(xpath: not(exists(f:low/f:value/@value)) or not(exists(f:high/f:value/@value)) or (number(f:low/f:value/@value) <= number(f:high/f:value/@value)))） • Inv-3: low和 high字段值都不能有comparator字段 Range 用于: Group, RiskAssessment, Observation, NutritionOrder 和 FamilyHistory。

7.8 Ratio比率 两个数量之间的关系，表示成分子和分母 Structure Name Flags Card. Type Description & Constraints Ratio I 1..1 Element 两个数量值的比- 分子和分母 numerator 和denominator字段要么同时存在，要么不存在 numerator 0..1 Quantity 分子值 denominator 0..1 Quantity 分母值 在分子和分母的共同因素是不会自动抵消。该比率的数据类型用于滴速（例如，“1：128”）和其他用比率来表示的实验室检验结果。比率不是简单的“结构化的数字” - 例如血压测量（例如，“120/60”）并不是比率。另外，比率所使用的地方，在分子和分母共同因素不抵消。这样做的最常见的例子是，其中的比率代表一个单位成本，而分子是一种货币（例如，50/10美元）。 一个适当的比率同时具有分子和分母;然而，这些都不是强制性的，可以使用扩展来表示一些额外的信息。 约束条件 • Inv-1:: numerator和denominator字段要么都存在，要么都不存在(xpath: count(f:numerator) = count(f:denominator)) 具体使用时分子或分母可能需要特定类型的Quantity。 Ratio 常用于: Substance, Observation, MedicationDispense, MedicationPrescription, MedicationStatement, Medication, MedicationAdministration and NutritionOrder

7.9 Period 时期 由开始和结束日期/时间所定义的时间段。 Period规定了时间间隔。具体使用时指定是否整个范围内都适用（例如，“病人在这段时间内住院”）或一段时间内的一个值（例如“2013年6月24日下面2点到4点给病人服药“）。 Structure Name Flags Card. Type Description & Constraints Period I 1..1 Element 时间范围内开始和结束日期/时间确定 如果存在，start的值应小于end的值 start I 0..1 dateTime 开始日期时间，闭区间 end I 0..1 dateTime 闭区间 截止日期时间 如果不存在start字段，该时间间隔的开始日期是未知的。如果end不存在，这意味着该时间间隔仍在进行中。 End的值包括了日期时间在内.比如2011-05-23 to 2011-05-27 包括了23和27号全天的时间。 Period常用于: Identifier, Supply, Coverage, EpisodeOfCare, Composition, NamingSystem, Practitioner, CarePlan, DeviceUseRequest, Schedule, PaymentReconciliation, RiskAssessment, Observation, RelatedPerson, ProcedureRequest, MedicationDispense, MedicationPrescription, MedicationStatement, DocumentReference, Provenance, Procedure, DiagnosticReport, MedicationAdministration, Encounter, PendedRequest, DeviceUseStatement, NutritionOrder, ReferralRequest, FamilyHistory, Contract, Specimen, Patient and CarePlan2

7.10 SampledData采样数据 由设备采集得到的一系列的值，有上限和下限。数据的维度可能不止一个。 A SampledData provides a concise way to handle the data produced by devices that sample a physical particular state at a high frequency. A typical use for this is for the output of an ECG or EKG device. SampledData类型提供了一个简洁的方式来处理由设备产生的对物理状态进行高频采用的数据。一个典型的使用便是心电图或心电图设备的输出。 Structure Name Flags Card. Type Description & Constraints SampledData 1..1 Element 设备所采集的一系列值 origin 1..1 Quantity 初始值和单位 period 1..1 decimal 采样间隔 factor 0..1 decimal 倍数，在把数据加到初始值上之前应该乘以多少倍 lowerLimit 0..1 decimal 能够检测的下限 upperLimit 0..1 decimal 能够检测的上限 dimensions 1..1 integer 每个时间点采样的个数(这里应该是频率) data 1..1 string 十进制值用空格分隔或“E”|“U”|“L” 一组由空格（Unicode字符U20）分隔的十进制值。除了十进制值之外，也可使用特殊值“E”（错误），“L”（低于检测限）和“U”（高于检测限）。如果多个维度互相交织在一起，则某个时间点的所有数据表示在一起。默认的倍数值factor字段为1。 SampledData类型常用于: Observation

7.11 Identifier标识符 在一个系统内，与一个单独的对象或实体相关联的数字或字母组成的数字串。典型地，标识符在资源中被用来将内容连接到其他框架或协议表示的外部内容。标识符与对象相关联，并且会随着人为或系统的处理和错误被改变或不再使用。 Structure Name Flags Card. Type Description & Constraints Identifier 1..1 Element 用于计算的标识符 use M 0..1 code usual | official | temp | secondary (If known) 标识符的用途，值取自字典IdentifierUse (Required) label 0..1 string 标识符的说明 system 0..1 uri 标识符的命名空间 value 0..1 string 标识符的唯一值 period 0..1 Period 标识符有效的时间区间 assigner 0..1 Organization 标识符的分配机构 system字段的值指向标识符如何定义的URI(怎么样确保标识符的唯一性等

等)。它可能是一个特定的应用程序或一个公认的标准/规范的所定义的唯一标识符。Value字段值在system定义的范围内应是唯一的，并具有一致的含义。system 和 value字段值都是大小写敏感的。

FHIR直接定义了一些有用的URI。如果内容是共享的或跨越机构边界交换，应该在HL7 OID注册库或任何公开的注册库中登记OIDs (urn:oid:) 和UUIDs (urn:uuid:)。如果标识符本身自然是全球唯一的（例如，一个OID，一个UUID，或没有尾随部分URI），那么system的值应该取“urn:ietf:rfc:3986”。

在某些情况下，可能不知道system的值 – 只知道value 字段值(例如：它是一个可以扫条形码的简易装置) 或该系统默认是知道的(在有限的情况下简单的交流，往往是由条码阅读器驱动)。在这种情况下，除非该系统可以根据上下文安全地推断出system的值，否则不能对value的值进行任何匹配。若信息在更大范围内共享，应避免这种做法，没有system值的values值实用性有限有地限于在使用中。

The assigner is used to indicate what registry/state/facility/etc. assigned the identifier. assigner 字段表示是什么注册中心、国家、机构/等负责分配标识符。 Identifier常用于: Condition, Supply, DeviceComponent, Communication, Group, ValueSet, OralHealthClaim, Coverage, Appointment, Slot, Contraindication, EpisodeOfCare, Composition, Profile, HealthcareService, OrderResponse, StatusResponse, PharmacyClaim, Reversal, Practitioner, CarePlan, Substance, DeviceUseRequest, Schedule, EligibilityRequest, QuestionnaireAnswers, PaymentReconciliation, ProfessionalClaim, ClaimResponse, CommunicationRequest, RiskAssessment, Observation, AllergyIntolerance, ExplanationOfBenefit, SupportingDocumentation, RelatedPerson, InstitutionalClaim, Alert, EligibilityResponse, StatusRequest, Person, ProcedureRequest, VisionClaim, DeviceMetric, Organization, Readjudicate, ImmunizationRecommendation, MedicationDispense, MedicationPrescription, PaymentNotice, MedicationStatement, AppointmentResponse, Questionnaire, Media, Other, VisionPrescription, DocumentReference, Immunization, ExtensionDefinition, ImagingStudy, Device, Order, Procedure, DiagnosticReport, DataElement, DocumentManifest, MedicationAdministration, Encounter, SecurityEvent, PendedRequest, List, DeviceUseStatement, Goal, NutritionOrder, ReferralRequest, FamilyHistory, EnrollmentRequest, Location, Contract, Basic, Specimen, EnrollmentResponse, Patient, CarePlan2 and DiagnosticOrder

7.12 HumanName 人名 一个人的名称可以是文字、不同的部分和用途信息。

名称可以改变或弃用。一个人在不同的场景下可以有名字。名称可以分割成不同的部分。对于人名，不同部分可以有一些暗含的意义，也可能没有；各种文化对于不同部分的重视程度各不相同。 Structure Name Flags Card. Type Description & Constraints HumanName 1..1 Element 一个人的名字 – 不同部分和用途 use M 0..1 code usual | official | temp | nickname | anonymous | old | maiden 姓名的用途，编码取自NameUse (Required) text 0..1 string 全名的文本表示 family 0..* string 姓（通常被称为'姓'）英文里的Family name / Surname

given 0.. *string* 名，包括中间名，英文里的*Given names (not always 'first').* 包括*middle names prefix 0..* string 名字的前缀 suffix 0..* string 名字的后缀 period 0..1 Period 名字在用活會用的时间区间

下表总结发现一个人的名字中又哪些常见的部分： 名称 示例 备注 Surname、姓 Smith Family Name First name、名 John Given Name Title、头衔 Mr Prefix Middle Name、中间名 Samuel Subsequent Given Names Patronymic bin Osman Family Name Multiple family names复姓 Carreño Quiñones Family Name. See note below about repeats Initials Q. Given Name as initial ("." recommended) Nick Name Jock Given name, with Use = common Qualifications PhD Suffix Honorifics Senior Suffix

text字段规定了整个姓名的文本表达方式。可以是整体或以不同部分的形式来表示。姓名的不同部分之间不应该包含空格。对于姓名中的姓，带连字符的姓名如“Smith-Jones”认为是一个名称，但带空格的如“Smith Jones”的名字则可分解成多个部分。对于姓名中的名而言，如果没有记录全名的话可以用首字母代替。其中given 类型的顺序是有意义的，必须记录其顺序，至于姓和名顺序的使用则取决于文化和使用环境。跨文化系统一般在呈现时应依赖text字段的值，并使用不同的部分来做索引/搜索功能。

在更新姓名时，系统应保证text和不同部分保持一致，或者之选其中一种。若系统不支持原始数据中的姓名的多个不同部分的话，可以使用空格将其拼接起来放在text字段中。

HumanName 常用于: NamingSystem, Practitioner, RelatedPerson, Person, Organization and Patient

7.13 Address地址 Address有多种世界各地的界定通信地址格式。通信地址通常用于记录可以访问找到的患者或人的位置。

Structure Name Flags Card. Type Description & Constraints Address 1..1 Element 通信地址 use M 0..1 code home | work | temp | old - purpose of this address 通信地址的用途，编码取自AddressUse (Required) text 0..1 string 地址的文本表示 line 0..* string 街道名称，门牌号，方向和P.O.盒等 city 0..1 string 城市名，镇等 state 0..1 string 国家之下第二级单位(州、省) postalCode 0..1 string 邮政编码 country 0..1 string 国家（可能是ISO31663字母代码） period 0..1 Period 曾用地址或现用地址的有效日期 当地址是时间段/在使用 text元素表示的是整个地址。可以是整体或以不同部分的形式来表示。在更新地址时，系统应保证text和不同部分保持一致，或者之选其中一种。 Address常用于: Practitioner, RelatedPerson, Person, Organization, Location and Patient

7.14 ContactPoint 联络方式 一个人或组织的各种联络方式，包括电话，电子邮件等详细信息。 Structure Name Flags Card. Type Description & Constraints ContactPoint I 1..1 Element 联系方式。value字段存在的话，system必须存在 system I 0..1 code phone | fax | email | url 联系方式的通讯方式，编码取自ContactPointSystem (Required) value 0..1 string 联系方式的详细信息 use M 0..1 code home | work | temp | old | mobile - purpose of this contact point 联系方式的用途，编码取自ContactPointUse (Required) period 0..1 Period 曾用或现用联系方式的时间区间

如果联系方式是电话、传真或其他类似的方式，value值的格式应遵循 ITU-T E.123. 由于遗留数据和采集方式的不同，这通常是难以实现的 约束条件 • Inv-2: value字段存在的话，system必须存在. (xpath: not(exists(f:value)) or exists(f:system))

ContactPoint常用于: ValueSet, Conformance, NamingSystem, Profile, HealthcareService, ConceptMap, Practitioner, OperationDefinition, RelatedPerson, Person, Organization, ExtensionDefinition, Subscription, Device, MessageHeader, DataElement, SearchParameter, Location and Patient

7.15 Timing Timing规定了可重复出现多次的某个事件，Timing schedule不适用于记录已经发生的事件，而是用于记录那些即将发生或要求发生的事件。Timing schedule既可以是说明了起止日期时间的事件列表，也可是说明了重复条件的单个事件，也可以是只是重复条件而没有具体的事件。 Structure Name Flags Card. Type Description & Constraints Timing I 1..1 Element timing schedule规定可能会发生多次的事件。如果没有event元素的话，也可以只包含repeat元素值

event 0..* Period 事件发生的时间 repeat I 0..1 Element event元素出现一次以下时才能使用该字段。 最多只能出现一个frequency或when元素，但二者不能同时出现 最多只能出现一个count或end元素。 frequency I 0..1 integer 时间段内事件发生的频率 when I 0..1 code HS | WAKE | AC | ACM | ACD | ACV | PC | PCM | PCD | PCV – 能够决定timing的那些所发生的日常生活，编码取自EventTiming (Required) duration I 1..1 decimal 每次重复应该持续的时间长度，应为正值 units 1..1 code s | min | h | d | wk | mo | a –时间的单位，编码取自UnitsOfTime (Required) count I 0..1 integer 待重复的次数 end I 0..1 dateTime 重复停止的截止日期时间 如果指定的事件，每个时间必须要有一个event.start字段值。如果event.end没有值，认为该时间持续了一段时间，但不知道何时停止。 如果timing包含了重复条件，在每个规定的时间间隔或与某个现实生活的事件关联起来，事件会重复发生多次。如果事件重复的话，可以规定其停止时间，可以通过时间重复的次数或者停止事件安排的截止日期时间来实现。如果没有指定截止日期时间，timing schedule会根据其他地方的一些条件停止。 约束条件 • Inv-1: event元素出现一次以下时才能使用repeat字段。 (xpath: not(exists(f:repeat)) or count(f:event) <2) •="" Inv-2:="" 对于 Timing.repeat:="" 最多只能出现一个frequency或when元素，但二者不能同时出现="" (xpath="" on="" f:Timing="" f:repeat:="" exists(f:frequency)="" !="exists(f:when))" Inv-3:="" 对于="" Timing.repeat:="" 最多只能出现一次count或end元素="" not(exists(f:count)="" and="" exists(f:end)))="" Inv-4:="" Timing.repeat.duration:="" duration应取正值="" f:repeat="" f:duration:="" @value=""> 0 or not(@value))

Timing常用于: CarePlan, DeviceUseRequest, ProcedureRequest, DeviceMetric, MedicationDispense, MedicationPrescription, MedicationStatement, Order, DeviceUseStatement and NutritionOrder

7.16 Open Type Element开放式元素

一些元素中并没有规定使用那种具体的数据类型.该类型由通配符 “*”表示。在这些情况下，数据类型可以是下列之一： integer decimal dateTime date instant time string uri boolean code base64Binary Coding CodeableConcept Attachment Identifier Quantity Range Period Ratio HumanName Address ContactPoint Timing Reference

元素名称为“[X]”结尾，可以用具体的数据类型的名称来替换“[X]”。 : DataElement 开放式元素常用于： DataElement 7.17 Other Types 其他类型

下列类型被定义为数据类型的部分，但在本说明书别处记载：

• Resource -资源 - 所有资源的概念基类 • Reference –引用 -从一个资源到另一个 • Extension -扩展 - 用来表达更多的数据 • Narrative –叙述性文本 – 表示资源内容中人可读的文本

1.17.1 数据类型的示例

这一部分包含了如下数据类型的示例

1.17.1 基本数据类型

布尔值：

```
<active value="true" />
```

负整数：

```
<score value="-14" />
```

高精度的小数：

```
<pi value="3.14159265358979323846264338327950288419716939937510" />
```

base64编码的字节流：

```
<data value="/9j/4...KAP//Z" /> <!-- covers many lines -->
```

unicode编码的字符串：

```
<caption value="Noodles are called ?? in Chinese" />
```

表示网站地址的uri:

```
<reference value="http://hl7.org/fhir" />
```

urn格式的uri:

```
<id value="urn:isbn:0451450523" />
```

出生日期:

```
<date value="1951-06-04" />
```

出生年月:

```
<date value="1951-06" />
```

临床文档创建的时间，包括时区:

```
<instant value="2013-06-08T10:57:34+01:00" />
```

精确到毫秒的UTC格式的临床文档创建时间:

```
<instant value="2013-06-08T09:57:34.2112Z" />
```

下午2点35分:

```
<time value="14:35" />
```

1.17.1.2 String Patterns

表示HL7的根oid的uri：

```
<root value="urn:oid:2.16.840.1.113883" />
```

uuid格式的uri:

```
<id value="urn:uuid:a5afddf4-e880-459b-876e-e4591b0acc11" />
```

A code:

```
<code value="acq4+acq5" />
```

带单空格的code:

```
<code value="Question 4b" />
```

数字型id:

```
<id value="314" />
```

字母型id:

```
<id value="alpha-gamma-14" />
```

1.17.1.3 Attachment

See also Base Definition, Detailed Descriptions and Mappings.

PDF文档:

```
<document>
  <contentType value="application/pdf" />
  <language value="en" />
  <data value="/9j/4...KAP//Z" /> <!-- covers many lines -->
  <title value="Definition of Procedure" />
</document>
```

```
document : {
  contentType :  { value : "application/pdf" },
  language : { value : "en" },
  data :  { value : "/9j/4...KAP//Z"},
  title :  { value : "Definition of Procedure" }
}
```

WADO协议的DICOM图像:

```
<image>
  <contentType value="application/dicom" />
  <url value="http://10.1.2.3:1000/wado?requestType=WADO&amp;wado_details..." />
  <hash value="EQH/..AgME" />
</image>
```

1.17.1.4 Identifier

See also Base Definition, Detailed Descriptions and Mappings.

Examples

系统主键:

```
<identifier>
  <use value="official" />
  <system value="urn:oid:2.16.840.1.113883.16.4.3.2.5" />
  <value value="123" />
</identifier>
`
```

院内的病人编号:

```
<identifier>
  <use value="official" />
  <system value="http://www.acmehosp.com/patients" />
  <value value="44552" />
```

```
    <period>
      <start value="2003-05-03" />
    </period>
  </identifier>
`
```

In this case, the period is used to track when the identifier was first assigned to the patient.

FHIR服务器中的患者标识:

```
  <identifier>
   <system value="urn:ietf:rfc:3986" />
   <value value="http://pas-server/xxx/Patient/443556" />
  </identifier>
```

This is not a resource reference - it's a logical reference by the patient identifier.

A UUID:

```
  <identifier>
    <use value="temp" />
    <system value="urn:ietf:rfc:3986" />
    <value value="urn:uuid:a76d9bbf-f293-4fb7-ad4c-2851cac77162" />
  </identifier>
`
```

UUIDs are often used for temporary identifiers, though this is not necessary.

A US SSN:

```
  <identifier>
    <use value="usual" />
    <label value="SSN" />
    <system value="http://hl7.org/fhir/sid/us-ssn" />
    <value value="000111111" />
  </identifier>
`
```

Notes:

- US SSNs are often presented like this: 000-11-1111, the dashes are for presentation and should be removed, as specified in the definition of ssn-us
- The use of "usual" means that this institution prefers to use SSN when identifying the patient (如果医院使用身份证号来标识患者 是否也是usual;如果院内有自己唯一的主索引，身份证号是用usual还是official)

表单编号或病案号:

```
  <identifier>
    <use value="usual" />
    <label value="MRN" />
    <system value="urn:oid:0.1.2.3.4.5.6.7" />
    <value value="2356" />
    <period>
      <start value="2009-07-05" />
    </period>
  </identifier>

`
```

1.17.1.5 Coding

See also Base Definition, Detailed Descriptions and Mappings.

Examples

头痛的ICD-10编码:

```
  <code>
    <system value="http://hl7.org/fhir/sid/icd-10" />
    <code value="G44.1" />
  </code>
`
```

A SNOMED CT expression:

```
  <problem>
    <system value="http://snomed.info/sct" />
    <code value="128045006:{363698007=56459004}" />
  </problem>
```

1.17.1.6 CodeableConcept

See also Base Definition, Detailed Descriptions and Mappings.

Examples

SNOMED-CT编码的头痛，转换成ICD-10编码:

```
  <concept>
    <coding>
      <system value="http://hl7.org/fhir/sid/icd-10" />
      <code value="R51" />
    </coding>
    <coding>
      <system value="http://snomed.info/sct" />
      <code value="25064002" />
      <display value="Headache" />
      <primary value="true" />
    </coding>
    <text value="general headache" />
  </concept>
`
```

本地编码的剂型单位，UCUM中找不到对应编码:

```
  <unit>
    <coding>
      <system value="urn:oid:2.16.840.1.113883.19.5.2" />
      <code value="tab" />
      <display value="Tablet" />
    </coding>
    <coding>
      <system value="http://unitsofmeasure.org" />
    </coding>
  </unit>
```

A SNOMED CT expression(由于该SNOMED-CT表达式不存在display，所以没有display元素):

```
  <diagnosis>
    <coding>
      <system value="http://snomed.info/sct" />
      <code value="128045006:{363698007=56459004}" />
    </coding>
    <text value="Cellulitis of the foot" />
  </diagnosis>
```

Using the valueset:

The results on a urinalysis strip:

```
  <valueCoding>
    <system value="http://example.org/codes/simple-grades" />
    <code value="+" />
    <valueSet>
      <reference url="ValueSet/simple-grades" />
    </valueSet>
  </valueCoding>
```

具体定义编码的value set如下所示:

```
<ValueSet xmlns="http://hl7.org/fhir">
  <text>
    <status value="generated"/>
    <div xmlns="http://www.w3.org/1999/xhtml">
      <p>Possible Clinistix codes: neg, trace, +, ++, and +++</p>
    </div>
  </text>
  <identifier value="http://hl7.org/fhir/vs/clinistix"/>
  <name value="Codes for Clinistix"/>
  <publisher value="HL7"/>
  <telecom>
    <system value="url"/>
    <value value="http://hl7.org/fhir"/>
  </telecom>
  <description value="Clinistix Codes"/>
  <status value="draft"/>
  <experimental value="true"/>
  <date value="2013-10-01"/>
  <define>
    <system value="http://hl7.org/fhir/clinistix"/>
    <caseSensitive value="false"/>
    <concept>
      <code value="neg"/>
    </concept>
    <concept>
      <code value="trace"/>
    </concept>
    <concept>
      <code value="+"/>
    </concept>
    <concept>
      <code value="+"/>
    </concept>
    <concept>
      <code value="++"/>
    </concept>
    <concept>
      <code value="+++"/>
    </concept>
  </define>
</ValueSet>
```

1.17.1.7 Quantity

See also Base Definition, Detailed Descriptions and Mappings.

Examples

A duration:

```
  <time>
    <value value="25" />
    <units value="sec" />
    <system value="http://unitsofmeasure.org" />
    <code value="s" />
  </time>
`
```

浓度超出范围:

```
  <result>
    <value value="40000" />
    <comparator value="&gt;" />
    <units value="mcg/L" />
    <system value="http://unitsofmeasure.org" />
    <code value="ug" />
  </result>
`
```

处方药的数量:

```
  <dose>
    <value value="3" />
    <units value="capsules" />
    <system value="http://snomed.info/sct" />
    <code value="385049006" />
  </dose>

`
```

A price (coded using currency codes defined in ISO 4217):

```
  <cost>
    <value value="25.45" />
    <units value="US$" />
    <system value="urn:std:iso:4217" />
    <code value="USD" />
  </cost>

`
```

1.17.1.8 Range

See also Base Definition, Detailed Descriptions and Mappings.

Examples

Range of Quantity (distance):

```
  <estimate>
   <low>
     <value value="1.6" />
     <units value="m" />
   </low>
   <high>
     <value value="1.9" />
     <units value="m" />
   </high>
  </estimate>
`
```

1.17.1.9 Ratio

See also Base Definition, Detailed Descriptions and Mappings.

Examples

滴速 (Ratio of integer:integer)

```
  <result>
   <numerator>
     <value value="1" />
   </numerator>
   <denominator>
     <value value="128" />
   </denominator>
  </result>

`
```

单位成本(Ratio of Money:Quantity):

```
  <charge>
   <numerator>
     <value value="103.50" />
     <units value="US$" />
     <code value="USD" />
     <system value="urn:std:iso:4217" />
   </numerator>
   <denominator>
     <value value="1" />
     <units value="day" />
     <code value="day" />
     <system value="http://unitsofmeasure.org" />
   </denominator>
  </charge>

`
```

1.17.1.10 Period

See also Base Definition, Detailed Descriptions and Mappings.

Examples

2011年5月23到27 包括27号:

```
  <coverage>
```

```
    <start value="2011-05-23" />
    <end value="2011-05-27" />
   </coverage>

`
```

1.17.1.11 SampledData

See also Base Definition, Detailed Descriptions and Mappings.

Example

EKG设备的输出:

```
 <sampledData>
  <origin>
   <value value="0"/>
   <units value="µV"/>
   <system value="http://unitsofmeasure.org"/>
   <code value="uV"/>
  </origin>
  <period value="2"/>
  <factor value="2.5"/>
  <dimensions value="1"/>
  <data value="-4 -13 -18 -18 -18 -17 -16 -16 -16 -16 -16 -17 -18 -18 -18 ...."/>
 </sampledData>
`
```

1.17.1.12 HumanName

See also Base Definition, Detailed Descriptions and Mappings.

A Simple example

```
<name>
  <family value="Everyman" />
  <given value="Adam" />
  <given value="A." />
</name>

`
```

Composite names

```
<name>
  <family value="Contrata" />
  <given value="Mary Jane" />
</name>

`
```

These cases can be quite ambiguous - is "Mary Jane" one name, or two? Different systems, and data enterers may treat this differently, and the person themselves may not know. Parts are allowed to contain spaces, but systems should consider how to treat these cases. Composite names separated by "-" should be treated as a single name part.

A common pattern: a person is called by a name other than that expected from their official name (first given name in most cultures).

```
<name>
  <use value="official" />
  <family value="Chalmers" />
  <given value="Peter" />
  <given value="James" />
</name>
<name>
  <use value="usual" />
  <given value="Jim" />
</name>
```

This same pattern is often encountered with immigrants, who retain their real name for official use, but adopt a localized name for everyday use:

```
<name>
  <use value="official" />
  <family value="Sczypinski" />
  <given value="Piotr" />
  <given value="Andre" />
</name>
<name>
  <use value="usual" />
  <family value="Skipper" />
  <given value="Jim" />
</name>
```

Karen van Hentenryck is of Dutch origin, and the "van" is a voorvoegsel.

```
<name>
  <use value="official" />
  <family value="van">
    <extension url="http://hl7.org/fhir/StructureDefinition/iso21090-EN-qualifier" >
       <valueCode value="VV" />
    </extension>
  </family>
  <family value="Hentenryck" />
  <given value="Karen" />
</name>
```

See the Extensibility Example for more information. Note that this name has multiple family name parts. Systems that do not support as many name parts as are provided in an instance they are processing may wish to append parts together using spaces, so that this becomes "van Hentenryck".

Complex example from Germany: Dr.phil. Regina Johanna Maria Gräfin Hochheim-Weilenfels, NCFSA. This example shows extensive use of multiple given names, prefixes, suffixes, for academic degrees, nobility titles, and professional designations.

```
<name>
  <use value="official" />
  <family value="Hochheim-Weilenfels" />
  <given value="Regina" />
  <given value="Johanna" />
  <given value="Maria" />
  <prefix value="Gräfin">
    <extension url="http://hl7.org/fhir/StructureDefinition/iso21090-EN-qualifier" >
      <valueCode value="NB" />
    </extension>
  </prefix>
  <prefix value="Dr. phil.">
    <extension url="http://hl7.org/fhir/StructureDefinition/iso21090-EN-qualifier" >
      <valueCode value="AC" />
    </extension>
  </prefix>
  <suffix value="NCFSA" />
```

```
</name>
<name>
  <use value="maiden" />
  <family value="Hochheim" />
</name>
```

This example makes use of the ISO 21090 extensions to carry the rare ISO 21090 qualifier attributes "AC" and "NB".

Japanese example in the three forms: ideographic (Kanji), syllabic (Hiragana) and alphabetic (Romaji).

```
<name>
  <family value="??" />
  <given value="??" />
  </name>
<name>
 <family value="???" />
 <given value="???" />
</name>
<name>
  <family value="KIMURA" />
  <given value="MICHIO" />
</name>
```

The three forms are differentiated by the character subset each contains.

Russian example in the two forms: cyrillic, and latin:

```
<name>
  <family value="??????" />
  <given value="????" />
  <given value="????????????" />
</name>
<name>
  <family value="EMELIN" />
  <given value="IVAN" />
  <given value="VLADIMIROVICH" />
</name>
```

In Russian usage, these names are known as the domestic and foreign names respectively. The two forms are differentiated by the character subset each contains.

Scandinavian example: Erikson is the family name. Jan Erik are the given names, and Östlund the family name of the mother, which is taken as a Mellannamn.

```
<name>
  <use value="official" />
  <family value="Erikson" />
  <given value="Jan" />
  <given value="Erik" />
  <given value="Östlund">
    <extension url="http://hl7.org/fhir/StructureDefinitioniso-20190#name-qualifier" >
      <valueCoding>
         <code value="MID" />
         <system value="http://hl7.org/fhir/v3/EntityNamePartQualifier2" />
      </valueCoding>
    </extension>
  </given>
</name>
```

This example makes use of the ISO 21090 extension to carry the culture specific ISO 21090 qualifier attribute "MID" for the Mellannamn.

Then Jan Erikson has a daughter, Karin, with his wife Margrete Hansen. The first communications of the new born name is "Margrete Jente" (Margrete's Girl) and the mother's family name, not the given name (Karin). The father's Family name is not used at all. This is a known temporary name assigned directly after the birth of the child.

```
<name>
  <use value="temp" />
  <!-- use could be OR+OLD, depends how record keeping is done -->
  <family value="Hansen" />
  <given value="Margrete Jente" />
</name>
```

The baby's name is subsequently changed to the fathers' family name, and to use the mother's name as mellomnamn.

```
<name>
  <use value="official" />
  <family value="Erikson" />
  <given value="Karin" />
  <given value="Hansen">
    <extension url="http://hl7.org/fhir/StructureDefinitioniso-20190#name-qualifier" >
      <valueCoding>
         <code value="MID" />
         <system value="http://hl7.org/fhir/v3/EntityNamePartQualifier2" />
      </valueCoding>
    </extension>
  </given>
</name>
```

Later, Karin gets married to Per Berg, and decides to adopts Berg as her family name, and also decides to use Erikson as the mellom navn. (Note: Karin could have chosen to use another mellom navn, e.g. the family name of her mother, her father or other family names as specified by naming laws of the country in question).

```
<name>
  <use value="old" />
  <family value="Erikson" />
  <given value="Karin" />
  <given value="Hansen">
    <extension url="http://hl7.org/fhir/StructureDefinitioniso-20190#name-qualifier" >
      <valueCoding>
        <code value="MID" />
        <system value="http://hl7.org/fhir/v3/EntityNamePartQualifier2" />
      </valueCoding>
    </extension>
  </given>
</name>
<name>
  <use value="official" />
  <family value="Berg" />
  <given value="Karin" />
  <given value="Erikson">
    <extension url="http://hl7.org/fhir/StructureDefinitioniso-20190#name-qualifier" >
      <valueCoding>
        <code value="MID" />
        <system value="http://hl7.org/fhir/v3/EntityNamePartQualifier2" />
      </valueCoding>
    </extension>
  </given>
</name>
<name>
  <use value="usual" />
  <family value="Berg" />
  <given value="Karin" />
</name>
```

1.17.1.12.1 W3C International Examples These examples are taken from the [W3C International Examples](http://www.w3.org/International/questions/qa-personal-names), which should be consulted for further information. A

patronymic is "The part of a name that links to the genealogy":

```
<name>
  <text value="Björk Guðmundsdóttir"/>
  <family value="Guðmundsdóttir"/>
  <given value="Björk"/>
</name>
```

A patronymic with a "son/daughter of" appellation:

```
<name>
  <text value="Isa bin Osman"/>
  <family value="bin Osman"/>
  <given value="Isa"/>
</name>
```

Note: The family name may also be given as two different family names.

A Chinese name with a generational name:

```
<name>
  <text value="???"/> <!-- left to right -->
  <family value="?"/>
  <given value="?"/>
  <given value="?"/>
</name>
<name>
  <text value="Mao Ze Dong"/> <!-- left to right -->
  <family value="Mao"/>
  <given value="Ze"/>
  <given value="Dong"/>
</name>
```

Todo: is there a need to identify the given name that is the generational name.

Additional Western name (see also example above):

```
<name>
  <use value="official" />
  <family value="Yao" />
  <given value="Ming" />
</name>
<name>
  <use value="usual" />
  <given value="Fred" />
</name>
```

Multiple Family names:

```
<name>
  <family value="Carreño" />
  <family value="Quiñones" />
  <given value="María-Jose" />
</name>
```

Brazilian Example:

```
<name>
  <family value="Eduardo" />
```

```
  <family value="Santos" />
  <family value="Tavares" />
  <family value="Melo" />
  <family value="Silva" />
  <given value="José" />
</name>
```

Russian Examples (using Cyrillic):

```
<name>
  <family value="??????????" />
  <family value="??????" />
  <given value="?????" />
</name>
<name>
  <family value="?????????" />
  <family value="???????" />
  <given value="?????" />
</name>
```

Example with Initial:

```
<name>
  <family value="Public" />
  <given value="John" />
  <given value="Q." />
</name>
```

Other Examples:

```
<name>
  <text value="Velikkakathu Sankaran Achuthanandan"/>
  <family value="Velikkakathu" />
  <given value="Sankaran" />
  <given value="Achuthanandan" />
</name>
<name>
  <text value="Kogaddu Birappa Timappa Nair"/>
  <family value="Nair" />
  <given value="Birappa" />
  <given value="Timappa" />
  <prefix value="Kogaddu" />
</name>
<name>
  <text value="Aditya Pratap Singh Chauhan"/>
  <family value="Singh" />
  <given value="Aditya" />
  <given value="Pratap" />
  <suffix value="Chauhan" />
</name>
<name>
  <text value="Madurai Mani Iyer"/>
  <given value="Mani" />
  <prefix value="Madurai" />
  <suffix value="Iyer" />
</name>
<name>
  <text value="Abu Karim Muhammad al-Jamil ibn Nidal ibn Abdulaziz al-Filistini"/>
  <family value="ibn Nidal" />
  <family value="ibn Abdulaziz" />
  <given value="Muhammad" />
  <given value="al-Jamil" />
  <prefix value="Abu Karim" />
  <suffix value="al-Filistini" />
</name>
```

Todo: need to discuss this with Indian / Arabic implementers. Note that collecting and storing the *text* element makes the primary purpose of the structured parts for index/searching, and fidelity of the name parts is not critical.

1.17.1.13

Address

See also Base Definition, Detailed Descriptions and Mappings.

Example

HL7 office's address.

```
  <address>
   <use value="work" />
   <text value="1050 W Wishard Blvd
RG
         5th floor
Indianapolis, IN 46240" />
   <line value="1050 W Wishard Blvd" />
   <line value="RG 5th floor" />
   <city value="Indianapolis" />
   <state value="IN" />
   <postalCode value="46240" />
  </address>
```

A UK example address.

```
  <address>
    <extension url="http://hl7.org/fhir/StructureDefinition/iso21090-ADXP-county" > <!-- todo: review this -->
      <valueString value="HUDDERSFIELD"/>
    </extension>
    <text value="1 Back Lane&#13;&#10;Holmfirth&#13;&#10;HUDDERSFIELD&#13;&#10;HD7 1HQ"/>
    <line value="1 Back Lane"/>
    <city value="Holmfirth"/>
    <postalCode value="HD7 1HQ"/>
  </address>
```

A Postal address - i.e. an address that it doesn't make sense to try and visit.

```
  <address>
    <extension url="http://hl7.org/fhir/StructureDefinition/iso21090-AD-use"> <!-- todo: review this -->
      <valueCode value="PST"/>
    </extension>
    <line value="PO Box 31445"/>
    <city value="Erewhon"/>
    <postalCode value="0001"/>
  </address>
```

1.17.1.14 ContactPoint See also [Base Definition](datatypes.html#ContactPoint), [Detailed Descriptions](datatypes-definitions.html#ContactPoint) and [Mappings](datatypes-mappings.html#ContactPoint). **Example** Home phone number:

```
  <telecom>
   <system value="phone" />
   <value value="+15556755745" />
   <use value="home" />
  </telecom>
```

如果是微信二维码的话，如何标识： #### 1.17.1.15 Timing See also [Base Definition](datatypes.html#Timing), [Detailed Descriptions](datatypes-definitions.html#Timing) and [Mappings](datatypes-mappings.html#Timing). **Example** 预约了一

个疗程的放疗？: ``` ````` 一天两次BID (twice a day) (no start or end specified): ``` ````` 早饭前半小时，从2011年12月23开始连续十天 1/2 an hour before breakfast for 10 days from 23-Dec 2011: ``` ````` Note that the end date is inclusive like the high date of a Period. ### 1.17.1.16 Signature See also [Base Definition](datatypes.html#Signature), [Detailed Descriptions](datatypes-definitions.html#Signature) and [Mappings](datatypes-mappings.html#Signature). **Example** todo

```
<signature>
  <!-- todo -->
</signature>
```


链接：试行版是什么 |版本更新情况 | 许可协议 |提交变更建议

code=====string oid=uri=string

基本类型 boolean integer string decimal uri base64Binary instant date dateTime time

扩展 | extension | ---------- | 否0..* | ---------- | | extension.url | uri | 是1..1 | ---------- | | extension.value[x] | ---------- | 否0..1 | x可以是Integer、Decimal、DateTime、Date、Instant、String、Uri、Boolean、Code、Base64Binary、Coding、CodeableConcept、Attachment、Identifier、Quantity、Range、Period、Ratio、HumanName、Address、ContactPoint、Timing、Signature、Reference |

| modifierExtension | ---------- | 否0..* | ---------- | | modifierExtension.url | uri | 是1..1 | ---------- | | modifierExtension.value[x] | ---------- | 否0..1 | x可以是Integer、Decimal、DateTime、Date、Instant、String、Uri、Boolean、Code、Base64Binary、Coding、CodeableConcept、Attachment、Identifier、Quantity、Range、Period、Ratio、HumanName、Address、ContactPoint、Timing、Signature、Reference |

coding 类型 | ResourceType.coding | |----------| 否0..* | ---------- | | ResourceType.coding.system | string | 否0..1 | xs:anyURI | | ResourceType.coding.version | string | 否0..1 | ---------- | | ResourceType.coding.code | code | 否0..1 | ---------- | | ResourceType.coding.display | string| 否0..1 | ---------- |

Narrative 类型 | ResourceType.Narrative | |----------| 否0..* | ---------- | | ResourceType.Narrative.status | string | 是1..1 | code | | ResourceType.Narrative.div | string | 是1..1 | xhtml |

Identifier 类型

| Identifier.use | code | 否0..1 | ---------- | | Identifier.type | CodeableConcept | 否0..1 | | | Identifier.type.coding | ----------| 否0..* | ---------- | | Identifier.type.coding.system | string | 否0..1 | xs:anyURI | | Identifier.type.coding.version | string | 否0..1 | ---------- | | Identifier.type.coding.code | code | 否0..1 | ---------- | | Identifier.type.coding.display | string| 否0..1 | ---------- | | Identifier.type.text | string | 否0..1 | ---------- | | Identifier.system | string | 否0..1 | uri | | Identifier.value | string | 否0..1 | | | Identifier.period | ---------- | 否0..1 | ---------- | | Identifier.period.start | dateTime | 否0..1 | ---------- | | Identifier.period.end | dateTime | 否0..1 | | | Identifier.assigner | ---------- | 否0..1 | Organization引用 | | Identifier.assigner.reference | string | 否0..1 | ---------- | | Identifier.assigner.display | string | 否0..1 | |

period 类型 | Period.start | dateTime | 否0..1 | ---------- | | Period.end | dateTime | 否0..1 | |

CodeableConcept 类型

| ResourceType.CodeableConcept.coding |----------| |----------| 否0..* | ---------- | | ResourceType.CodeableConcept.coding.system | string | 否0..1 | xs:anyURI | | ResourceType.CodeableConcept.coding.version | string | 否0..1 | ---------- | | ResourceType.CodeableConcept.coding.code | code | 否0..1 | ---------- | | ResourceType.CodeableConcept.coding.display | string| 否0..1 | ---------- | | ResourceType.CodeableConcept.text | string | 否0..1 | ---------- |

Reference 类型 | Reference.reference | string | 否0..1 | ---------- | | Reference.display | string | 否0..1 | |

基本resource和domainResource的字段整合 | 字段名称 | 数据类型 | 必须存在 | 说明 | | ResourceType | ---------- | ---------- | ---------- | | ResourceType.id | string | 否0..1 | ---------- | | ResourceType.meta | ---------- | 否0..1 | ---------- | | ResourceType.meta.versionId | string | 否0..1 | ---------- | | ResourceType.meta.lastUpdated | instant | 否0..1 | ---------- | | ResourceType.meta.profile | uri | 否0.. | ---------- | | *ResourceType.meta.security* |---------- | 否*0..* | ---------- | | ResourceType.meta.security.system | uri | 否0..1 | ---------- | | ResourceType.meta.security.version | string | 否0..1 | ---------- | | ResourceType.meta.security.code | code | 否0..1 | ---------- | | ResourceType.meta.security.display | string| 否0..1 | ---------- | | ResourceType.meta.security.primary | boolean | 否0..1 | ---------- | | ResourceType.meta.tag | |----------| 否0.. | ---------- | | *ResourceType.meta.tag.system* | *uri* | 否*0..1* | ---------- | | *ResourceType.meta.tag.version* | *string* | 否*0..1* | ---------- | | *ResourceType.meta.tag.code* | *code* | 否*0..1* | ---------- | | *ResourceType.meta.tag.display* | *string*| 否*0..1* | ---------- | | *ResourceType.meta.tag.primary* | *boolean* | 否*0..1* | ---------- | | *ResourceType.implicitRules* | *uri* | 否*0..1* | ---------- | | *ResourceType.language* | *string* | 否*0..1* | 默认为*zh* | | *ResourceType.text* | ---------- | 否*0..1* | ---------- | | *ResourceType.text.status* | *string* | 是*1..1* | *code* | | *ResourceType.text.div* | *string* | 是*1..1* | *xhtml* | | *ResourceType.contained*

| *ResourceType* | 否*0..* | ---------- | | ResourceType.extension | ---------- | 否0.. | ---------- | | *ResourceType.extension.url* | ---------- | 是*1..1* | ---------- | | *ResourceType.extension.value[x]* | ---------- | 否*0..1* | *x*可以是*Integer*、*Decimal*、*DateTime*、*Date*、*Instant*、*String*、*Uri*、*Boolean*、*Code*、*Base64Binary*、*Coding*、*CodeableConcept*、*Attachment*、*Identifier*、*Quantity*、*Range*、*Period*、*Ratio*、*HumanName*、*Address*、*ContactPoint*、*Timing*、*Signature*、*Reference* | | *ResourceType.modifierExtension* | ---------- | 否*0..* | ---------- | | ResourceType.modifierExtension.url | ---------- | 是1..1 | ---------- | | ResourceType.modifierExtension.value[x] | ---------- | 否0..1 | x可以是Integer、Decimal、DateTime、Date、Instant、String、Uri、Boolean、Code、Base64Binary、Coding、CodeableConcept、Attachment、Identifier、Quantity、Range、Period、Ratio、HumanName、Address、ContactPoint、Timing、Signature、Reference |

FHIR and OAuth2

原文链接：FHIR and OAuth2

文中提到的**request type**到底有哪些 所说的**code type**和**'implicit' flow**到底该如何理解 一些基本概念的解释.

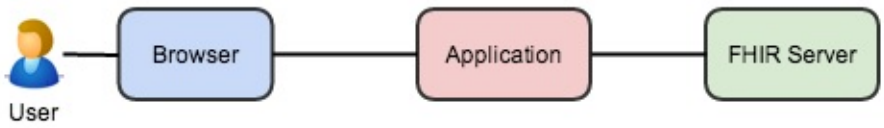
- *Identity*实体的属性(诸如姓名、性别、出生日期等)之一，一个人可以有多个标识.
- *Authentication*认证就是向服务器证明自己是自己所声称的那个实体。最常见的用户名密码就是其中一种方式.
- *Authorization*授权就是确定和控制访问特定类型的资源.比方说 访问患者临床数据用户的角色必须是医师.
- *Audit*系统中完成了哪些操作，是谁执行的的跟踪记录。FHIR中利用SecurityEvent来表示, 是基于 IHE ATNA 规范的.

这些概念之间是互相独立的。比方说在授权之前你需要认证用户的身份， 在进行一些操作时新建一份审计记录。

FHIR quite explicitly并不是安全协议, 其他的一些组织机构负责构建此类的标准。但FHIR中对安全相关的问题也给出了一些参考意见(e.g. tags for privacy)， 也有一些很有用的资源 (such as SecurityEvent 和 Provenance),当使用HTTP/S就可以使用诸如TLS (Transport Layer Security), OAuth2 and OpenID Connect等标准 后续的文章中我们将重点讨论后面2个.

OAuth2 是一个能够让应用程序访问位于另外的服务器上用户数据的框架， 应用程序无需在本地保存用户凭证。

例如,你的web应用程序能够让患者从FHIR服务器上访问他们的数据， 系统架构可能如下:

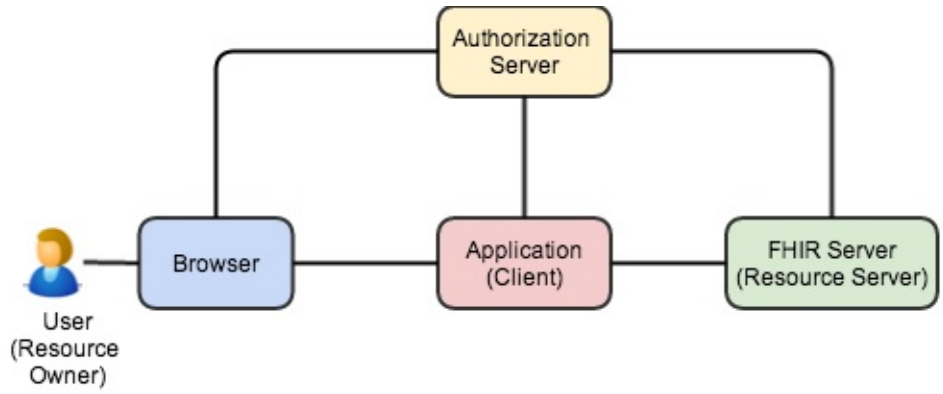


用户通过界面与网络上的应用程序进行交互， 应用程序服务器端的程序从FHIR服务器中获取患者数据，比如说体重。.

假如用户通过用户名密码登录了应用程序的系统，但FHIR服务器又如何知道该传输患者的数据呢？ 一种方式是用户在FHIR服务器上也有一个帐号， 应用程序的系统之中保存了这些凭据， 这样应用系统在请求数据时可以将其传输给FHIR服务器,这样做的问题在于：

- 应用系统本身会保存用户凭据，提高了信息泄漏的风险(Weakest links in a chain and all that stuff)
- 如果需要访问其他FHIR服务器上的数据，要么用户在所有服务器上的凭据都是一样的，要么就需要保存一个用户的多份凭据.
- 如果用户不希望应用程序继续访问服务器上的数据，用户必须更改所有服务器上的账户信息
- 每个服务器和应用程序都需要保存每个用户的凭据。极容易在变更、启动过程中产生混乱，由此增加泄漏的风险

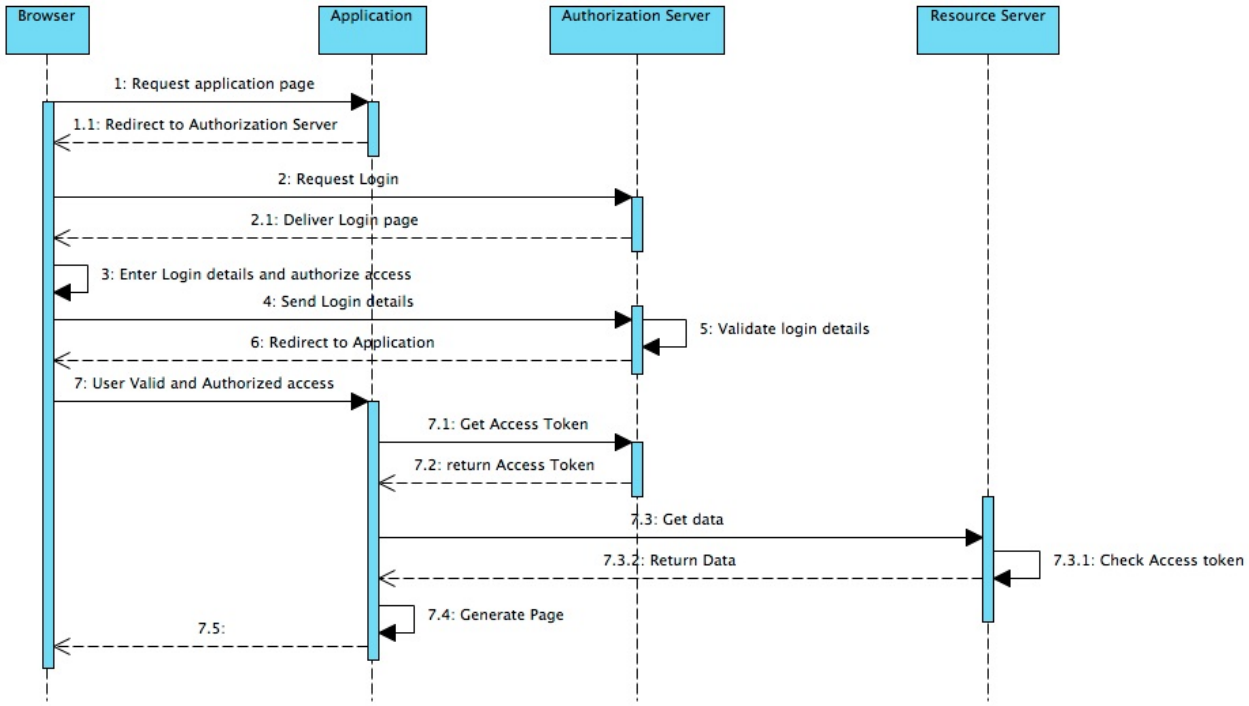
OAuth2的方式通过份认证和授权角色与提供信息的橘色分离解决了此类问题.



授权服务器是一个单独的组件，用户凭据保存在这之中。其他组件如应用程序、FHIR服务器通过授权服务器来标识确定用户，表明调用程序可以访问那些数据.

一般而言，授权服务器是一些可靠的系统，用户在这些系统上已经有了帐号-诸如google、亚马逊，但也可以是一个单独的构件或者是FHIR服务器的一个功能.

工作机理如下.



1. 用户向应用程序(客户端)发送请求(通过浏览器)，应用程序需要从FHIR服务器中获取数据，应用程序发送了一个“浏览器重定向”给用户的浏览器，用户的浏览器会向可信的授权服务器发送登录请求.
2. 授权服务器发送登录界面给用户.
3. 用户填写好登录信息，同意应用程序访问授权服务器上的数据，包括了数据及其相应的访问权限.
4. 将登陆界面提交给授权服务器 .
5. 授权服务器验证登录信息
6. 授权服务器发送浏览器重定向给用户浏览器，跳转到应用程序的某个节点。重定向链接中包含了授权码，允许访问用户数据
7. 用户浏览器重定向到应用程序的回调函数，这样应用程序就能:

    i. 利用授权码调用授权服务器，授权服务器返回Access token
    ii. 应用程序向FHIR服务器发送查询请求，请求参数中包含access token.
    iii. FHIR服务器确认access token的合法性，并完成查询请求，将数据返回给应用程序.

iv. 应用程序利用数据 展示在web浏览器中

备注:

- 这只是对整体流程的大概描述，省略了大量的细节，大多数情况会使用某个库来实现OAuth2.
- 需要OAuth2授权的FHIR服务器通常称之为 'OAuth2 protected'.
- 资源服务器校验访问令牌的机制取决于开发人员，有诸多方案可供选择:
    - 假如授权服务器是FHIR服务器的一部分，可以直接进行校验.
    - FHIR服务器可以单独调用授权服务器来确认这些信息
    - 授权服务器可以给访问令牌签名，以此向FHIR服务器证明令牌是合法的.
- 在授权服务器返回访问令牌的同时也要返回一个单独的刷新令牌，刷新令牌保存在应用程序本地。访问令牌有一定的生命周期，通常是一个小时，当其过期后，应用程序可以使用刷新令牌来获取一个新的访问令牌。这样设计的原因在于授权服务器可以废除刷新令牌，这样应用程序就无法获取新的访问令牌，无法在访问令牌过期之后继续访问FHIR服务器。没有这种机制，FHIR服务器就得每次都先与授权服务器核对一下是否访问令牌已经失效.

- 上述的流程称之为 'Code' request type, 仅适用于应用程序可以保障刷新令牌的安全的情况，因为刷新令牌是保存在服务器上。对于桌面端或移动端的应用程序而言，不能采用这样的流程，但可以选用'implicit' flow .
- 在这种流程下，应用程序需要与授权服务器单独进行注册。在这个过程中，应用程序收到一个编码，用这个编码可以向授权服务器证明自己的身份，提供应用程序回调地址，这样子在用户登录后就可以重定向至浏览器.
- OAuth2的安全性完全取决于安全的传输机制-因此向SSL之类的TLS是绝对需要的.
- 上述只是个大概的描述，详细请参阅标准英文，最好使用如下的一些库来实现 标准库列表.

参考资料:

The OAuth2 Specification

A post on Grahames Blog. This points out some issues with delegating authorization..

A tutorial by Jakob Jenkov

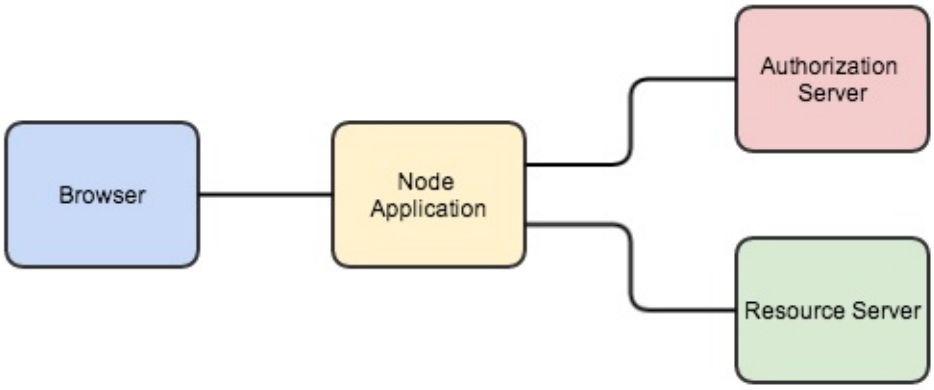
Blue Button

SMART on FHIR

A simple OAuth client

A simple OAuth client

选用的技术：Node.js Javascript，构建一个node服务器，作为OAuth客户端来调用其他的OAuth服务器。整体架构如下图所示



一个基本的web应用程序，包含一个HTML页面，通过HTTPS与 OAuth服务器通讯，由于代码是允许在服务器端而不是客户端，因此可以 选用"授权码"流程。

选用node模块simple-outh2’

实现拿Google来试验. Google使用OAuth2来授权访问自己的服务-实际上，Google在OAuth模型中包含了授权服务器和资源服务器。第一步，获取客户端用以进行授权服务器认证的凭据，可以在Developer website中找到.

app代码如下:

```
var request = require('request');   //https://github.com/mikeal/request
var path = require('path');
var express = require('express'),
    app = express();

app.use(express.cookieParser());
app.use(express.session({secret: '1234567890QWERTY'}));
app.use(express.static(path.join(__dirname, 'public')));

var OAuth2;

var credentialsGoogle = {
    clientID: "<myclientid>",
    clientSecret: "<mysecret>",
    'site': 'https://accounts.google.com/o/oauth2/',
    'authorizationPath' : 'auth',
    'tokenPath' : 'token'
};

// Initial call redirecting to the Auth Server
app.get('/auth', function (req, res) {
    OAuth2 = require('simple-oauth2')(credentialsGoogle);
    authorization_uri = OAuth2.AuthCode.authorizeURL({
        redirect_uri: 'http://localhost:3001/callback',
        scope: 'openid email https://www.googleapis.com/auth/drive https://www.googleapis.com/auth/tasks',
        state: '3(#0/!~',
        access_type: "offline"      //causes google to return a refresh token
```

```
    });

    res.redirect(authorization_uri);
});

// Callback endpoint parsing the authorization token and asking for the access token
app.get('/callback', function (req, res) {
    var code = req.query.code;

    OAuth2.AuthCode.getToken({
        code: code,
        redirect_uri: 'http://localhost:3001/callback'
    }, saveToken);

    function saveToken(error, result) {
        if (error) {
            console.log('Access Token Error', error.message, error);
            res.json({'Access Token Error': error.message});
        } else {
            //see what we've got...
            console.log(result);
            //this adds the expiry time to the token by adding the validity time to the token
            token = OAuth2.AccessToken.create(result);
            //save the response back from the token endpoint in a session
            req.session.token = result;

            //perform the res of the processing now that we have an Access Token
            gotToken(req,res);
        }
    }

    //Now the token has been received and saved in the session, return the next page for processing
    function gotToken(req, res) {
        res.sendfile(__dirname +"/public/main.html");
    };

});

//get the lists for the current user...
app.get("/tasks", function (req, res) {
    //Need a token to access the google services
    if (req.session.token) {
        var AT = req.session.token["access_token"];
        var url = "https://www.googleapis.com/tasks/v1/users/@me/lists";

        var options = {
            method: "GET",
            headers: {
                "content-type": "application/json+fhir",
                "authorization": "Bearer " + AT
            },
            rejectUnauthorized: false,      //to allow self-signed cetificates
            uri: url
        };

        request(options, function (error, response, body) {
            res.json(body);
        });
    } else {
        res.json({err:"Not logged in"});
    }

});

app.listen(3001);

console.log("OAuth Client started on port 3001");
```

1. 首先调用/auth endpoint.使用凭据初始化OAuth2对象，授权和令牌endpoint的位置;然后利用如范围、状态等参数重定向至授权服务器，详细信息可参考google官网
2. Google提供一个登录界面，用户进行登录，对该请求进行认证。
3. 如果一切顺利，浏览器会重定向至本地的调用endpoint节点， (http://localhost:3001/callback 从响应中拿到授权码，向google再次请求访问令牌

4. 获得访问令牌之后，打开一个HTML页面。.
5. 这样，用户就可以访问Google的资源了，比如/task节点能够获得当前用户的所有任务列表.

第27行代码中access_type 的值设为 ‘offline’，要求Google在响应中包含一个刷新令牌，但似乎并没有成功，这里还需要再研究研究。 在获取授权令牌的响应中 你可以看到 id_token，因为我们在scope的属性中包含了 ‘openid’Google也实现了OpenId Connect，同时也能够返回标识信息

FHIR, Oauth2 and the Mobile client

原文链接:FHIR, Oauth2 and the Mobile client

FHIR和OAuth2中讨论了web应用程序(浏览器端运行 代码在服务器端)中最常见的流程，但如果是桌面和移动应用程序如何处理？要考虑以下几点：

- 对于应用程序而言，用户有哪些，是否需要存储刷新指令，
- 在OAuth2的处理流程当中，它是依赖HTTPS的。主要用在登录界面和浏览器重定向。

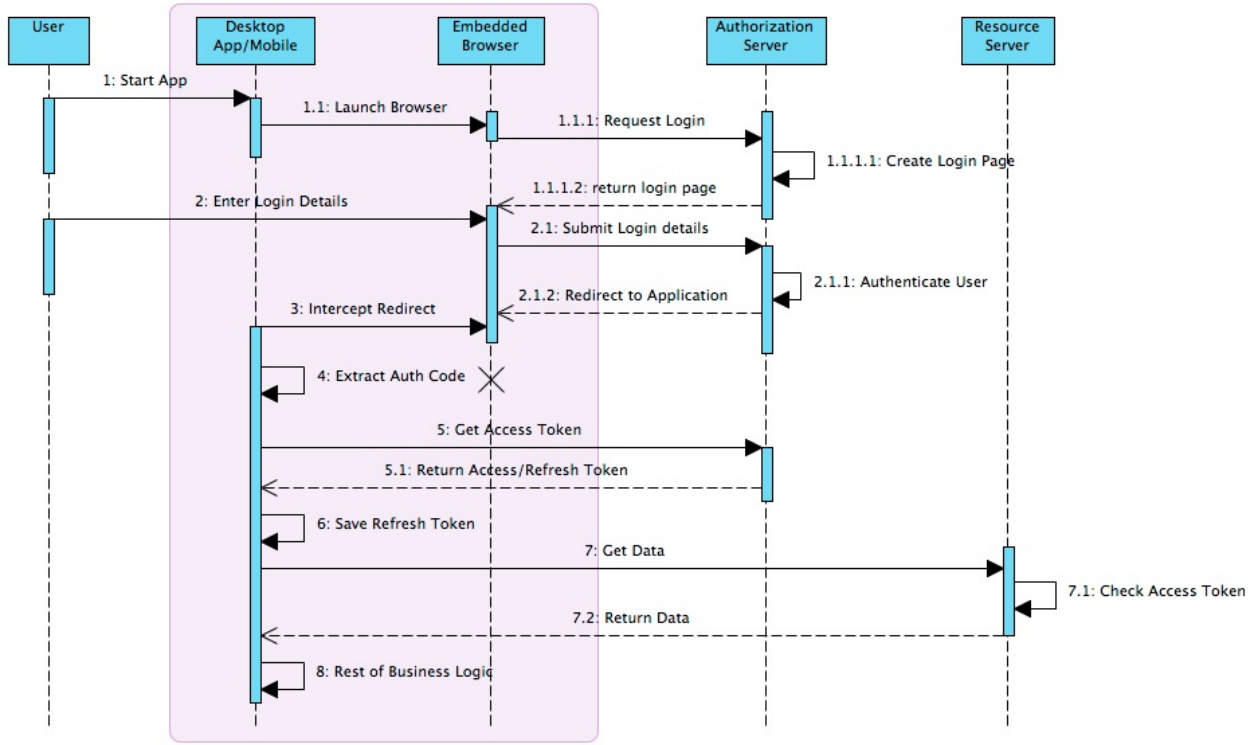
刷新令牌的处理

理论上，这些应用程序都不能安全地保存刷新令牌。原因在于该刷新指令将会存储在终端用户的移动设备或电脑上，黑客如果想获取该刷新指令是很容易的。这么讲的话，应用程序是不能存储刷新指令的。用户每次需要使用刷新令牌的时候，必须与授权服务器进行认证，这个session的生命周期与访问指令的生命周期一样长。事实上，针对这种情况，OAuth2描述了一种这样的特殊流程Implicit workflow,在用户认证通过之后授权服务器直接返回访问指令，但不会返回一个刷新指令。这是针对于用户不想要长期保存凭据的这种'公开式'场景.

这样是很好理解,但对于用户而言,如果是用户自己使用的移动设备的话,就不是那么友好.这种情况下,你除了需要在本地存储刷新令牌之外没有其他可选的解决方案:

- 你必须尽可能做到更安全.
- 你必须应对多用户使用的APP可能性,比方说与当前用户的关联
- 你必须提供一种简单的机制能够让用户废除访问令牌.标准本身并没有说明如何处理这种情况,一种方法是授权服务器会有一个重置或设备遗失的功能.当用户登录之后,会出现一个正常的web界面,授权服务器会撤销所有与用户相关的刷新指令.当用户使用新的设备时需要重新认证,整个流程正常继续.

以这种方式的话,也能够理解如何在桌面端和移动端应用程序使用OAuth2 一种方法是在应用程序中启动一个内嵌式的浏览器窗口来与授权服务器进行交互,如下图所示



1. 用户以某种方式登录并启动了APP,APP调用浏览器组件并直接调用授权服务器的登录请求节点,授权服务器生成并返回一个登录界面.
2. 用户输入信息并将页面提交给授权服务器.授权服务器认证用户通过之后,生成一个认证码,发送一个重定向给内嵌式浏览器,浏览器会重定向至应用程序的回调节点
3. 应用程序拦截了内嵌式浏览器的重定向
4. 应用程序从中抽取出认证码并终止了浏览器组件
5. 应用程序发送一个HTTPS请求给授权服务器,认证码作为参数传输过去,服务器返回一个访问令牌和刷新令牌.
6. 以最安全的方式将刷新令牌保存在本地
7. 应用程序发送HTTPS请求给资源服务器,将访问令牌作为参数传递过去,资源服务器对访问令牌进行校验,通过之后则返回数据.
8. 应用程序完成其他的业务逻辑.

对于废弃的访问令牌如何处理,访问令牌都有自己的生命周期,但如果在处理过程中访问令牌失效了该怎么办.流程类似如下:

1. 应用程序向资源服务器发送请求.
2. 资源服务器发现令牌已失效,拒绝了这次请求 标准中并未对状态码进行描述,可能是401-Unauthorized
3. 应用程序从本地存储库中获取刷新令牌,将其发送给授权服务器.授权服务器再次向应用程序提供一个有效的访问令牌.
4. 应用程序利用新的访问令牌再次向资源服务器发送请求.

这篇文章描述了桌面端和移动端应用程序如何实现标准的OAuth2协议来对用户进行认证,授权用户访问数据.当然,这并不是唯一的方法,但是最直接最易于实现一种方法.

可以利用 开源库!来实现

FHIR and OpenID Connect

原文链接:FHIR and OpenID Connect

fhir-and-oauth2和fhir-oauth2-and-the-mobile-client中探讨了OAuth2的用法，这样，我们就了解到用户如何授权一个应用程序无需再暴露自己的登录凭据的情况下从其他服务器上获取数据。授权服务器是唯一需要拥有用户名和密码或者所采用的其他授权方式相关信息的组件。

但除了这些登录相关内容，应用程序如何保证用户是谁？当向FHIR服务器查询数据时，返回的数据是给谁了呢？标识用户的流程常常被称之为应用系统间的语境传递，也有诸如CCOW之类的方法可以实现。

要解决此类问题，我们需要更多有关用户身份相关的信息，也就是一到多个应用程序和FHIR服务器都能够识别出来的属性，能够确保这个人是同一个人。在FHIR中，我们认为这就是一个Identifier，很多资源、实体都有这个属性，Patient也不例外。

从另一种角度来看，应用程序的查询应该是这样的：

```
GET [base]/fhir/Condition?subject.identifier=PRP1660
```

比如查询患者标识为PRP1660的所有condition。

这里的Identifier说的是业务上的标识，比如社保卡号，身份证号，而不是服务器上该资源实例的标识，或者说是主键之类。

一个Identifier由2部分组成：标识的字符串和保持标识字符串唯一性的系统，或者说标识字符串的类型。

如何获得此类标识，与授权服务器类似，我们需要的是一个身份标识服务器，可以从中获取患者的标识-这正是OpenID Connect的作用.

OpenID Connect是一种基于OAuth2的协议，能够提供标识服务，也就是说它和OAuth2使用同样的组件和流程，但是增加了与标识相关的内容，比如除了登录到授权服务器的信息之外，包含更多与人相关的属性，诸如姓名、email、出生日期和身份标识等!

OpenID Connect中定义了与user用户相关的一些标准属性，可以在此处查看,尽管也允许你定义一些其他属性，但是identifier并不在其中。由于identifier并不是标准属性，在所有采用这种方式的应用程序之间我们需要一份协议来来表示identifier。常常将这称之为 交易双方协议或者实施指南的一部分。

从另一种角度来OpenID Connect，它利用了OAuth现有的组件，使用了现有的流程并增加了一些额外的功能。但是其中组件名称与OAuth2中叫法不一样，Client-Relying Party，Authorization server-OpenID connect provider

与 OAuth2标准流程极其相似，如下图，注意其中的不同点

1. 浏览器与APP进行交互，重定向至授权服务器。入参包括了scope参数，其中罗列了app需要的identity claim，还有一个“openid”用来标识这是一个OpenID Connect请求
2. 重定向浏览器调用授权服务器，渲染出一个登录界面。页面上常常包含了app所请求的claim，这样用户能够对其进行授权
3. 用户输入了登录凭据之后，对信息发布进行授权.
4. 详细信息发送至授权服务器，对用户进行认证，并向APP返回一个授权码
5. 应用程序拿到重定向请求和授权码之后，利用授权码作为参数发送请求到授权服务器的访问令牌节点，授权服务器校验该授权码，返回三种令牌：
    i. 访问令牌
    ii. 刷新令牌

iii. ID令牌 6.其他流程如上所述

上面省略了诸多细节，详细信息请参考 OpenID Connect标准

当然在上面的第七布也可以有另外一个单独的 'Claims Provider' – 其中只返回访问令牌/刷新令牌，可以单独调用这个claim provider，这种情况可以使用标准中定义的*UserInfo*节点。

ID令牌是OpenID Connect特有的，它是 JSON Web Token形式的，用 "*a compact URL-safe means of representing claims to be transferred between two parties*"的方式来描述 这里的claim其实指的是user的属性。其中就包含了identifier属性，格式为JSON string.

如下例所示 摘自标准) .

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

{
"access_token": "SlAV32hkKG",
"token_type": "Bearer",
"refresh_token": "8xLOxBtZp8",
"expires_in": 3600,
"id_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ.ewogImlzcyI6ICJodHRwOi8vc2VydmVyLmV4YW1wbGUuY29tIiwKICJzdWIiOiA
}
```

JSON object响应中包含了三类令牌，id令牌是JWT格式的.其中包含了三部分，用 '.'作为分隔符, 每一部分都是base64编码。中间部分包含了claim 也就是user的属性。

综上所述，OpenID Connect在OAuth2之上增加了服务器存储和传输身份标识数据的功能，APP可以在管理患者信息时可以使用。至于说服务器中最初怎么获得标识，并将其与登录的凭据相关联则取决于具体的实现。

从客户端的角度来讲，

1. 使用OAuth2
2. 在第一步调用授权服务器时，选择你所需要的claim/属性，外加一个"openid"字符串
3. 在获得访问令牌时就能够拿到所需数据

OpenID Connect的基本应用就是这样子，但是一些未解决的问题包括：

- 当用户授权APP可以访问自己的数据时，如何控制其中的粒度，比如 检验结果可以给你看，但其他数据就不行
- 鉴于OpenID Connect的'claim'对于identifier而言，我们如何表示我们需要这个字段/属性?

如何利用这些标准来保证数据源和数据服务的安全性。

FHIR: Securing an ecosystem

原文链接:FHIR: Securing an ecosystem

编者注：请关注以下原文下面的评论，**Trond Elde**提到当患者作为自身数据的使用者，**OAuth2**和**OpenID**的方式是使用的，但医疗中很多情况下，在患者授权的前提下，医疗机构扮演了患者数据使用者/用户的角色，可能需要**XACML**和**IHE Secure Retrieve (SeR)**解决此类问题

前面一篇文章fhir-and-openid-connect 中探讨了 OAuth2 和 OpenID Connect, 其中介绍了授权服务器如何认证用户，为资源服务器提供访问令牌，资源服务器根据访问令牌，确定它所能提供的信息和服务。FHIR标准 中也提到了如何解决此类问题，同时指出可以使用IHE IUA 规范 (可在此下载).

每个服务器都拥有自己的认证机制是最直接了当的，但当要在不同服务器之间形成一个生态圈的话就会变得异常复杂，可能会存在如下问题：

- 资源服务器如何保证某个访问令牌的有效性
- 授权服务器如何告诉资源服务器的访问范围，也就是患者允许你访问的数据范围

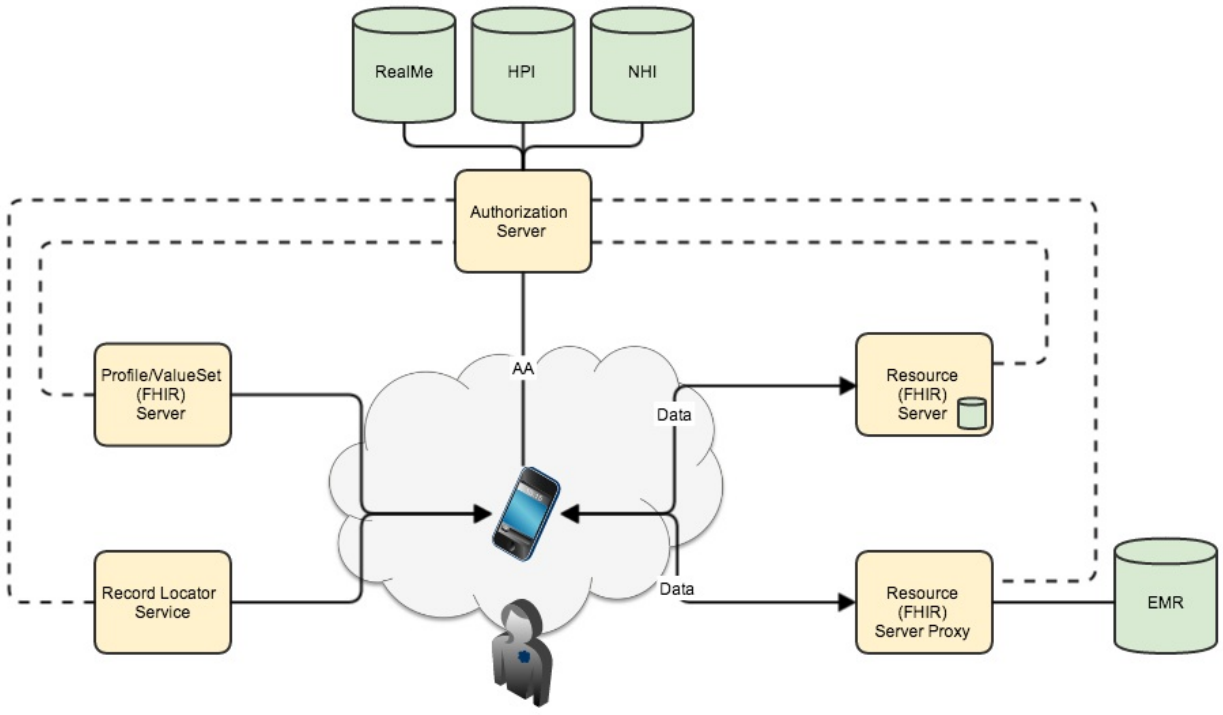
简单说明一下生态圈，也就是有哪些单独的服务器，扮演哪些角色

处于简化问题的考虑，假设我们的生态圈内只有一个授权服务器。与IHE XDS Affinity domain concept的概念类似:

“一些医疗机构，以公共的策略和公用的架构来协同工作”

区别之处在于affinity domain都有自己单独的注册库，我们这里只有一个授权服务器，

生态圈示意图:



- 授权服务器依托一些数据库来标识患者和医务人员/医疗机构.
- FHIR Profile和ValueSet 注册库/registry. profiles, ValueSets和扩展.当然也可以使用其他的注册库，主要是HL7所定义的一些扩展。

- 记录定位服务/Record Locator Service. 它扮演的是XDS Registry 的角色，上面保存了DocumentReference资源getting-documents-from-a-fhir-xds-infrastructure.
- 一些资源服务器，一个保存资源本身，其他都作为现有医疗信息系统的代理.

可信的访问令牌

授权服务器在拿到一个访问令牌后如何确保其有效性，以及用户所同意的数据发布的范围。

当授权服务器和资源服务器是同一个应用时就很简单, 在处理访问令牌的过程中，授权服务器能够在本地缓存中保存详细信息.当多个不同服务器请求该访问令牌时就变得尤为复杂.

有一些方法可供选择：

- 授权服务器通过安全的后台通道向资源服务器提供缓存有效的访问令牌 (比如Redis服务器内存中保存了有效的访问令牌). 当资源服务器接收到新的请求，先向内存中以确认其有效性和获取范围)
- 实际上访问令牌的结构比Bearer Token要复杂的多 – 比如可以是JWT (JSON Web Token)，JSON web令牌中包含了失效时间和授权服务器的签名。这就要求资源服务器要有授权服务器的公钥.

选择上面的2种的哪一个，取决于具体的实现(“Affinity Domain”的协议). 无论哪种情况，复杂度要么在服务端 要么在客户端。

范围定义

起初用户与授权服务器进行认证时，其中包含了他们允许应用程序能够以用户的身份从资源服务器上访问哪些信息.也就是请求的范围，可以在 OAuth2 标准3.3章有详细描述:

scope 参数的值是以空格为分隔符，大小写敏感的字符串列表。这些字符串是由授权服务器所定义的。 如果值中包含了多个空格分隔的字符串，它们的次序并不重要，每个字符串都限定了请求的访问范围.

访问粒度的控制如何做到，并没有共同的标准，参考Grahame提供的一些方案:

- 读取任何数据
- 读取任何未标记为保密的数据
- 读/写操作
    - 用药
    - 过敏
    - 症状
    - 其他临床信息

很明显，原则上，你可以使用一到多种策略，但在实现时具体要看情况选择.

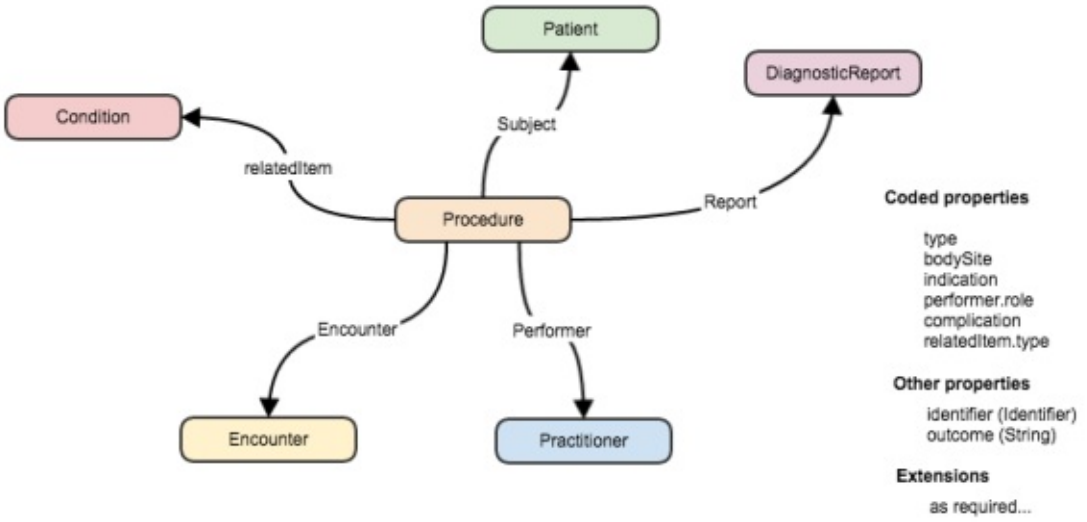
原文链接:Clinical resources in FHIR

Clinical resources in FHIR

**译者注:大多数技术标准尤其医疗健康领域的通常情况下为人所诟病，要么说不贴近临床实际的应用场景，要么说技术过于晦涩难懂，不利于实现，本文作者尝试从临床的角度去阐释FHIR中的资源，应该怎么样使用FHIR中的资源(医学概念)来表述一个完整的临床流程。主要是针对作者最近在Melbourne的一个FHIR研讨会上的PPT的介绍。

核心的理念在于FHIR资源，就如同web页面的构件一样组装起来就能够表述一个完整的临床流程，下图就是一个简单的范例,为大家展示了FHIR资源的核心特征：

- 资源之间是相互关联的 像一张网，或者说图，每个资源都是一个点
- 资源中有一些可以编码的属性 会用到各种字典
- 资源中也会有一些简单数据类型的属性、字段
- 资源也会存在扩展



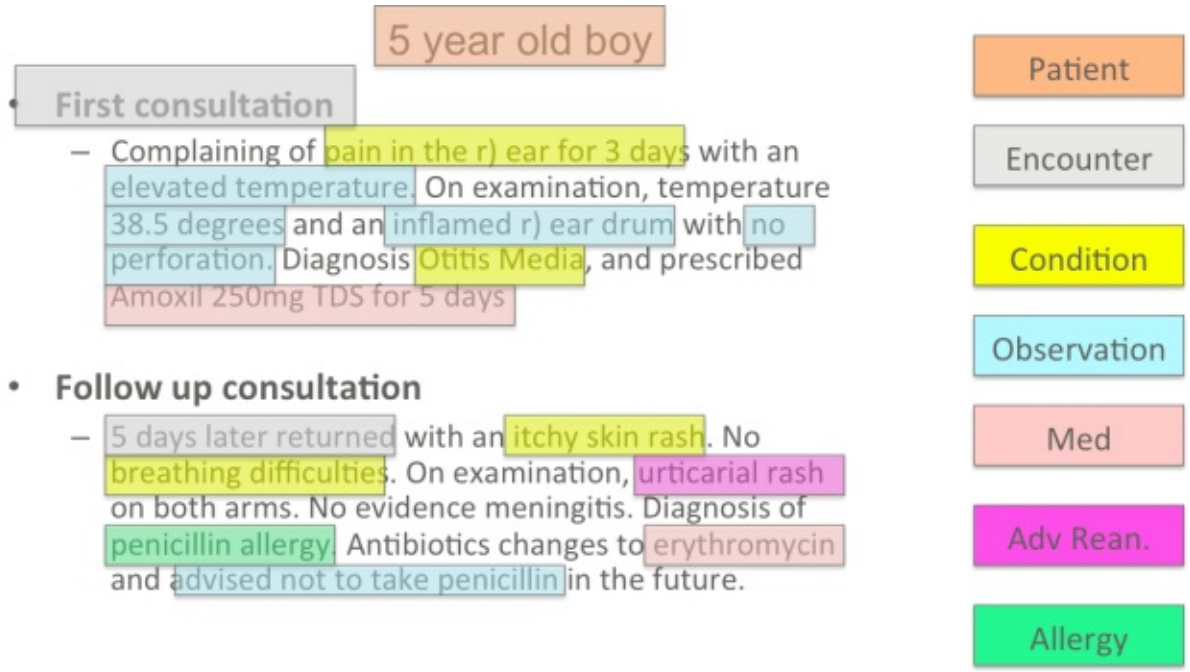
下图给大家看一个相对简单，但在全科诊疗中也很罕见的场景

5 year old boy

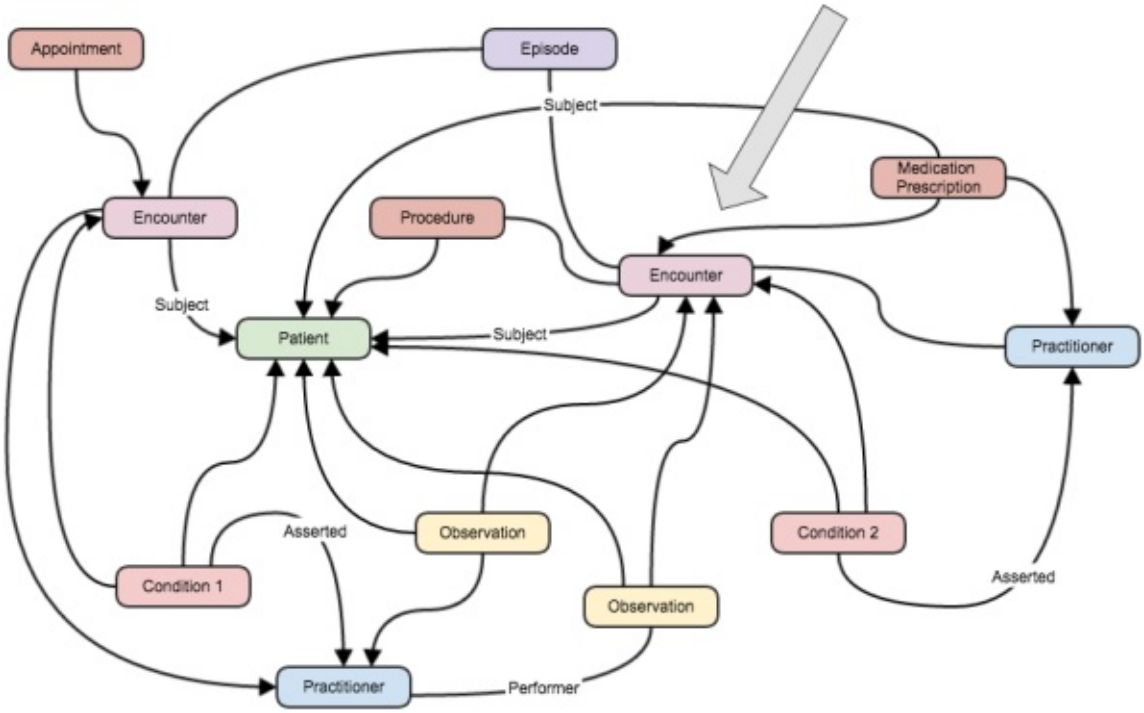
- **First consultation**
  - Complaining of pain in the r) ear for 3 days with an elevated temperature. On examination, temperature 38.5 degrees and an inflamed r) ear drum with no perforation. Diagnosis Otitis Media, and prescribed Amoxil 250mg TDS for 5 days

- **Follow up consultation**
  - 5 days later returned with an itchy skin rash. No breathing difficulties. On examination, urticarial rash on both arms. No evidence meningitis. Diagnosis of penicillin allergy. Antibiotics changes to erythromycin and advised not to take penicillin in the future.

我们在看一下上面的场景中我们需要用到哪些资源



要注意的是 上面颜色标注的也不全是对的，详情可以在Patient Care workgroup的邮件列表中进行探讨. 如果用图的形式来表示的话,可能会是下面的样子


Appointment
Episode
Subject
Medication Prescription
Encounter
Procedure
Encounter
Subject
Patient
Subject
Practitioner
Asserted
Observation
Condition 2
Condition 1
Asserted
Observation
Practitioner
Performer

原文链接:FHIR Medication lists revisited

FHIR Medication lists revisited

**译者注:用药信息是每个医疗信息标准都迈步过去的坎儿，能不能准确便捷的表达临床流程的信息流转，满足大多数医疗信息系统的需求，作者也绕着这个梗给大家分享了一些自己的想法。

如何使用List来表达患者的用药列表信息
如何使用transaction来对患者用药列表信息进行更新操作
区域范围内该共享哪些用药信息

业务需求

这里面主要是探讨如何对用药信息进行评估(medication review 查阅了一些中文材料 这里最合适的翻译是临床用药评估，见参考资料1)(当然可能是一种理想化的情况)，又大致可以分为：全面的临床评估，出院前的变更，患者个人反映自己正在服用的某些药物并不在最近的处方列表内。

对于要使用此类信息的系统而言，怎么样最简单最快的能够获得"最近的列表"(最近的一次评估，或许是一种或几种类型)以及既往的变更记录。如果能够保证此类信息及时可靠的话能够极大的降低成本。

对于每一次用药评估而言，我们需要记录：

- 评估的日期 时间
- 评估的人员
- 评估的类型(新增药物的更新操作还是说整体全面的所有用药的评估)
- 评估的地点 医院，全科诊所，药房
- 评估的赞助人员
- 评估之前的用药列表
- 评估之后的用药列表，发生了哪些变化以及发生的原因，停用的药物啊，现用药的剂量等的变化呀

对于其中每一种药物，需要记录：

- 药品名称，编码
- 药品通用名和编码
- 剂量信息
- Patient instructions
- 开药的原因
- 诊疗的时间区间

为了获得如上的评估信息，我们要能够：

- 检索病人一种或多种最近的List(用药列表)
- 根据类型，拿到list变更的摘要信息
- 检索某个List的既往变更记录
- 更新这个List
- 其他。。。。

这里我们主要探讨如何对List进行建模，后续的文章里再来看看技术上如何实现。

之前的文章里已经使用过MedicationPrescription资源，其中包括用药的原因，剂量，如何服用，治疗的时间区间等等，这个主要是下医嘱时来用(患者还没有用这些药 针对某次就诊事件而言 有限定的事件点)，这里我们使用MedicationStatement (用过的，还没用的用药记录 范围更广)。

选了MedicationStatement 意味着我们得添加一些扩展才能满足我们的需要；

- Generic medication (#generic – Reference to Medication resource). The medication resource has an ‘isBrand’ so we can tell if we need the extension.
- Reason for prescription (#reason – String)
- Patient Instructions (#patientinstructions – String)
- Duration of treatment (#duration – Period)

至于说如何在profile里面定义上面提到的扩展，取什么名字，选什么数据类型。

接下来看一看如何使用List。这里面要有 patient (List.subject), who did it (List.source) and date/time (List.date)，List.code表示List的类型(loinc 10160-0 – history of medication use).

这里可能也要定义一些扩展，

- 评估进行的地点，可以使用Location资源，定义一个ResourceReference类型的扩展即可
- 赞助者，可以使用Organization 定义一个ResourceReference类型的扩展即可

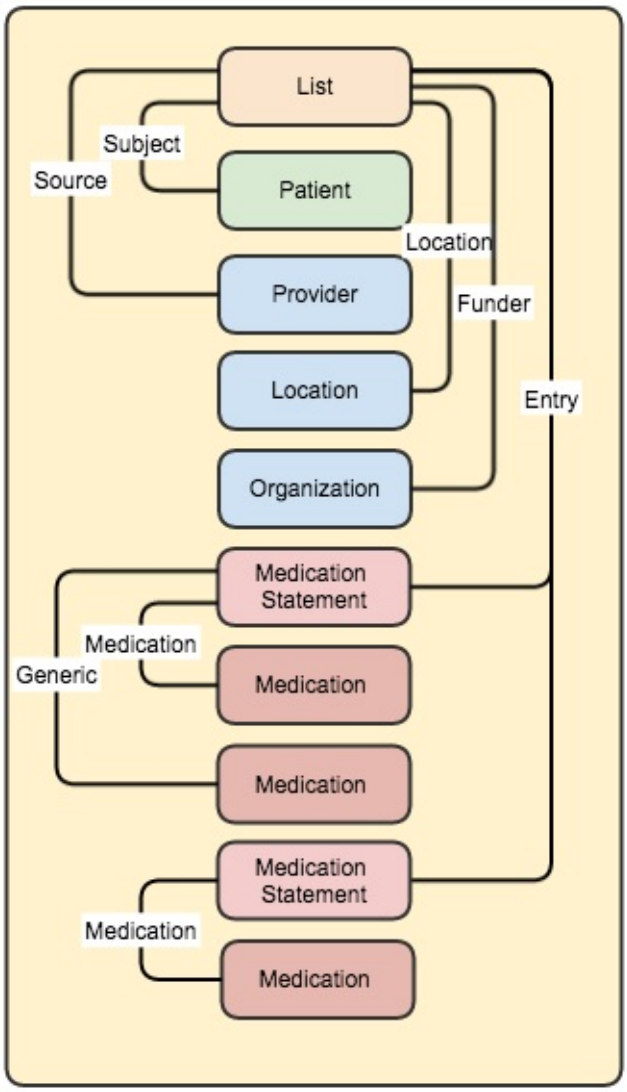
接下来，考虑如何表示变化前和变化后的List

- 可以通过将最新的List与之前的List进行比较得知变化。但如何记录变更的原因是很复杂的，同样，如何得知前一个List的信息是评估人员选择的那个List
- 一种更合理的方式是使用List.entry元素配合扩展，entry.flag用来记录工作流相关的信息，比如说用药是新增的，还是和之前一样，还是说变化了或者终止了。entry.deleted用来表示用药是否停止了，该元素是一个‘modifier’ 元素，客户端必须理解。entry.changeReason (extension)用该扩展来记录变更的理由，可以使用CodeableConcept 数据类型，预设几种理由的编码即可。

Scenario	Flag	Deleted	ChangeReason
Existing Medication	existing	false	
Changed Medication	changed	false	Reason for change
New medication	new	false	Why started
Stopped Mediation	stopped	true	Why stopped

entry.flag的数据类型为CodeableConcept，标准中给出了一个字典的例子，1.14.2.1.89.1 Patient Medicine Change Types,当然我们可以根据自己的需要进行扩展，或自己造一个字典ValueSets 。

仍需考虑的List的mode，如果是要记录一系列的评估，可能该mode值为changes， 综上所述，我们得到如下的结果



参考资料

1. 临床用药评估制度

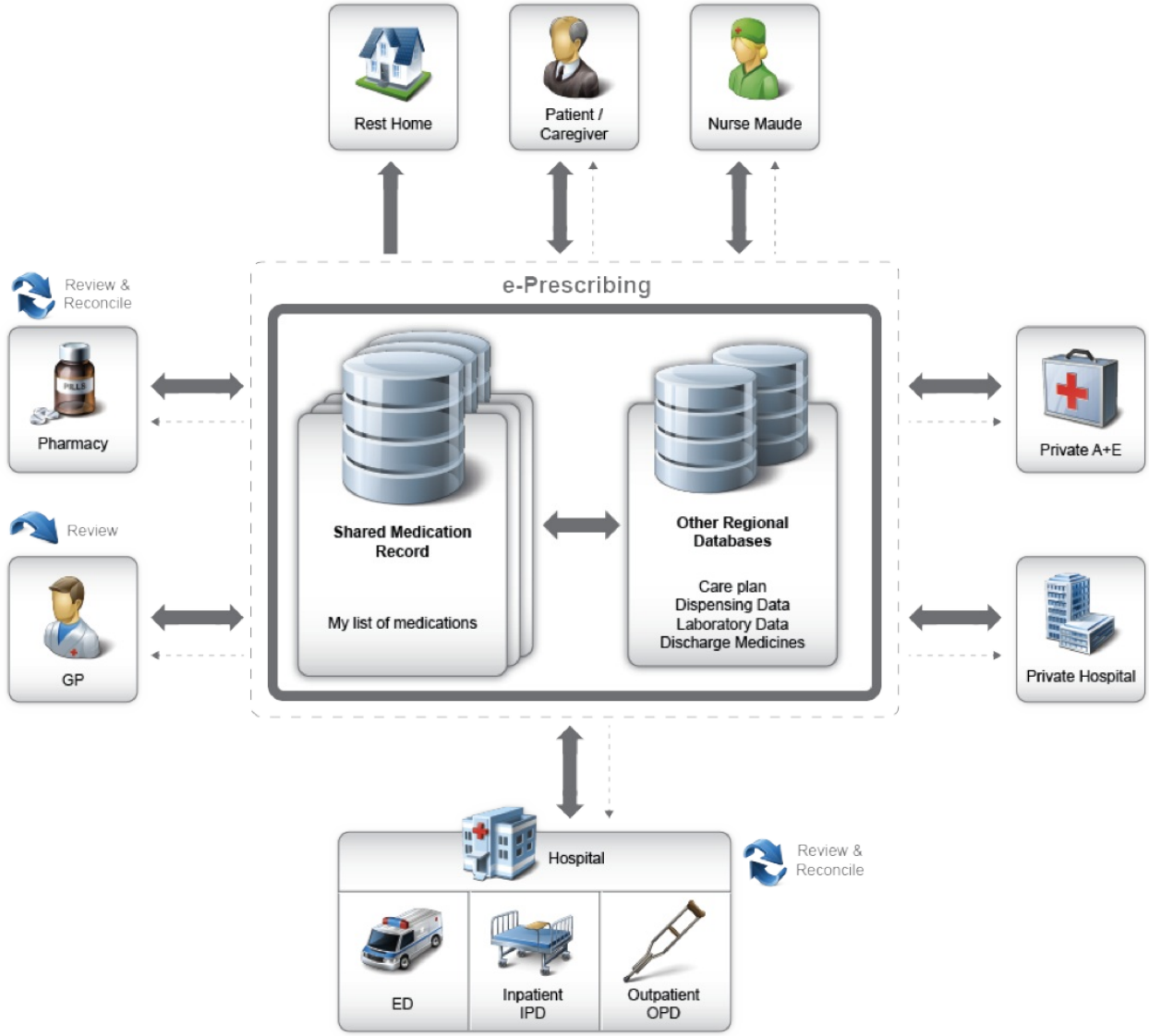
原文链接:Regional Shared Medications with FHIR

Regional Shared Medications with FHIR 区域范围内共享的用药信息

**译者注:用药信息是每个医疗信息标准都迈步过去的坎儿，作者围绕着这个topic的第一篇。这里面主要是介绍一个区域性的居民的用药信息库 ,其他的医疗信息系统在对自己系统中患者的用药信息进行操作时,都事先与用药库进行比对同步. 当然围绕着用药库可以做一些面向患者,医生的门户,适合与移动医疗的场景.

如何使用List来表达患者的用药列表信息
如何使用transaction来对患者用药列表信息进行更新操作
区域范围内该共享哪些用药信息
FHIR Medication lists revisited

整体架构图如下



用药信息库旨在让所有的医务人员,患者都能够使用.其中包含大量不同类型的用药信息.比如:

- 现用药列表(FHIR List表示)
- 现用药列表中引用的MedicationPrescription资源
- MedicationDispense资源

应用场景

考虑如下:

- 存储药房的药物配送记录 药房每次配药的时候,就生成一个MedicationDispense资源并将数据保存到用药库
- 检索一段时间内某个患者的配药数据.主要是在医师reconciling the patients' medication list的情况下使用
- 获取患者现用药列表.任何人在患者的诊疗过程中包括患者和护理人员,比如康复中心,全科诊所以及急诊室
- 获取患者现用药列表的变更记录,和前一个版本的用药列表
- 更新患者的现用药列表

这里面之涉及到用药信息的记录,并没有下医嘱(会用到MedicationPrescription资源)的功能.

Security安全

系统间的通讯间只能走SSL连接.使用oAUTH来认证和标识用户.简单起见,任何注册用户均可以访问每个患者的记录.后续会考虑一些隐私安全策略-尤其是在更新用药信息的情况.

FHIR接口

Patient相关接口 仅提供一个已有的注册服务的接口,比如说区域平台或者社交平台的帐号 tx sina之类的 Find Patient (eg GET /Patient?name=eve) Get Patient by identifier (eg GET /Patient?identifier=PRP1660)

Practitioner相关接口 与patient类似. Find Practitioner (eg GET /Practitioner?name=smith) Get Practitioner by identifier (eg GET /Practitioner?identifier=PRP1660)

MedicationDispense相关接口 从药房获取到数据,然后客户端可以和对这个列表是否正确,也就是reconciling列表的过程,列表中是否包含了患者已经拿到的药 假设药房在提交数据前先查询患者ID,然后将该MedicationDispense数据发送到用药库.因此,需要以下两个接口 Submit a dispense resource (POST /MedicationDispense) Get dispense records in the past (say) month (GET / MedicationDispense?patient={patientID}&whenHandedOver < {1 month ago} 另外,MedicationDispense 和 MedicationPrescription 资源都会调用Medication 资源,其中包含了具体的药品信息, medication资源有一个字段为code,可以使用某种药典来表示具体的药物. For example in New Zealand we have the ULM (Universal List of Medications) – a terminology based on SNOMED – thus the code 44362701000116107 refers to a 100mg tablet of aspirin

假设客户端根据这个字典来进行查询检索,我们需要code和code system.出于这样的考虑,我们在MedicationDispense 和 MedicationPrescription 资源直接包含Medication资源,而不是间接的引用它.

MedicationPrescription相关接口 MedicationPrescription主要用来记录患者服用的每种药物的详细情况,比方说药物名称,剂量,原因等.在另一篇文章里我们讨论了使用 "transaction"来批量更新药品信息,也就是说我们只需要一个根据资源标识来检索资源数据的接口 Get a single MedicationPrescription (GET /MedicationPrescription/{ID})

List相关接口

```
Get a patients list of medications (GET /Patient/{patientID}/List?code=10160-0)
Update a patients list of medications. This will be a transaction update as described earlier.
GET  /List/{listID}/_history
GET  /List/{listID}/_history/{versionID}
```

SecurityEvent相关接口 以下情况自动生成审计事件资源

- 当要获取患者的用药列表
- 当要更新患者的用药列表

  Get a bundle of SecurityEvent resources for a patient over a given time period (GET /SecurityEvent?patientId={patientID}&date > {startDate} & date < {endDate}

上面描述了一些接口功能,相较于文档存储的方式,这种方式更有利于信息的查,诸如那些人服用了某种药物,多少人有糖尿病但在过去半年内没有进行HBA1C检查

附件是一个conformance的例子

```
<?xml version="1.0" encoding="utf-8"?>
 <Conformance xmlns="http://hl7.org/fhir">
   <text>
     <status value="generated"/>
     <div xmlns="http://www.w3.org/1999/xhtml">
       <p>This conformance statement supports the Shared Medication repository, and specifies the following endpoints</
       <p>Person: Read and Search on name and identifier</p>
       <p>Practitioner: Read and Search on name and identifier</p>
       <p>MedicationDispense. Create, and search on patient,whenHandedOver</p>
       <p>MedicationPrescription. Read. </p>
       <p>List. Create and search on code,patient. Version read.</p>
       <p>SecurityEvent. Search on patient,date</p>
       <p>Transaction interfaces to update the Medication List</p>
     </div>
   </text>

   <identifier value="68D043B5-9ECF-4559-A57A-396E0D452311"/>
   <version value=".1"/>
   <name value="My List Of Medicines (MLOM) Conformance Statement"/>
   <publisher value="Elbonian MOH"/>
   <telecom>
     <system value="email"/>
     <value value="wile@elbonia.govt"/>
   </telecom>
   <description value="The FHIR endpoints required to support a regional Medication repository - My List Of Medicines"/
   <date value="2012-10-14"/>
   <software>
     <name value="MLOM"/>
     <version value="0.34.76"/>
   </software>
   <fhirVersion value="0.12"/>
   <acceptUnknown value="false"/> <!--   this system does not accepts unknown content in the resources   -->

   <!--   this system can do either xml or json. (Listing both implies full support for either, with interconversion)
   <format value="xml"/>
   <format value="json"/>
   <!-- We only support REST interfaces at this time. This includes transaction to the server root to update the List--
   <rest>
     <mode value="server"/>

     <!-- SecurityEvent record -->
     <resource>
       <type value="SecurityEvent"/>
       <operation>
         <code value="read"/>
       </operation>
       <searchParam>
         <name value="patient"/>
         <type value="reference"/>
         <documentation value="Lookup by patient."/>
       </searchParam>
       <searchParam>
         <name value="date"/>
         <type value="date"/>
         <documentation value="Lookup by date the event occurred."/>
       </searchParam>
     </resource>

     <!-- MedicationDispense record -->
     <resource>
       <type value="MedicationDispense"/>
       <operation>
         <code value="create"/>
       </operation>
       <operation>
         <code value="read"/>
       </operation>
       <searchParam>
         <name value="patient"/>
         <type value="reference"/>
```

```
            <documentation value="Lookup by patient."/>
          </searchParam>
          <searchParam>
            <name value="whenHandedOver"/>
            <type value="date"/>
            <documentation value="Lookup by date the medication was given to the patient."/>
          </searchParam>
        </resource>

        <!-- MedicationPrescription resource. The prescription records are all created through the 'transaction' process s
        <resource>
          <type value="MedicationPrescription"/>
          <operation>
            <code value="read"/>
          </operation>
        </resource>

        <!-- List resource. Used to support the List of Medications. -->
        <resource>
          <type value="List"/>
          <operation>
            <code value="create"/>
          </operation>
          <operation>
            <code value="read"/>
          </operation>
          <operation>
            <code value="vread"/>
          </operation>
          <searchParam>
            <name value="patient"/>
            <type value="reference"/>
            <documentation value="Lookup by patient."/>
          </searchParam>
          <searchParam>
            <name value="code"/>
            <type value="token"/>
            <documentation value="Lookup by code - this will be for the MLOM"/>
          </searchParam>
        </resource>

          <!-- The Practitioner resource endpoint -->
          <resource>
            <type value="Practitioner"/>
            <operation>
              <code value="read"/>
            </operation>
            <searchParam>
              <name value="name"/>
              <type value="string"/>
              <documentation value="Lookup by practitioner name. All parts of the name are searched."/>
            </searchParam>
            <searchParam>
              <name value="identifier"/>
              <type value="token"/>
              <documentation value="Lookup by identifier. Both active and inactive practitioners will be returned."/>
            </searchParam>
          </resource>

        <!-- The Patient resource endpoint -->
        <resource>
          <type value="Patient"/>
          <operation>
            <code value="read"/>
          </operation>
          <searchParam>
            <name value="name"/>
            <type value="string"/>
            <documentation value="Lookup by patient name. Only active patients will be returned. All parts of the name are
          </searchParam>
          <searchParam>
            <name value="identifier"/>
            <type value="token"/>
            <documentation value="Lookup by identifier. Both active and inactive patients will be returned."/>
          </searchParam>
          <searchParam>
            <name value="birthDate"/>
            <type value="date"/>
```

```
          <documentation value="Lookup by patient birts date. Supports the :before and :after modifiers to allow for age
        </searchParam>
      </resource>
    </rest>
  </Conformance>
```

原文链接:Updating the Medication List

Updating the Medication List 如何更新用药列表信息

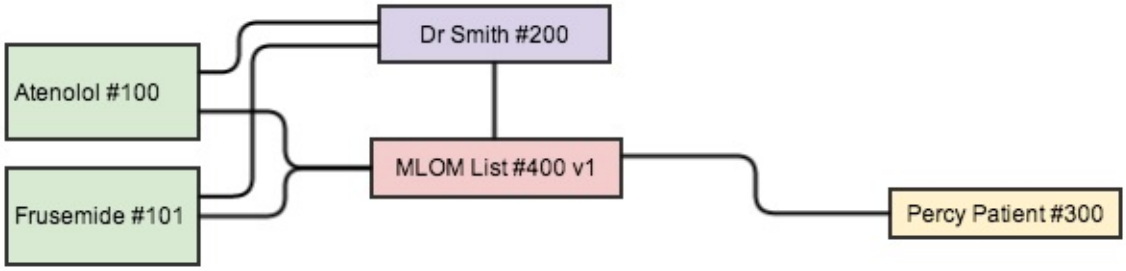
**译者注:用药信息是每个医疗信息标准都迈步过去的坎儿，作者围绕着这个topic的第二篇。在知道了如何用List资源来表示患者的用药列表之后,这里主要探讨如何对列表中的信息进行更新操作.比方说,医师调整了患者正在服用的药物,想要记录这些变更.

如何使用List来表达患者的用药列表信息 如何使用transaction来对患者用药列表信息进行更新操作 区域范围内该共享哪些用药信息 FHIR Medication lists revisited

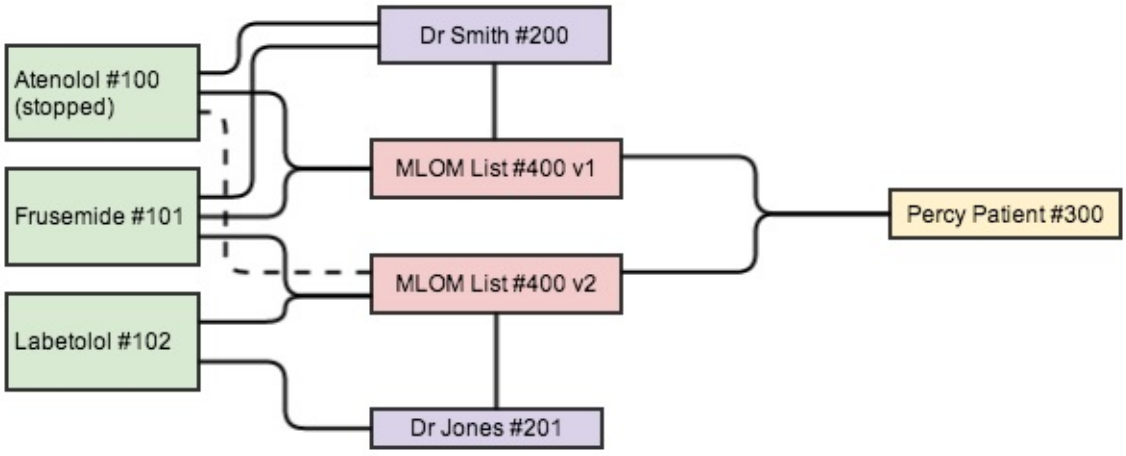
List资源用来表示其他资源的集合,其中可能会包含大量的存储于其他地方的资源数据.

以患者正在服用2种药物的简单场景为例:

- 一个Practitioner资源 ,下达医嘱的药师和用药列表的作者/来源
- 一个Patient资源
- 2个MedicationPrescription资源,分别描述药物信息
- 一个List资源引用了上述资源



资源间的线表示资源引用. 如果Atenolol停用之后,用Labetolol替换,又会如下



注意:

- 这样List资源就产生了一个新版本 ID相同
- 新版本的List的医生是Dr Jone
- 一般而言,下医嘱的和List作者是同一个人,但不异地给你是
- 新版本的List保留了对Atenolol的引用,事实上是不必要的,但这样做能够表示在开始服用Labetolol的时候停用了Atenolol(及其原因)

假设我们拿到了患者ID,可以通过如下流程来实现

1. 获取已有的用药列表GET /Patient/100/List?code=http://loinc.org|10160-0 which is a FHIR query that will return a

bundle containing the List
2. 获取我们要停用的medicationPrescription GET操作,将其status字段值设为'nullified',执行PUT更新操作.
3. 新增一个MedicationPrescription资源,Post新增
4. 在List资源中对应的要停用药物的entry中 flag字段设为‘cancelled’,deleted字段设为true,可以添加一个extension来表示停药的原因
5. 在List为新增的药物添加一个entry
6. 对新的List数据进行PUT操作 获得新版本的List

这个流程整体来看并不复杂,但是后面的步骤是以前面的步骤为前提的,要么都成功,要么都失败,这实际上就是数据库操作中的transaction的概念

一种方式是

- 客户端按照上述流程依次操作,和对每一步是否成功,是否需要在过程中进行回滚操作,是否同时有其他客户端对同样的数据进行更新操作
- 在客户端生成或更新所有资源数据,将所有资源打包成一个bundle,通过transaction请求发送到服务器上,服务器负责保证整个操作是否成功失败

目前FHIR DSTU1中bundle是Atom feed格式的,可以是XML或JSON.每个entry表示一个FHIR资源. Entry.id表示的是资源的逻辑ID,非特定版本相关的ID,每个资源都只有一个逻辑ID Entry.link表示的特定版本相关的ID,可选字段,但在针对版本的更新中能够避免不不要的冲突 使用Bundle之后的流程如下:

1. 生成新的Bundle,
2. 获取已经存在的用药列表 拿到List的ID值
3. 获取我们要停用的medicationPrescription GET操作,将其status字段值设为'nullified',将变更后的资源添加至bundle当中,将entry.id值设为 medicationPrescription的ID.
4. 新增一个MedicationPrescription资源,添加到bundle当中,使用CID scheme的方式分配一个临时值到entry.id字段中
5. 在List资源中对应的要停用药物的entry中 flag字段设为‘cancelled’,deleted字段设为true,可以添加一个extension来表示停药的原因
6. 在List资源中新增一条List.entry来保存对新增药物的引用,ID使用3中的临时值
7. 将更新后的List资源添加至bundle当中,entry.id设为List的ID
8. 通过Post操作提交bundle到服务器上

最后提交的bundle中包含

- 2 MedicationPrescription resources (1 new & 1 updated) and
- 1 List resource

服务器拿到bundle之后,处理过程和单独提交时的操作一致,返回成功或失败的结果.服务器也可以自行实现其他的一些业务处理逻辑或校验逻辑 比如,核对bundle中List资源的版本和服务器上最新的版本一致.

对于客户端而言,这种方式简化了工作量,将复杂度抛给了服务端.

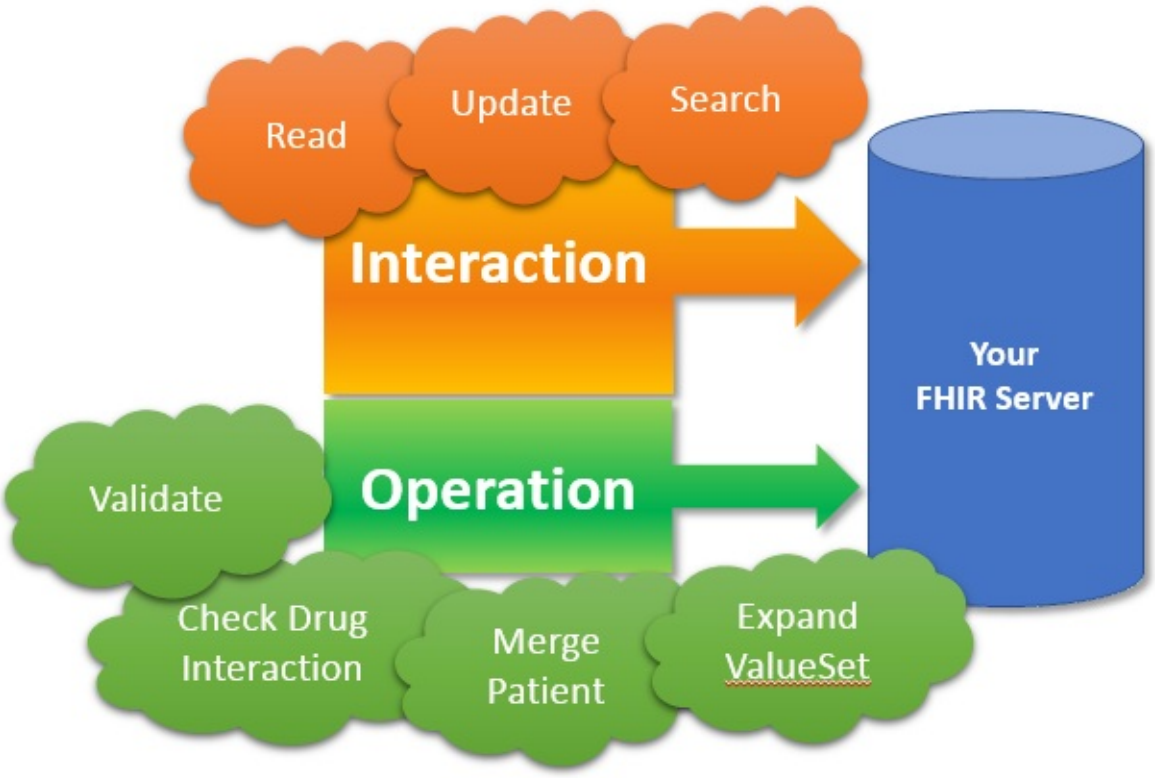
原文链接：The new DSTU2 Operations framework

operation 框架 的诞生

众所周知，你可以使用 **extension** 来对FHIR中核心数据模型进行自由扩展，以满足系统和特殊场景下的数据需求。同时可以在FHIR 服务器上 发布这些扩展定义的详细信息，这样大家就可以找到你所定义的扩展的具体含义和表达的是那类数据。DSTU2 中 FHIR 引入了另一种扩展机制 **operation** 框架.除了FHIR 中定义好了的REST接口之外，你可以添加额外的操作/接口，这样就可以添加一些 只是针对某个系统的功能。同样的，你也可以将这些operation/操作/接口定义发布到 FHIR服务器中，大家就可以找到你所定义的扩展的具体含义和如何调用这些 operation/操作/接口。

那么到底在FHIR的REST接口中，“operation”具体指的是什么呢？DSTU1中对于这个问题还不明朗，FHIR中只是包括了一些定义好的interaction/接口，诸如read、search、validate等。 另外，DSTU1中有一个叫“ *query*”的查询参数，用于表示自定义的查询。 有趣的是，标准中所定义的唯一一个查询是值集的扩展，由此引发了 值集扩展到底算不算查询的争论。 另外，仍然有必要添加一些有副作用的*operation*/操作/接口，也要能使用复杂类型的参数。“ query”的接口不能满足所有这些要求，由此产生了operation框架.

当你调用 FHIR 服务器中的一些方法时，要么调用的是最基本的核心交互，也就是CRUD 操作，查询和变更记录(CRUD operations, search and history)。 要么就是operation/操作/接口。标准中会包括一些预先定义好的operation/操作/接口，当然你也可以自行定义新的operation/操作/接口。



operation 的调用

Operations大多数是使用POST方法到FHIR endpoint，Operations的名称是以 “$” 号为前缀，比如 `POST http://fhir.someserver.org/fhir/Patient/1/$everything` 不管operation是不是幂等的，都可以使用GET方法。 可以在四种不同类型的FHIR endpoint中调用 operation ：

- FHIR 服务的根节点：(e.g. http://fhir.someserver.org/fhir)，这些operation是在整个服务器层面上执行的。比如：返回服务器中已知的所有扩展
- 某个资源类型的节点：(e.g. http://fhir.someserver.org/fhir/Patient)，这些operation是在所有该类型的资源实例层面上执行的

- 某个类型的某个资源实例的节点：(e.g. http://fhir.someserver.org/fhir/Patient/1) 这些operation是在该类型的某个资源实例层面上执行的，比如上例中的$everything operation
- 某个类型的某个资源实例的某个版本的节点： (http://fhir.someserver.org/fhir/Patient/1/_history/4)。这些operation是在某个类型的某个资源实例的某个版本层面上执行的，要注意的是，这些操作不应该改变实例的数据，主要是为了管理 profile 和tag等元数据。

调用请求的body中包含了一个特殊的资源parameters，该资源是一些键值对的集合， 其中的值可以是任意的基本数据类型或是复杂数据类型，亦或是完整的某个资源。除此之外，还可以使用格式化成查询参数类型的字符串。 调用完成之后，operation 返回另一个 Parameters 资源。其中包含一到多个输出 Parameters资源。也就是说 FHIR operation 可以接收任意输入，返回一些输出参数。 由于引入了 Parameters 资源， 这样POST请求的body和返回的结果总是一个资源。如果没有这种机制，我们需要对FHIR 接口和 FHIR的序列化格式进行扩展来满足那些body中包含复杂数据类型的情况。

operation 定义的发布

了解了如何调用之后，我们来看看如何发现新的 operations 。这里要用到OperationDefinition resource,与extension一样，它是一种对 operation 定义结构化的表达方式，包含了一些元数据和输入输出值参数的详细信息。由于 OperationDefinition 本身也是一种资源，可以在FHIR 服务器或FHIR仓库中进行发布。 开发工具可以利用该定义，生成API 接口，如我们的发布工具生成的这样 ，亦可以利用它来生成相关的文档。

DSTU2中包含了一些预先定义好的 operations，在DSTU1中的一些interaction，在DSTU2中被作为operations：

- Validation ($validate) –取代了DSTU1中的 _ validate interaction。可以根据资源内容You can send in a Resource (or a reference to a resource) and a reference to a StructureDefinition to get your resource validated.
- Tag operations ($meta, $meta-add, $meta-delete) – 取代了DSTU1中的tag operations,用于给某个资源实例添加 profile, tag and security tag metadata 。
- Mailbox endpoint ($mailbox) – Delivers a message to a FHIR server. 除此之外，spec中现在定义了完整的ValueSet operation，这样子FHIR 服务器就相当于术语服务器,能够提供一些CTS的功能。

每类资源介绍的"operation"标签下面包括了定义好的所有operation。

原文链接：Responsibilities of a FHIR client

FHIR 客户端的职责

当在我们Orion Health 公司开始进行实质性的FHIR开发工作的时候，我们就FHIR接口做了大量的限制和约束，这些决定无疑会影响客户端使用我们所提供的服务。 通常，我们使用Conformance资源来记录这些限制和约束的决策，这也让我不禁想起一个FHIR 客户端总体上应具备的职责。

尽管对FHIR是一种比其他标准更易于开发的观点不存在过多分歧，但它仍然是为了解决医疗信息的交换而生的，而问题本身是极其复杂的。 尽管设计人员已尽其所能将复杂度转移至服务器端而非客户端，实现同样的目的仍然存在多种方式，这也就意味着除了遵循spec中资源和接口的规定，FHIR 客户端仍然有很多额外的职责。

让我们逐一来看一看。

扩展

首先是扩展，FHIR的一大卖点就是扩展，有了它，基础的资源才能够足够简单，FHIR的内在扩展机制使得具体系统的开发可以按需添加额外的属性字段到资源中去，对于客户端而言，要能够理解它所不知道的扩展，也就是说，客户端在处理资源内容时需要特别地查找扩展，根据 客户端的了解程度来做出决策。

对于标化扩展(字段名称为extension)，其中包含的信息对客户端来讲可能是有用的，但忽略其中的内容从临床上来讲也是安全的。比如我们要在Patient中记录患者的宗教信仰，可能是很有意思的信息，但对于临床来讲不是很重要。

spec中并没有指明客户端该如何处理未知的扩展，其中强烈建议将扩展的含义包含在资源的文本描述之中(narrative)，这样至少可以将其展示给用户，但这取决于具体的开发实现。

同样需要注意的是，扩展可以出现在资源的任何未知：

- 作为一个新字段
- 作为一个已有字段的修饰
- 扩展某个数据类型

对于Modifier类的扩展(字段名称为 modifierExtension)，则是另外一种情况。从临床安全的角度来讲，这类信息对于客户端来讲是必须理解的。比如，你如果想表达某个患者并没有某种病情，那么就新建一个negative的扩展。另一个例子是表达患者并没有得truncal rash。很显然，此类扩展完全改变了资源内容的含义，从临床上来讲，客户端忽略此类信息是不安全的。spec中明确指出：

- 理解此类扩展的含义
- 拒绝处理包含此类扩展的资源
- 以警告形式展示给用户来做出决定

Modifier类的扩展出现的位置相对受限，尤其是它不能更改包含数据的属性/字段，也就意味着着它只能作用于资源类型或类的层面上。

```
客户端必须查找扩展，尤其是modifierExtension
```

资源间的引用

FHIR 中充满了资源间的链接，了解如何理解引用对于客户端是很重要的。大体上，也是有2类引用。

最常用的是对独立资源的引用，可以是同一个服务器上的本地资源，也可以是不同服务器上的资源。每个资源都有一个唯一id，但获取资源的具体 机制则大不相同。 可能是客户端中已经拥有的bundle中存在的资源内容的一个copy副本，或者是客户端需要直接从某个服务器中获取。

另一种则是内嵌资源，这种常用于被嵌入的资源不是独立存在的，比如服务器中只使用药品编码的话，MedicationPrescription中可能包含一个内嵌的Medication 资源，内嵌资源的ID是以#开头的，内嵌的资源除了不能有text字段之外，和其他资源都是一样的。 (内嵌的资源不能再内嵌资源)

profile

客户端不需要过多操心的是Profile，由于profile基本上就是一个如何使用资源来满足某个特殊案例/场景需求的声明，在不知道资源遵循哪个profile的前提下， 客户端应该能处理任意的资源,如果资源中不存在某个值，客户端不能根据profile得到默认值。

也就是说客户端不能做预设，不管资源声称遵循哪个profile，客户端都要进行核对/校验。比如资源遵循某个profile(其中将一个可选字段改为必选) 客户端必须要核对，而且要能处理存在错误或省略的情况。可编码字段是另一种情况，不管profile有没有强制规定使用某个特殊字典或值集，不意味着资源中数据必须遵循这个约束。

综上所述，有3点：

- 查找extension modifierExtensions，定好处理无法识别扩展的策略
- 要知道被引用的资源可以出现在不同的地方
- 对资源中的数据不要做预设

FHIR标准与V2 消息

主要是对已经大量使用V2消息进行数据交换的遗留系统向FHIR 标准迁移的思路和过程中遇到的问题进行探讨。

原文链接:Mapping HL7 Version 2 to FHIR Messages

Mapping HL7 Version 2 to FHIR Messages FHIR 消息：第一篇——V2 消息与 FHIR 消息的映射

译者注:消息是医疗信息交换的一种重要模式，从**HL7 V2 V3**到**X12**等。对于已经应用了**HL7 V2** 消息的系统来讲，如何迁移到 **FHIR** 消息中来，是我们接下来要探讨的话题**.** 由于原文是**2014**年**10 05**日撰写的，但是**FHIR Dstu**在过去的日子里发生了较大的变化，所以我根据最新的版本对原文中的一些内容进行了修正。 全文以**ORU^R01**消息为例，简述了该**V2** 消息应该包含哪些字段，字段的含义，如果用**FHIR** 消息表示该**V2**消息又要使用到哪些**FHIR** 资源，资源的字段又如何赋值。

HL7 V2消息和FHIR 消息说明

为了简便起见，这里我们使用 HL7 V2.4 中的ORU^R01消息作为案例，利用它得到一个FHIR 消息， 然后将该消息提交到 FHIR 服务器中，最后将实际的数据存储到一个 Observation里去。

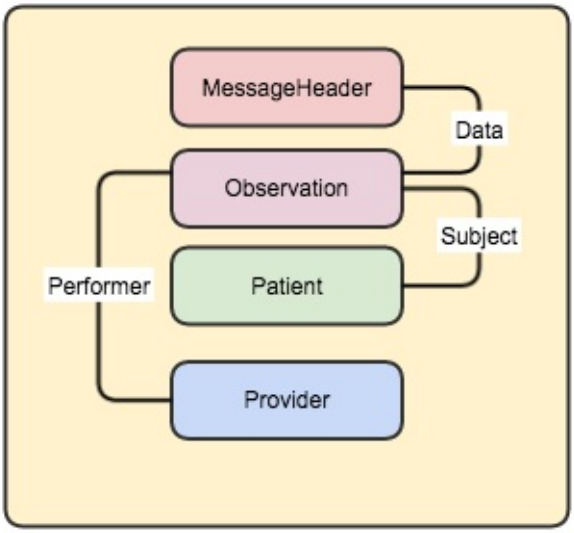
实际的应用中可能会存在多个 OBX 区段segment，每个 OBX 区段里都包含了单独的观察结果，每个OBX 区段都可以表示成消息中的一个Observation。我们也忽略了一些区段，这里只展示我们的数据会用到的一些区段。假如要能够实现双向的转换，也就是说要能够从 FHIR 消息转换成 V2 消息的话，我们可能需要更多的数据，甚至要用到一些扩展。这里的案例中只考虑根据一个 Radiology observation和与之相关的内容来创建一个 FHIR 消息。

如下就是一个 HL7 V2 消息的范例

```
MSH|^~\&|Amalga HIS|BUM|New Tester|MS|20111121103141||ORU^R01|2847970-201111211031|P|2.4|||AL|NE|764|ASCII|||
PID||100005056|100005056||Dasher^Mary^""^^""|""|19810813000000|F||CA|Street 1^""^""^""^34000^SGP^^""~""^""^""^""^Danlin
PV1||OPD   ||||""^""^""||||CNSLT|||||C|VIP|||6262618|PB1|||||||||||||||||||||||||20101208134638
PV2|||^Unknown|""^""||||""|""|0||""|||||||||||||||||||||||||||||HP1
ORC|NW|""|BMC1102771601|""|CM||^^^^^""|||||||||""^""^^^""
OBR|1|""|BMC1102771601|""^Brain (CT)||20111028124215|||||||||||||||||CTSCAN|F||^^^^^ROUTINE|||""||||||""||||||||||^"'
OBX|1|FT|""^Brain (CT)||++++ text of report goes here +++|||REQAT|||F|||20111121103040||75929^Gosselin^Angelina
`
```

其中包含了7个区段：

区段名称	用途	对应的**FHIR** 资源
MSH	消息头 消息的元数据	MessageHeader
PID	患者的身份标识	Patient
PV1	就诊信息	该案例中未使用到
PV2	就诊的额外信息	该案例中未使用到
ORC	医嘱信息	该案例中未使用到
OBR	观察的申请	Observation
OBX	Observation观察项的值	Observation.performer



而对应的 FHIR 消息结果大体如下所示

其中包含了如下的资源

- MessageHeader
- Observation
- Patient (Subject of the Observation)
- Provider (Performer of the Observation)

有了上面的知识之后，我们看一下如何根据 V2 消息中的数据来得到这样一些对应的资源。

如何生成 **FHIR** 消息

Bundle

在以前的版本中 Bundle 是遵循atom feed标准的，但最新的版本中已经将其结构调整为 FHIR 自定义的一个资源。

> FHIR 中， Bundle.type=message， Bundle.entry.resource[1]=MessageHeader。每个消息包含2个标识符，第一个Bundle.id指的是表示这个消息的资源的标识，而第二个MessageHeader.identifier指的是消息自身的标识,这个标识最好使用OID或者是UUID，保证它是全球唯一的。而Bundle.id则随消息的发送而变化，也就是说同一条消息的内容，第一次发送和第二次发送Bundle.id是不同的。

MessageHeader

在表示 FHIR 消息的 Bundle 里， MessageHeader 是Bundle.entry.resource的第一个资源，MSH 区段 与 MessageHeader 字段间的映射关系如下

字段名称	**V2** 区段名称	描述
MessageHeader.Identifier	MSH-10	消息自身的标识符
MessageHeader.Timestamp	MSH-7	消息的发送时间
MessageHeader.Event	MSH-9.2	消息的类型 详细取值请参考http://hl7.org/fhir/message-events
MessageHeader.Source.name	MSH-3	发送消息的系统名称
MessageHeader.Source.software	MSH-3	发送消息的系统、软件名称

MessageHeader.Source.endpoint	MSH-24	发送系统的网络IP
MessageHeader.Destination.name	MSH-5	接收系统的名称
MessageHeader.Destination.endpoint	MSH-5	接收系统的名称
MessageHeader.data		References to the 'root' resource of the message.

- MessageHeader.Identifier是由发送系统设置的，应该在发送系统的范围内是唯一的，理论上应该是全球唯一的，但对于V2来说是没有办法保证这点的。
- MessageHeader.Source.software字段值建议使用SFT区段，但该区段只有在V2.5以后的版本中才存在，这里就直接用了source.name
- MessageHeader.Event这里我们使用 observation-provide ，如果在 FHIR 定义的消息类型中找不到合适的，我们也可以使用V2的消息类型
- enterer, author and receiver中可以表示消息是由谁生成的，发送给谁的，如果有数据的话就可以赋值。
- MessageHeader.data是对具体的 Observation 的引用，如果我们有多个 Observation的话，我们可能要用到一个List来组合这些Observation。

Observation

Observation 中包含了放射检查结果的数据。

字段名称	**V2** 区段名称	描述
Observation.code	OBX-3	观察项的类型
Observation.valueString	OBX-5	观察项的值
Observation.interpretation	OBX-8	对观察项的值的解释 高、低、正常等
Observation.comments	NTE-3	对观察值的评论、批注
Observation.appliesDateTime	OBX-14	The time or time-period the observed value is asserted as being true.
Observation.issued	OBR.22	观察的日期时间
Observation.status	OBX-11	观察项的状态 到底是临时报告还是最终报告 到底是审核和终止
Observation.reliability	OBR-25	质量问题对观察值的影响程度的估计
Observation.identifier	OBX-21	观察项的标识
Observation.subject		观察的对象 人、物件、地理位置
Observation.performer		观察的执行者.

- OBX-3中保存的是观察项的名称 类型 数据类型为CE,
- Observation.value[x]中表示的观察项的具体值，在V2 中，我们要考虑下面三个部分：
  - OBX-2 中是观察项的类型，对于放射学检查结果，可能是FT或者ST区段，Observation.value[x]可采用string类型. 具体数据类型的选择要根据OBX-2来确定。
  - OBX-5 中是观察项的具体值。
  - OBX-6是观察项值的单位。
- 我们使用 OBR-22 中的数据来表示报告的发布日期时间
- OBX-8 (Abnormal Flags) and OBX-9 (Probability) 所表示的数据均是说结果是期望的范围之内，而非结果是否可信。这里使用OBR-25 (Result Status) 中的数据来给Observation.reliability赋值
- V2中如果要对OBX 中的结果进行批注，需要在OBX后面紧跟一个 NTE区段来表示。
- Observation.text的值可以从OBX-2中拿到，但具体的处理暂不讨论，

Patient

FHIR 中 Patient 用一个单独的资源来表示，甚至可能与 Observation 都不在同一个服务器上, 在Observation 中要引用 这个 Patient 资源，那么首先要找到Patient 的url。如果找不到的话我们要新建一个 Patient 资源。

首先根据 PID 区段中的值在服务器中进行查找，PID 区段中的患者标识都是 CX 数据类型的，与FHIRs 中 identifier数据类型是等同的。如果服务器中能够找到对应的记录，直接引用该URL即可，如果找不到的话就使用 PID 区段中的值新建一个患者资源。

查询患者资源的请求如下所示:

```
GET [patientserver]/patient?identifier={identifier}
```

- 如果正好找到一条记录， Observation.subject引用即可。
- 如果找到不止一条记录，我们认为存在错误并拒绝该消息或者进行认为干预，视具体情况来定
- 如果找不到一条记录，我们要先排除 Patient 和Observation 是不是在同一个服务器上
  - 如果在同一个服务器上且找不到记录，直接使用PID 区段中的值新建一个患者资源，这时候要给patient分配一个格式如 cid: prefix的标识
  - 如果不在同一个服务器上且找不到记录，我们新增一条记录，保存到服务器上并获取ID。用返回的ID构建新的 patient 资源替换消息中的内容即可 我们也可以使用transaction机制来将内部检索隐藏到服务器上去。

Provider

observation 有一个performer字段用于记录完成、执行观察的设备、人。我们的案例中Provider是一个设备。

与Patient 中很类似，我们要找到这个具体的资源，然后引用即可。但有一些额外的考虑因素：

- performer 的数据可能存在于 V2 消息的多个地方，具体得视 V2 的消息类型来定
- 根据我们选择的情况，可能会出现信息不足以构建一个新的资源的情况出现。

在我们的案例中，选择 OBX-16-responsible observer.该字段的数据类型为XCN (Extended Composite ID Number and Name for Persons) –也就是说如果要新建资源的话数据是足够的。

总结

V2 具体实现的差异可能会使得 V2 消息 到FHIR 消息的转换异常困难，具体情况还是得具体分析

原文链接:FHIR Messages – part2

FHIR Messages – part 2 FHIR 消息：第二篇——V2 消息与 FHIR 消息的工作流程和架构

译者注:消息是医疗信息交换的一种重要模式，从**HL7 V2 V3**到**X12**等。对于已经应用了**HL7 V2** 消息的系统来讲，如何迁移到 **FHIR** 消息中来，是我们接下来要探讨的话题**.** 由于原文是**2014**年**10 06**日撰写的，但是**FHIR Dstu**在过去的日子里发生了较大的变化，所以我根据最新的版本对原文中的一些内容进行了修正。 本文以放射学影像报告为例，说明 了系统如何从**V2** 消息转换**FHIR** 消息的整个流程

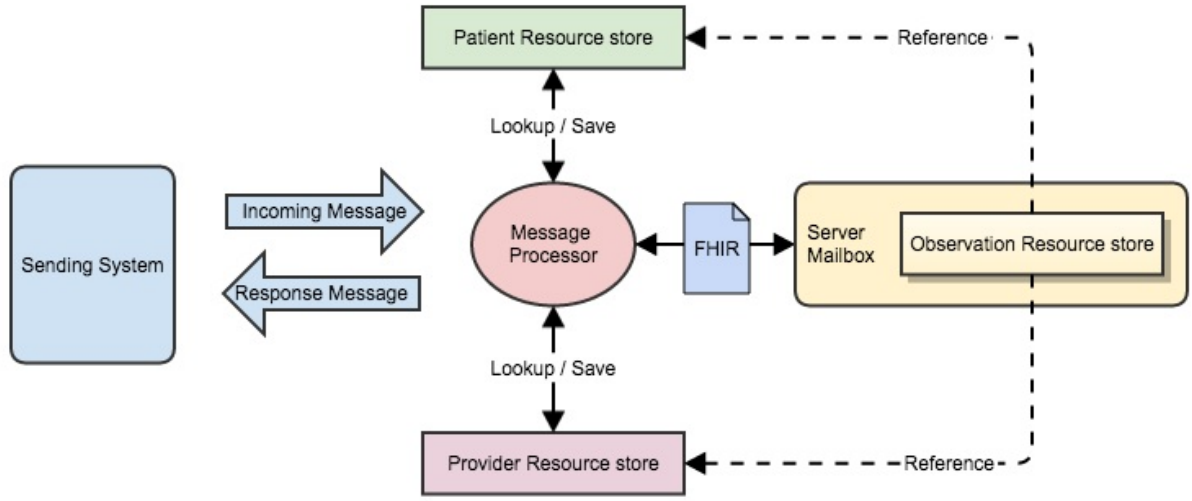
整体流程说明

在第一篇里面我们给了一个简单的例子，演示了 V2 消息和FHIR 消息的字段间的具体映射关系，这篇里着重探讨一下工作流程和架构相关的一些内容。

本文中仍然沿用第一篇中的放射学报告的例子来进行说明.

FHIR 消息架构中介绍了消息架构的前提条件和消息交换的模式，这里不在赘述。

简易的架构图如下所示:



消息处理器接受源系统发送的V2消息，将其转换成FHIR消息提交至 'Mailbox'服务器节点，该服务器节点处理之后将反馈给消息处理器一个FHIR 消息，消息处理器紧接着将该FHIR 消息转换成 V2消息反馈给源系统。整个架构中我们认为系统间的通信是异步的，但mailbox服务器内部的处理是同步的。

"Server Mailbox" 也就是实际上存储 Observation 的服务器。接口'mailbox'负责处理消息。 图中 Patient and Provider的存储是与"Server Mailbox" 分离的，但实际上可能是同一个。

假设Message Processor 到 mailbox 服务器间是同步的 HTTP 请求，整体流程总结如下：

- RIS系统生成一条v2 消息将其发送给消息处理器.
- 消息处理器根据映射关系生成FHIR 消息，转至'Mailbox'服务器节点。要给bundle and MessageHeader分配 ID（MessageHeader.identifier表示的是业务层面上该消息的标识符，而bundle ID 指的就是这一条消息 这个bundle资源自身的标识）
- 'Mailbox'服务器节点根据MessageHeader.event code的值识别出具体的消息类型，完成预设的处理。这里也就是完成数据存储。

- 在处理完成之后，‘Mailbox’服务器节点新建一个反馈消息，也就是新建一个bundle(分配唯一的bundle.id)，其中包含了另一个MessageHeader，MessageHeader中包含了服务器节点的信息也有自己的标识符，其中response元素的值表示它是哪条消息的响应消息(类似于V2中的 ACK消息)。并将反馈消息发送给消息处理器。response元素包含如下元素:
  - an identifier – 也就是请求消息的MessageHeader.identifier。该字段在同步模式下不是必要的，异步模式则是必须存在。
  - a code 表示处理的结果
  - a details是一个OperationOutcome资源，表示错误的具体信息
- 消息处理器收到反馈消息后，根据它构建一条HL7 V2 ACK消息
- 消息处理器将ACK消息反馈给RIS系统.

  HTTP 状态码反映了处理的结果。

架构方面的考虑因素

- Patient & Provider 在哪里存储? 如果和Observation没有存在同一个服务器上，就需要消息处理器负责构建新的Patient & Provider 资源(‘Mailbox’服务器节点能够做的话就另说了)。
- 消息处理器与‘Mailbox’服务器节点如何通信？可以是同步，或者异步。如果是异步的话，MessageHeader.response.identifier字段必须赋值。
- 消息处理器如何管理错误，比如消息映射过程中或者是'Mailbox’服务器节点返回的处理错误

Server Mailbox 方面的考虑因素

- ‘Mailbox’服务器节点是本地存储 Observation还是存储到其他服务器上
- 如果Patient & Provider resources是存储在‘Mailbox’服务器节点本地的，‘Mailbox’服务器能够新建资源，处理好引用。
- 如果Patient & Provider resources是存储在其他服务器上，‘Mailbox’服务器能够进行查询并在外部服务器上新建资源。
- 处理存储数据之外，‘Mailbox’服务器是否还要完成其他处理，比如有新的待评审结果时发送通知
- 在返回的bundle中‘Mailbox’服务器是否应包含已更新的Observation数据。

策略方面的问题

- Patient & Provider资源的标识符查询的命名空间是什么？加入V2消息中没有命名空间的话，是否应假设它们就是存储在本地的
- Patient & Provider如果在本地存储中没有找到该如何处理？到底是报错还是新建，如果是报错的话，该如何管理？
- 如果多个Patient & Provider拥有同样的标识该如何处理？是报错还是择其一使用？

消息转换方面的问题

- 消息中source and destination endpoints到底应该是什么，第一篇中我们只是简单的认为消息处理器是发送方，‘Mailbox’服务器是接收方。假如‘Mailbox’服务器能够直接接收其他系统的发送的消息呢？
- responsible字段该如何赋值？尤其是医嘱和医嘱响应中该怎么使用

总结

原文链接:More FHIR Messaging: ADT messages 原文链接:FHIR Messaging: Clinical data from ADT Messagess

More FHIR Messaging: ADT messages FHIR 消息：第三篇——不同类型的V2消息所对应的FHIR 消息中涉及的资源

译者注:消息是医疗信息交换的一种重要模式，从**HL7 V2 V3**到**X12**等。对于已经应用了**HL7 V2** 消息的系统来讲，如何迁移到 **FHIR** 消息中来，是我们接下来要探讨的话题。 本来是两篇，我们这里整合成一个.一开始给出了患者管理域**ADT**消息的常见字段，随后分析了系统在收到不同的**ADT**消息该做出哪些处理**(**增删改哪些**FHIR**资源**)**

#

V2 消息始于1987年，其中定义了一些触发事件，以及这些触发事件所引起的消息，规定了在收到消息之后应该完成的一些特殊的行为。

比如患者入院是其中一个常见的，患者的详细信息通过Patient Administration System (PAS)(门诊住院医生工作站、挂号系统)录入，遂即产生一条 V2 消息发送至每个订阅系统-诸如RIS和LIS系统。

触发事件的编码为 ‘A01’, 消息类型为 ‘ADT’ (Admission/Discharge/Transfer) – 也就是俗称的 ADT^A01 消息

一旦接收系统处理完消息之后，消息自身就没用了，可以丢弃，但大多数系统都保留一份用作审计。与CDA不同，它不是用于持久化的临床文档，与REST不同，它没有明确定义系统的处理行为(GET 读 POST 新增)。

V2标准应用很广泛，这里只拿一小部分来看看如何使用 FHIR 来实现 V2 消息.

FHIR messaging 框架不像 REST接口那样成熟，这里我们要先做一些预设，后续的FHIR 中可能会发生变更。具体实现的时候请参考最新的 FHIR spec即可。

我们以患者管理域为例– 也就是‘ADT’ (Admission/Discharge/Transfer) 消息。我们要建一个库，其中包含了医生想要查看的从V2 消息中抽取而成的 FHIR resources. 先看一下我们到底需要抽取哪些资源.

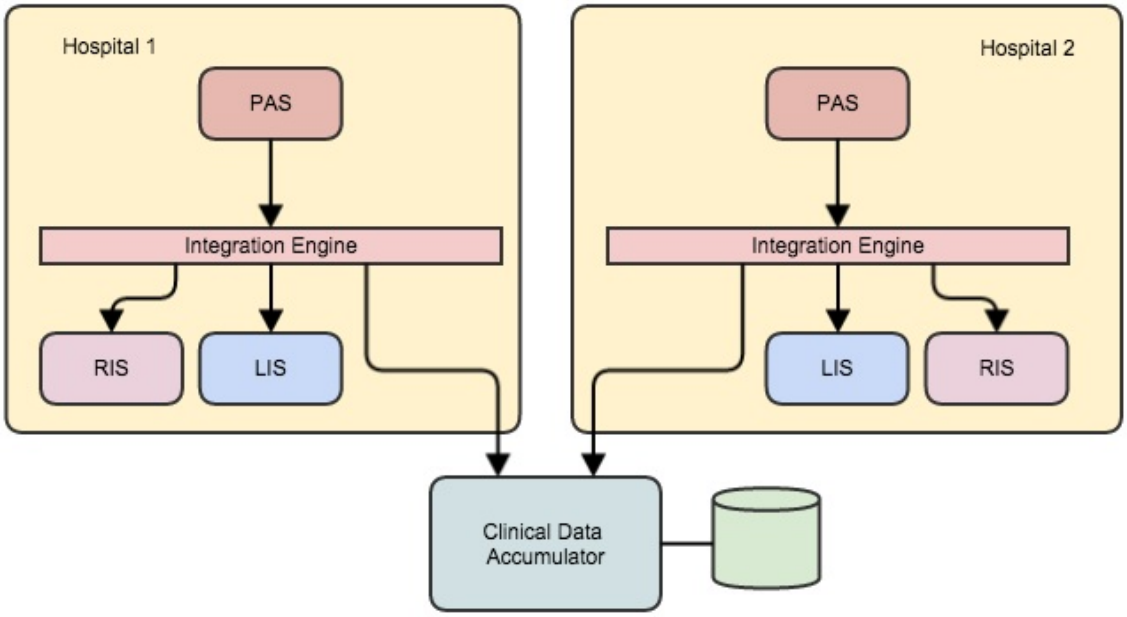
第一条消息ADT^A01,也就是入院通知消息。通常是由门诊住院医生工作站、挂号系统产生的，包含大量的区段，大部分都是可选的区段。下表罗列了一些我们要用到的(不是全集),和我们可以从这些数据得到的 FHIR 资源。

区段名称	资源名称	讨论意见
MSH	MessageHeader	包含了发送和接收系统，事件类型，标识符和日期等
PID	Patient	是哪个患者的消息
PD1	Patient	患者的额外信息
PV1	Encounter	就诊信息
PV2	Encounter	就诊信息的详细信息
OBX	Observation	
PR1	Procedure	入院通知是面向管理的而非临床，这里假设这个手术指的是是入院的原因，计划中的手术 不能简单的就认为患者已经做了该手术
AL1	AllergyIntolerance	过敏史 主要是入院时 患者自己陈述的
DG1	Condition	诊断 入院原因
DRG	Condition	病情的额外信息

这样我们就知道了我们的库里应该保存哪些资源的数据。后续几篇文章中将详细讨论每个资源。

由于V2 是表示现实世界中事件的消息标准，从消息到资源的转换并非那么简单直接，我们要考虑工作流程，不同类型的消息将如何影响资源。

常见的模式就是下游的诸如RIS LIS系统订阅医生工作站的入院消息。如果是多院区之间的话，又该如何处理？

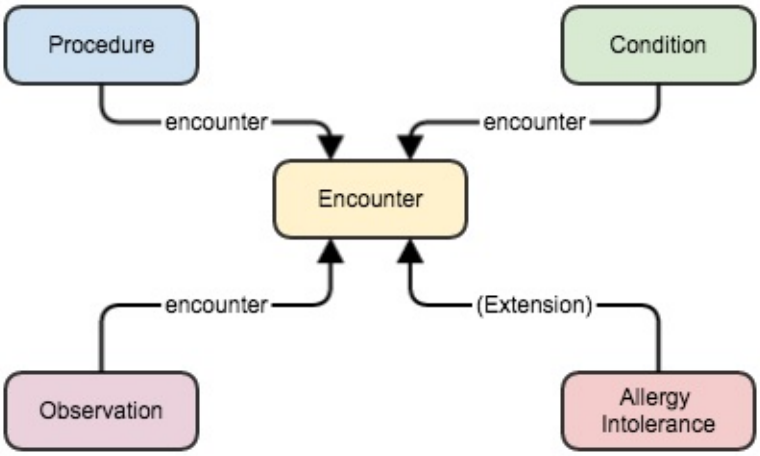


我们可以从这些消息生成如下资源：

- Encounters (既往就诊信息，是否住过院)
- Conditions – 就诊时的诊断. 不是正式的problem list
- Allergies –可能不如临床类消息中的过敏信息可靠，但也是很有用的
- Procedures – well, you would expect a hospital to know what procedures they are performing.
- Observations. Not quite sure what these represent in the context of an ADT message, but let's store them.

在消息模式中，我们需要考虑下面的问题：

- 如何将同一个患者的消息关联起来
- 不同的消息类型对数据模型的影响



数据模型的草图如下：

其中每个临床类资源都与某次就诊相关联，这样才能将工作流问题引起的变更管理起来。

大多数资源中都有这样一个属性：

- Condition: Encounter when condition first asserted. 将Condition.category 设为'diagnosis', and 给每次就诊新增一个 Condition (as the diagnosis is about that visit).
- Observation: 观察所对应的医疗事件
- Procedure: The encounter when the procedure was performed
- AllergyIntolerance: 这是一个扩展,没有现成的对reference的引用，that there should 'encounter first asserted' property – if you feel strongly about it, then raise a change request on the spec – we'll just use an extension.

这里我们并没有提到 Patient，每个资源都要引用Patient。

这样 模型就有了，来看一下每个消息对我们数据模型的影响

V2.4 中有不止60个消息需要考虑，我们只选其中一些来进行说明。 下表罗列了一些消息以及对数据模型的影响。简便起见用A01来代表ADT^A01

消息每次	目的	是否有临床数据	描述
A01	Admit/Visit Notification	yes	入院、分配病床后产生的消息。收到该消息后，根据临床数据产生一个新的 Encounter resource
A02	Transfer a Patient	No	转科转院消息。更新encounter resource，不更新临床数据。值得注意的是如果有临床数据发生变化，应该使用A08消息
A03	Discharge/End Visit	some	出院产生的消息。收到该消息后更新 Encounter resource，可能包括手术和诊断洗洗脑，如果要更新相关临床数据，应该使用A08消息
A04	Register Patient	yes	与A01类似，这条消息只是指患者目前在医院而非正式入院。收到该消息后，产生一个新的 Encounter resource
A05	Pre-Admit a Patient	yes	产生一个新的 Encounter resource。在实际入院之前，患者处于检查阶段
A06	Change an Outpatient to an Inpatient	yes	门诊病人办理住院时使用。可以更新现有的Encounter resource，或是创建一个新的Encounter resource这里面涉及到如何区分一个Encounter resource，视具体情况来定
A07	Change an Inpatient to an Outpatient	yes	住院病人转成门诊病人。要看系统怎么处理了，是不是当成出院来对待，那样的话就要么是 更新现有的Encounter resource，或是创建一个新的Encounter
A08	Update Patient Information	yes	更新患者的信息。 encounter resource不受影响，其他临床类资源可能会影响（过敏）
A11	Cancel Admit / Cancel Visit Notification	no	因为某种原因取消了入院通知。根据情况来设置encounter的状态，但临床类数据如何处理？待定
A12	Cancel Transfer	no	更新Encounter的状态。临床类数据不变
A13	Cancel Discharge / Cancel End Visit	yes	更新Encounter的状态
A28	Add Person or Patient Information	yes	与A08类似，主要是用于更新跨患者主数据库的汉子信息。Encounter 不变，临床类可能变
A31	Update Person Information	yes	主要是用于更新EMPI中的患者信息

A37	Unlink Patient Information	no	参考A40
A38	Cancel Pre-admit	no	与A11类似
A40	Merge Patient – Patient Identifier List	no	合并两个患者信息(要么是2个关联起来，要么是新的替换掉旧的)。
A45	Move Visit Information	no	Encounter.subject 的对象变了，至于说患者的标识本身的问题很含糊 是否需要把所有涉及到原来的患者标识的临床类数据的subject都改过来呢？
A50	Change Visit Number	----	This actually changes the visit number – the identifier that indicates that a sequence of messages is about the same visit. The actual effect for us will simply be to update the encounter

下面的几条要格外注意，不同的医院情况可能各异：

- 如何区分一次“visit”?-比方说接收到一条消息，到底这条消息会影响那个 Encoounter？
- 哪个消息会新建一个 Encoounter 那个会更新现有的 Encoounter
- 患者与 visit 如何关联？ 患者账号 和患者标识如何关联？
- 当 A45 中 Encoounter的患者标识变化时，对应的临床数据如何处理？
- 当一些消息中包含了临床类数据时，我们如何处理单次visit内临床类数据的更新？
  - 是假设每个消息的数据都是完整的还是它们只是待更新的？假如A01中又一个特殊诊断A02没有，我们是不是要移除掉？
  - 如果A02中包含了A01中没有的特殊诊断，是不是要添到列表里面去？
  - 是不是要从A01中生成临床类数据，要求系统在临床类数据变化时发送A08或A28消息？

我们这里做如下假设：

- PV1-19 中包含了就诊的标识号，这样就能知道消息到底属于哪次就诊。使用MSH-4 (sending facility) and MSH-3 (sending application)来区分不同医院 不同系统发送的消息
- PID-3 中包含了所有的患者标识符
- 使用 A01, A04 or A05 来创建临床类数据，使用A08 or A28 来更新此类数据，更新是一个快照操作，也就是说会替换该次就诊中原有的数据
- 支持一个患者有多个标识符，在接收数据时保证标识符和数据一致。在读取数据时处理多个标识符的问题。比如查询使用某个标识符的相关数据，使用EMPI来管理所有标识，然后返回所有匹配的数据
- 如果患者信息是由于A45而发生了改变，我们认为后面会有一条A08消息负责更新临床类信息。这个消息将会改变所有资源的subject。
- 能够接收对于乱序的消息，使用message time (MSH-7) 来决定如何处理这些消息。比如，收到一个出院消息A03，将 encounter.status 设为 ‘finished’，然后收到一条换床消息A02，具体操作将取决于相应的日期时间，如果日期在出院日期之后，则将 encounter.status 设为‘in-progress’，其他情况则忽略该A02消息。

实际上，我们要保证源系统也遵循这样的约定。如果它们不遵循的话，要么改造我们的系统，要么使用一个集成引擎来特殊处理这些不遵循约定的消息。 For example, if a particular hospital does not emit A08, but does include a snapshot of clinical data in all update messages (like an A02) then the Integration Engine could create an additional A08.

下表则是我们的系统能够支持的消息列表和对应的处理：

Message	Purpose	Action in our system
A01	Admit/Visit Notification	Create a new encounter and all related clinical data
A02	Transfer a Patient	Update the Encounter properties
A03	Discharge/End Visit	Update the Encounter properties

A04	Register Patient	Create a new encounter and all related clinical data
A05	Pre-Admit a Patient	Create a new encounter and all related clinical data
A06	Change an Outpatient to an Inpatient	Update the Encounter properties
A07	Change an Inpatient to an Outpatient	Update the Encounter properties
A08	Update Patient Information	Update the clinical data associated with this encounter. Note that this can change the subject associated with those resources as well.
A11	Cancel Admit / Cancel Visit Notification	Update the Encounter properties
A12	Cancel Transfer	Update the Encounter properties
A13	Cancel Discharge / Cancel End Visit	Update the Encounter properties
A28	Add Person or Patient Information	Update the clinical data associated with this encounter
A31	Update Person Information	Not quite sure about this one yet. It's not really related to a particular encounter, so we'll think about it a bit more when we start to consume other messages. For now, we'll ignore it.
A37	Unlink Patient Information	Our patient identity system will no longer assume that these two identifiers refer to the same person.
A38	Cancel Pre-admit	Update the Encounter properties
A40	Merge Patient – Patient Identifier List	Mark the two identifiers as being the same person. (The message contains the 'surviving' identifier). We won't actually merge them – we'll just link them together. That way we can 'unmerge' – A37 – them later.
A45	Move Visit Information	Change the encounter.subject to reference the new patient identity
A50	Change Visit Number	Change the encounter.identifier property. Of course, an encounter may have multiple identifiers so we'll need to make sure to change the correct one!

简短的两篇文章希望解释了为什么工作流程是很重要的。

FHIR标准库-HAPI-FHIR

主要是对FHIR 标准库HAPI-FHIR在应用过程中遇到的问题进行探讨

原文链接:Processing FHIR Bundles using HAPI

另外一篇文章:FHIR transactions: the Search functionality

FHIR 事务:search功能

FHIR中是通过transaction来实现资源批量处理的.服务器对每个资源都单独处理,看起来就像是每个资源都是单独发送/提交的(除了批量更新的情况之外 要么都成功 要么都失败). 对于服务器而言,要处理bundle中每个资源的引用是很困难的,不论是新添加的资源或者是已经存在的. 其中有一点,服务器首先要判断这个资源服务器上存在不存在. 用例描述: 通过FHIR标准将某个可穿戴式设备(家用血糖仪)采集的血糖数据导入到数据仓库当中去. 血糖仪能够采集血糖数据,通过USB接口,利用桌面应用程序把数据取出来,发送给数据仓库.数据仓库期望 所存储的是结构化数据,这样子便于展示和临床决策支持之用. 思路: 大体上可以有多次操作和单次操作.

1. 首先向数据仓库发送查询请求,确保Patient和Device存在,不存在就新增.然后利用Observation来传输(POST方法) 每个血糖结果,在observation中加上对Patient和device的引用即可.
2. 用一个bundle来表达整个与血糖相关的信息,其中包含一个Patient资源,一个Device资源,一个Observation资源 (用于记录每一项结果). Observation.subject 是patient , Observation.performer是Device.最终URL类如 POST [base] {?_format=[mime-type]} 客户端生成这个bundle的时候,Patient的ID和Device的ID应该如何赋值,鉴于血糖结果总是新添加的,Observation的ID使用cid:id即可. 对于Patient资源和Device资源而言,我们是已经给他们了一个id,但这个id又不是资源中的患者或设备的标识,假设我们给ID赋值为cid:ID的话 ,服务器会不断的新增一条患者记录,设备记录, 我们无法与可能之前就存在的记录进行关联.在标准中有如下一段话,提出了解决这个问题的方法:

   ```
   The application constructing a bundle may not be sure whether a particular resource will already exist at the time
   ```

   也就是说.遇到这种情况,资源的id赋值为cid:id,另外在atom的link属性中增加一条如下

   ```
   <link href="http://localhost/Patient?[parameters]" rel="search"/>
   ```

   服务器接收到这样的一条记录,首先依据href中那个的url和参数在服务器上进行检索有无匹配记录,如果有, 就直接使用服务器上的匹配记录,没有的话,就按原来的思路处理.如果发现多条记录,不进行处理,直接报异常. 如下是patient的实例片段:

   ```
   <entry>
   <title>Patient details</title>
   <id>cid:patient@bundle</id>
   <updated>2014-05-28T22:12:21Z</updated>
   <link href="http://localhost/Patient?identifier=PRP1660" rel="search"/>
   <content type="text/xml">
     <Patient xmlns="http://hl7.org/fhir">
         <text>
             <status value="generated"/>
             <div xmlns="http://www.w3.org/1999/xhtml">Joe Bloggs</div>
         </text>
         <identifier>
             <value value="PRP1660"/>
         </identifier>
         <name>
             <text value="Joe Bloggs"/>
         </name>
     </Patient>
   </content>
   </entry>
   ```

3. 利用mailbox节点,我们给方法2中的bundle添加一个MessagHeader资源,使用MessageHeader.event 来 表示bundle是用来干嘛的.mailbox就能够正常处理了.
4. 利用transaction节点,也就是服务器根节点,通过profile资源来定义bundle的内容.在事务transaction过程中, 可以识别出profile,使用预定义的处理方法,可能包含了校验和资源的更新操作.

方法2是一种自定义方法,需要发送方接受方的协调,FHIR中暂时没有定义和描述此类服务的方式. 方法3 你要定义消息事件类型,添加消息头资源,发送方和接受方需要协调 方法4POSTing to the root (#4) is attractive –provided that the server knows how to perform the ‘search processing’ (i.e. recognize the search link for a resource in the bundle and process accordingly. We could possibly use a profile so that the server can validate the bundle first. 通用型的事务处理过程是很复杂的,一方面包括了ID的重写操作,另一方面也要照顾bundle中需要删除操作的资源.通过profle资源来约束限制事务的处理过程的行为,比方说,服务器识别出profile,将其作为某次transaction的基准,而不需要实现所有功能.

不论选取那种方式,都需要在响应时返回一个bundle,包含了各个资源,每个资源都会分配一个ID. 下面的代码是方式2的范例,仅供参考.

- GlucoseBundleProcessor迭代所有资源,更新其cid:id的引用,核对是否存在Patient Device资源.
- 该例中忽略了所有服务器想要的资源 显然是不科学的

类的示例：

```
public class GlucoseBundleProcessor {

    private MyMongo _myMongo;   //the database helper class

    public GlucoseBundleProcessor(MyMongo myMongo){
        _myMongo = myMongo;
    }

    //process with a Bundle...
    public List<IResource> processGlucoseBundle(Bundle bundle) {
        //generate a list of resources...
        List<IResource> theResources = new ArrayList<IResource>();    //list of resources in bundle
        for (BundleEntry entry : bundle.getEntries()) {
            theResources.add(entry.getResource());
        }
        return this.process(theResources);
    }


    //process with a List of resources
    public List<IResource> processGlucoseUploads(List<IResource> theResources) {
        return this.process(theResources);
    }


    //process a bundle of glucose results, adding them to the repository...
    private List<IResource> process(List<IResource> theResources) {
        Patient patient = null;     //this will be the patient resource. There should only be one....
        Device device = null;       //this will be the device resource. There should only be one....
        List<IResource> insertList = new ArrayList<IResource>();    //list of resource to insert...

        //First pass: assign all the CID:'s to a new ID. For more complex scenarios, we'd keep track of
        //the changes, but for this profile, we don't need to...
        //Note that this processing is highly specific to this profile !!! (For example, we ignore resources we don't e
        for (IResource resource : theResources) {
            String currentID = resource.getId().getValue();
            if (currentID.substring(0,4).equals("cid:")) {
                //Obviouslym the base URL should not be hard coded...
                String newID = "http://myUrl/" + java.util.UUID.randomUUID().toString();
                resource.setId(new IdDt(newID));    //and here's the new URL
            }

            //if this resource is a patient or device, then set the appropriate objects. We'll use these to set
            // the references in the Observations in the second pass. In real life we'd want to be sure there is only o
            if (resource instanceof Patient) {
                patient = (Patient) resource;
                //we need to see if there's already a patient with this identifier. If there is - and there is one,
                //then we use that Patient rather than adding a new one.
                // This could be triggered by a 'rel=search' link on the bundle entry in a generic routine...
                IdentifierDt identifier = patient.getIdentifier().size() >0 ? patient.getIdentifier().get(0) : null;
                if (identifier != null) {
                    List<IResource> lst = _myMongo.findResourcesByIdentifier("Patient",identifier);
                    if (lst.size() == 1) {
                        //there is a single patient with that identifier...
                        patient = (Patient) lst.get(0);
```

```
                        resource.setId(patient.getId());    //set the identifier in the list. We need to return this...
                    } else if (lst.size() > 1) {
                        //here is where we ought to raise an error - we cannot know which one to use.
                    } else {
                        //if there isn't a single resource with this identifier, we need to add a new one
                        insertList.add(patient);
                    }
                } else {
                    insertList.add(patient);
                }
            }

            //look up a Device in the same way as as for a Patient
            if (resource instanceof Device) {
                device = (Device) resource;
                IdentifierDt identifier = device.getIdentifier().size() >0 ? device.getIdentifier().get(0) : null;
                if (identifier != null) {
                    List<IResource> lst = _myMongo.findResourcesByIdentifier("Device", identifier);
                    if (lst.size() == 1) {
                        device = (Device) lst.get(0);
                        resource.setId(device.getId());    //set the identifier in the list. We need to retuen this...
                    } else {
                        insertList.add(device);
                    }
                } else {
                    insertList.add(device);
                }
            }

            if (resource instanceof Observation) {
                //we always add observations...
                insertList.add(resource);
            }
        }

        //Second Pass: Now we re-set all the resource references. This is very crude, and rather manual.
        // We also really ought to make sure that the patient and the device have been set.....
        for (IResource resource : theResources) {
            if (resource instanceof Observation) {
                Observation obs = (Observation) resource;

                //this will be the correct ID - either a new one from the bundle, or a pre-existing one...
                obs.setSubject(new ResourceReferenceDt(patient.getId()));

                //set the performer - there can be more than one in the spec, hence a list...
                List<ResourceReferenceDt> lstReferences = new ArrayList<ResourceReferenceDt>();
                lstReferences.add(new ResourceReferenceDt(device.getId()));
                obs.setPerformer(lstReferences);
            }
        }

        //Last pass - write out the resources
        for (IResource resource : insertList) {
            _myMongo.saveResource(resource);
        }

        //we return the bundle with the updated resourceID's - as per the spec...
        return theResources;
    }
}
```

代码示例

```
@WebServlet(urlPatterns= {"/fhir/service/glucoseprocessor"}, displayName="Process Glucose bundle")
public class GlucoseResultServlet extends HttpServlet {

    private MyMongo _myMongo;
    private FhirContext _fhirContext;

    @Override
    public void init(ServletConfig config) throws ServletException {
        //get the 'global' resources from the servlet context
        ServletContext ctx = config.getServletContext();
```

```
        _myMongo =  (MyMongo) ctx.getAttribute("mymongo");
        _fhirContext = (FhirContext) ctx.getAttribute("fhircontext");
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExceptio
        PrintWriter out = response.getWriter();
        //first parse into a fhir bundle (need to check the mime type here. Should also check for _format paramters as
        //if we have to to this a lot, then a separate utility class would be good...

        Bundle bundle = null;
        IParser parser = null;
        String ct =  request.getHeader("content-type");
        if (ct.equals("application/xml+fhir")){
            parser = _fhirContext.newXmlParser();
            bundle = parser.parseBundle(request.getReader());
        } else if (ct.equals("application/json+fhir")){
            parser = _fhirContext.newJsonParser();
            bundle = parser.parseBundle(request.getReader());
        } else {
            OperationOutcome operationOutcome = new OperationOutcome();
            operationOutcome.getText().setDiv("Invalid content-type header: " + ct);
            parser = _fhirContext.newXmlParser();
            response.setStatus(415);    //unsupported media type
            out.println(parser.encodeResourceToString(operationOutcome));
            return;
        }

        //now we have a bundle we can process it...
        GlucoseBundleProcessor glucoseBundleProcessor = new GlucoseBundleProcessor(_myMongo);
        //process the bundle, and get back the list of resources with updated ID's...
        List<IResource> resources = glucoseBundleProcessor.processGlucoseBundle(bundle);
        Bundle newBundle = new Bundle();
        newBundle.getTitle().setValue("Processed Glucose results");
        newBundle.setId(new IdDt(java.util.UUID.randomUUID().toString()));
        newBundle.setPublished(new InstantDt());
        for (IResource resource : resources) {
            newBundle.addResource(resource,_fhirContext,"http://localServer/fhir");
        }

        response.setStatus(201);    //created
        out.println(parser.encodeBundleToString(newBundle));

    }



}
```

FHIR标准与SMART ON FHIR

主要是对基于FHIR 标准的SMART ON FHIR项目的一些相关技术点和应用的探讨，比如Oauth2、openID和profile的应用，其中还包括了SMART ON FHIR所提供的Swift和javascript库的应用

原文链接:SMART on FHIR: Part 1

另外一篇文章:SMART on FHIR – adding OAuth2英文版

另外一篇文章:SMART on FHIR – adding OAuth2中文版

第七届connectathon眼看着就要开始了, 之前的一篇里我们讨论了其中第一个场景(Patient的访问)，其中利用一些开源库进行了简单的实现(btw 你完全可以不使用这些开源框架，使用自己熟悉的平台和编程语言即可).

这篇主要探讨一下第三个场景如何实现SMART on FHIR. 对于这个场景具体的描述请参考7thConnectathon_tracks场景说明 – 其核心是构建一个标准，不同的独立开发APP/应用程序能够安全的访问存储在任何支持这种标准的服务器上的数据-电子病历、患者门户、区域平台相当于这里的应用程序，而类如苹果的HealthKit就是服务器，而苹果定义的HealthKit的接口就是我们所说的标准，这个比方打的不太恰当，但大致上是这么个意思。

第一步，我们先实现一个简单的服务器，主要是参考Smart Server快速入门, 如需深入了解，请细读该文档. 这篇文章中的服务器没有考虑安全性，后续的SMART on FHIR adding OAuth2中会详细介绍如何添加安全性.

这里假设我们的服务器是一个电子病历系统, 用户可以通过SMART团队所开发的儿童成长曲线的APP来访问其中的数据.

最简单的做法，比如在我们的电子病历系统的界面上添加一个按钮"查看儿童成长曲线".当用户点击该按钮:

1. 第一步，弹出一个界面，到这个页面上，指向儿童成长曲线的APP启动界面(类似你在Quaro上登录时选择使用google账号登录那样). 我们会把参数如患者ID还有APP所需要用到的数据以URL的形式暴露出去(FHIR中定义的Patient 和 Observation接口URL/endpoint),
2. 启动界面在我们的界面上加载之后，通过ajax调用我们服务器上的FHIR endpoint来获取服务器上的数据 (安全性主要是在这一步中需要考虑).
3. 拿到数据之后，就可以进行展示，或者向服务器回写数据.

在这篇里面，我们要做如下内容:

- 一个HTML页面，用来模仿电子病历系统，当然也要有一个启动按钮
- FHIR Patient endpoint，能够返回患者的基本/人口统计学信息
- FHIR Observation endpoint，能够返回患者的生命体征等观察项的信息(如身高、体重、BMI等).

HTML页面很简单，加入如下的Jquery代码即可。 其他的内容需要大家自己动手。

```
$(document).ready(function(){
      //the url to launch the app (hardcoded patientID, and local server)
      var url = "https://fhir.smartplatforms.org/apps/growth-chart/launch.html?";
      url += "fhirServiceUrl=http://localhost:8080/fhir";
      url += "&patientId=100";
      //var iframe = &amp;amp;quot;&amp;amp;quot;; // this line should contain the html for an iframe, if wordpress wou
      //the jQuery handler to create an iFrame and launch the app in it
      $("#launchApp").on('click',function(){
        //$('#launchFrameDiv').append(iframe);
        $("#launchIframe").attr("src",url);
      })
});
```

服务器端的话稍微要复杂一些。这里我们使用HAPI开源库 和Tomcat服务器.IDE的话选IntelliJ IDEA IDE.

利用HAPI构建服务器端,首先要为你所支持的每种资源定义一个resource provider , 然后利用 server class来发布服务 ；.

Patient Provider如下:

```
public class PatientResourceProvider implements IResourceProvider {
    //return a single Patient by ID
```

```
    @Read()
    public Patient getResourceById(@IdParam IdDt theId) {
        Patient patient = new Patient();
        Calendar cal = Calendar.getInstance();
        cal.add(Calendar.YEAR, -18);         //18 years old
        patient.setBirthDate(new DateTimeDt(cal.getTime()));
        patient.addName().addFamily(&amp;amp;quot;Power&amp;amp;quot;);
        patient.getName().get(0).addGiven(&amp;amp;quot;Cold&amp;amp;quot;);
        patient.setGender(AdministrativeGenderCodesEnum.M);
        return patient;
    }
}
```

为了简单起见，这里的赋值都是硬编码的形式，当然你可以使用数据库来set get.

Observation Provider如下:

```
public class ObservationResourceProvider implements IResourceProvider {
    @Search()
    public List&amp;amp;lt;Observation&amp;amp;gt; getObservationBySubject(@RequiredParam(name = Observation.SP_SUBJECT
                                                @RequiredParam(name = Observation.SP_NAME) TokenOrListParam theObs
        List&amp;amp;lt;Observation&amp;amp;gt; lstObservations = new ArrayList&amp;amp;lt;Observation&amp;amp;gt;();

        //emulates calling an existing non-FHIR REST service and getting a JSON object back...
        //we'd probably pass across the theObsNames to only return the list that we want
        InputStream is = null;
        try {
            URL url = new URL(&amp;amp;quot;http://localhost:4001/dataplatform/obs/&amp;amp;quot;+theSubject);
            URLConnection conn = url.openConnection();
            conn.connect();
            is = conn.getInputStream();
            JsonReader rdr = Json.createReader(is);
            JsonObject obj = rdr.readObject();
            //assume that the json structure is {data[{id:,unit:,code:,value:,date: }]}

            JsonArray results = obj.getJsonArray(&amp;amp;quot;data&amp;amp;quot;);
            SimpleDateFormat format = new SimpleDateFormat(&amp;amp;quot;yyyy-MM-dd'T'HH:mm:ss&amp;amp;quot;);
            for (JsonObject result : results.getValuesAs(JsonObject.class)) {
                //get the basic data for the Observation
                String id = result.getJsonString(&amp;amp;quot;id&amp;amp;quot;).getString();
                String unit = result.getJsonString(&amp;amp;quot;unit&amp;amp;quot;).getString();
                String code = result.getJsonString(&amp;amp;quot;code&amp;amp;quot;).getString();
                String display = result.getJsonString(&amp;amp;quot;display&amp;amp;quot;).getString();
                double value =  result.getJsonNumber(&amp;amp;quot;value&amp;amp;quot;).doubleValue();
                Date date = format.parse(result.getJsonString(&amp;amp;quot;date&amp;amp;quot;).getString());

                //create an Observation resource
                Observation obs = new Observation();
                obs.setId(id);
                obs.setApplies(new DateTimeDt(date));
                obs.setName(new CodeableConceptDt(&amp;amp;quot;http://loinc.org&amp;amp;quot;, code));
                QuantityDt quantityDt = new QuantityDt(value).setUnits(unit).setSystem(&amp;amp;quot;http://unitsofmeas
                obs.setValue(quantityDt);
                obs.setStatus(ObservationStatusEnum.FINAL);
                obs.setReliability(ObservationReliabilityEnum.OK);

                //the text...
                obs.getText().setDiv(result.getJsonString(&amp;amp;quot;date&amp;amp;quot;).getString() + &amp;amp;quot
                obs.getText().setStatus(NarrativeStatusEnum.GENERATED);

                //and add to the list...
                lstObservations.add(obs);
            }
        } catch (Exception ex) {
            System.out.println(&amp;amp;quot;Error during REST call &amp;amp;quot; + ex.toString());
        }
        finally {
            try {
                if (is != null) {
                    is.close();
                }
            } catch (Exception ex){              //was going to throw an exception - but that seems to need to be in a
                System.out.println(&amp;amp;quot;Exception closing stream &amp;amp;quot;+ ex.getMessage());
            }
```

```
        }
        return lstObservations;

    }
}
```

这个provider类似于一个代理，现有的FHIR服务器可能支持类似的功能，但接口不是FHIR形式的.对于系统的改造 大多数情况应该是这样的。

server servlet如下:

```
@WebServlet(urlPatterns= {&amp;amp;quot;/fhir/*&amp;amp;quot;}, displayName=&amp;amp;quot;FHIR Server&amp;amp;quot;)
public class OrionRestfulServlet extends RestfulServer {
    public OrionRestfulServlet() {
        List&amp;amp;lt;IResourceProvider&amp;amp;gt; resourceProviders = new ArrayList&amp;amp;lt;IResourceProvider&am
        resourceProviders.add(new ConditionResourceProvider(_myMongo));
        resourceProviders.add(new ObservationResourceProvider(_myMongo));
        resourceProviders.add(new PatientResourceProvider(_myMongo));
        setResourceProviders(resourceProviders);
    }
}
```

剩下的就只需要打成war包，发布到tomcat中去即可. 输入URL，点击按钮，即可查看到生长曲线图!

后续的例子我们会使用OAuth2来给服务器添加安全保护。

原文链接:SMART on FHIR – adding OAuth2

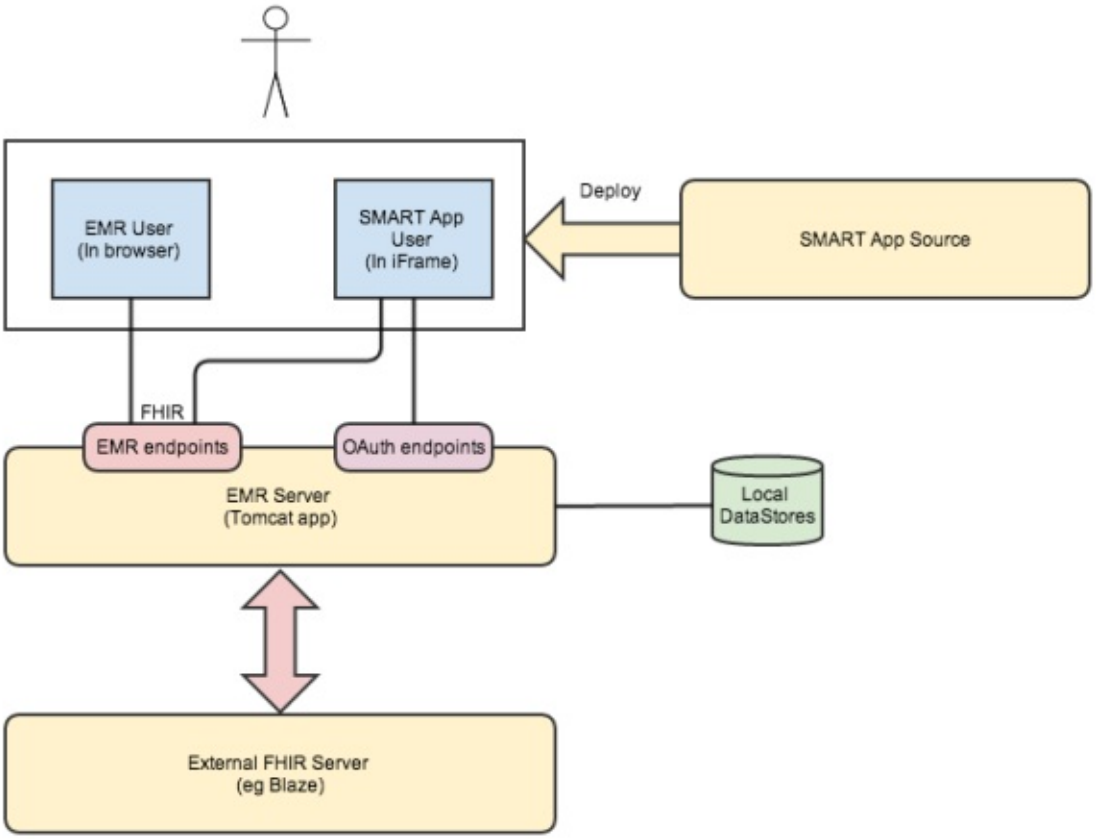
相关文章链接:SMART on FHIR: Part1英文版

相关文章链接:SMART on FHIR: Part 1中文版

在SMART on FHIR: Part 1中文版 基础上 ,利用SMART版本 的OAuth2 标准为其提供安全保障.这里还是沿用 HAPI开源库 、Tomcat 和IntelliJ IDEA IDE.

这里的应用场景是假设我是已经登录了电子病历/区域平台的用户，我想利用一些外部的应用程序来实现一些其他功能，比如调用SMART团队开发的儿童生存曲线的APP。

整体的架构如下所示：



这是一个web应用程序， 发布在Tomcat上，其中暴露了FHIR data endpoints 和 OAuth endpoints . SMART app部署在其他服务器上，在我们的页面上以iFrame的形式来调用. Tomcat server 上的数据可以存储在本地，但是这里调用的其他外部的FHIR服务器，其实就相当于外部FHIR服务器的一个代理。

使用 Maven 来管理项目代码.安装配置略。

在IDE中创建一个 “maven-archetype-webapp” 的maven项目.

由于SMART客户端是一个运行在其他domain的JavaScript程序， 对于我们的服务器而言，需要允许它访问我们的数据， 为此需要添加对 CORS 的支持，按照HAPI中的说明 ， 只需在pom.xml 添加dependency 来下载filter即可,修改web.xml file，修改其中的权限，可以直接从HAPI官网上的例子中复制.

接下来考虑如何SMART进行集成. SMART服务器端快速入门提供了很详细的资料 – 总结一下:

1. 在EMR web 界面上, 新建一个 iframe, 将其指向 SMART应用程序, 进行一些基本的配置即可. 比如

OpenMRS-FHIR

- Happiness
- Valentine

2. SMART 应用程序从服务器上加载一致性声明/conformance statement，这样就知道如何获取授权以及token的endpoint

3. SMART 应用程序调用authorization end point
4. 假设通过授权, endpoint会重新返回给 SMART 应用程序一个授权码
5. SMART 应用程序使用获取的授权码再次调用 token endpoint,获取 Authorization Token.
6. SMART 应用程序利用Authorization Token调用 EMR FHIR endpoints 来获取所需的数据，并进行展示.

下一步构建 Authorization and Token end points, 告知SMART 应用程序如何找到它们。 利用对Conformance资源的扩展可以实现.

首先:

在我们演示用的电子病历系统中新增一个登录功能，用户发送登录信息到login endpoint,对登录信息进行校验，创建一个user token并保存在一个context对象当中，同时将其发送给本地的APP，这样在启动时即可使用这个user token。由于系统用户同时也需要访问一些受限的资源，用户也会得到一个access token。

Login servlet示例:

```
@WebServlet(urlPatterns= {&amp;quot;/auth/login&amp;quot;}, displayName=&amp;quot;Login&amp;quot;).
public class Login extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExceptior
        System.out.println(&amp;quot;login&amp;quot;);
        PrintWriter out = response.getWriter();
        //Here is where we check and validate username &amp;amp; password. We're going to cheat right now...
        //create a user object. This would ultimately be a FHIR obect I suspect...
        Person person = new Person();
        person.userName = request.getParameter(&amp;quot;username&amp;quot;);
        person.userToken = java.util.UUID.randomUUID().toString(); //generate a user token

        //save the user details in the context - we previously created this map...
        ServletContext context =  getServletContext();
        Map&amp;lt;String,Person&amp;gt; usertokens = (Map&amp;lt;String,Person&amp;gt;) context.getAttribute(&amp;quot;usertokens&amp;quot;);
        //save the access token for later use - in production persistent store...
        usertokens.put(person.userToken,person);

        //create an access token for this person ...
        Map&amp;lt;String,JsonObject&amp;gt; oauthtokens = (Map&amp;lt;String,JsonObject&amp;gt;) context.getAttribute(&amp;quot;oauthtoker
        JsonObject json = Json.createObjectBuilder()
                .add(&amp;quot;access_token&amp;quot;, person.userToken)
                .add(&amp;quot;token_type&amp;quot;, &amp;quot;bearer&amp;quot;)
                .add(&amp;quot;expires_in&amp;quot;, 3600)
                .add(&amp;quot;scope&amp;quot;, &amp;quot;patient/*.read&amp;quot;)
                .build();
        oauthtokens.put(person.userToken,json);

        response.addHeader(&amp;quot;Content-Type&amp;quot;,&amp;quot;application/json+fhir&amp;quot;);
        out.println(person.getJson().toString());
    }
}
```

新建一个end-point来启动SMART app 如下:

```
@WebServlet(urlPatterns= {&amp;quot;/auth/launch&amp;quot;}, displayName=&amp;quot;Launch SMART application&amp;quot;)
public class Launch extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExceptior
        System.out.println(&amp;quot;launch endpoint accessed&amp;quot;);
        String userToken = request.getParameter(&amp;quot;usertoken&amp;quot;);
```

```
        String patientId = request.getParameter(&quot;patientid&quot;);

        //make sure this is a logged in person (the have a valid token)
        ServletContext context =  getServletContext();
        Map&lt;String,Person&gt; usertokens = (Map&lt;String,Person&gt;) context.getAttribute(&quot;usertokens&quot;);

        if (usertokens.containsKey(userToken)) {
            //yep, this is a valid user...

            //retrieve the user object and update with the patient they have in context. We'll need this for the access
            Person person = (Person) usertokens.get(userToken);
            person.currentPatientId = patientId;

            //the re-direct URL. In reality the url and 'iss' would come from config...
            String url = &quot;https://fhir.smartplatforms.org/apps/growth-chart/launch.html?&quot;;
            url += &quot;iss=http://localhost:8081/fhir&quot;;

            //we'll use the user token as the launch token as we can use that to validate the Auth call..
            url += &quot;&amp;launch=&quot; + userToken;

            response.sendRedirect(url);
        } else {
            response.setStatus(403);    //forbidden.
            PrintWriter out = response.getWriter();
            out.println(&quot;&lt;html&gt;&lt;head&gt;&lt;/head&gt;&lt;body&gt;&lt;h1&gt;User not logged in&lt;/h1&gt;&
        }
    }
}
```

接下来是**authorization** end-point. 这里用用户token做启动token来确保请求方的可信，重定向到 *redirect_url*.

代码如下

```
@WebServlet(urlPatterns= {"/auth/authorize"}, displayName="Authorize endpoint for FHIR Server")
public class Authorize extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExceptior
        System.out.println("auth check...");

        String response_type = request.getParameter("response_type");
        String client_id = request.getParameter("client_id");   //the id of the client
        String redirect_uri = request.getParameter("redirect_uri");
        String scope = request.getParameter("scope");   //what the app wants to do
        String state = request.getParameter("state");

        //the scope parameter includes the launch token - eg patient/*.read launch:7bceb3c6-66e9-46c9-8efd-9f87e76a5f9a
        //so we would pull out both scope and token, check that the token matches the one we set (actually the patient
        //and that the scope is acceptable to us. Should move this to a function somewhere...
        String[] arScopes =  scope.split(" ");
        String launchToken = "";
        for (int i = 0; i < arScopes.length; i++){
            System.out.println(arScopes[i]);
            if (arScopes[i].substring(0,7).equals("launch:")) {
                launchToken = arScopes[i].substring(7);
            }
        }

        ServletContext context =  getServletContext();
        Map<String,Person> usertokens = (Map<String,Person>) context.getAttribute("usertokens");

        if (usertokens.containsKey(launchToken)) {
            //we'll assume that the user is OK with this scope, but this is where we can check...
            //so, now we create an auth_code and re-direct to the redirect_url...
            String auth_code = java.util.UUID.randomUUID().toString();
            //we'll save the auth code in a previously defined context variable. In real life you'd use a
            //persistent store of some type, and likely save more details...
            Map<String,Person> oauthcodes = (Map<String,Person>) context.getAttribute("oauthcodes");

            Person person = (Person) usertokens.get(launchToken);

            oauthcodes.put(auth_code,person);
            //and re-direct to the 'authenticated' endpoint of the application
            response.sendRedirect(redirect_uri + "?code="+auth_code+ "&state="+state);
```

```
        } else {
            response.setStatus(403);    //forbidden.
        }

    }
}
```

我们可以使用一个注册过程来存储应用程序的ID和回调URL来进一步确保安全性

这样我们就可以交换授权token的授权码。 如下是**Token endpoint**的范例:

```
@WebServlet(urlPatterns= {"/auth/token"}, displayName="Token endpoint for FHIR Server")
public class Token extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExceptio
        PrintWriter out = response.getWriter();

        String code = request.getParameter("code");

        //the map containing auth_code that was set during the authorization phase...
        ServletContext context =  getServletContext();
        Map<String,Person> oauthcodes = (Map<String, Person>) context.getAttribute("oauthcodes");

        //is this a valid access code?
        if (oauthcodes.containsKey(code)) {
            Person person = (Person) oauthcodes.get(code);
            String access_token  = java.util.UUID.randomUUID().toString();
            JsonObject json = Json.createObjectBuilder()
                    .add("access_token", access_token)
                    .add("patient", person.currentPatientId)
                    .add("token_type", "bearer")
                    .add("expires_in", 3600)
                    .add("scope", "patient/*.read")
                    .build();
            response.addHeader("Content-Type","application/json+fhir");

            //save the access token for later use - like the codes, you'd use a persistent store...
            Map<String,JsonObject> oauthtokens = (Map<String,JsonObject>) context.getAttribute("oauthtokens");
            oauthtokens.put(access_token,json);
            //and return the token to the applciation
            out.println(json.toString());

        } else {
            //the auth codes don't match.
            response.setStatus(403);    //forbidden.
            out.println("{}");
        }
    }
}
```

有了auth token之后就可以调用实际的FHIR endpoint.

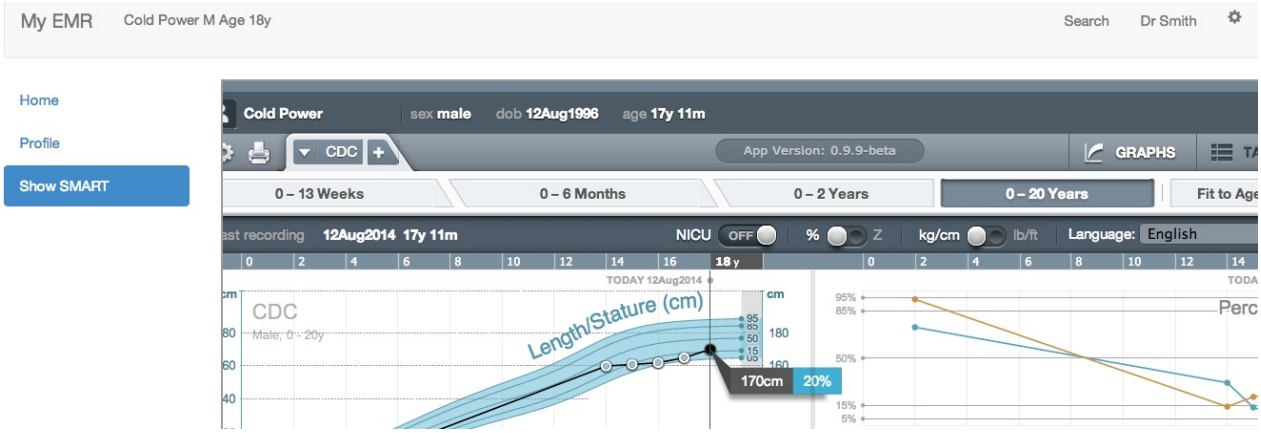
```
@Override
public void handleRequest(SearchMethodBinding.RequestType theRequestType,
                          javax.servlet.http.HttpServletRequest theRequest,
                          javax.servlet.http.HttpServletResponse theResponse)
        throws javax.servlet.ServletException,
        IOException {

    String uri = theRequest.getRequestURI();

    //anyone can access metadata...
    if (uri.equals("/fhir/metadata")) {
        super.handleRequest(theRequestType,theRequest,theResponse);
    } else {
        //but you need to be authorized to access clincial data...
        String auth = theRequest.getHeader("Authorization");
        if (auth != null){
            auth = auth.substring(7);//get rid of the 'Bearer ' at the front
```

```
            ServletContext context =  getServletContext();// request.    .setAttribute("oauthtokens", oauthtokens);
            Map<String,JsonObject> oauthtokens = (Map<String,JsonObject>) context.getAttribute("oauthtokens");
            if (oauthtokens.containsKey(auth)) {
                //we could pull out the actual access token, and apply security logic there...
                super.handleRequest(theRequestType,theRequest,theResponse);
            } else {
                theResponse.setStatus(403);    //forbidden.
            }
        } else {
            theResponse.setStatus(403);    //forbidden.
        }
    }
}
```

成功之后就会出现下图的效果 有点丑



后续也应该考虑如何判断 Auth tokens是否过期。

FHIR Connetathon 活动

主要是介绍FHIR Connectathon活动的议题和相关内容

第七届FHIR connectathon活动：FHIR_Connectathon7_for_Java_Dummies

原文链接:FHIR Connectathon 7 for Java Dummies

FHIR Connectathon 7中罗列了三个应用场景.

- 一是对Patient资源的处理 包括了新增,修改和查询等
- 二是对Profile资源的处理 包括了新增和校验等
- 三是基于FHIR的SMART APP的server端和client端的编程
- 四是对FHIR在DICOM中的应用演示 主要是DICOMweb协议与FHIR中ImageStudy资源的整合: 对于FHIR服务器而言,能够暴露DICOMWEB接口 提供ImageStudy资源 并能够将其转换为DICOM 模型的xml json格式; 对于DICOMWEB 服务器构建出ImageStudy资源,暴露FHIR 的rest接口.

让我们看一下利用现有的FHIR开源代码如何来实现这些场景. 所选场景:第一个 开发语言:JAVA(安装配置略) IDE:IDEA community版本 开源库:HAPI-FHIR 测试系统:UBUNTU14.04 32位

1. 新建maven工程,假设为FHIRConnectation7UseCase1,
2. 在pom文件中按照[HAPI-FHIR官网[()上的说明进行配置即可 为了偷懒起见,我直接copy了源码附带的例子中的pom
3. 将main类中的代码用如下代码替换

```
import ca.uhn.fhir.context.FhirContext;
import ca.uhn.fhir.model.api.Bundle;
import ca.uhn.fhir.model.dstu.resource.Conformance;
import ca.uhn.fhir.model.dstu.resource.Patient;
import ca.uhn.fhir.model.dstu.valueset.AdministrativeGenderCodesEnum;
import ca.uhn.fhir.rest.client.IGenericClient;
import ca.uhn.fhir.rest.server.exceptions.ResourceNotFoundException;

//Code copied very substantially from http://jamesagnew.github.io/hapi-fhir/doc_rest_client.html

public class Main {

    public static void main(String[] args) {

        //create the FHIR context
        FhirContext ctx = new FhirContext();
        String serverBase = "https://fhir.orionhealth.com/blaze/fhir";
        IGenericClient client = ctx.newRestfulGenericClient(serverBase);

        // Retrieve the server's conformance statement and print its description
        Conformance conf = client.conformance();
        System.out.println(conf.getDescription().getValue());


        //Read a single patient
        //String id = "1";        //this is not found on the server, and will throw an exception
        String id = "77662";
        try {
            Patient patient = client.read(Patient.class, id);
            //Do something with patient

            // Change the patient gender
            patient.setGender(AdministrativeGenderCodesEnum.M);
            //and save...
            client.update(id, patient);

        } catch (ResourceNotFoundException ex) {
            System.out.println("No patient with the ID of " + id);
        } catch (Exception ex){
            System.out.println("Unexpected error: " + ex.getMessage());
        }

        //Create a new Patient
```

```
        Patient newPatient = new Patient();
        newPatient.addIdentifier("urn:oid:2.16.840.1.113883.2.18.2", "PRP1660");
        newPatient.addName().addFamily("Power").addGiven("Cold");
        client.create()
                .resource(newPatient)
                .encodedJson()
                .execute();

        //do a simple search
        Bundle bundle = client.search()
                .forResource(Patient.class)
                .where(Patient.NAME.matches().value("eve"))
        .execute();
        //bundle will be a HAPI bundle of patients
        System.out.println(bundle.getTitle());
        System.out.println(bundle.getEntries().size() + " matches.");

    }
}
```

原文链接:FHIR Connectathon 7

Connectathon tracks

编者注:为了直观的了解目前**FHIR**标准的进展以及其现阶段能用来做什么，特地从**wiki**上摘录下来 这部分介绍第7届connectathon中将要进行演示的场景.大致可分为两类，Track 1 是针对不熟悉FHIR的新手. Track 2适合已经对FHIR有所了解的开发人员.

Track 1 - Patient

前置条件: 无

1. 注册一个新的**patient**

- 操作: (Patient Demographics consumer) 新建一个patient，调用Patient Service进行保存. 客户端可以为患者分配Id.
- 前置条件: 操作之前该Patient并不存在
- 成功条件: 服务器上成功创建该Patient(use browser to inspect Patient)
- 加分项: Patient资源包含扩展

>>备注: 客户端不一定非要给资源的ID赋值. 但如果是服务器分配的ID,要求客户端能够在服务器的响应获取该id .

2. 更新**patient**信息

- 操作: (Patient Demographics consumer) 更新了场景 #1中所创建的患者信息，调用Patient Service更新服务器端的信息. 通过id来调用patient资源.
- 前置条件: 服务器上已经创建了Patient资源
- 成功条件: 服务器上的Patient资源成功更新 (use browser to inspect Patient)
- 加分项 #1: 更新的患者资源中包含了扩展,但并不对扩展项进行修改 .
- 加分项 #2: 更新的患者资源中包含了扩展,同时对扩展项进行修改.

3. 检索**Patient** 历史记录

- 操作: (Patient Demographics consumer) 调用patient Service 查询Patient的历史记录
- 前置条件: 已经存在一个patient资源，并对其进行了至少一次更新
- 成功条件: 能够成功获得patient历史记录. (use browser query Patient Service)
- 加分项: UI上可以展示患者的历史记录

4. 根据姓名查询**patient**

- 操作: (Patient Demographics consumer) 调用patient Service，根据患者的given name来查询患者
- 前置条件: 服务器上存在Patients，且name字段有值
- 成功条件: 界面上展示出patients(use browser query to confirm)

参考资料:

- Java客户端实例.
- .net 客户端实例.

Track 2 - Profile

分三个步骤:

- 生成 profiles 和 valuesets
- 创建服务端，用以测试一致性
- 实现测试一致性的过程(服务器端和客户端都可以)

这里也尝试使用了profile tags标签.

1. 创建 Profiles 和 Value Sets

只要能够生成Profiles and/or value sets就好了

- 创建一个profile and/or valuesets
  - 构建方法不限-可以手动编码，也可以从其他格式转换得到，以可以使用某些编辑工具
- profile应包含的属性:
  - Profile on Observation
    - 固定observation.name为某些LOINC code
    - 固定observation.value为某个类型
    - 固定observation.value.units
    - 固定一到多个属性的基数
    - 固定reliability 和 status的值
  - 将构建好的profile提交个profile库
    - 取保profile库中的profile/value set 引用准确
  - 创建一个可以通过和失败的资源实例，利用FHIR中提供的校验工具根据profile来校验它们 (可以从FHIR DSTU中下载)

2. 创建服务器，供测试一致性之用

- 已经有了一些observation resources实例 (zip file to be posted here)
- 已经创建好了一个profile (see below)
- 调用服务器上的校验操作，对提交的资源进行校验
  - 并不规定如何提交资源
  - 优先使用该服务器 http://fhir.healthintersections.com.au/open 也可使用如下#3中的一些服务器
  - 校验操作要求profile中的tag为profile所在的的服务器完整URL
  - 客户端必须能够正确处理响应 (成功 | 失败)
- the profile master is here (link to be provided). Consult the server administrator for the correct profile tag for the test profile

3. 服务器校验 Validation

- host a Profile and value set registry
- 实现校验操作 (只需在资源类型层面，无需到资源实例层面)
- 在校验过程中正确处理profile tags
- 根据可标识的profile正确的校验提交的资源
- 成功|失败的结果必须与FHIR校验工具的结果保存一致

Track 3 - SMART on FHIR

实验性质，关注面向用户的可在EHR/PHR系统界面中启动/调用APP. SMART on FHIR使用开放标准(FHIR, OAuth2, OpenID Connect)来构建了一个医疗APP的平台，这个平台能够与现有的医疗信息系统进行集成.

- 大概情况请参考 http://docs.smartplatforms.org/
- 有问题请访问SMART on FHIR Google Group.

1. 构建SMART服务器

一个服务器需要实现如下功能:

1. 支持 `/Patient` and `/Observation` end points, APP可以获取人口统计学和生命体征信息.
2. 支持SMART on FHIR的启动和授权，APP可以获得许可，利用OAuth2得知患者的语境信息.

SMART servers快速入门 ，其中提供了URL、参数、LOINC编码和数据负载的实例.

- 其他参考资料
  http://fhirblog.com/2014/08/02/smart-on-fhir-part-1/

  http://fhirblog.com/2014/08/12/smart-on-fhir-adding-oauth2/

2. 构建 SMART App

客户端可以是web app或者是mobile app，可以在SMART on FHIR testing server上运行. 应满足如下功能:

- 能够与SMART's sandbox server 通信
- 通过授权获取患者数据
- 获取诸如人口统计学、体征和实验室检查检验等数据
- 提供一些患者数据的展示，最简单的表格也可以

SMART apps快速入门包含了一些代码和范例.

如需更多信息，可查看源代码 sample apps on GitHub.

最开始可以在本地进行开发 `[http://localhost](http://localhost)` 无需进行注册 --如果你想在网络上运行你的APP， 只需按照app 注册指南操作即可.

Track 4 - DICOMweb (Joint with DICOM)

DICOMweb是医学影像的web标准. 主要是一些RESTFUL服务接口，能够让web应用开发人员充分利用既有的工具来实现医学影像的功能.

尽管DICOMweb's REST接口和序列化格式与 FHIR不同, FHIR中的 ImagingStudy是联合DICOM一起开发的，为的是能够让DICOM服务器利用FHIR ImagingStudy通过FHIR REST api暴露服务器上的数据,反过来，对于FHIR服务器而言，能够利用ImagingStudy Resources 通过DICOMweb's REST api暴露服务器上的数据.

利用WADO-RS:

- 对于FHIR servers: 通过DICOMweb's REST interface来暴露FHIR ImagingStudy资源,将ImagingStudy转换为DICOM model的xml和或json 格式
- 对于DICOMweb servers: 将一系列study实例整合成一个FHIR ImagingStudy资源, 通过 FHIR's REST interface (basically, a read operation)暴露资源，将FHIR的ImagingStudy转换为DICOM serialization model的 xml和或json格式 .