A GraphQL approach to Healthcare Information Exchange with HL7 FHIR

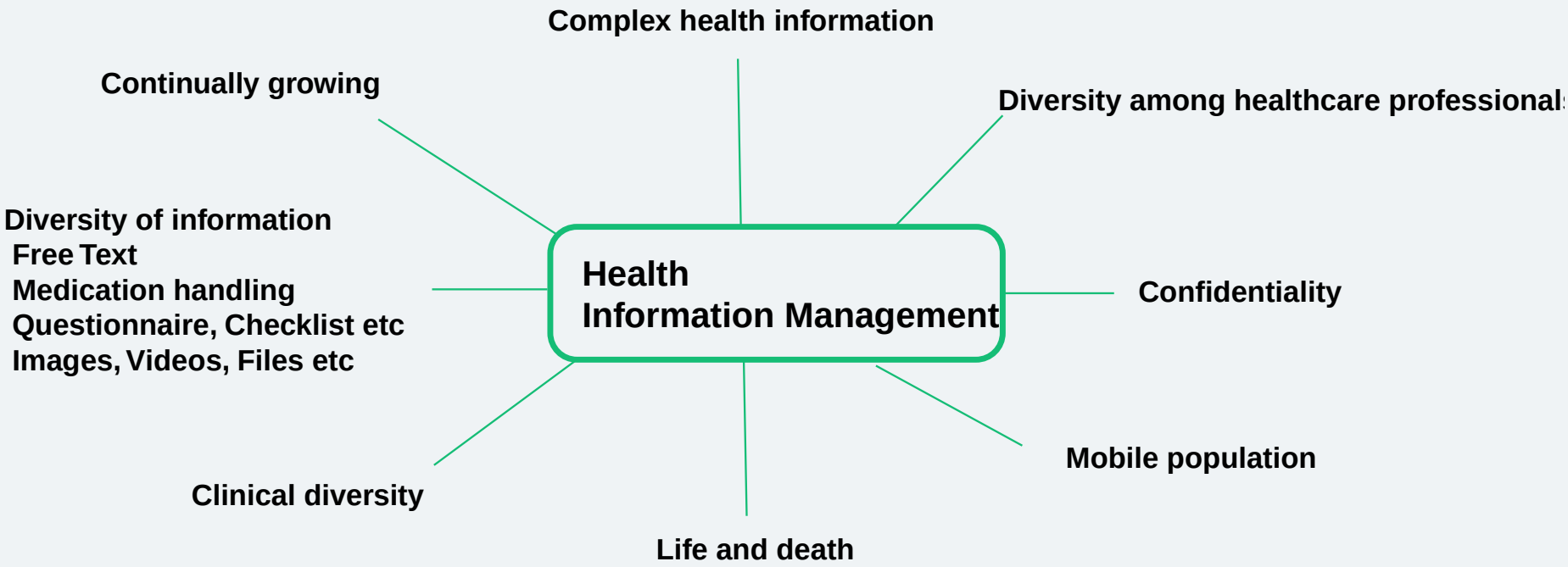
Suresh Kumar Mukhiya, Fazle Rabbi, Ka I Violet Pun,
Adrian Rutle, Yngve Lamo
skmu@hvl.no

November 5, 2019

Who are we?

- Our aim is to improve public mental health with Adaptive technologies (ICT) and psychological treatments.
- Team of Domain experts, Software Engineers, ML engineers, HCI engineers and Industry.

Why is Health Information(HI) management so hard?



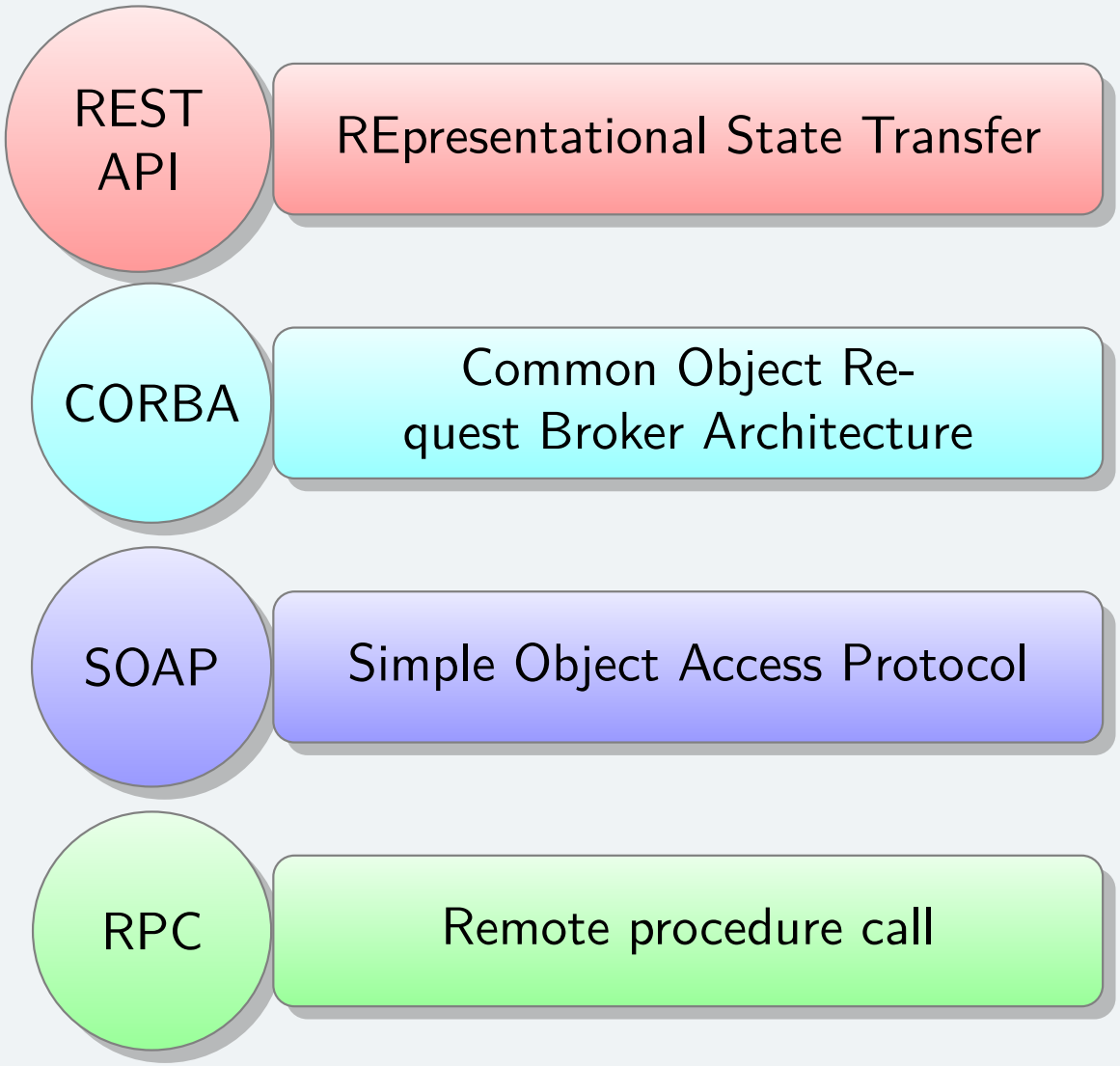
Why do we need to structure HI?

- Avoid repetitive data entry
- Retrieval and overview
- Reuse of record info
- **For technical integration**
- **For information exchange**
- Clinical decision support
- Quality indicators
- Management data

How can we structure HI?

By using of consistent standards defining syntactic and semantic meaning of information being exchanged.

Health Information Interchange



Issues with RESTful API

- Query Complexity
- Overfetching
- Under-fetching and `n+1` request problem
- Modifiability
- API versioning

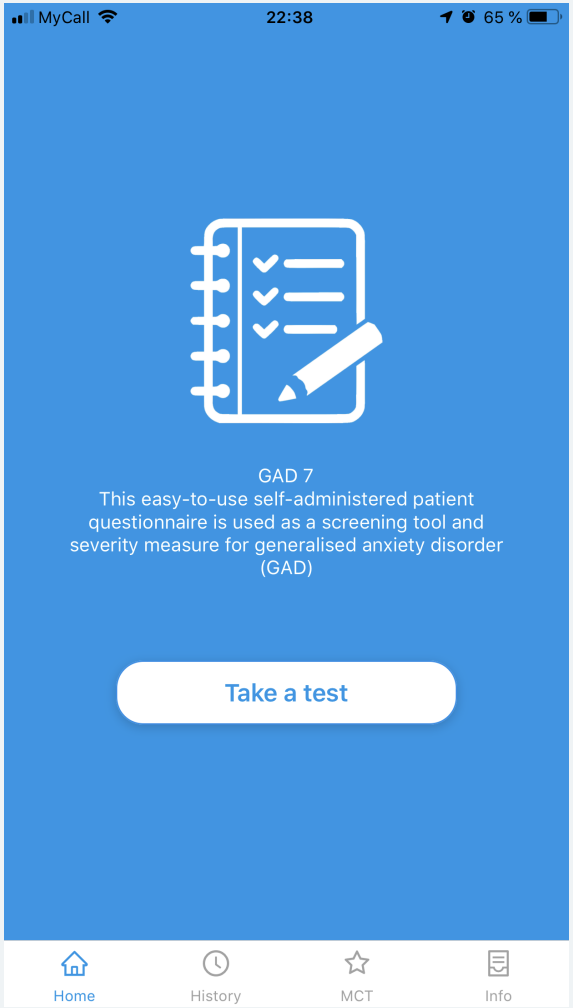
RESTFul approach - Patients

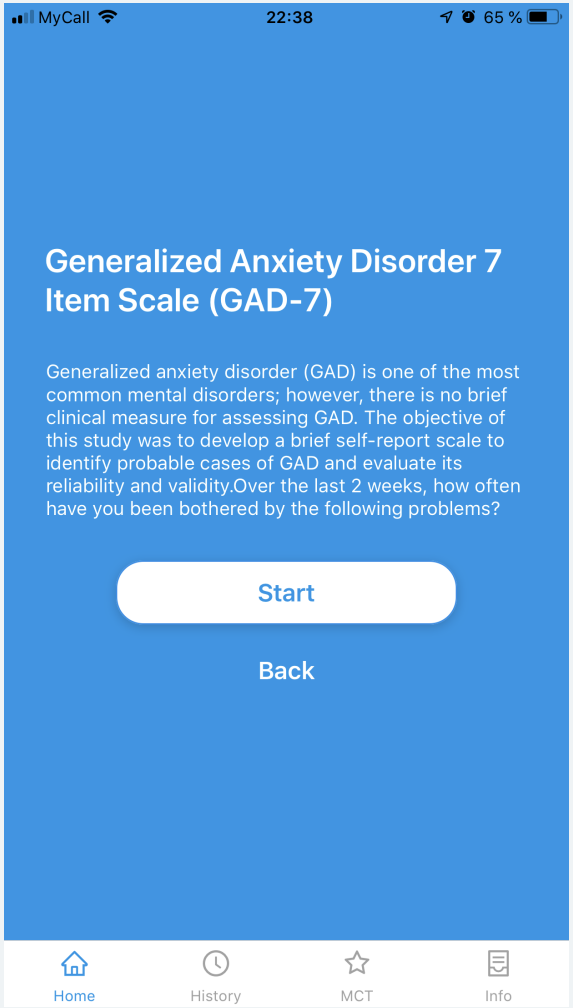
- Base URL: http://hapi.fhir.org/baseDstu3

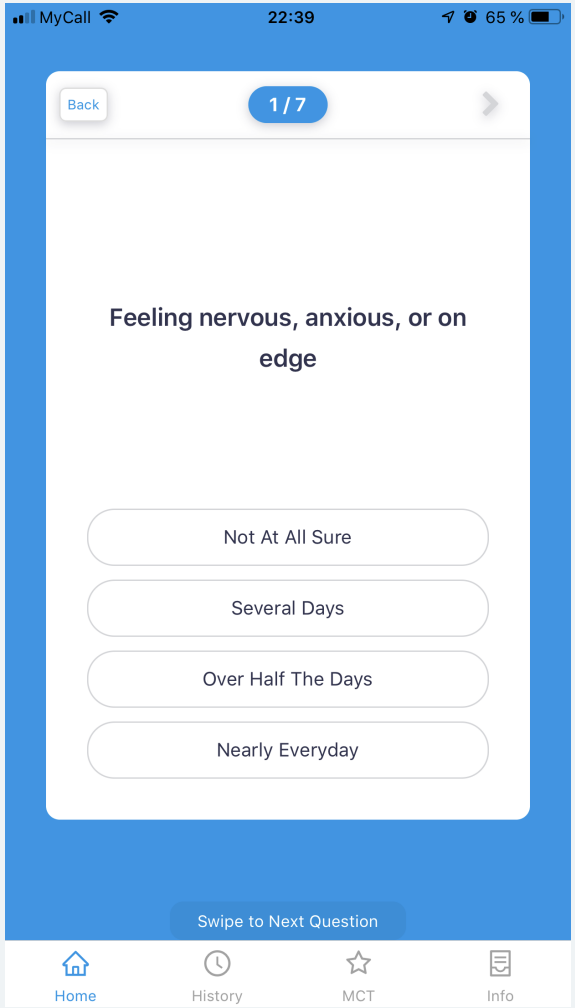
GET	/Patient	All Patients
POST	/Patient	Create Patient
GET	/Patient/649227	Single Patient
PUT	/Patient/649227	Update Patient
DELETE	/Patient/649227	Delete Patient

- https://www.hl7.org/fhir/resourcelist.html lists 143 `HL7 FHIR` resources.
- Assuming **Best Case**, each resources needs 5 endpoints (CRUD) resulting (143 x 5 = 715) endpoints.
- **Worst Case**: Custom endpoints based on custom requirements. Eg. all patients over 30 years, all patients only from Bergen etc.

Self Assessment App

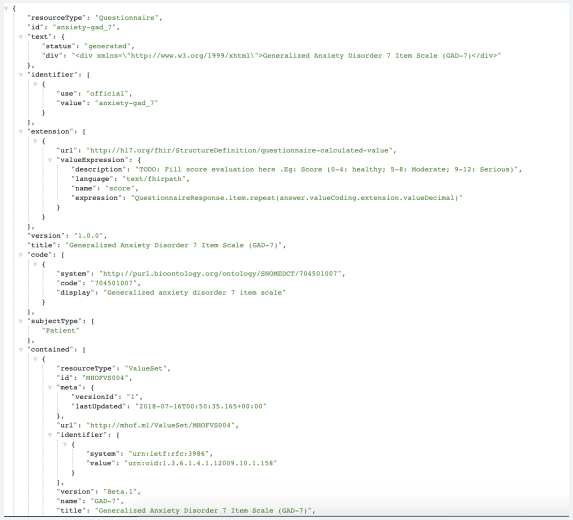






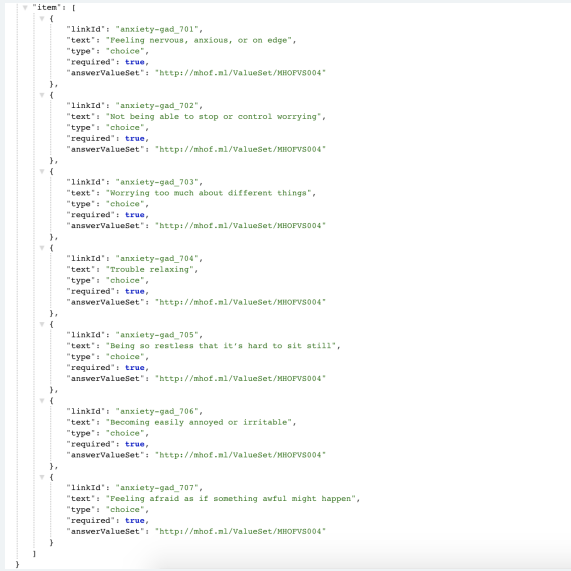
1. (name, description), 2. (name, description, title), 3. Question and options

Response From REST - Overfetching



```
{
  "resourceType": "Questionnaire",
  "id": "anxiety-gad_7",
  "text": {
    "status": "generated",
    "div": "<div xmlns=\"http://www.w3.org/1999/xhtml\">Generalized Anxiety Disorder 7 Item Scale (GAD-7)</div>"
  },
  "identifier": [
    {
      "use": "official",
      "value": "anxiety-gad_7"
    }
  ],
  "extension": [
    {
      "url": "http://hl7.org/fhir/StructureDefinition/questionnaire-calculated-value",
      "valueExpression": {
        "description": "TODO: Fill score evaluation here .Eg: Score (0-4: healthy; 5-8: Moderate; 9-12: Serious)",
        "language": "text/fhirpath",
        "name": "score",
        "expression": "QuestionnaireResponse.item.repeat(answer.valueCoding.extension.valueDecimal)"
      }
    }
  ],
  "version": "1.0.0",
  "title": "Generalized Anxiety Disorder 7 Item Scale (GAD-7)",
  "code": [
    {
      "system": "http://purl.bioontology.org/ontology/SNOMEDCT/704501007",
      "code": "704501007",
      "display": "Generalized anxiety disorder 7 item scale"
    }
  ],
  "subjectType": [
    "Patient"
  ],
  "contained": [
    {
      "resourceType": "ValueSet",
      "id": "MHOFVS004",
      "meta": {
        "versionId": "1",
        "lastUpdated": "2018-07-16T00:50:35.165+00:00"
      },
      "url": "http://mhof.ml/ValueSet/MHOFVS004",
      "identifier": [
        {
          "system": "urn:ietf:rfc:3986",
          "value": "urn:oid:1.3.6.1.4.1.12009.10.1.158"
        }
      ],
      "version": "Beta.1",
      "name": "GAD-7",
      "title": "Generalized Anxiety Disorder 7 Item Scale (GAD-7)",
```


```
        ]
      }
    ],
    "compose": {
      "include": [
        {
          "system": "http://mhof.ml",
          "concept": [
            {
              "code": "MHOFVS004-1",
              "display": "Not At All Sure",
              "extension": [
                {
                  "url": "http://hl7.org/fhir/StructureDefinition/valueset-ordinalValue",
                  "valueDecimal": 0
                }
              ]
            },
            {
              "code": "MHOFVS004-2",
              "display": "Several Days",
              "extension": [
                {
                  "url": "http://hl7.org/fhir/StructureDefinition/valueset-ordinalValue",
                  "valueDecimal": 1
                }
              ]
            },
            {
              "code": "MHOFVS004-3",
              "display": "Over Half The Days",
              "extension": [
                {
                  "url": "http://hl7.org/fhir/StructureDefinition/valueset-ordinalValue",
                  "valueDecimal": 2
                }
              ]
            },
            {
              "code": "MHOFVS004-4",
              "display": "Nearly Everyday",
              "extension": [
                {
                  "url": "http://hl7.org/fhir/StructureDefinition/valueset-ordinalValue",
                  "valueDecimal": 3
                }
              ]
            }
          ]
        }
      ]
    }
  ],
```

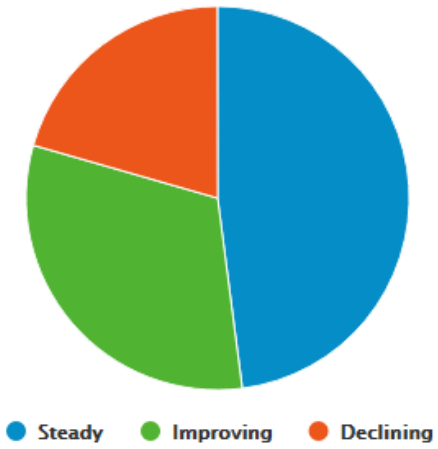



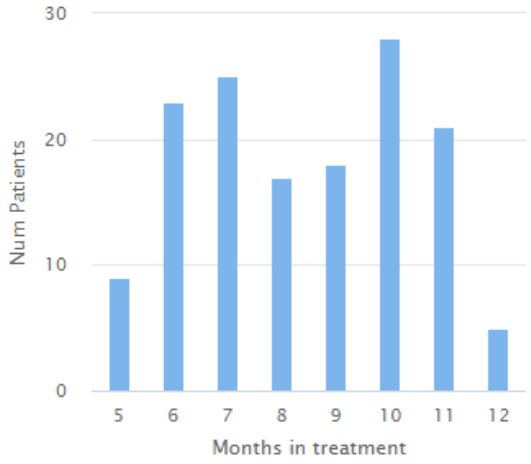
```
  "item": [
    {
      "linkId": "anxiety-gad_701",
      "text": "Feeling nervous, anxious, or on edge",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    },
    {
      "linkId": "anxiety-gad_702",
      "text": "Not being able to stop or control worrying",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    },
    {
      "linkId": "anxiety-gad_703",
      "text": "Worrying too much about different things",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    },
    {
      "linkId": "anxiety-gad_704",
      "text": "Trouble relaxing",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    },
    {
      "linkId": "anxiety-gad_705",
      "text": "Being so restless that it's hard to sit still",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    },
    {
      "linkId": "anxiety-gad_706",
      "text": "Becoming easily annoyed or irritable",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    },
    {
      "linkId": "anxiety-gad_707",
      "text": "Feeling afraid as if something awful might happen",
      "type": "choice",
      "required": true,
      "answerValueSet": "http://mhof.ml/ValueSet/MHOFVS004"
    }
  ]
}
```


- Getting more information than required - overfetching.
- Getting less information than required - Underfetching
- Need for addition `HTTP` request to get all the required resource - `n+1 request` problem

Response From REST - Visualization

Dashboard





Warnings

Name	Reason
Aaron697 Herzog843	Suicidal thoughts 5
Adrian111 Metz686	Suicidal thoughts 5
Arlette667 Effertz744	Suicidal thoughts 5
Blair400 Langosh790	Suicidal thoughts 6
Calvin845 McGlynn426	Suicidal thoughts 6
Carla633 Sandoval902	Suicidal thoughts 5

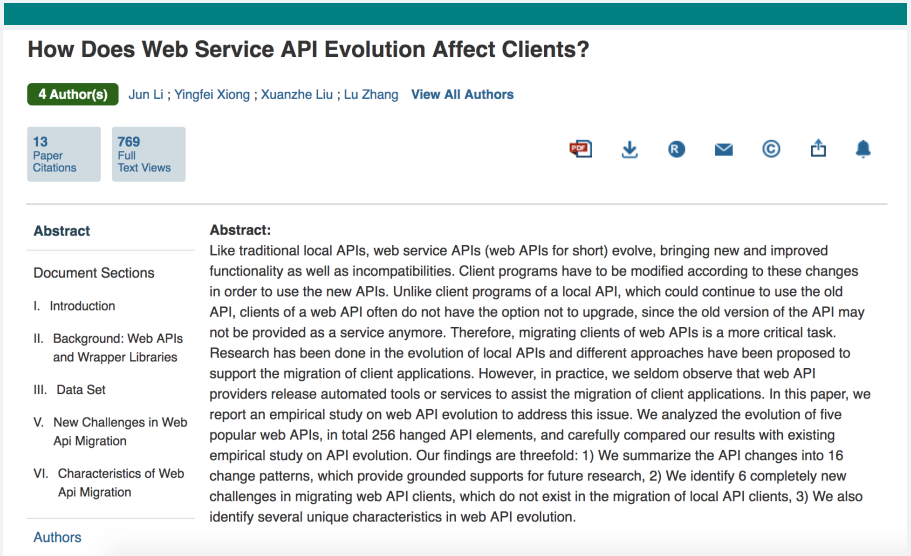
Your patients

Search..

ALL | NOT CHECKED | CHECKED

Name	Time spent in program	Flags	Progression	Last checked	Urgency score
Aaron697 Herzog843	9 months	Suicidal thoughts +2 (5)	declining	1 days	40
Adrian111 Metz686	10 months		steady	7 days	35
Alfonzo975 Schumm995	6 months		steady	7 days	42
Aline709 Mayert710	8 months		improving	7 days	34
Alphonso102 Wisoky380	10 months		steady	11 days	33
Anisha16 Hartmann983	5 months	Concentration difficulties +2 (5)	steady	5 days	38
Annis955 Reinger292	8 months		steady	Never	35

API versioning



How Does Web Service API Evolution Affect Clients?

4 Author(s) Jun Li ; Yingfei Xiong ; Xuanzhe Liu ; Lu Zhang View All Authors

13 Paper Citations | 769 Full Text Views

Abstract

Document Sections

I. Introduction

II. Background: Web APIs and Wrapper Libraries

III. Data Set

V. New Challenges in Web Api Migration

VI. Characteristics of Web Api Migration

Authors

Abstract:
Like traditional local APIs, web service APIs (web APIs for short) evolve, bringing new and improved functionality as well as incompatibilities. Client programs have to be modified according to these changes in order to use the new APIs. Unlike client programs of a local API, which could continue to use the old API, clients of a web API often do not have the option not to upgrade, since the old version of the API may not be provided as a service anymore. Therefore, migrating clients of web APIs is a more critical task. Research has been done in the evolution of local APIs and different approaches have been proposed to support the migration of client applications. However, in practice, we seldom observe that web API providers release automated tools or services to assist the migration of client applications. In this paper, we report an empirical study on web API evolution to address this issue. We analyzed the evolution of five popular web APIs, in total 256 hanged API elements, and carefully compared our results with existing empirical study on API evolution. Our findings are threefold: 1) We summarize the API changes into 16 change patterns, which provide grounded supports for future research, 2) We identify 6 completely new challenges in migrating web API clients, which do not exist in the migration of local API clients, 3) We also identify several unique characteristics in web API evolution.

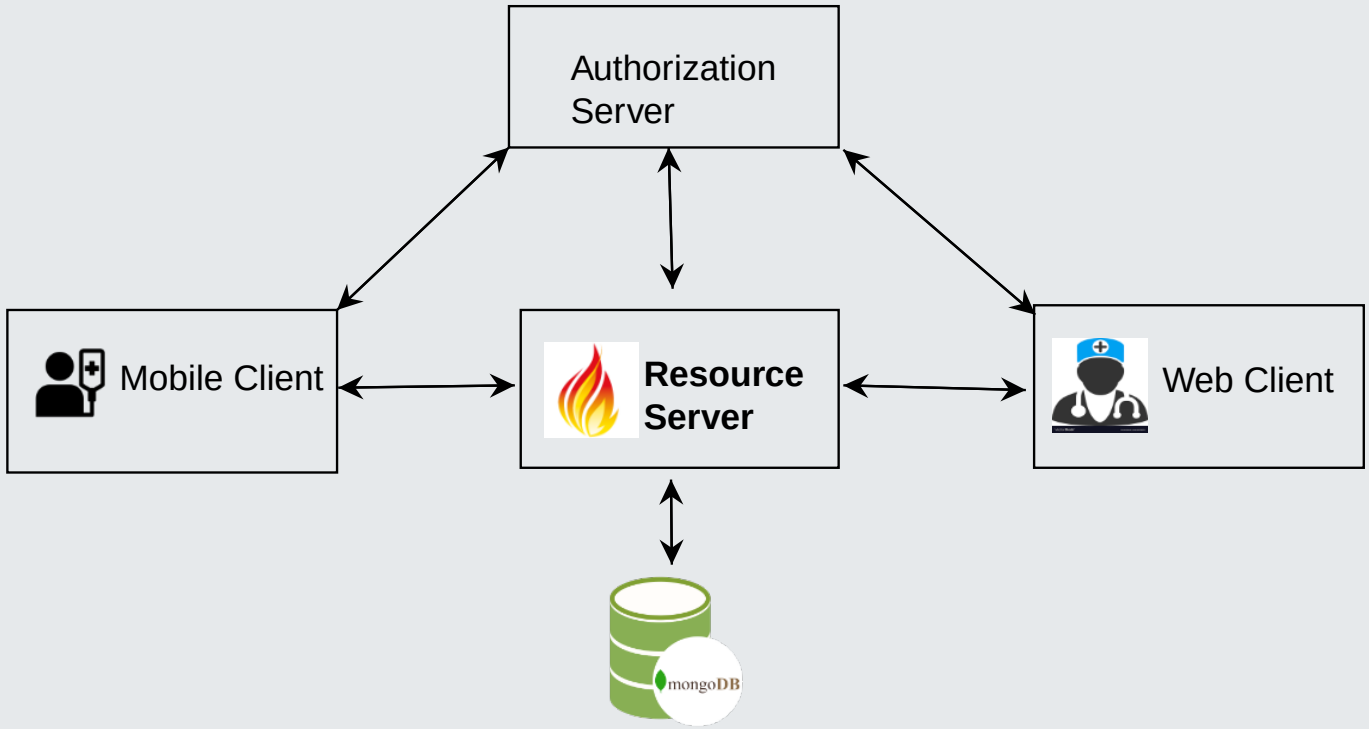
- the requirements of clients are often dynamic.
- can modify the existing endpoints or create endpoints to only fetch required resources. BUT requirements changes quickly.
- modification quickly becomes inflexible as one need to think about how to support existing customers while providing new functionality.

Our solution

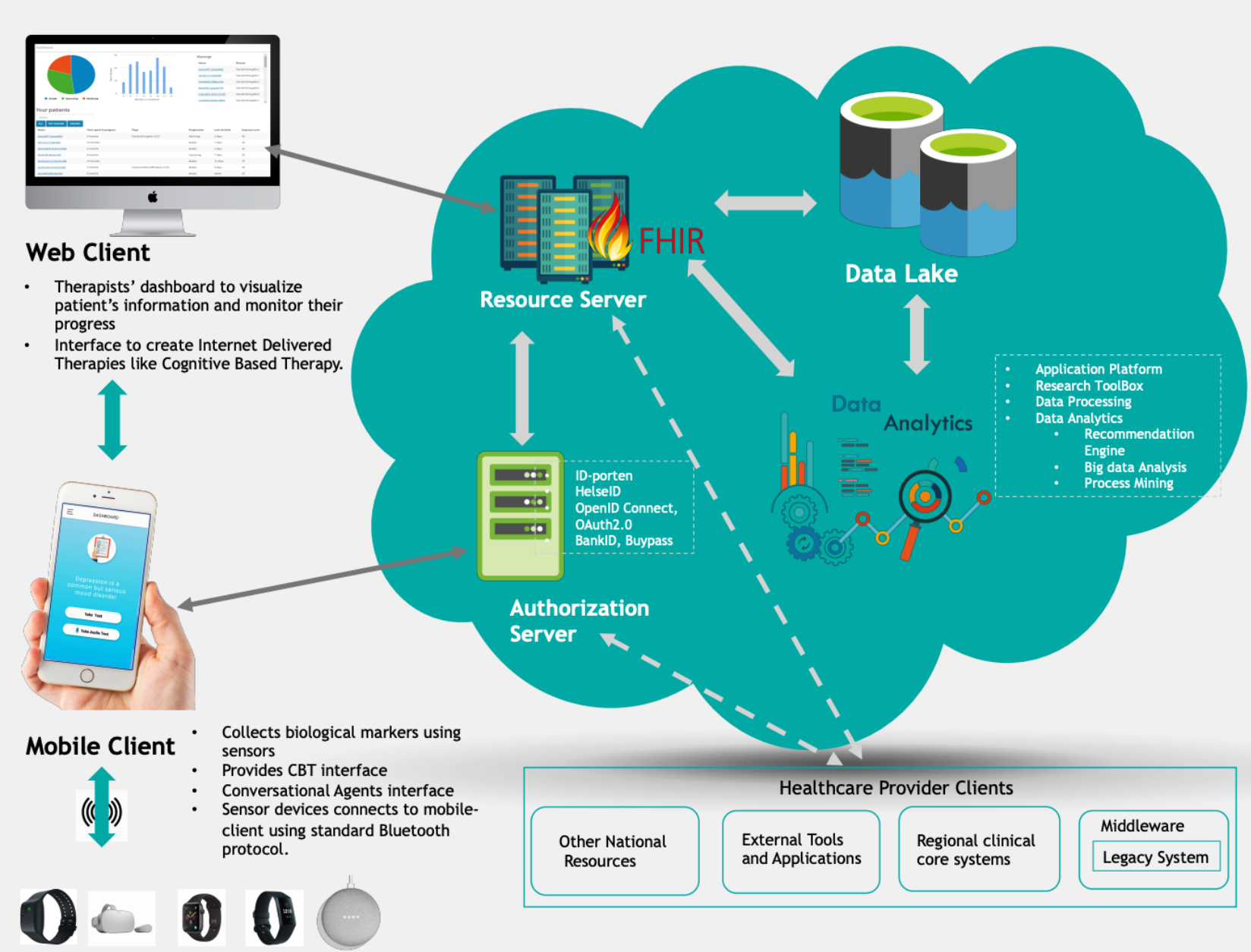
GraphQL

GraphQL developed by Facebook and has been embraced users including Coursera, GitHub, Pinterest, Neo4J, PayPal and others.

Prototype



Architecture Centric Development (ACD)



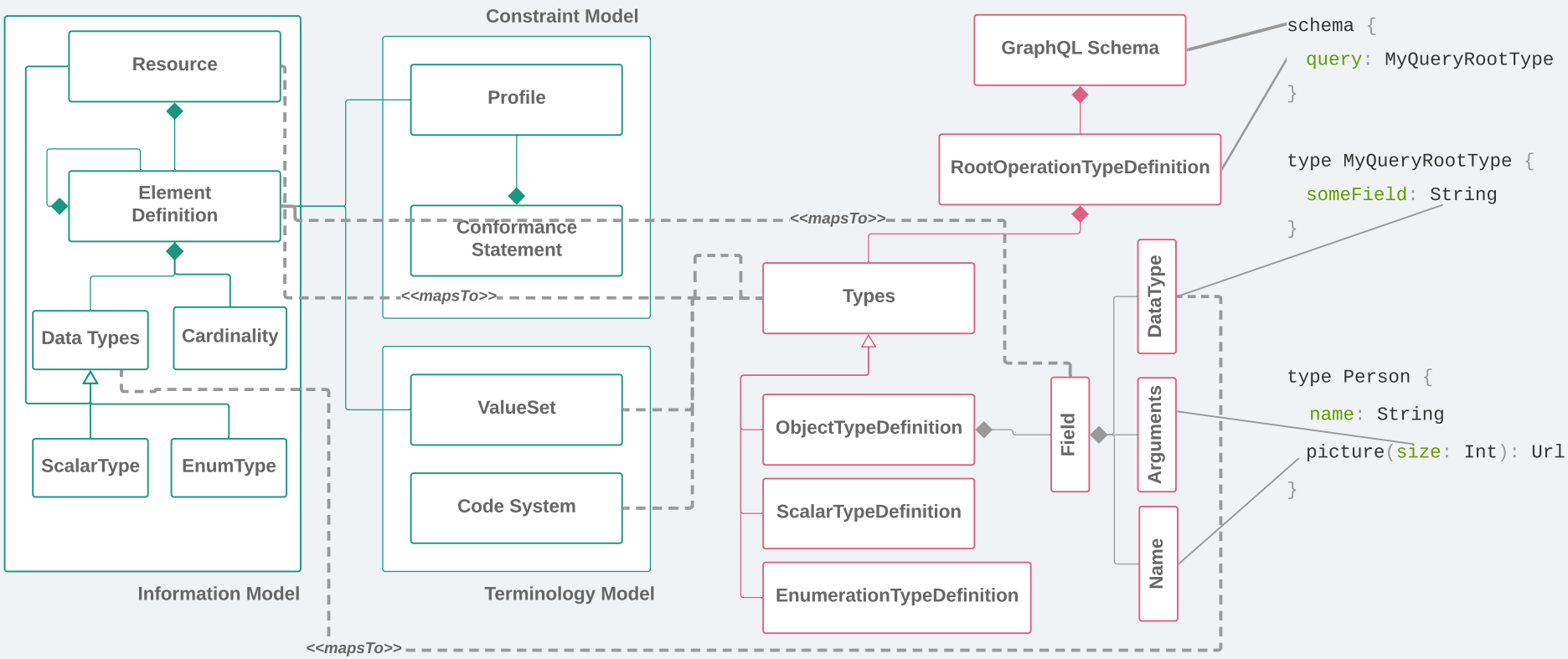
Mapping HL7 FHIR resources to GraphQL schema

Algorithm 1: Mapping HL7 FHIR resources to GraphQL schema

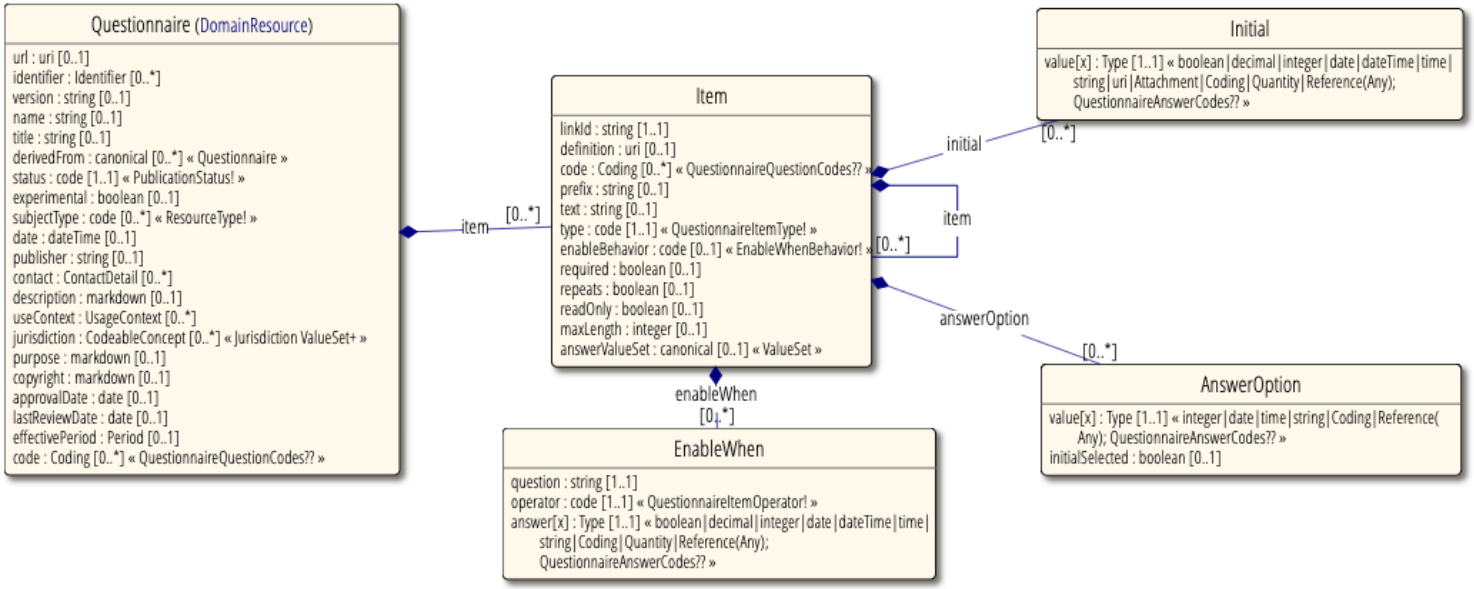
Input: HL7 Resource
Output: GraphQL Schema

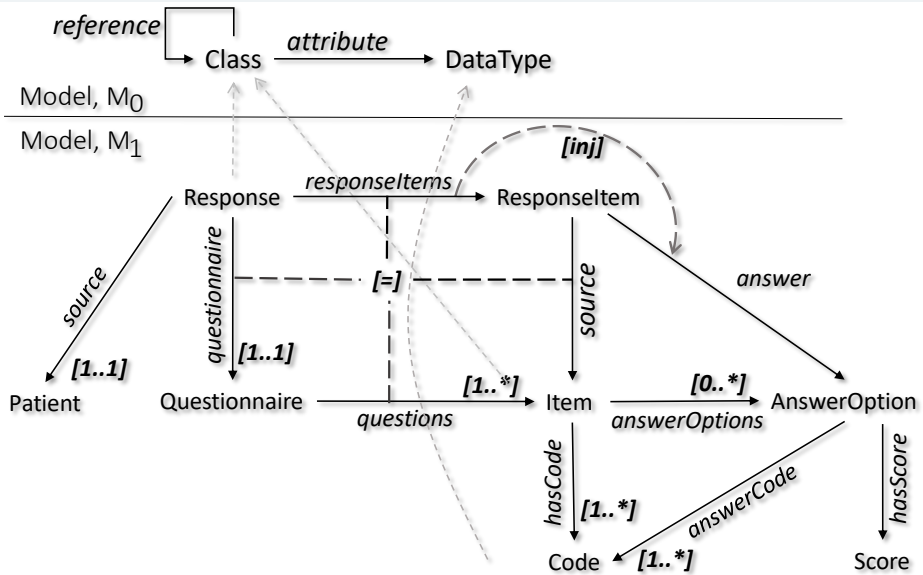
```
1  function recursive_hl7fhir_graphql_mapper (Resource)
2   schema = {};
3  foreach field ∈ HL7 Resource.fields do
4  switch Resource.field do
5      case field.Type is ScalarTypeDefinition do
6          if field.cardinality is 0,1 then
7              add_to_schema(field, type)
8          end if
9          if field.cardinality is 0,* then
10             add_to_schema_as_list(field, type)
11         end if
12     end case
13     case field.Type is EnumTypeDefinition do
14         if EnumTypeDefinition already exists then
15             - reference to schema
16         else
17             - define_new_type_enum(**args)
18             - reference to schema
19         end if
20     end case
21     case field.Type is Custom OR field.Type is HL7 FHIR Resource do
22         if Custom OR Resource already exists then
23             - reference to schema
24         else
25             - define new type Resource
26             - reference to schema
27             - recursive_hl7fhir_graphql_mapper(Resource)
28         end if
```

Mapping HL7 FHIR with GraphQL schema



Case Study - Questionnaire





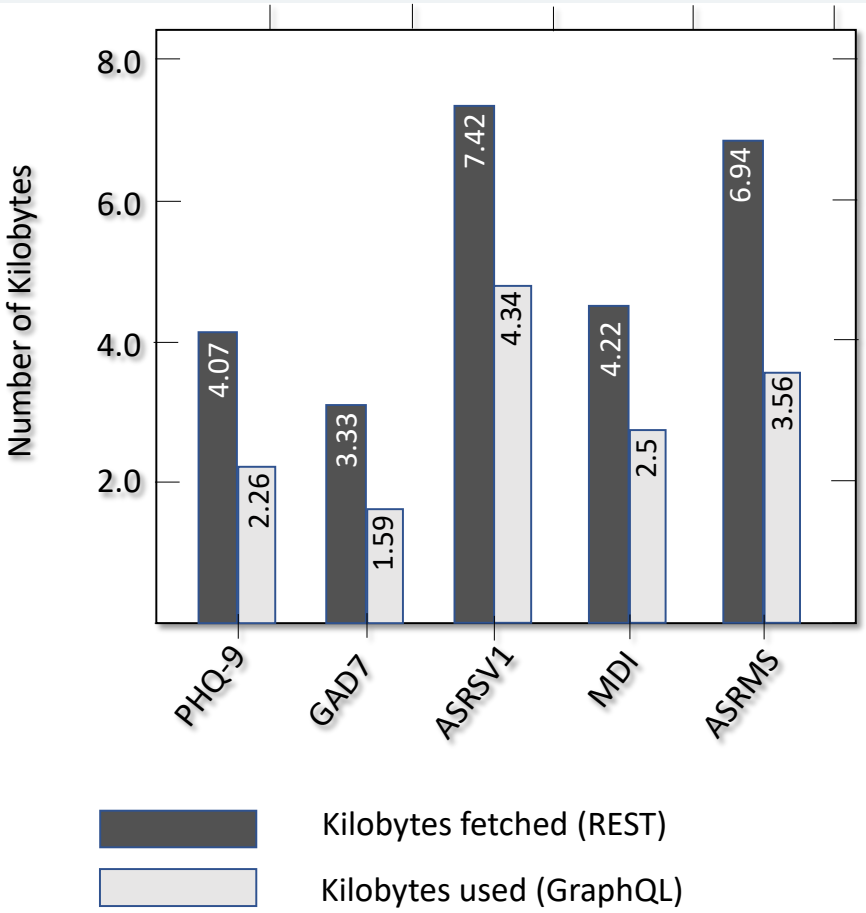
Schema Definition

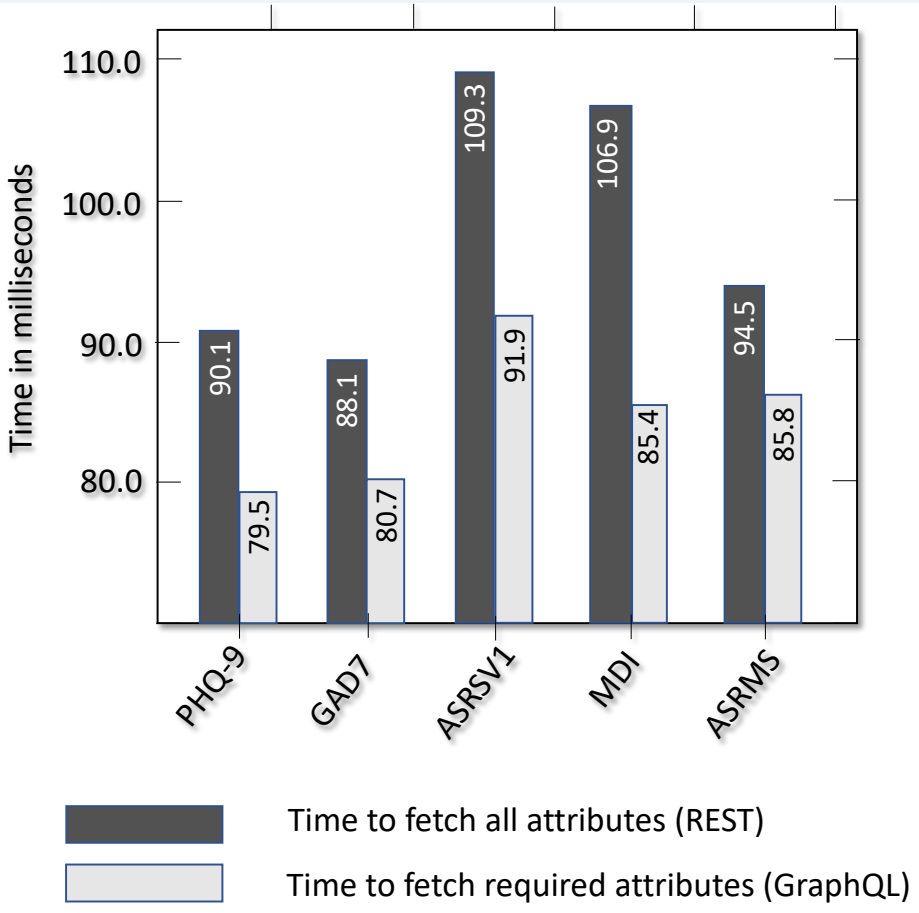
```
type Questionnaire {
    resourceType: String
    url: String
    identifier: [Identifier]
    version: String
    name: String
    title: String
    status: statusEnumType
    experimental: Boolean
    publisher: String
    description: String
    item: [QuestionnaireItem]
  }

  type Identifier {
    system: String
    value: String
    use: indentifierEnumType
  }

  type QuestionnaireItem {
    linkId: String
    prefix: String
    text: String
    type:
          QuestionnaireItemTypeEnum

    required: Boolean
    answerValueSet: [ValueSet]
  }
```

```
type ValueSet {
    resourceType: String
    url: String
    identifier: [Identifier]
    version: String
    name: String
    title: String
    status: statusEnumType
    experimental: Boolean
    publisher: String
    description: String
    compose: Compose
  }

  type Compose {
    inactive: Boolean
    include: [ComposeInclude]
  }

  type ComposeInclude {
    concept: [
          ComposeIncludeConcept
          ]
  }

type ComposeIncludeConcept {
    code: coding
    display: String
    extension: [Extension]
  }
```

```
    type Extension {
    uri: String
    valueDecimal: Float
  }

  enum statusEnumType {
    draft
    active
    retired
    unknown
  }

  enum indentifierEnum {
    usual
    official
    temp
    secondary
    old
  }

  enum TypeEnum {
    group
    display
    boolean
    decimal
    integer
    date
    dateTime +
  }
```

Evaluation of Response size and Time





Throughput and response time with RESTful approach

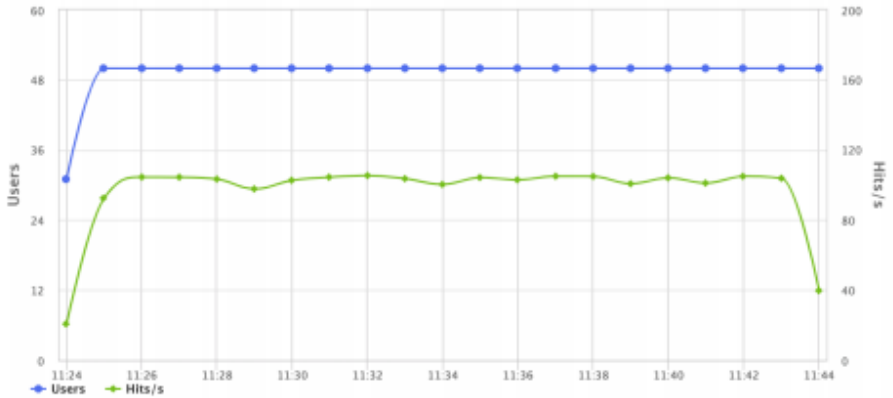


Fig. 4: Concurrent users and number of hits per second when fetching all the available attributes from the `Questionnaire`

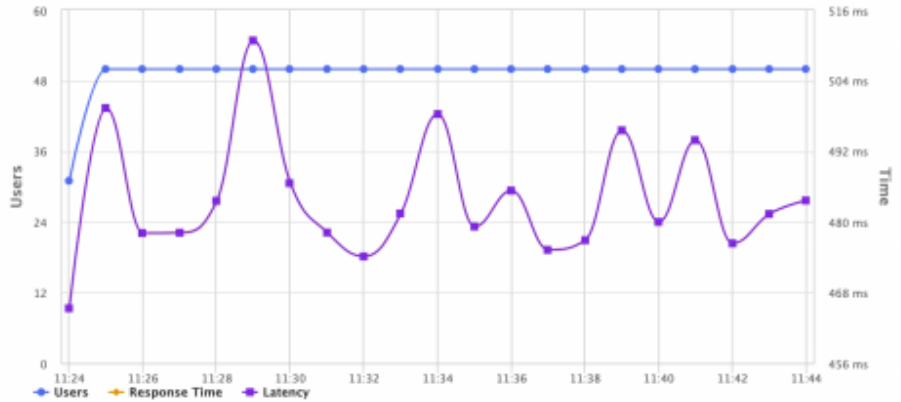


Fig. 5: Concurrent users and response time (milliseconds) when fetching only the required attributes from the `Questionnaire`

Throughput and response time with GraphQL

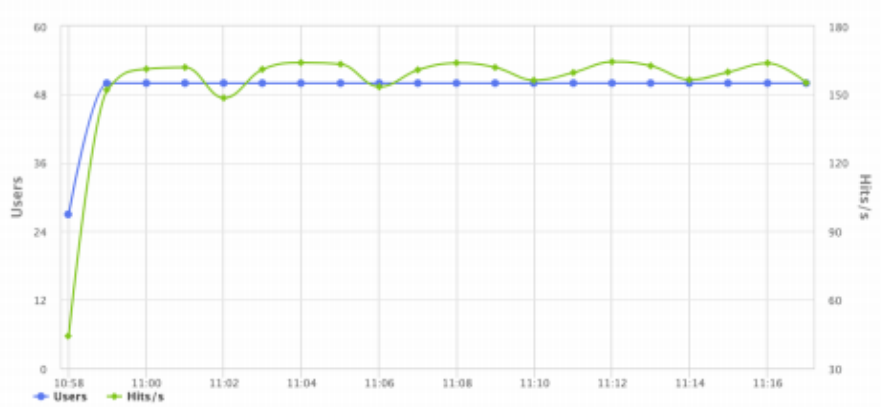


Fig. 6: Concurrent users and number of hits per second when fetching only the required attributes from the `Questionnaire`

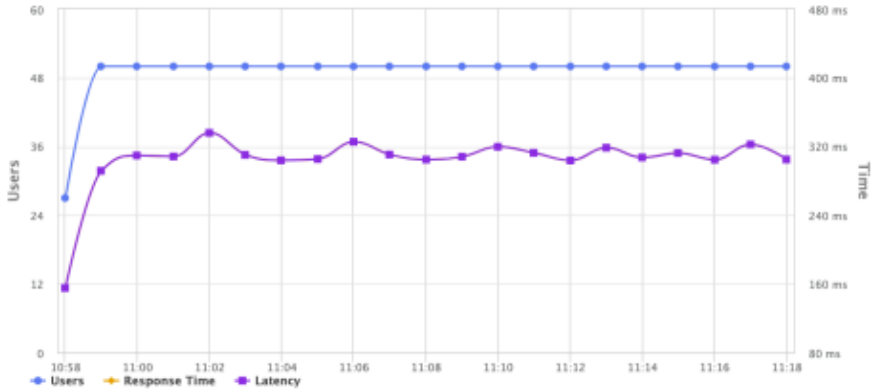


Fig. 7: Concurrent users and response time (milliseconds) when fetching only the required attributes from the `Questionnaire`

Performance Test statistics

Description	All attributes	Required Attributes
Average Throughput	100.6 hits/second	157.7 hits/second
Average Response Time	484 millisecond	308 millisecond
Test Start time	Mon, 06/03/2019 - 11:24	Jun 03, 2019, 10:58:29 AM
Test Stop Time	Mon, 06/03/2019 - 11:44	Jun 03, 2019, 11:18:54 AM
Time Elapsed	20 minutes	20 minutes
Concurrent Users	50	50

Table 1: Performance test meta-data for fetching GAD-7 `Questionnaire` resource. Column 1: description of the meta data, column 2: meta data for fetching all attributes from the endpoints. column 3: meta data when fetching only required attributes

- Throughput :average number of HTTP/s requests per second generated by the test
- Response Time: average amount of time from first bit sent to the network card to the last byte received by the client.

Challenge: Circular relationship complexity

```
type ThreadDefinition {
  title: String
  text(first: Int, after: String): [
        MessageDefinition]
}

type MessageDefinition {
  text: String
  thread: ThreadDefinition
}

type Query {
  thread(id: ID!): ThreadDefinition
}
```

```
query complicatedQuery {
  thread(id: "ID") {
    text(first: 100000) {
      thread {
        text(first: 100000) {
          thread {
              # ...repeat 100000 (n) times
                    ...
            }
          }
        }
      }
    }
  }
}
```

Challenge: Schema Duplication

- schema definition based on the choice of the database being used (this project uses mongoDB, so schema are based on mongoose [1]);
- schema definition for a `GraphQL` endpoint.
- Good thing is community is already aware of both issues. And there are solutions being worked out currently.

[1] https://mongoosejs.com/

Future Directions

- Schema Stitching or Schema federation to promote interoperability between current health systems and legacy systems.
- Creation of adaptive Internet-Delivered Psychological Interventions (IDPT) using GraphQL based HIE.
- Create a comprehensible dashboard for better visualization for therapists.
- Further research in both development of Adaptive system and clinical trials are required.

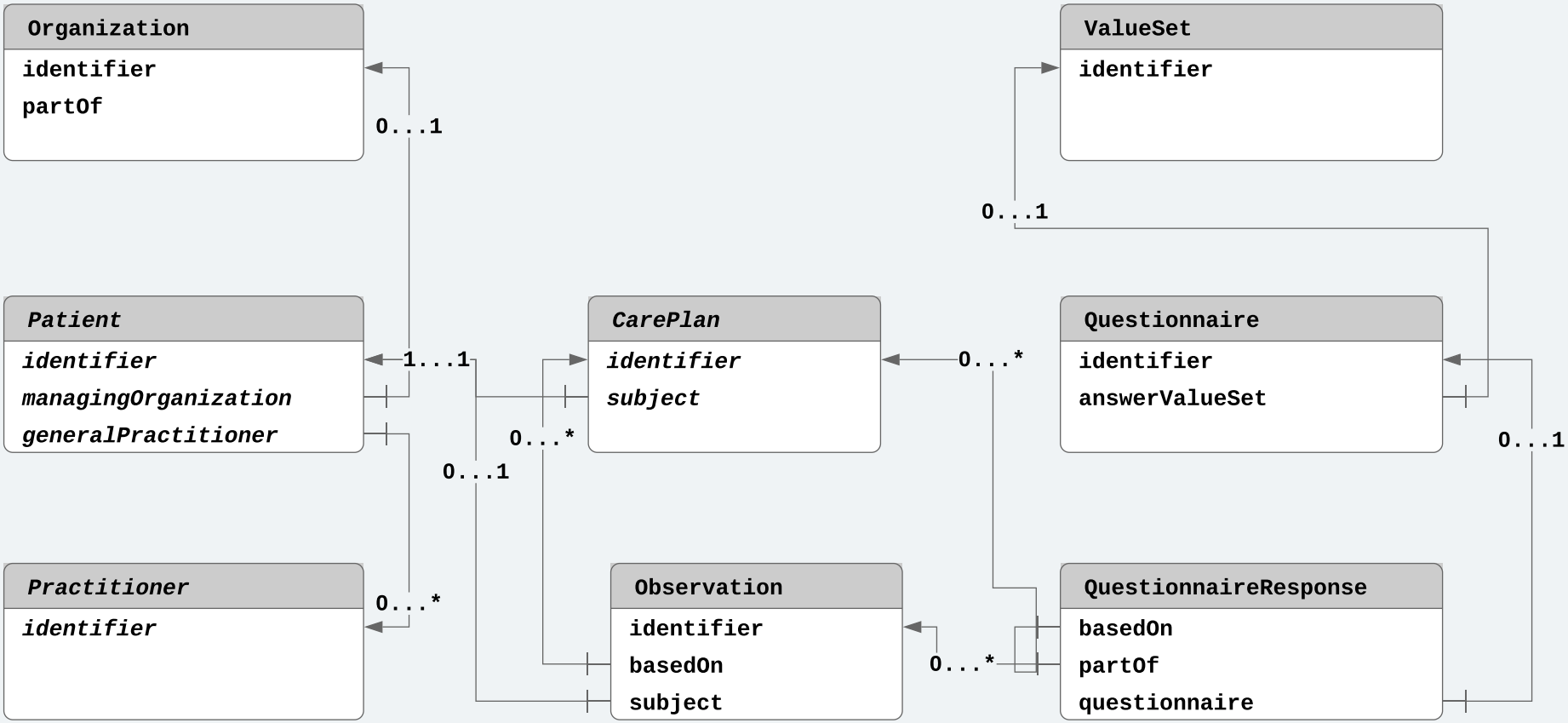
Conclusion: ISO/IEC 25000.ISO/IEC 25000:2005

- Interoperability ✓`HL7 FHIR`
- Security ✓`SMART on FHIR`
- Modifiability ✓`SOA-Oriented architecture, GraphQL`
- Scalability ✓`SOA-Oriented architecture`
- Testability ✓`TDD Approach`

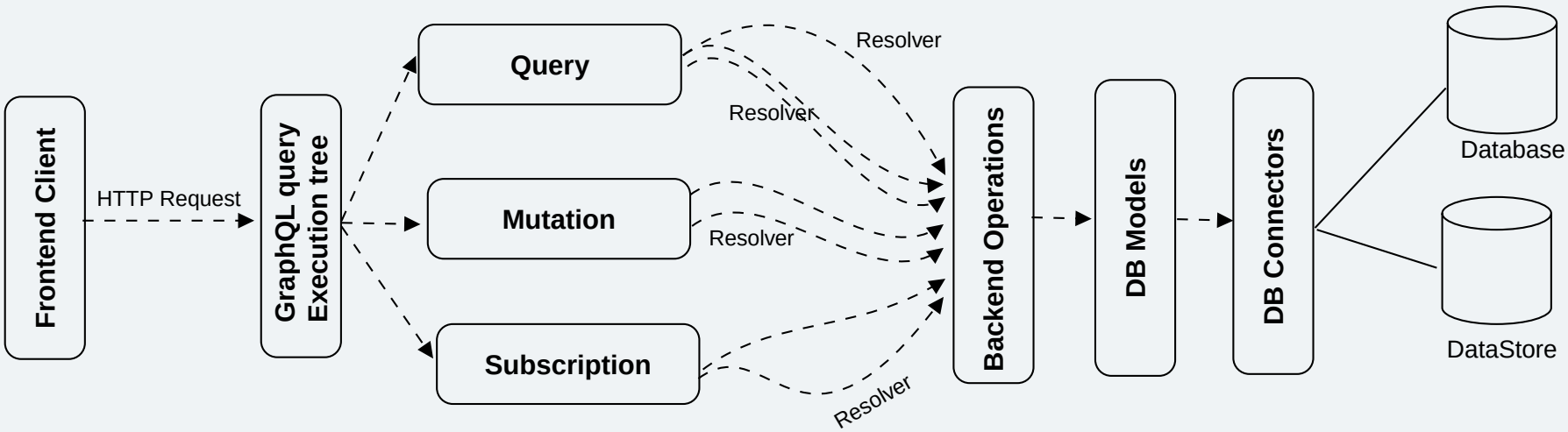
Thank You

skmu@hvl.no

HL7 FHIR ER Diagram



How does GraphQL work?



Resolvers

What is a resolver?

A resolver is a function that resolves a value for a type or field in a schema. If an Object is returned, execution continues to the next child field. If a scalar is returned (typically at a leaf node), execution completes. If null is returned, execution halts and does not continue.

Executing queries

- **Parse:** A query is parsed into an abstract syntax tree (or AST).
- **Validate:** The AST is validated against the schema. Checks for correct query syntax and if the fields exist.
- **Execute:** The runtime walks through the AST, starting from the root of the tree, invokes resolvers, collects up results, and emits JSON.

References I

Duane Bender and Kamran Sartipi.

HL7 FHIR: An agile and RESTful approach to healthcare information exchange.
In *Proceedings of CBMS 2013 - 26th IEEE International Symposium on Computer-Based Medical Systems*, 2013.

M. Bryant.

Graphql for archival metadata: An overview of the ehri graphql api.
In *2017 IEEE International Conference on Big Data (Big Data)*, pages 2225–2230, Dec 2017.

ISO/IEC 25000.

ISO/IEC 25000:2005 Software Engineering - Software product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE, 2005.

R. Srinivasan.

RPC: Remote Procedure Call Protocol Specification Version 2.
RFC 1831, DDN Network Information Center, 1995.

Roy Fielding.

Architectural Styles and the Design of Network-based Software Architectures.
PhD thesis, 2000.

HI7 FHIR SMART app launch, Retrieved December 28.

Roberto Rodriguez-Echeverria, Javier Luis Cánovas Izquierdo, and Jordi Cabot.

Towards a UML and IFML Mapping to GraphQL.
In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018.

Facebook.

Graphql: A query language for apis, Retrieved April 11, 2019.

References II

Ka I Pun Yngve Lamo Suresh Kumar Mukhiya, Fazle Rabbi.

An architectural design forself-reporting e-health systems.
In *ICSE 2019 Proceedings in the IEEE Digital Library*, 2018.

Hevner, March, Park, and Ram.

Design Science in Information Systems Research.
MIS Quarterly,, 2004.

H. Ulrich, J. Kern, D. Tas, A. K. Kock-Schoppenhauer, F. Ückert, J. Ingenerf, and M. Lablans.

QL 4 MDR: A GraphQL query language for ISO 11179-based metadata repositories.
BMC Medical Informatics and Decision Making, 2019.

Jun Li, Yingfei Xiong, Xuanzhe Liu, and Lu Zhang.

How does web service API evolution affect clients?
In *Proceedings - IEEE 20th International Conference on Web Services, ICWS 2013*, 2013.

Khalil Khoumbati and Marinos Themistocleous.

Integrating the IT Infrastructures in Healthcare Organisations : a Proposition of Influential Factors.
Electronic Journal of e-Government, 2006.

Yigang Xu, Dominique Sauquet, Eric Zapletal, David Lemaitre, and Patrice Degoulet.

Integration of medical applications: The 'mediator service' of the SynEx platform.
International Journal of Medical Informatics, 2000.

Olaf Hartig and Jorge Pérez.

Semantics and Complexity of GraphQL .
2018.

References III

Nat Sakimura, John Bradley, Michael B. Jones, Breno Medeiros, and Chuck Mortimore.

Final: OpenID Connect Core 1.0 incorporating, 2014.

P. Ferguson and D. Senie.

Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing.
Technical report, 2000.

Cross-origin Resource Sharing.

Cross-Origin Resource Sharing.
Options, 2011.

Suresh Kumar Mukhiya and Khac Hoang Hung.

An Architectural Style for Single Page Scalable Modern Web Application.
5(4):6–13, 2018.

ISO/IEC 25000.

ISO/IEC 25000:2005 Software Engineering - Software product Quality Requirements and Evaluation (SQuaRE) - Guide to SQuaRE, 2005.

Scott Ambler.

Process Patterns Building Large-Scale Systems Using Object Technology.
1998.

A GraphQL approach to Healthcare Information Exchange with HL7 FHIR

Suresh Kumar Mukhiya, Fazle Rabbi, Ka I Violet Pun,
Adrian Rutle, Yngve Lamo
skmu@hvl.no

November 5, 2019