

openIMIS

Gumzo ya Mwezi 01/07/2019



Agenda

- Bluesquare:
 - who we are
 - our engagement towards openIMIS community
 - our methodology
- Achieved
 - what we delivered this month
- Roadmap
 - what we will deliver and what are our dependencies





www.bluesquarehub.com

Delivering
innovative
technology for
better lives.

Fall 2018 —

what we do

COUNTRY LEVEL DATA SYSTEMS 24 COUNTRIES

We build technologies that enhance governmental health data systems with a focus on three markets:

HEALTH FINANCING DATA SYSTEMS

- Data systems for purchasers, health insurance, Ministries of Health
- Example: Develop a Pay for Performance data system in Kyrgyz hospitals

GOVERNMENT HEALTH DATA WAREHOUSES

- Example: The health data warehouse in Morocco

DISEASE OR THEMATIC DATA SYSTEMS

- Diabetes
- HIV
- Tuberculosis
- Malaria
- Immunization systems
- Vector Borne eradication systems (i.e. sleeping sickness)
- Family Planning
- Emergency Obstetric Care

Bluesquare develops these data systems based on a suite of in-house software products connected to DHIS2 a popular open source data management platform used by over 40 governments.

How we do IT products and data services

We deliver technologies and services that strengthen governmental health data systems, mainly:

Hesabu (aka ORBF)

- An open sourced rule engine that allows complex calculations in DHIS2, a popular open source data management platform. This is particularly useful for health financing data systems.

Data Viz

- A public web dashboard that allows showcasing results.

Modeling and data science

- Statistical analysis, Data cleaning, Modeling & machine learning and analysis automation to help customers bring value out of their health data.

Bluesquare suite of in-house software products and services allow collecting, computing, analyzing and visualizing data in a intelligent and friendly manner.



24

COUNTRIES

Bluesquare: our engagement towards openIMIS community

We believe that health insurance will be at the heart of the UHC agenda in many countries.

openIMIS modular transformation is an opportunity to develop code that can be used at scale to help provide health services “for the global good” (i.e. exact DNA of Bluesquare).

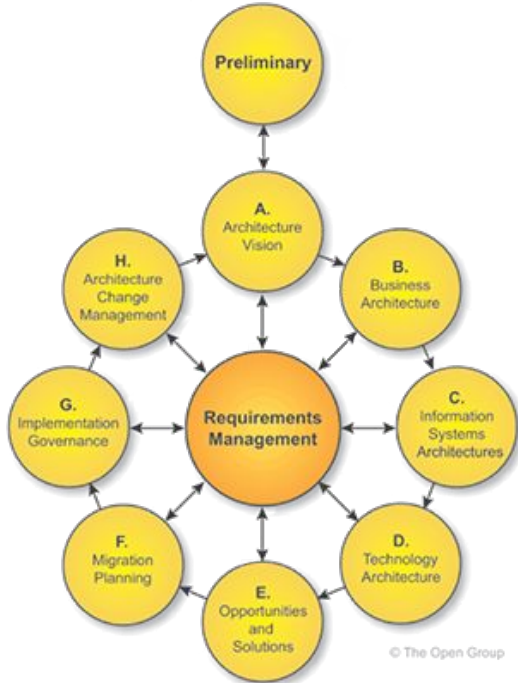
Creating synergies with our existing and future health-financing portfolio, promoting the tool in the countries where we operate.



Bluesquare: our methodology

Our approach to deliver the openMIS modules borrows several concepts from TOGAF, most important one being the ADM (Architecture Development Method):

- Iterative, ensuring pragmatism and responsiveness in delivered solution
- We strive to keep things simple: we aim to use the TOGAF framework as a guide not a rule book. Where we feel it will serve this project we will make use of it. However, our proposed approach is much lighter than a traditional TOGAF implementation effort.
- It helps any community member to find/contribute to the appropriate part of the system.



Agenda

- Bluesquare:
 - who we are
 - our engagement towards openIMIS community
 - our methodology
- Achieved
 - what we delivered this month
- Roadmap
 - what we will deliver and what are our dependencies

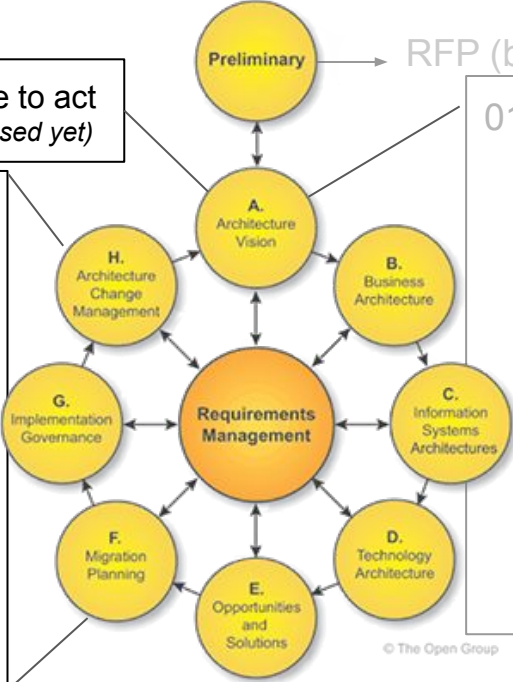


Achieved (Iteration 1): FHIR API

No architecture change to act
(but deliverable not really used yet)

01/2019: start of work

- Brainstorming (blsq internal)
- Conceptual architecture documentation (in openMIS wiki)
- Architecture Presentations (16/01 and 23/01 + follow-up calls) 6 m/d
- Technology stack proposal
- Migration strategy & roadmap
- PoC on proposed technology stack (preparation 'hands on' session at Bonn Workshop) 14 m/d



Under progress:
Objective iteration 1: FHIR API
03/2019 13 m/d
04/2019: 12 m/d
05/2019: 4 m/d
06/2019: 0,5 m/d
Workaround for small bug (rowid)

TODO (Bluesquare):
follow up, bug fix and adaptations

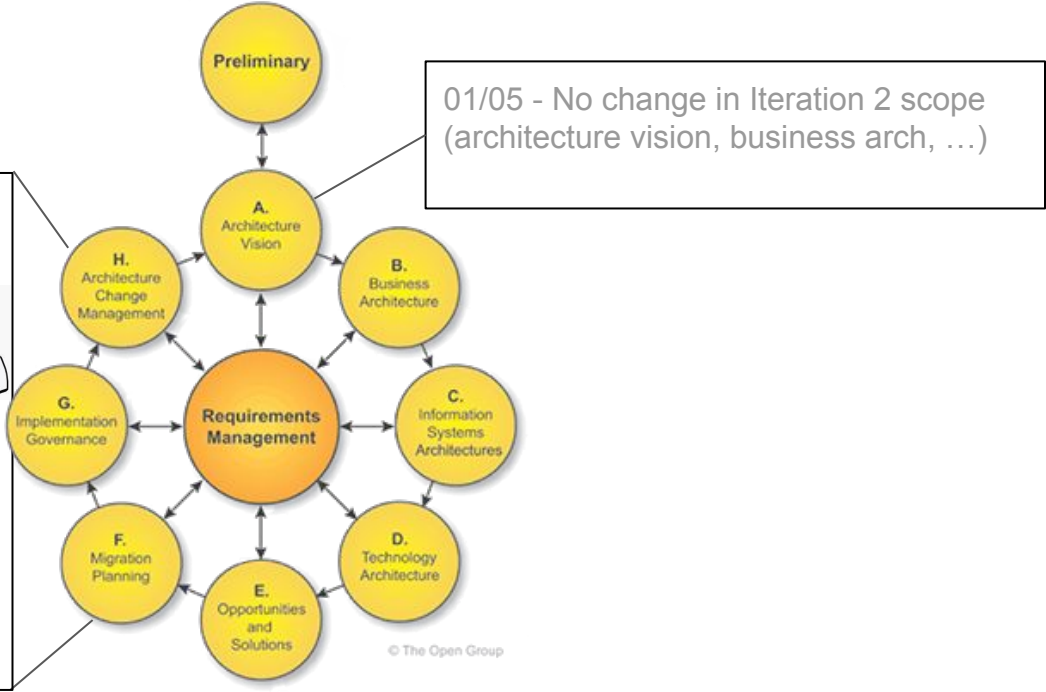

Bonn Workshop
- Demonstrated technology stack and migration strategy
- Agenda alignments
4 m/d

Achieved (Iteration 2): All Proxies

Under maintenance:
Objective iteration 1: All proxy (frontend) setup
05/2019: 10 m/d
06/2019: 8,5 m/d

Delivered (Bluesquare):
- test server: <http://openimis.bluesquare.org>
- login workaround (without login API) switched from nginx to OpenResty
- translations mechanism (modular contributions and language packs)
- First remarks (from Dragos) on main menu implemented: Menu in AppBar and responsive

Enquiry screen moved into the claim module scope!



TODO (Bluesquare):
follow up, bug fix and adaptations





Select Criteria

Claim Details

Region

HF Name

Visit Date From To

District

Review Status

Claim Date From To

HF Code

Feedback Status

Main Dg.

Claim Administrator

Claim Status

Batch Run

Insurance Number

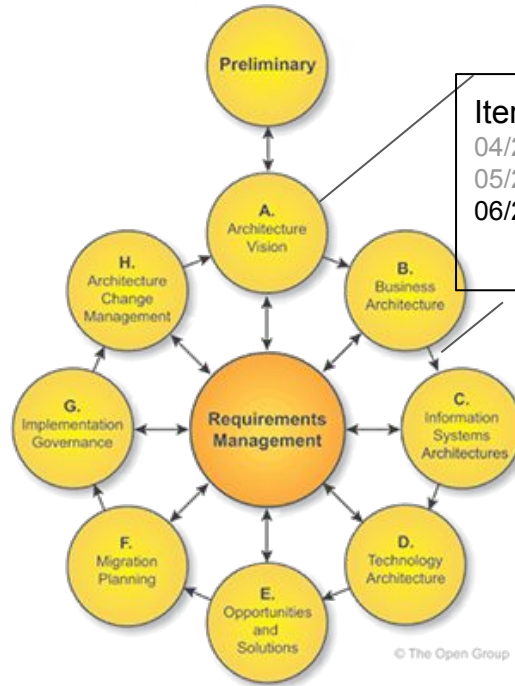
Claim No.

Visit Type

Claims Found

Claims Found	
--------------	--

Achieved (Iteration 3): Claim Module



Iteration 3:

04/2019 : Claim module scope document (draft) 2 m/d

05/2019 : Mapping attempt to JLN business processes 0.5 m/d

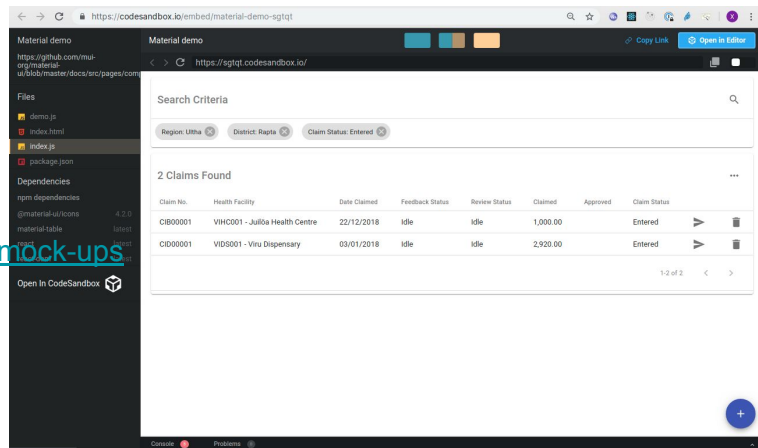
06/2019 : Claim main screen mockups (+ sessions with implementors) 4,5 m/d

06/2019: Django Dynamic Rest > Graphene (GraphQL) 4 m/d

Screen mockups

<https://openimis.atlassian.net/wiki/spaces/OP/pages/859308033/Claim+Screens+mock-ups>

- Eligibility (enquiry): <https://codesandbox.io/embed/material-demo-b4bw1>
- Health Facility Claims: <https://codesandbox.io/embed/material-demo-sgtqt>
- Claim Edit: <https://codesandbox.io/embed/material-demo-5059z>
- Review: <https://codesandbox.io/embed/material-demo-36swk>
- Claim Feedback: <https://codesandbox.io/embed/material-demo-3erif>
- Claim Review: <https://codesandbox.io/embed/material-demo-ji6dx>
- Search Service: <https://codesandbox.io/embed/material-demo-9oxrx>



Graphene

- Data structure exposed to API
- Schema published
- REST based
- Easy developer UI with debugging & performance tools

Graphene

GraphiQL ▶ Prettify Merge History

```
1 {
2   allClaims {
3     id
4     approved
5     approvalStatus
6     dateProcessed
7     insuree {
8       id
9       lastName
10      healthFacility {
11        name
12      }
13    }
14    claimitemSet {
15      id
16      priceAsked
17      qtyProvided
18      status
19    }
20  }
21 }
22
```

QUERY VARIABLES

1

< ClaimType **HealthFacilityType** x

🔍 Search HealthFacilityType...

No Description

FIELDS

id: ID!

legacyId: Int

code: String!

accCode: String

name: String!

level: String!

address: String

phone: String

fax: String

email: String

careType: String!

validityFrom: DateTime!

validityTo: DateTime

offline: Boolean!

auditUserId: Int!

insureeSet: [InsureeType]

claimadminSet: [ClaimAdminType]

claimSet: [ClaimType]

Graphene

GraphiQL



Prettify

Merge

History

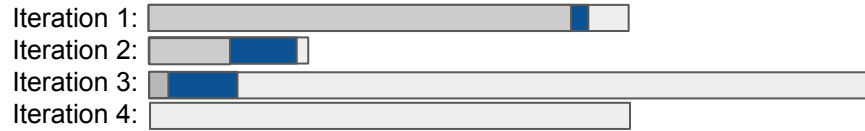
< Docs

```
1 {
2   allClaims {
3     id
4     approved
5     approvalStatus
6     dateProcessed
7     insuree {
8       id
9       lastName
10      healthFacility {
11        name
12      }
13    }
14    claimitemSet {
15      id
16      priceAsked
17      qtyProvided
18      status
19    }
20  }
21  _debug {
22    sql {
23      rawSql
24      duration
25      isSlow
26    }
27  }
28 }
29
```

QUERY VARIABLES

1

```
    },
    "claimitemSet": []
  }
},
"_debug": {
  "sql": [
    {
      "rawSql": "SELECT [tblClaim].[ClaimID], [tblClaim].[LegacyID], [tblClaim].[ClaimCategory],
[tblClaim].[InsureeID], [tblClaim].[ClaimCode], [tblClaim].[DateFrom], [tblClaim].[DateTo],
[tblClaim].[ClaimStatus], [tblClaim].[Adjuster], [tblClaim].[Adjustment], [tblClaim].[Claimed],
[tblClaim].[Approved], [tblClaim].[Reinsured], [tblClaim].[Valuated], [tblClaim].[DateClaimed],
[tblClaim].[DateProcessed], [tblClaim].[FeedbackID], [tblClaim].[Explanation], [tblClaim].[
FeedbackStatus], [tblClaim].[ReviewStatus], [tblClaim].[ApprovalStatus], [tblClaim].[
RejectionReason], [tblClaim].[ValidityFrom], [tblClaim].[ValidityTo], [tblClaim].[AuditUserID],
[tblClaim].[ValidityFromReview], [tblClaim].[ValidityToReview], [tblClaim].[HFID], [tblClaim].[
SubmitStamp], [tblClaim].[ProcessStamp], [tblClaim].[Remunerated], [tblClaim].[GuaranteeID],
[tblClaim].[ClaimAdminId], [tblClaim].[ICDID], [tblClaim].[ICDID1], [tblClaim].[ICDID2], [tblClaim].[
ICDID3], [tblClaim].[ICDID4], [tblClaim].[VisitType], [tblClaim].[AuditUserIDReview], [tblClaim].[
AuditUserIDSubmit], [tblClaim].[AuditUserIDProcess], [tblClaim].[RowID] FROM [tblClaim]",
      "duration": 0.0018610954284667969,
      "isSlow": false
    },
    {
      "rawSql": "SELECT [tblInsuree].[InsureeID], [tblInsuree].[LegacyID], [tblInsuree].[CHFID],
[tblInsuree].[LastName], [tblInsuree].[OtherNames], [tblInsuree].[Gender], [tblInsuree].[DOB],
[tblInsuree].[IsHead], [tblInsuree].[Marital], [tblInsuree].[passport], [tblInsuree].[Phone],
[tblInsuree].[Email], [tblInsuree].[CurrentAddress], [tblInsuree].[GeoLocation], [tblInsuree].[
CurrentVillage], [tblInsuree].[PhotoDate], [tblInsuree].[CardIssued], [tblInsuree].[HFID],
[tblInsuree].[ValidityFrom], [tblInsuree].[ValidityTo], [tblInsuree].[isOffline], [tblInsuree].[
AuditUserID], [tblInsuree].[RowID] FROM [tblInsuree] WHERE [tblInsuree].[InsureeID] = %s",
      "duration": 0.001592874526977539,
      "isSlow": false
    },
    {
      "rawSql": "SELECT [tblInsuree].[InsureeID], [tblInsuree].[LegacyID], [tblInsuree].[CHFID],
[tblInsuree].[LastName], [tblInsuree].[OtherNames], [tblInsuree].[Gender], [tblInsuree].[DOB],
[tblInsuree].[IsHead], [tblInsuree].[Marital], [tblInsuree].[passport], [tblInsuree].[Phone],
[tblInsuree].[Email], [tblInsuree].[CurrentAddress], [tblInsuree].[GeoLocation], [tblInsuree].[
CurrentVillage], [tblInsuree].[PhotoDate], [tblInsuree].[CardIssued], [tblInsuree].[HFID],
[tblInsuree].[ValidityFrom], [tblInsuree].[ValidityTo], [tblInsuree].[isOffline], [tblInsuree].[
AuditUserID], [tblInsuree].[RowID] FROM [tblInsuree] WHERE [tblInsuree].[InsureeID] = %s",
      "duration": 0.0015869140625,
      "isSlow": false
    }
  ],
}
```

Roadmap (Bluesquare)

- Iteration 1 (04/2019): Building blocks for FHIR API

Dependencies:

- ✓ Identified data to be mapped (cfr. xls of Soldevelo) & stored proc to be called
- ✓ Module boundaries (started with the one documented in wiki and shown @Bonn)
- follow up, bug fix and adaptations

- Iteration 2 (05/2019): “All proxy” openIMIS

Dependencies:



Login API (adaptation in current openIMIS)



Screen layout (mainly the menu structure) >> still under validation (<https://openimis.bluesquare.org>)

- Iteration 3 (09/2019): Claim module

Dependencies:



Final specifications (starting from existing process) >> feedback on screen layouts (country variations as module parameters [with ceilings,...]?!?)

- Acceptance criteria (test plan,...)

- Iteration 4 (11/2019): 2nd module

