OpenIMIS System Architecture & Technical Roadmap

Living Document

Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ) GmbH

Report Cover Page

Version History

Version	Delivered	Feedback
1	2017-04-24	Feedback from Alicia Spengler, Julius Murke
2	2017-05-09	Additional feedback from Uwe Wahser (GIZ/ NHIF Kenya) Kelvin Hui (GIZ Cambodia), Carl Leitner (PATH), Knut Staring (UiO). Most of the feedback referred to the three system options. Several comments (Wahser, Hui, Leitner) hinted at the complexity of option 3 (DHIS2 based system architecture). Picking up these comments, two major streams of thoughts have been included: • Phased implementation approach, module development, proof of concepts • Approach for step-by-step transitioning from MS IMIS Implementation to openIMIS (Based on option 2 or 3) Feedback from Knut Staring on mapping of MS IMIS data objects to DHIS2 configuration objects.
3	2017-06-21	An excerpt of draft version 3 - the section "technical analysis of MS IMIS" - was pre-released to GIZ Nepal and SwissTPH earlier in June, the feedback is included in this version 3. On this basis, some technical inaccuracies were corrected. Some of the comments, which were of a more generic nature, describing and appreciating technical and functional approaches of IMIS were also included directly in the chapter. Other comments were already implicit part of some of the discussions in other chapters of the document. The full SwissTPH comments are available upon request.
4	2018-01	Included July 2017 comments from Swiss TPH. Development of synergistic approach, combining advantages of 3 proposed options.
5	2018-05	Included openIMIS Technical Workshop Comments from February 2018
6	2018-06	Restructured and updated the document. While version 5 was more of a documentation of the discussion of options that lead to the TRM, version 6 aims to focus on the results of the discussion, including the contributions of the technical workshop from 2018-02.
7.1	2018-07-25	- added document references - added classification according to UN-SDG and WHO-DHI - added digital development principles - added chapter on code review sessions - sharpened focus as basis for tender document - added chapter on code sharing - added chapter on mid-term strategy.
7.2	2018-10-01	- proof-reading

Table of Contents

List of Figures

List of Tables

List of Abbreviations

API	Application Program Interface
BMZ	German Federal Ministry Economic Cooperation and Development (Bundesministerium für Wirtschaftliche Zusammenarbeit und Entwicklung)
CHF	Community Health Fund (Tanzania)
CRDM	Collaborative Requirements Development Methodology
CRM	Customer Relationship Management
DHI	Digital Health Intervention
DHIS2	District Health Information System Version 2
DIP	Digital Investment Principles
EHR	Electronic Health Record
EMR	Electronic Medical Record
FHIR	Fast Health Interoperability Resources
GIS	Geographic Information System
GIZ	Gesellschaft für Internationale Zusammenarbeit
HL7	Health Level Seven
HMIS	Health Management Information System
HSC	Health System Challenge
ICT	Information and Communication Technology
ID	Identification
IT	Information Technology
JLN	Joint Learning Network
LMIS	Logistics Management Information System
NHIF	National Health Insurance Fund (Tanzania) National Hospital Insurance Fund (Kenya)
IMIS	Insurance Management Information System
MoH	Ministry of Health
MoU	Memorandum of Understanding
MS IMIS	Microsoft based Insurance Management Information System
OpenHIE	Health Information Exchange
OpenHIMIS	Open Health Insurance Management Information System (transitory naming for openIMIS in 2017)
openIMIS	Open Source Insurance Management Information System
OpenLMIS	Open Logistics Management Information System
OpenMRS	Open Medical Record System
openRBF	Open Result Based Financing
OS	Open Source
PDD	Principles for Digital Development
RfC	Request for Change
SDC	Swiss Agency for Development (Direktion für Entwicklung und Zusammenarbeit DEZA)
SDG	Sustainable Development Goal
Swiss TPH	Swiss Tropical and Public Health Institute

TRM	System Architecture and Technical Roadmap
UHC	Universal Health Coverage
UiO	University of Oslo
UN	United Nations
WHO	World Health Organization
XML	eXtensible Mark-up Language

1 Executive Summary

This “System Architecture and Technical Roadmap” document (TRM) serves to describe the future design of openIMIS, an open source (OS) licensed software package for the management of health insurance organisations. It is based on options that have been translated into a rough development and implementation roadmap. This also includes the possibility to transition the organisations that are currently using Microsoft based predecessor version IMIS (MS IMIS) step-by-step to the new version.

The application should be built based on certain principles, which were agreed upon during the inception workshop in 2016 (GIZ2016Workshop), and have been further detailed since. They include using open source technology to benefit from the advantages of OS communities and to overcome challenges that arise from the use of proprietary technologies. In addition, the principles foresee scalability to support also larger insurance organisations and plan for interoperability, creating a system that is modularized and can communicate with other systems through clearly defined Application Program Interfaces (APIs). Furthermore adhering to data standards should prepare also for future interoperability scenarios, respecting data security aspects. In addition, components of existing standard software/platform solutions should be re-used to benefit from technical, functional and community synergies.

An important aspect is the status of the user requirements. Before making significant software investments, the user requirements need to be well defined to make sure that the software functionality covers the current and future customer needs. The requirements are considered to be sufficiently well documented, stable and predictable to proceed (SwissTPHUserRequiements). The current MS IMIS version contains functionality that has proven to be fitting to the needs of three target countries, largely corresponding to requirements defined by a group of countries in the Joint Learning Network JLN Collaborative Requirements Development Methodology (CRDM) in 2011/2012 (JLN2012).

A key activity for developing the roadmap is a technical and functional analysis of the current MS IMIS. The analysis starts off with an asset inventory, describing the completeness and quality of the technical and functional documentation. In general, the existing documentation is quite thorough and complete, giving partners a technical understanding to adapt the software to their needs. What is missing are hands-on installation and implementation guidelines, an important basis for creating an ecosystem where several partners can assume responsibility in the implementation process.

The technical analysis shows how MS IMIS was originally developed for Tanzania, and then modified to offer further functionality from Cameroun and Nepal. Due to its functional flexibility, MS IMIS is coping well with the country requirements, but it also has some limitations regarding modularity and interoperability. This makes it difficult to expand MS IMIS in a modular way, restraining the possibilities of external partners to contribute new functions. Finally, the technology is tied to the Microsoft ecosystem, requiring a Microsoft operating system, databases and thus generating license costs.

As a response to these limitations, three alternative system architecture options were developed and discussed:

- **Option 1** seeks to stick to the current **MS IMIS,** exploring options how to upgrade it to overcome above discussed limitations.
- **Option 2** discusses **re-writing** entirely the application using open source technology, at the same time introducing a more modular design and using micro-services and possibly also existing library components, for example from Health Level 7 Fast Health Interoperability Resources (HL7 FHIR).

- **Option 3** aims to build a system architecture using existing OS solutions, such as District Health Information System Version 2 (DHIS2) or other components from the open Health Information Exchange openHIE ecosystem.

Above described options were then discussed in the context of their resource needs. One of the general observations is that in any case, a minimum permanent team of 3-5 staff will be required to maintain the software and to support countries, independent of the chosen technical approach. When comparing the three options, some differences become apparent. Continuing with MS IMIS as the only system would have the lowest cost in the beginning, but will possibly lead to rising costs if the limits of the system architecture show. As opposed to that, working in the openHIE context promises important synergies due to the existing communities. It will however require more significant initial investment in order to integrate new functionalities into the existing framework.

Taking into account these options and corresponding resource needs, a phased development approach is suggested.

In a next step or in parallel, technical concepts will be elaborated to verify specific assumptions of the three options:

- For MS IMIS, the concept should outline the technical upgrade path, especially how to create more modularity and interoperability
- For the open source re-write, the concept should clarify how innovative system elements (micro-services, pre-built standard libraries) can be integrated into a development and implementation timeline, overcoming the stability and quality problems that usually come with re-writes
- For the openHIE-integrated system architecture, cooperation approaches with openHIE partners should be explored to maximise technical and budget synergies.

A focus is also to benefit from leveraging synergies between the three options. During the discussions, considerable advantages of all three options were identified. An implementation approach that seeks to **combine advantages of the different options** is to build on a gradual transition from MS IMIS to option 2 and 3, where MS IMIS assumes for a defined period or a specific setting a specified function in a new system architecture. As a consequence, this TRM also describes some of the migration challenges that come with a modular development path, ensuring that both existing and future user organisations always have access to viable system options.

2 Introduction

2.1 Background of the openIMIS Initiative

The Open Source Insurance Management Information System (openIMIS) initiative seeks to provide a comprehensive health information system, linking beneficiary, health service provider and payer data. This system shall be managed by a dedicated software team and continuously improved by an open source (OS) software community. OpenIMIS shall have a customisable design, allowing it to be adapted to the needs of a variety of low- and middle-income countries on their path to universal health coverage (UHC). Through a modular and interoperable build, it shall provide the basis for a seamless integration into existing digital health landscapes. The initiative also seeks to provide capacity development services and foster a community of practice for implementers and users. The OS approach has been chosen over the development of a proprietary solution to benefit from the many advantages of OS communities and to overcome challenges that arise from the use of proprietary technologies.

The technological basis of the initiative is the Microsoft based Insurance Management Information System (MS IMIS), designed by the Swiss Tropical and Public Health Institute (Swiss TPH) and funded by the Swiss Agency for Development and Cooperation (SDC), who is the software license holder. MS IMIS was first implemented in 2012 as the IT backbone for district-based pre-payment schemes in Tanzania. The system was then adopted for a mutual health insurance scheme in Cameroon and since 2015 – supported by the German Development Cooperation – for operating Nepal's national health insurance scheme. MS IMIS has grown organically and has been adapted to different types of health financing systems moving towards UHC.

The current phase of the openIMIS Initiative is jointly funded through SDC and the German Federal Ministry Economic Cooperation and Development (BMZ). As members of the international donor community, both agencies adhere to international standards for development in the health sector. The most important ones are explained in the following chapters:

- the United Nations' (UN) sustainable development goals (SDGs) (chapter OpenIMIS Support to the Sustainable Development Goals2.2)
- the WHO classification of digital health interventions (chapter 2.3)
- the UN principles for digital development and the digital investment principles (chapter 2.4)

2.2 OpenIMIS Support to the Sustainable Development Goals

The openIMIS initiative contributes to the achievement of the following Sustainable Development Goals (SDG)s of the UN (United Nations, 2015):

	Goal	Contribution
1 NO POVERTY	Goal 1: End poverty in all its forms everywhere	Support health financing mechanisms that reduce catastrophic health expenses of beneficiaries.
3 GOOD HEALTH AND WELL-BEING	Goal 3: Ensure healthy lives and promote well-being for all at all ages	Support health financing mechanisms that promote universal access to health care (UHC).
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE	Goal 9: Build resilient infrastructure, promote sustainable industrialization and foster innovation	Strengthen sustainable digital health infrastructures and create employment opportunities for local implementers.
10 REDUCED INEQUALITIES	Goal 10: Reduce inequality within and among countries	Support health financing mechanisms that promote universal access to health care (UHC).
16 PEACE, JUSTICE AND STRONG INSTITUTIONS	Goal 16: Promote just, peaceful and inclusive societies	Increase transparency of health finance services.
17 PARTNERSHIPS FOR THE GOALS	Goal 17: Revitalize the global partnership for sustainable development	Foster international cooperation of digital health interventions through active involvement in global digital health communities.

Table 1: Supported Sustainable Development Goals by openIMIS

2.3 Classification of openIMIS as Digital Health Intervention

In 2018, the World Health Organization (WHO) has released a scheme for the classification of digital health interventions (DHI) to provide a common language according to the addressed health system challenge (HSC), the type of DHI and the ICT system category (World Health Organisation, 2018). According to this scheme, a Health Insurance Management System ...

- ... addresses a broad variety of health system challenges (compare Table 2)
- ... requires services from multiple other system categories and often takes over the roles of these systems (compare Table 3)
- ... supports very many different DHIs (compare Table 4)

Placing openIMIS into the Digital Health ecosystem means that it is destined to play a central role within this system. This calls for a very active communication with other players in that scenery and provides challenges and chances in terms of integration and sharing of resources.

	Health System Challenges
1	**Information**
1.*	All sub-categories
4	**Acceptability**
4.1	Lack of alignment with local norms
6	**Efficiency**

6.1	Inadequate workflow management	
6.2	Lack of or inappropriate referrals	
6.3	Poor planning and coordination	
7	**Cost**	**core challenge**
7.*	All sub-categories	
8	**Accountability**	
8.5	Poor accountability between the levels of the health sector	
8.6	Inadequate understanding of beneficiary populations	

Table 2: Health System Challenges that could be addressed with openIMIS

	ICT System Categories	
E	Clinical terminology and classifications	
G	Data interchange interoperability and accessibility	
M	Health finance and insurance information system	**core role**
P	Identification registries and directories	
V	Public health and disease surveillance system	

Table 3: ICT System Categories that openIMIS could cover

	Digital Health Interventions	
1	**Clients**	
1.7	**Client financial transactions**	
1.7.*	All sub-categories	
2	**Healthcare Providers**	
2.1	**Client identification and registration**	
2.1.*	All sub-categories	
3	**Health System Managers**	
3.1	**Human resource management**	
3.1.2	Monitor performance of healthcare providers	
3.1.3	Manage certification/registration of healthcare providers	
3.3	**Public health event notification**	
3.3.1	Notification of public health events from point of diagnosis	
3.5	**Health financing**	**core intervention**
3.5.*	All sub-categories	

3.7	Facility management
3.7.*	All sub-categories

Table 4: Digital Health Interventions that could be supported by openIMIS

2.4 Global Standards for Digital Development

The increased use of digital technologies for data collection and analysis in the health sectors of low and middle income countries in the last years has resulted in a high degree of fragmentation, duplication, and lack of interoperability of digital health systems in many of those countries. Major international partners in development have recognized the need for greater coordination among donors that fund digital systems and agreed on an ethical framework for the investment in and the development of digital systems.

Currently two frameworks are existing:

- the Principles for Digital Development (PDD) aim at the actual development of digital systems through donors. (United Nations, 2012)
- the Digital Investment Principles (DIP) encompass the PDD plus additional principles that aim at implementing digital systems. (Health Data Collaborative, 2018)

Together with all of the major partner organizations for the development of the health sector, SDC and BMZ have formally endorsed these frameworks or are in the process of endorsing them. As a consequence, these frameworks also apply for the openIMIS Initiative which is funded by the two agencies.

2.5 Aim of the Technical Roadmap Document

This TRM document serves to identify options for the future development of openIMIS. These options will then be translated into a rough development and implementation timeline. This also includes the possibility to transition the countries currently using MS IMIS to the new version.

3 Design Principles

The application should be built based on certain principles, which were agreed upon during the inception workshop in 2016 (GIZ2016Workshop). They and have been further detailed since and aligned to the Principles for Digital Development (compare chapter 2.4). As a result of this synthesis, the future openIMIS technology and system architecture should follow certain principles:

1. It should be built using **OS** technology.
2. It should be **scalable** to support also larger countries.
3. **Interoperability and modularity**: It should have a modular structure and be able to communicate with other systems through clearly defined Application Program Interfaces (APIs).
4. **Data standards**: Adhering to data standards prepares also for future interoperability scenarios, including data security.
5. It should re-use components of **existing standard software**/platform solutions, to benefit from technical, functional and community synergies.
6. It should take into account the ongoing support needs of existing **users** and implementations (Tanzania, Nepal, Cameroun)

In the next sections, the different principles are discussed in more detail.

3.1 Open Source

In the space of global health software, OS approaches have gained considerable traction due to a number of advantages compared to the use of proprietary software:

- efficiency gains of being able to build on other countries' experiences and software developments
- the progress towards international standardization of health systems data
- the avoidance of vendor lock-in
- lower administrative and operational cost
- (technical) support by an international community of practice
- the opportunity of capacity development in the areas of health system management and governance, evidence-based decision-making, and ICT systems for health (including software development).

(SRCOpenSourceAdvantages)

One of their main advantages is that they can be developed and maintained without license fees. This refers to server operating systems, user clients, databases and the application itself. However, although open source can be free, it still has a cost, e.g. human resources to develop and implement the software. Just as with commercial software, it is important to have a concept that foresees resources for the maintenance of all involved pieces of software at the different levels.

3.2 Scalability

Health insurance software applications can be amongst the most resource demanding applications in the global health space. The system ought to be able to respond to different implementation scenarios. On the one hand, it should be able to quickly support simple usage scenarios, where a basic implementation is required, that can later grow to more elaborate usage. On the other hand, openIMIS should also be prepared to support large scale scenarios in the long run, for example a beneficiary group of more than 100 million beneficiaries which may result in more than a billion yearly claimed line items.

3.3 Interoperability and Modularity

National health system architectures in low resource settings are often marked by the existence of many independent systems. Vertical disease programs, such as vaccination, communicable disease surveillance, TB, HIV/Aids, Malaria often have different funding sources leading organisations to set up isolated systems and creating information silos. A similar tendency can be observed for functional approaches, such as logistics, human resources (health worker registries) finance management and social health insurance systems.

For several years, there has been considerable effort to integrate these systems through interoperability. Interoperability builds on existing applications, tying them together through data exchange (which normally requires an active technical maintenance from both sides). In the area of health insurance a number of interfaces are relevant, offering potential for the streamlining of operations, improvement of data quality and gaining analytical insight:

- Receive updated reference or master data from a central repository (Facility IDs, medical services and items, civil registration data, …)
- Receive enrolment data and contribution payment information from a mobile app
- Receive claiming data from claiming apps or medical record systems
- Share approved cumulated claim data to finance management systems for payment
- Share reporting data to National HMIS for integrated reporting

A major approach for interoperability has been defined in the openHIE framework (Open Health Integration, 2016).

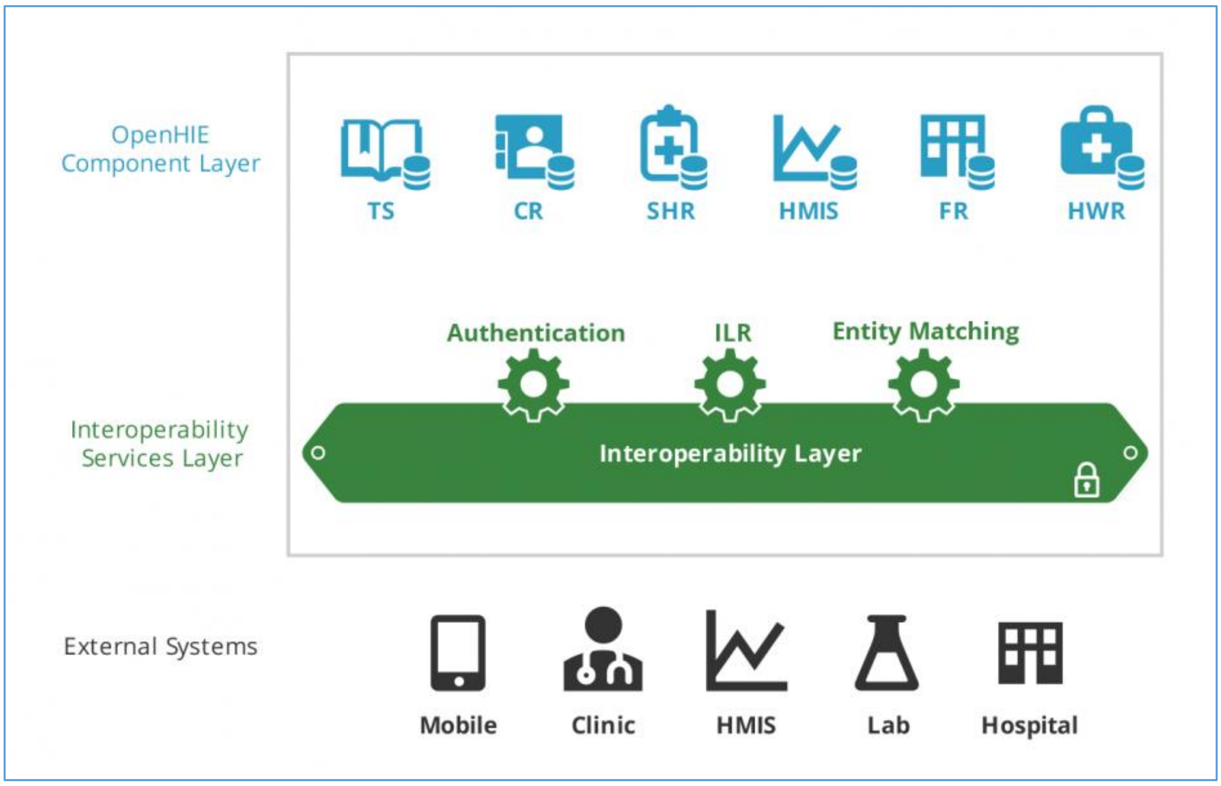


Figure 1: OpenHIE architecture diagram

This approach, which has recently been becoming the main reference for modelling national health system architectures aims to connect the different existing health software solutions into a connected architecture, proposing to standardise interfaces and exchange data through a dedicated system interoperability layers, that allows to exchange data connecting through a dedicated mapping layer. In the current openHIE design, health financing services are not explicitly mentioned. It was due to the intervention of the openIMIS initiative, that a respective workgroup was created where openIMIS now participates in defining the inclusion of health financing systems.

An important aspect when building interoperable approaches is also to have a **modular** software design. This allows to modify components of one system without affecting the other component and provides implementing partners with more options to develop additional functionality.

3.4 Adherence to Standards

Whenever possible, the application should adhere to internationally established data standards. This is also an important basis to be prepared for future interoperability scenarios. An important data interoperability standard is **HL7 FHIR**. Supported by the large Electronic Medical Record (EMR) and medical device makers, it is becoming the quasi standard for storage and interchange of data objects, such as claims or basic patient data.

The application should also adhere to established standards of **data security and data confidentiality**. The general challenges are that the application has to deal with two sensitive elements: Financial data and personal health data. In an interoperability context this is especially important: when data is shared across systems and institutions, clear access rules need to be defined and enforced.

3.5 Build on Existing Software

3.5.1 Customise and Implement Existing Software

By using components of **existing standard software/platform solutions**, costs can be reduced during development and especially during implementation and maintenance. Furthermore, the implementation of existing components can help to connect to existing implementer and practitioner communities, which can accelerate adoption and integration with existing country solutions.

Over the last years, the health domain has seen an impressive growth of standard software solutions or flexible platforms, most of themOS, targeted at low resource settings. These systems have made implementations much more predictable, allowing countries to draw on the experiences of communities of practitioners and the availability of experienced implementers.

Some of the most important solutions in the health space are discussed in Table 5, with a special focus on their relevance for the health insurance sector.

Solution	Description	Relevance for openIMIS
DHIS2	Leading software for HMIS. While it started out managing mainly aggregate data, development focus has recently shifted to DHIS2 Tracker, which allows capturing event or patient related data on a transactional level. DHIS2 is used by Ministries of Health, NGOs and international agencies/donors. It is in use in an estimated >200.000 facilities in openIMIS target countries.	In this document, we will explore the option of building openIMIS in DHIS2 in greater detail. Although there are some challenges in the database structure of DHIS2, the wide usage of DHIS2 and its flexible platform approach make it an interesting option. A crucial asset of DHIS2 is its strong community, which counts hundreds of implementers and support staff in all target countries.
OpenMRS (Open Medical Record System)	Leading open source EMR in target countries. Used in almost 2.000 facilities. Tends to be used only in larger facilities (Specialized / National /	Larger Hospitals that are using openMRS may want to convert their encounter data into claims and transmit them electronically to OpenIMIS.

	Tertiary/University Hospitals), sometimes also in Regional or District Hospitals.	
Bahmni	Hospital management distribution that combines openMRS for medical record keeping, openELIS for lab services and Odoo (OpenERP) for financial management.	OpenIMIS can learn from closely integrating a Finance Management into the system set-up. In addition, the experiences from tightly bundling different solutions into an application package may be an interesting reference. Eventually open-IMIS could also find its place in the Bahmni distribution.
Resource Map	Specialized Health Facility Registry tool. Designed to feed facility master data to other systems. Used in approx. 5 countries. CommCare claims to be open source although the source code is nowhere to be found.	OpenIMIS needs to synchronize facility metadata with a central facility register. In many countries, DHIS2 is the most reliable source of the facility data.
CommCare	Mobile data collection tool, well established in the area of pharmaceutical logistics	Additional (mobile) data entry client
ODK	Mobile data collection tool, can be easily configured based on MS XLS tools.	Additional (mobile) data entry client
RapidPro	Mobile data collection tool, using SMS and USSD	Additional (mobile) data entry client
OpenSRP	Open Smart Register Platform (OpenSRP) is an OS mobile health platform that allows frontline health workers to electronically register and track the health of their entire client population. Its data model is based on openMRS.	Additional (mobile) data entry client.
OpenLMIS	Enterprise-class, open source electronic logistics management information system (LMIS) purpose-built to manage health commodity supply chains	An IMIS can profit from synchronizing master data on medical supplies with an LMIS to generate product catalogues. Potential learnings to be drawn from system architecture and community management activities.
OpenRBF	OpenRBF is a solution to facilitate the daily management of RBF schemes, such as data entry, strategic purchasing, and generation of payment order. OpenRBF is not open source.	OpenRBF started out as stand-alone product and then moved towards a close integration with DHIS2. OpenRBF deals with financial data since RBF schemes are part of Strategic Purchasing.

Table 5: Common digital health packages in low and middle income countries.

There are different ways to leverage existing systems. Figure 2 shows an approach that uses many different existing digital health systems for the purpose of data entry at facility level. This approach can help to reduce the effort of rolling out new software tools at the facility level. The schema also includes a software that serves as a dedicated interoperability layer, as discussed as part of the openHIE concept. To reduce the number of APIs, external modules can send their data to a mapping layer before data is send on to the actual openIMIS database.

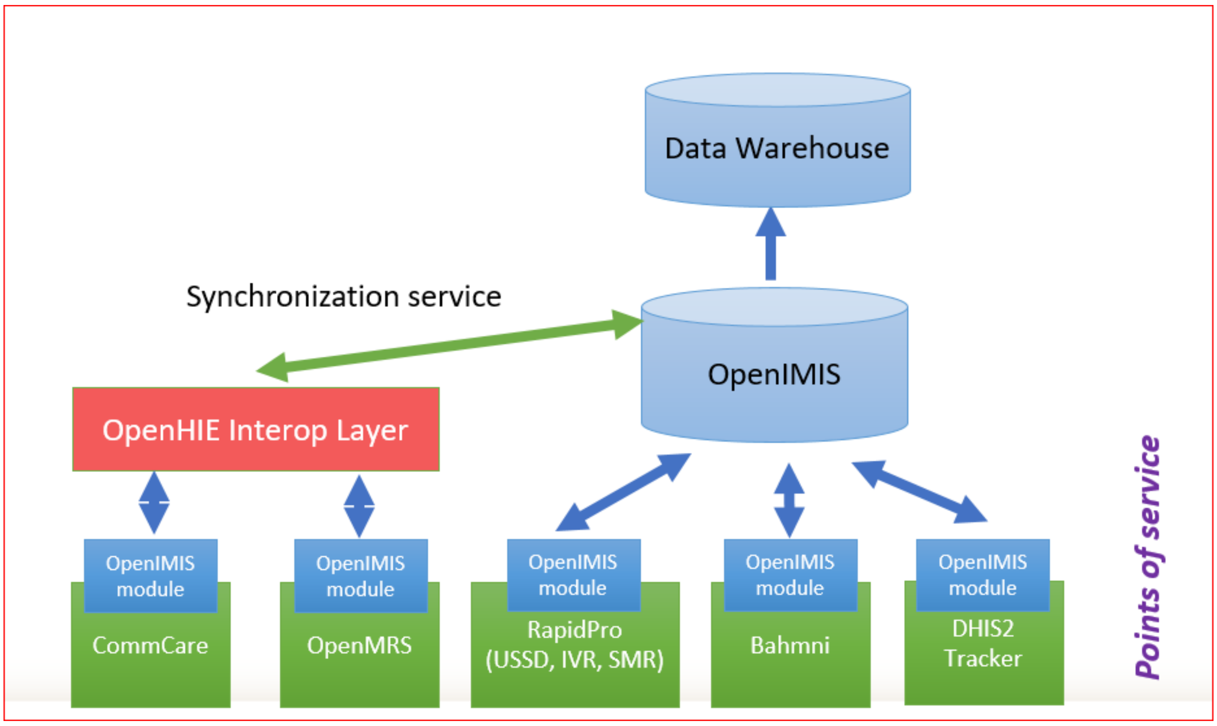


Figure 2: Re-using existing software solutions for facility data entry

3.5.2 Share Programming Code and Libraries with other OS Projects

Chapter 3.5.1 highlights benefits that arise from the use and implementation of existing software packages from the digital health ecosystem. However, the openIMIS development can already profit from other OS developer communities in the sector by re-using existing software code from their repositories and vice versa. This can help kick-start the developments of new functionalities in openIMIS, but can also help to set a few standards within the digital health community as a whole: common functionalities could be developed jointly, enhancing the quality and the user experience in all projects.

At the same time, all communities profit from an exchange of experiences and an extended developers community. Although the number of digital health developers has significantly risen, their total number is globally still quite small. This symbiosis works of course only with projects that use the same programming languages and on the basis of the same programming frameworks.

Table 6 gives an overview of the core programming languages that are being used by the major digital health projects:

Name	Role	Core Programming Language
Bahmni	Integrated Hospital Information System	Java
DHIS2	Analytics, Tracking	Java
iHRIS	Human Ressources	PHP
Odoo	Administration	Python
OpenELIS	Laboratory Services	Java
OpenLMIS	Supply Chain	Java
OpenMRS	Medical Records, Hospital Management	Java
RapidPro	Patient Communication	Python

Table 6: Programming languages used by major digital health OS projects

3.6 Support to Existing Users/ IMIS Implementations

One of the current main assets of the openIMIS program are the existing users in Tanzania, Nepal and Cameroun. In this TRM, we want to identify the ideal and sustainable target architecture for an OS system that can support the development of a strong and stable user base and become the standard in as many target countries as possible. For each of the options, then user transition and data migration scenarios will be discussed, to ensure that the new system represents an attractive option for the existing users.

4 User Requirements

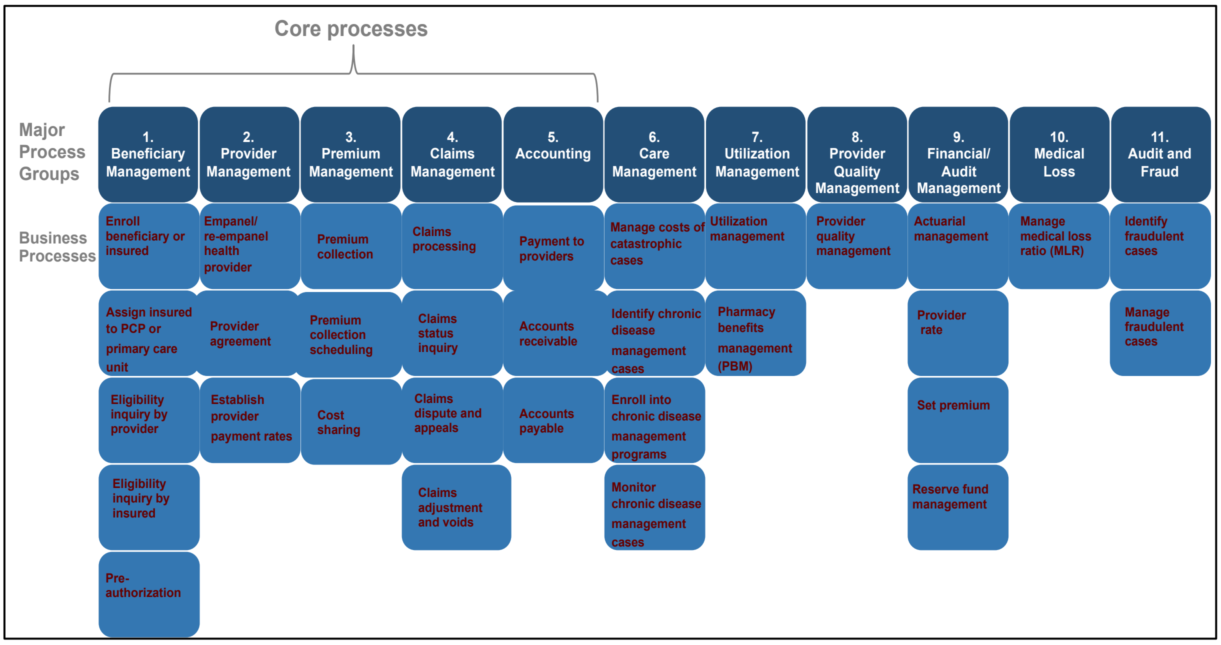
Before designing the technical system architecture, it is necessary to look at the potential functional range the application needs to cover. For this procedure, we rely on already existing synthesized requirements from several projects, including JLN, which have defined generic requirements identified through a multi-country collaborative process and the existing MS IMIS application, in use in three countries. This requirement overview serves to determine the software approach based on the following expectations:

- How can we build software that has a technological base allowing to accommodate all requirements in the long run?
- How can we accommodate the most urgent requirements quickly without losing the possibility to gradually expand the functional range?

4.1 Generic Requirements

The most relevant collection of generic user requirements for an IMIS in low and middle income countries was created by JLN in the years 2010 to 2012. The JLN requirements provide an excellent basis because they summarize the requirements from a range of countries, representing different regions, cultures and socio-economic situations (Ghana, India, Indonesia, Kenya, Malaysia, Mali, Nigeria, the Philippines, Thailand, and Vietnam). It is based on the CRDM. (Joint Learning Network, 2012)

Figure 3 shows a summary of all processes identified as “common” amongst most IMIS in low and middle income countries.



Core processes
Major Process Groups
Business Processes
1. Beneficiary Management
2. Provider Management
3. Premium Management
4. Claims Management
5. Accounting
6. Care Management
7. Utilization Management
8. Provider Quality Management
9. Financial/ Audit Management
10. Medical Loss
11. Audit and Fraud
Enroll beneficiary or insured
Assign insured to PCP or primary care unit
Eligibility inquiry by provider
Eligibility inquiry by insured
Pre-authorization
Empanel/ re-empanel health provider
Provider agreement
Establish provider payment rates
Premium collection
Premium collection scheduling
Cost sharing
Claims processing
Claims status inquiry
Claims dispute and appeals
Claims adjustment and voids
Payment to providers
Accounts receivable
Accounts payable
Manage costs of catastrophic cases
Identify chronic disease management cases
Enroll into chronic disease management programs
Monitor chronic disease management cases
Utilization management
Pharmacy benefits management (PBM)
Provider quality management
Actuarial management
Provider rate
Set premium
Reserve fund management
Manage medical loss ratio (MLR)
Identify fraudulent cases
Manage fraudulent cases

Figure 3: Generic requirements for an IMIS (Joint Learning Network, 2012)

The requirements focus on the following five core processes:

- **Beneficiary Management** includes all beneficiary interactions, including the interactions with the Insurance agency and providers.
- **Provider Management** focuses on interactions between the payer (insurance agency) and the care providers.

- **Premium Management** covers the beneficiary interaction (premium collection) and administrative procedures at the insurance agency.
- **Claims Management** ranges from claims transmission to acceptance/rejection.
- **Accounting** covers ledger systems and payment mechanisms. This functionality is typically handled by a third party software.

The remaining processes are not being discussed in all detail in the JLN requirements documents and range from care management, which relates to care or coverage specific schemes (e.g. catastrophic or chronic disease) to functions such as fraud management.

However, there are also some limitations to using results from this approach for the openIMIS TRM:

- The requirements were compiled in 2011, some recent trends may not have been fully considered.
- The approach aims to help countries build their own requirements based on these general requirements. Following this logic, the requirements can only serve as a first step towards the determination of openIMIS requirements.

4.2 Current MS IMIS Functionality

The functions currently covered in the different MS IMIS versions serve as the major requirement basis. MS IMIS has successfully covered different use cases and its users constitute a quasi-community of practice. Also, it is an important option to offer the current customers the possibility to **migrate** to the future openIMIS version. MS IMIS functionalities will also serve to structure the modular analysis in the next chapter. The overview in Table 7 shows a simplified modular view on the existing MS IMIS modules and the main user roles for each module, “X” representing the main use case, “(X)” a secondary use case.

Module	Facility	Insurance Agent	Insurance HQ
Reference data			X
Enrolment		X	
Registration, Pre-authorization	X		
Claiming	X	(X)	
Operational Agent Functions		X	X
Analytics functions	(X)	(X)	X

Table 7: Modules and actors

There are currently three relevant main different versions: Nepal, Tanzania and a master version. Further development is planned in both countries. Table 8 gives a general overview on main workflows and functions:

Module	Workflow / Functions
Reference data	• Configuration of reference data (Persons/ Families, Facilities, Payment methods, Services and goods, price lists, insurance products) • Definition at HQ level, transfer to mobile apps
Enrolment	• Two-step enrolment: Mobile Android app for pre-enrolment; Full enrolment at Insurance Office. Enrolment on paper forms with pre-

	printed QR-Code, Pre-printed card included. Card does not have a photo. Scan code, take picture, send via app to database • Contribution collection, recording (Mpesa Mobile payment planned for Tanzania)
Facility Enquiry	• Beneficiary goes to Facility. Shows card. Facility uses app to identify via QR code or number (Ideally online, but also possible offline) • Pre-authorization – checks in online DB for eligibility / remaining credit • Access through mobile app or browser
Claims submission	• Encounter details can be recorded on paper form, then entered for claiming into mobile app or browser (by facility or Insurance agent/District Office) • IDs: Facility Code, User code, Patient, Claim Form Number, several diagnosis codes, Serval service codes (drop down list), Emergency/Referral
Head Office Operations	• Claims management (Verification, Approval, Transfer for Payment) • Operator Office (HQ, some functions also at District Offices)
Analytics	• Standard reports (Transaction lists, aggregations) • Available in system directly and in external MS BI tool.

Table 8: Modules and functions (simplified structure based on MS IMIS master documentation)

In addition, the MS IMIS application has an API to a web based SMS service for communication with insurance agents. In addition, customer relationship management (CRM) functions are planned (direct messaging to beneficiary via USSD codes or SMS).

An up-to-date list of planned future functions can be found in the JIRA issue queue, more general information on the strategic direction of the functional roadmap can be found in the wiki:

- **openIMIS issue queue**: https://openimis.atlassian.net/projects/OP/issues/
- **openIMIS wiki**: https://openimis.atlassian.net/wiki/spaces/OP/overview

4.3 Requirements Summary

The discussions above indicate that the existing JLN requirements and the functions build into MS IMIS can serve as a solid foundation for defining the system's architecture and approach of openIMIS.

5 Technical Analysis of MS IMIS

All versions of MS IMIS (country specific versions and the master version) were developed by Swiss TPH in cooperation with the Tanzanian software company Exact. The Nepal version was then modified by the local GIZ team. All implementing organisations (Tanzania, Nepal, Cameroon and Congo-South Kivu as the first French MS IMIS implementation) received a free MS IMIS licence from SDC. Some of the outputs were also the thorough technical and functional documentation. This analysis serves to identify how this investment can best be leveraged for the development of an OS openIMIS.

The value of the investment is not only represented by the software itself. Beyond the software version available for implementation, further assets have been generated, such as:

- Consolidated multi-country requirements that have been transformed into software functionality, which have been tested in different settings and documented
- Data model and analytics structure, technical documentation and test cases
- Implementation experiences: Time plans, resource needs and evaluation reports for implementation and maintenance.

5.1 System Documentation

A good documentation is an important basis for system deployments in different contexts. We are currently aware of the following documents:

Type	Documents	Version
Specification	In 2012, software specifications were documented, including use cases. These specifications represented the original expectations towards the software. The documents have been updated for each country. Requests for change are referenced to these specifications. • **2012-07 IMIS_Specification_Phase_1_v4.doc** • **2012-07 IMIS_Specification_Phase_2_v2.doc** A separate specification was created for the analytics and reporting system: • **2016-03 HPSS_IMIS_Specification_Analytics_v1.6.doc** The original document naming divides the software specification into phases (which correspond to modules).	2012
Specification	In 2017, a specification document of the consolidated master version was created: • **HPSS IMIS Specification Master version_3. 3.5.pdf** The document covers all aspects of the master version, including detailed overviews and data fields for the three country versions. In section 4.4 “Additional functionality” the requests for change are lists that were the basis for the integration of country functions into the master version.	2017-01
Technical documentation	The main technical documentation consists of a description of the system architecture, programming approach, standards and methods, the menu structure and interface flow, functional area descriptions and database design. Furthermore, web services and the mobile phone apps are described. • **Technical Document 24042017.pdf**	24 April 2017 Version 17.3.11
Missing documentation	• Release history: Including all detailed requests for changes (RfC**)** • Compilation & installation guide: Including Server & DB Setup • Test cases • Implementation experiences: Time plans, resource needs and evaluation reports for implementation and maintenance. This material does not exist in an up-to-date version or has not been made available for analysis.	

Table 9: Documentation

As of beginning of 2018, this documentation is being transferred step-by-step to publically available documentation repositories such as http://openimis.readthedocs.io/. Up-to-date references to the documentation can be found on https://openimis.org .

5.2 Technical Analysis from July 2017

The technical analysis serves to describe the current system and to discuss aspects that help to evaluate how sustainable the current system approach is. The analysis is based on system documentation, additional information from SwissTPH and feedback from GIZ Nepal. (Swiss TPH, 2017)

Item	Description of IMIS features	Comments on sustainability of approach
1. Programming languages	The programming language is VB.NET (Microsoft Visual Studio 2012). VB and C# are the two major programming languages used in MS.NET. Since 2017, Microsoft seems to prioritise C# over VB.Net, raising fears that support for VB.NET will be eventually phased out.[1]	In many developing countries Microsoft is still the most widely used developing environment. Especially VB is widely used amongst developers.
2. DB Version, OS Version	• Mobile app: Android 2.3 • Minimum: MS Server 2008, MS SQL Server 2008 MS SQL Server 2014 is required for running parts of Analytics and Reporting tool.	
3. Licenses	The application can only run on a Microsoft (Server) Operating System, which requires a license. For the database, there is the choice between the MS SQL Server Express or the full version. Countries can start with the free Express version, once usage grows it is useful to switch to the commercial version (As Nepal and Tanzania have done). The approximate licences cost is around 4000 EUR.	License cost is low taking into account the scope of most projects. Small implementations can be realized license free
4. Performance / Scalability	The current countries have relatively small member basis and claiming numbers compared to other countries.[2] Tanzania has witnessed no performance issues (with exception of running long extracts for off-line installations). Nepal witnessed challenges while the full ICD code list (ca. 12.000) was loaded as list of values in 4 fields. The problem disappeared with the reduction of codes.	In general, the application seems fit for volume scaling. Some performance tuning may be useful in association with national roll-outs in larger countries.
5. Extensibility and parametrization options	IMIS allows to parametrize options, such as enrolment and claiming workflows, configuration of insurees, products, prices and policies. Additional parameters can be changed at the database level. The business logic and database layer of all country specific versions are identical, differing only at the presentation layer by well-defined features (labels of data fields, drop down lists for data fields, the second language of user interface, displaying of specific data fields, mandatory or optional entry of data into specific fields). Super users can modify data entry screens by excluding a field, but not by adding.	IMIS covers a wide range of the domain specific functionality that should respond to many future requirements. In case a country wants to include unforeseen functions, it should normally request the centralized development team to make the changes and wait for the changes to be released (to the country or as part of the general master). Countries attempting to change functionality themselves need to create a separate version and have upgrade problems.

[1] https://en.wikipedia.org/wiki/Comparison_of_C_Sharp_and_Visual_Basic_.NET#Language_history - "The primary reason for the decision is that with time VB has become less popular with professional developers relative to C#."

[2] NHIF Kenya, 2016:
- 6 Mio Members, Monthly 60.000 new members (10 Mio Dependants, 200.000 new monthly)
- Outpatient visits > 2 Mio, monthly 110.000, 5 Mio Impatient admissions (60.000 per month)
- > 17 Mio Contribution records, 300.000 monthly
- Individual accounts upload: > 140 Mio, 3 Mio monthly

Philippines (2015): 93 Mio Beneficiaries (40 Mio Members, 53 Mio Dependents)

	The core is maintained centrally and development of diverging country versions is not desired. Functional enhancement based on core functionality should be built into core after approving requirements and testing compatibility with existing functions. This approach is chosen to control the complex transactional business logic and data integrity. The Master Version has some predefined exit points for incorporating the so-called escape procedures for modifying business logic of IMIS, e.g. verification of insurance numbers, communication with SMS portals and mail servers. The scope of exit points can be extended. In addition in Tanzania the development of an API is underway for the enrolment data collection process	The possibilities for countries to add functions to the core functionality by themselves, e.g. through additional modules or APIs are thus limited.
6. Internationalization	Internationalization features allow a system to adapt to local requirements. • IMIS allows to maintain two languages via resource sheets • IMIS currently does not have a repository of all national calendars, in order to support Nepali date of Birth, a workaround at the presentation layer is in place (also due to budget reasons). • Typical requirements additional for internationalization are currencies, numeral systems, time zones, regulatory requirements, name structures, national IDs, ZIP codes, …[3]	IMIS was first developed for a project in Tanzania and later adapted to different national settings. It relies on standard internationalization features of Microsoft tools, which cover a wide range of typical requirements. An analysis of SwissTPH shows that IMIS should be prepared to handle most future requirements regarding currencies, time zones or languages.
7. Data import and interoperability	Claims and enrolment data can be imported via eXtensible Mark-up Language (XML) files. This method is used by the Android apps to push data to the MS file server. For inserting large administrative data lists (e. g. territorial/administrative units, medical items/services) SQL scripts are used. IMIS can export analytics data to a DWH via an ETL. Also payment data can be exported to accounting system to initiate payment processing. In Nepal this is done through custom made SQL report.	The current IMIS master version does not provide a far-reaching API, which would allow other systems to submit or receive all data after authentication (e.g. like DHIS2 does via a RESTful Web API, where practically all data is covered, including metadata such as facility lists). API development has been initiated, there is now a stateless API under development (Tanzania).
8. Database abstraction	The system architecture has a special database tier. This layer is used to connect application and database.	As of our understanding, it is currently not possible to connect open source databases (because of MS restrictions), leaving as only (hypothetical option) to replace the MS database with an Oracle database.
9. Security concept	IMIS has usual security functions (user roles, auditable log files, historical records, session expiry after a defined time with automatic logout,)	Microsoft applications and databases normally can provide a high level of security. Database vulnerabilities are quickly fixed. The data security will depend on how the application is implemented.
10. Browser access	The user interfaces of the main application are well displayed on typical PC or laptop screens (e.g. Width of 1024). The screen design is not mobile responsive in the sense that it does not adapt its layout to smaller screens (tablets, mobile phones). Mobile access takes place via Android apps. The screens have a typical professional Microsoft look. Countries can adapt the interface, by omitting fields that should not be displayed.	Mobile responsiveness will become more important since the diversity of user devices will continue to increase, especially in different settings (facilities size, use case, countries, etc.).

Table 10: Elements of the technical analysis from July 2017

[3] https://en.wikipedia.org/wiki/Internationalization_and_localization

5.3 Code Review Sessions from June/July 2018

In June / July 2018 a more profound technical review of the software architecture and code base of the openIMIS package was done by a group of technical experts from GIZ, SwissTPH and independent consultants to support the strategic and technical roadmaps for openIMIS. Several influencing aspects were analysed for the whole application (crosscutting) or according to functional areas of the application. The results of the technical review are available in a living on-line document for further reading. (openIMIS Initiative Product Team, 2018)

5.3.1 Review of Functional Areas / Modules

The reviewed functional areas, which could form a basis for a future modularisation, are defined as Master Data Management, Insurance Product Definition, Insuree Management, Health Facility Registration, Claiming, Client Feedback, and Analytics. The aspects for the analysis of each area included:

- the adaptability of the application to user defined business processes,
- the adaptability of the application to user defined data needs
- the adaptability of the application to local settings such as the support for local languages.
- the complexity of modularisation, if one functional area was to be isolated as an independent module.

While some functional areas do offer a good amount of flexibility within existing parameters (for example product definitions), the implementation of user defined business processes or additional data needs require the interventions of developers in all functional areas. Two languages are currently supported but further development work is needed to fully support internationalization. While there are isolated aspects in the current architecture (mobile apps, analytics), significant amount of work is needed to reveal the dependencies between different functional areas and modularise the application. Table 11 gives a condensed overview of the analysis results, the hyperlinks lead to a detailed discussion in the online-document.

Functional Area	Business Processes.	Data Needs	Localisation	Modularity.
Score: 1 (nothing to be done) - 4 (complete overhaul needed)				
Master Data	4	4	3	3
Insurance Product	3	4	3	3
Insurees	4	4	3	3
Registration	4	4	3	3
Claiming	4	4	3	3
Feedback	4	4	3	2
Analytics	4	4	3	2

Table 11: Code review results of openIMIS modules

5.3.2 Review of Cross-Cutting Aspects

The analysis of cross-cutting aspects related to security, authentication but also to the standards expected from OS applications (technology stack, openness to collaboration, ease of installation, development and code quality). There is no urgent need for an immediate “big-bang” rewrite to move away from the currently used .NET Framework, but there are severe security issues that need to be urgently addressed in the current code base. There are also issues in the public code repository, which can impact the perception of openIMIS by new developers, which includes a clean code-base split into well-defined boundaries, regular releases with documented changelogs, improvements to the easiness of setting up a development environment and packaging and distribution of new versions of the application.

5.3.3 Summary of the Code Review Sessions

In summary, the results from the code review sessions support the overall strategy for the further development of openIMIS. The review has helped identify the requirements for modularization of the existing system without changing the entire code base, which in turn will allow for gradually introducing functional modules based on a technology stack more in-line with the openHIE community. Additionally there needs to be changes to foster the collaborative development expected by OS solutions. The seamless and successive integration of new modules based on a different technology stack and their distribution will require immediate action in the current code base. As a whole these changes will open a path towards a more maintainable and modular system.

5.4 Country Feedback from Nepal

In May 2017, the GIZ consultants visited the GIZ team in Nepal to get a better view of the state of the MS IMIS implementation in Nepal. The general feedback was that GIZ Nepal is satisfied with the overall functioning of the application. However, there are also some areas where GIZ Nepal feels that the application does not adequately cover all needs.

GIZ Nepal has been modifying the application locally, which makes it more difficult to upgrade to the next version. GIZ Nepal has not yet fully tested or implemented the master version.

GIZ Nepal has made some enhancements to the application that might also be interesting for other countries. For example, on the start page a dashboard built on Google Graphs has been integrated, built on a web API.[4] With the current system architecture, such enhancements are more difficult to share between countries and difficult to migrate in case of version updates.

5.5 Conclusion

Over the last years, MS IMIS has developed into a flexible health insurance solution that is ready for deployment and should be able to accommodate most immediate needs of many potential users. However, MS IMIS also has some limitations regarding parametrization, modularity and interoperability. The use of MS .NET programming framework is quite unusual in the digital health ecosystem, adding more complexity to an integration into a common platform / distribution such as Bahmni. Therefore, it is useful at this stage to evaluate the continuation of MS IMIS development against other options.

[4] It should be noted that it is also possible (and maybe more convenient) to use an external app such as DHIS2 to build such dashboards.

6 Main System Architecture Strategies

6.1 Overview of Three Different Model Options

As part of this analysis, three major approaches for the further advancement of openIMIS have emerged, which are roughly outlined in Table 12:

No.	Approach	Main points
1	OS the existing MS IMIS master and continue its development	• In the short term, it is the most attractive option to build on this working system. • Yet it needs to be determined whether the long-term technical constraints can be overcome. • In any case a transition strategy for existing MS IMIS users' needs to be defined.
2	Transform MS IMIS into openIMIS, using open source code and tools	• Starting an application from scratch allows to define and apply best practices approaches for modularity, interoperability and usage of existing standards • Allows for a better exchange of code and knowledge with other OS projects in the openHIE arena. • As with any entirely new application, the starting phase will require a lot of testing and error corrections, especially when trying to apply innovative concepts.
3	Build on existing platforms	• OpenHIE offers a growing ecosystem of interrelated components • DHIS2 offers a well-supported software platform with a strong community • Connecting existing systems may result in a technically more demanding solution and will require a clear approach to manage external dependencies

Table 12: Principal software architecture and technology options

These options have to be understood as "pure", simplified model strategies to elaborate the underlying ideas. For reasons of clarity, these ideas are discussed in chapters 6.2 to 6.4. An intensive discussion process was moderated among the major stakeholders in 2017 and 2018, including workshops and in depth email discussions. The synthesis of the discussion process led to an overall strategy for the further development of openIMIS, which is outlined in chapter 7 "Way forward".

6.2 Option 1: Continue with MS IMIS Development

The first option suggests to actively continue the development of the MS IMIS, maintaining the Microsoft based approach. The obvious advantage of this approach is to build on an existing, stable software, which would make development activities and required resources more predictable. Although VB.Net is seeing a slight degrade against C# as a programming language, it would still be a solid basis for further development for the next years.

Figure 4 outlines the current system architecture, which would remain unchanged in its structure, only adding additional existing applications, such as DHIS2 as data warehouse and potentially other data entry applications.

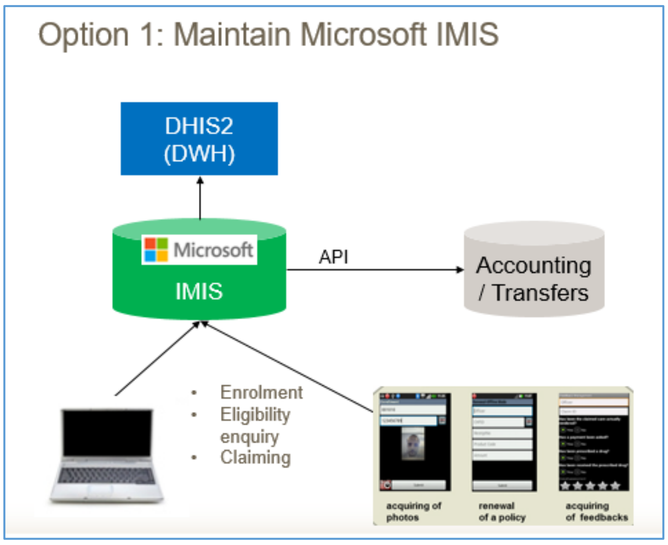


Figure 4: Option 1: Maintain MS IMIS

However, as shown in the technical analysis, there are also some conceptual challenges to transform MS IMIS into a more sustainable solution. The development could focus on areas such as interoperability, implementing standard methods such as RESTful/ASP.NET Web API to enlarge the data exchange options. It would also be useful to allow for code contributions to core functions via apps or visualization of external apps in browser. Also, scalability would have to be tested to make sure the application is fit for larger countries with bigger member groups and higher claiming volume.

In most scenarios, the licensing cost can be contained to around 5.000 Euro per country, which in most scenarios only represents a negligible fraction of required overall investment. More important regarding costs is that the initial implementation efforts seem to have been quite high for some of the countries. As of now, detailed implementation effort information from projects has not been collected. However, some observations point toward these high efforts. It is not clear which part of this effort is due to the earlier immaturities of the software, the country-specific deployment situation and country-specific requirements. Still, as part of the next steps, one activity could be to systematically evaluate implementation experiences and develop technical and organisational approaches to make country customization and implementation smoother and less costly.

6.3 Option 2: Rewrite openIMIS

The second approach would rely on the functionality of MS IMIS but transfer it into an OS environment, basically replacing the code basis of the core module of MS IMIS under the Microsoft .NET framework with core modules in a pure open source framework as shown in Figure 5. This is a quite predictable and low-risk technical effort. The application code and database scheme can be converted or rewritten from scratch respecting functionality, database structure and business rules.

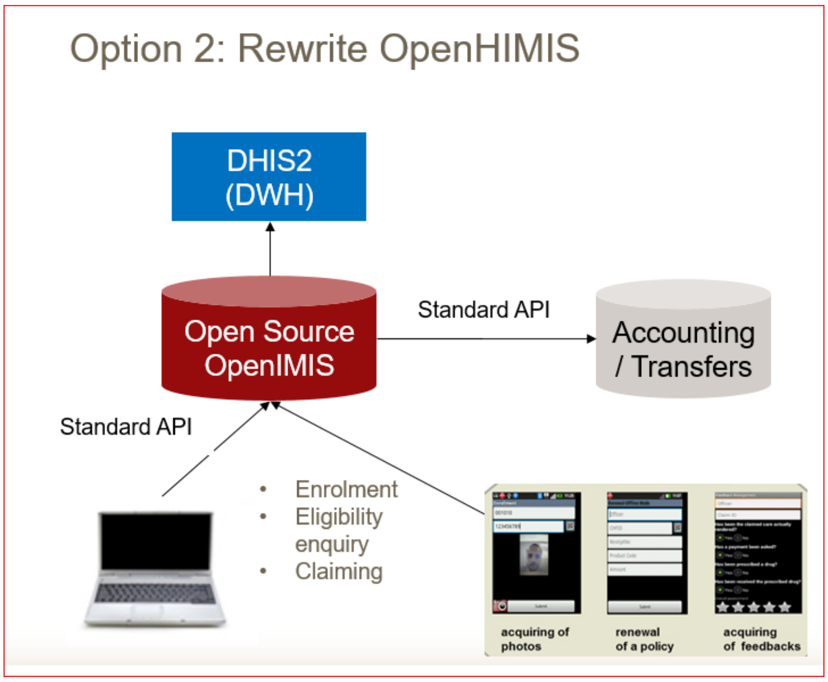


Figure 5: Option 2: Rewrite MS IMIS using OS technology

When the rewrite is aligned with programming frameworks and programming languages that are being used in the context of the openHIE digital health ecosystem, openIMIS developers can profit from experiences and existing code samples of the other OS projects in that arena and vice versa. This would help to bundle scarce developer resources and to standardise shared functionalities. According to the overview in Table 6 "Programming languages used by major digital health OS projects", the vast majority of the other projects uses JAVA. It is therefore reasonable to target a migration into the JAVA framework, while at the same time ensuring compatibility to Microsoft and Linux platforms.

However, in order to have a lasting impact and justify the investment, the rewrite should go beyond simply reproducing the current system. Some of the conceptual challenges to be tackled are to produce a software that fully corresponds to the concepts of an international standard software, allowing for a full parametrization of functionalities, including all typical language, calendar, currency requirements. In addition, standard interoperability approaches would be integrated, while modularizing the design.

A modular architecture would allow to apply internationally established data standards, such as HL7 FHIR or frameworks such as openHIE. HL7 already provides well defined data models, libraries and APIs for some important data objects, such as basic patient data (client demographics), health facility data (clinical and administrative), patient records, claims, invoices, etc.

Enhanced use of standard APIs would in turn facilitate building a **micro-service** oriented architecture, where each of these services have their own database structure, offering the possibility to develop the micro-services more independently from each other. This independency can enable development partners or countries to develop different versions or functional enhancements for specific use cases, without interfering with the rest of the core software. It also supports a slow transition process into OS development frameworks, successively migrating one service after the other without affecting the end-users.

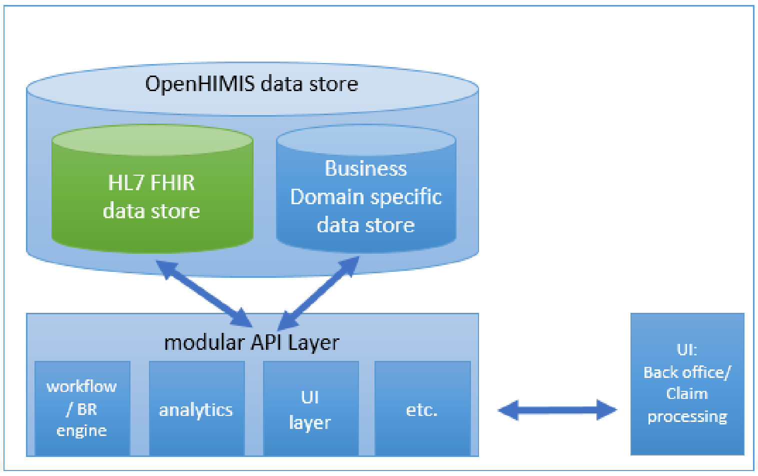


Figure 6: Micro-service based implementation of HL7 FHIR data model

The HL7 approach would allow to be compatible with many other health information initiatives, including many EMR solutions and solutions of private health insurance providers that openIMIS may have to interact with. However, this would also require further exploration. While HL7 is gaining importance in the sphere of large commercial EMR and medical device makers, it needs to be evaluated whether it is mature enough to be used as a central concept of openIMIS in the context of low-resource countries.

The initial investment would certainly be higher than with the MS IMIS solution, and getting the solution to the stage of maturity and stability where MS IMIS is now, will probably take 1-2 years. The more innovation is applied as part of this process, the higher the potential rewards, but also the uncertainties regarding cost, time and software quality.

6.4 Option 3: Customize openIMIS into Existing Platforms

A fundamentally different approach would be to build on the basis of an existing framework or platform. As discussed in the section on software principles, in the health software space there are several widely known solutions that are fit for many related use case scenarios in low resource settings. **OpenMRS** is a medical record system that is mainly used at facility level, but also supports some national scale programs. **DHIS2** started out as a national statistics system for aggregate health service data (HMIS). Recently it has gained considerable traction and functionality related to patient tracking, gradually integrating the basis for transactional approaches.

Both systems offer common advantages over a software development that starts with a proper codebase, both in the area of community access and technical synergies. DHIS2 for example offers a strong Data Warehouse, flexible organisational hierarchies as well as a framework approach. It already plays an important role for many centralized government agencies. Furthermore, it has a well-established training and support structure that has allowed it to act as an anchor point for many national health information architectures.

For DHIS2, some explorative architecture conceptualization was carried out, as shown in Figure 7.

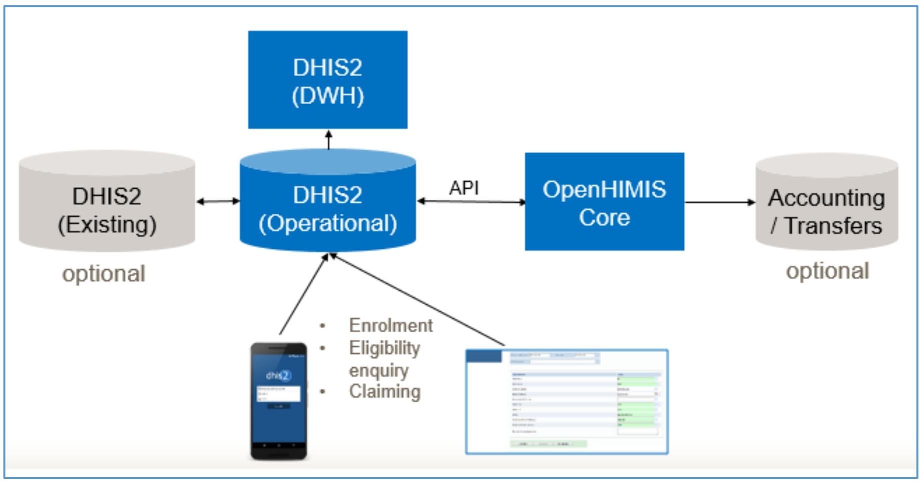


Figure 7: Option 3: OpenIMIS based on DHIS2

7 Way forward

7.1 Synthesis of the Strategy Options

Discussions between GIZ, SDC and Swiss TPH in July and August 2017 pointed towards a step-wise integration of modularity, interoperability and OS components. During the technical workshop in February 2018, these elements were discussed and agreed upon by a wider team of stakeholders.

None of the three approaches was selected in a pure form. Option 2 was seen as a clean and desirable approach, but with an eye on limited resources in terms of time and finance this option was not considered to be feasible, especially when keeping in mind that the current implementations and users need ongoing support.

While option 3 was initially discussed as a cheap alternative, it was eventually discarded because it appeared to be difficult to align the openIMIS objectives with the development path of the DHIS2 core. Therefore, discussions focused more on a closer cooperation with partners of the openHIE framework. These modules cannot replace the current openIMIS, but they can lower the development burden for specific functions that are already available in existing modules of the openHIE ecosphere.

Instead, a consensus was reached concerning the combination of the three approaches:

- the MS IMIS core will still be supported for another 2 - 3 years (option 1), while new functionality is subject to migration, if applicable.
- a gradual, step-wise transformation "from the outside in" towards a modular and inter-operable platform based on OS components (option 2) will be target.
- a tight integration into the existing Digital Health ecosystem especially in the context of the openHIE community is desired as a result of the discussion on option 3. This falls in line with Digital Investment Principles (see chapter 2.4 "Global Standards for Digital Development")

A short-term strategy was defined as a result of the technical workshop in February 2018 in the context of the current SDC/BMZ funding package (compare chapter 7.2 "Short-Term Strategy: Maintain Current MS IMIS Core")

In the further course of 2018, additional funds were generated through BMZ and supplemented through the Digital Square. These additional funds give room for initial steps towards a slow migration of the MS IMIS core to a full OS platform (compare chapter 7.3 "Mid-Term Strategy: Slow Migration to Open Source Frameworks").

7.2 Short-Term Strategy: Maintain Current MS IMIS Core

7.2.1 Short-Term Target Architecture

During the technical workshop in February 2018, an approach was discussed and agreed upon, that aims to combine elements of the three options discussed into a short-term target architecture, as shown in Figure 8:

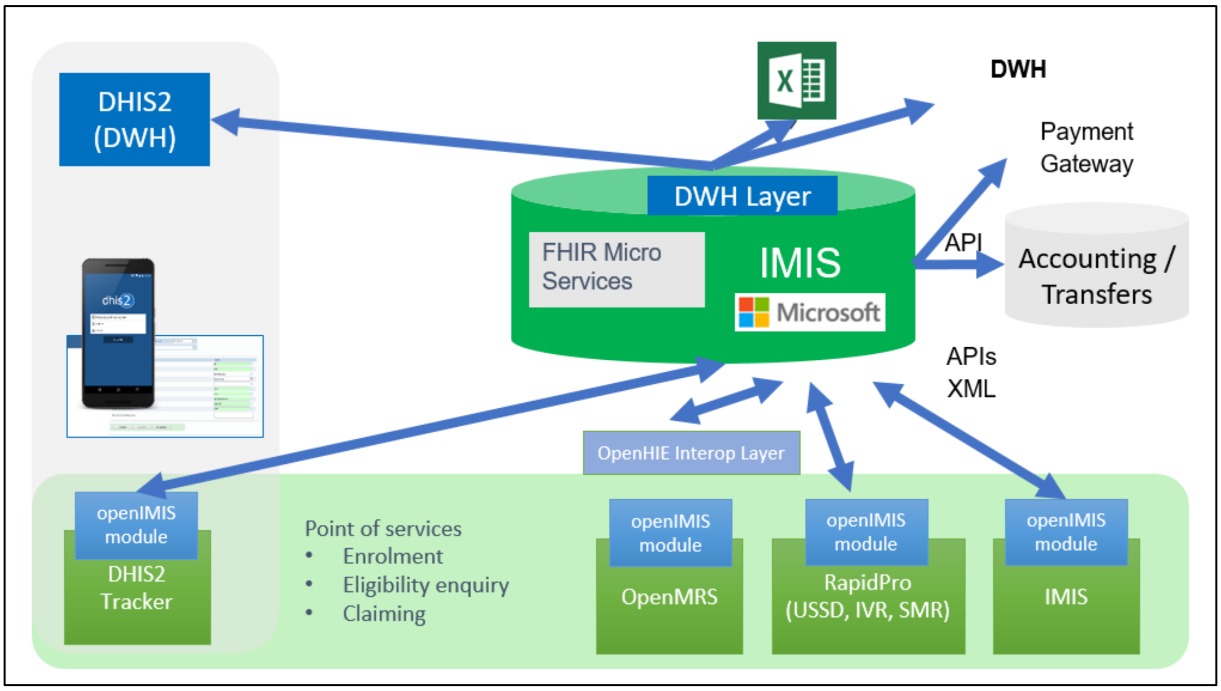


Figure 8: Synergistic system approach (Step 1)

The major elements from the three different options are included in this approach:

- **Option 1:** For the time being, the current MS VBA code based application will be maintained. However, investment into VBA code will be limited. A special focus will be set on quickly developing further interoperability features of openIMIS, opening up the legacy system to modular enhancements. This is based on the assumption that the current MS (?) IMIS core will not need to be replaced in the coming 2-3 years. To confirm this assumption, some more analysis would be needed to be certain that the core holds up to the expectations regarding interoperability, modularity and accessibility to an open source community.
- **Option 2:** For all major functional updates, it will be verified whether this can be done based on OS technology or existing HL7 standards or modules. A pre-condition of that is to modularize MS (?) IMIS core components step-by-step. A complete replacement of the current MS IMIS core will not be aimed for during the next two years but rather at a later stage.
- **Option 3:** Integration with elements of the openHIE architecture will offer important opportunities for re-using existing technology, enable strategic partnering with other software communities, and achieving more sustainable funding. We believe that modular technology allows to pursue these synergies, at the same time respecting data security, privacy aspects and other organisational boundaries.

A major aspect of the **investment case** is to preserve the current customer base. We believe we should be offering options to upgrade the existing systems and to continue strengthening it by integrating it further with the national **health system architecture**. We believe that actively integrating openIMIS with established OS systems can broaden the future customer base. We also acknowledge that the **current MS IMIS core functionality** (especially the management of policies, beneficiary contributions and transactional processing of claims) represents one of the major assets of MS IMIS and can also be part (?) of a future openIMIS.

7.2.2 Short Term Technical Action Points

An important component of the discussions during the workshop in February 2018 were technical suggestions made by SwissTPH. They are summarized in Table 13, complemented by some of the major comments given by GIZ.

Summarized suggestion by SwissTPH	Comments by openIMIS Initiative
1. OpenIMIS should be developed based on the current MS IMIS, using MS VBA.	Agreed in the sense that for the time being all existing core functions should remain in Microsoft Technology. For new functions it should be analysed case-by-case whether new elements can be used (micro services, open source, existing libraries), thereby stepwise modularizing the core.
2. MS IMIS will be modified in such a way that it will support besides MS SQL Server **database** also other databases (e.g. open source or license free database). This will also allow running the data base server on Linux OS and gives partners technology choices. Migration tools will be prepared.	Agreed, based on specific customer demands. *(This has a lower priority)*
3. The data warehouse of MS IMIS will be integrated with alternate front–end besides the currently used front-end MS Excel. Priority would be DHIS2, but it should also be possible for other reporting/DWH/BI products.	Agreed. Pro-actively pursuing DHIS2 integration corresponds to requests of different partners. Still, vendor-independent approach is important.
4. The current **RESTful API** layer will be completed (and made fully compatible with the relevant standards) in order to facilitate full replacement of web form functionality in case it is needed for ensuring online communication with adjacent software systems.	Agreed. It has been a new information for us, that MS IMIS uses RESTful API.
5. A **data format** in conformance with HL7 FHIR will be provided for data exchange on claims as an alternative to the current native XML based format.	Agreed. In a later step, it can be analysed how micro-service based FHIR libraries could also contribute to long-term core modularization.
6. Data formats based on CSV and/or Excel will be offered for initial loading of selected **registers** (it is already available but only for some registers and it is not published yet).	Agreed. This will speed up implementation and help transfer maintenance responsibility to countries. Also continuous data exchange with specialized registers, such as facility lists, can be explored in line with openHIE approaches.
7. The existing mechanism of **exit points and escape procedures** will be enriched to allow country specific modification of more aspects of the business logic in case an explicit configuration on the user level is not advantageous for a specific case.	Agreed. This seems like a promising approach to allow countries to inject further functionality. Since this can deeply affect core stability, the procedures should be well structured and documented. *(This has a lower priority)*

Table 13: Technical suggestions for immediate action.

The code review sessions from June 2018 unearthed further needs for immediate action regarding data security. In addition, the whole area of packaging and distributing the application in current and future versions needs to be addressed urgently (build & release system). This affects the current repositories and documentation.

In order to enhance the maintainability of the code and to enable future migration steps in the mid-term strategy, the step-wise refactoring of the code into a modular structure based on functional areas has to be started.

7.3 Mid-Term Strategy: Slow Migration to Open Source Frameworks

As an overall mid-term strategy the term migration "from the outside in" was coined in the discussions. This means that the application is being prepared for a slow migration from the Microsoft environment into something that can also run in a pure OS environment as it is the basis for most of the software packages in the openHIE ecosystem.

The application shall be migrated successively from the Microsoft based development framework .NET and the Microsoft server platforms (data-base and application server) into an OS framework with a platform independent architecture. To enable a tight integration into the openHIE community, the JAVA development framework seems to be the most appropriate, as it is being used by almost all of the other communities. However, other options are being discussed.

A slow migration "from the outside in" would start adding new functionality to the overall package on the basis of the new framework and then slowly add updated core functionality into the new framework. This needs to be done in a way that the new services can run on the same server in parallel to the old core, but already be integrated into the overall user-interfaces. The user will not see a change, but from release to release, the software package will have a higher number of services on the new framework.

A decision on the needed architecture has not yet been made. However, a micro-service architecture seems to be most promising for a smooth transition.

8 References

(n.d.).

(n.d.).

Health Data Collaborative. (2018, 05). *Digital Investment Principles: Donor Alignment Principles in Digital Health*. Retrieved 07 22, 2018, from Digital Investment Principles: http://digitalinvestmentprinciples.org/

Joint Learning Network. (2012).

Open Health Integration. (2016, 04). *OpenHIE Architecture Diagram*. Retrieved 06 25, 2018, from OpenHIE: https://ohie.org/wp-content/uploads/2016/04/201603-Architecture-Diagram-940x600.png

openIMIS Initiative Product Team. (2018). *openIMIS Code Review Meeting.* openIMIS Initiative, Product Team. Bonn: openIMIS Initiative. Retrieved 07 25, 2018, from https://docs.google.com/document/d/1Ui9G7RsZRX2NNmFVJJYYE7Wn0wqSdCbw10An05LjXl4/edit?usp=sharing

Swiss TPH. (2017). *Technical Evaluation of OpenIMIS (22.07.2017).*

United Nations. (2012). *Principles | Principles for Digital Development*. Retrieved 06 25, 2018, from Principles for Digital Development: https://digitalprinciples.org/principles/

United Nations. (2015, 09 25). *Sustainable Development Goals: 17 Goals to Transform Our World*. Retrieved 06 25, 2018, from Welcome to the United Nations: https://www.un.org/sustainabledevelopment/

World Health Organisation. (2018, 05 01). *WHO | Classification of digital health interventions v1.0.* Retrieved 06 25, 2018, from WHO | World Health Organization: http://apps.who.int/iris/bitstream/10665/260480/1/WHO-RHR-18.06-eng.pdf