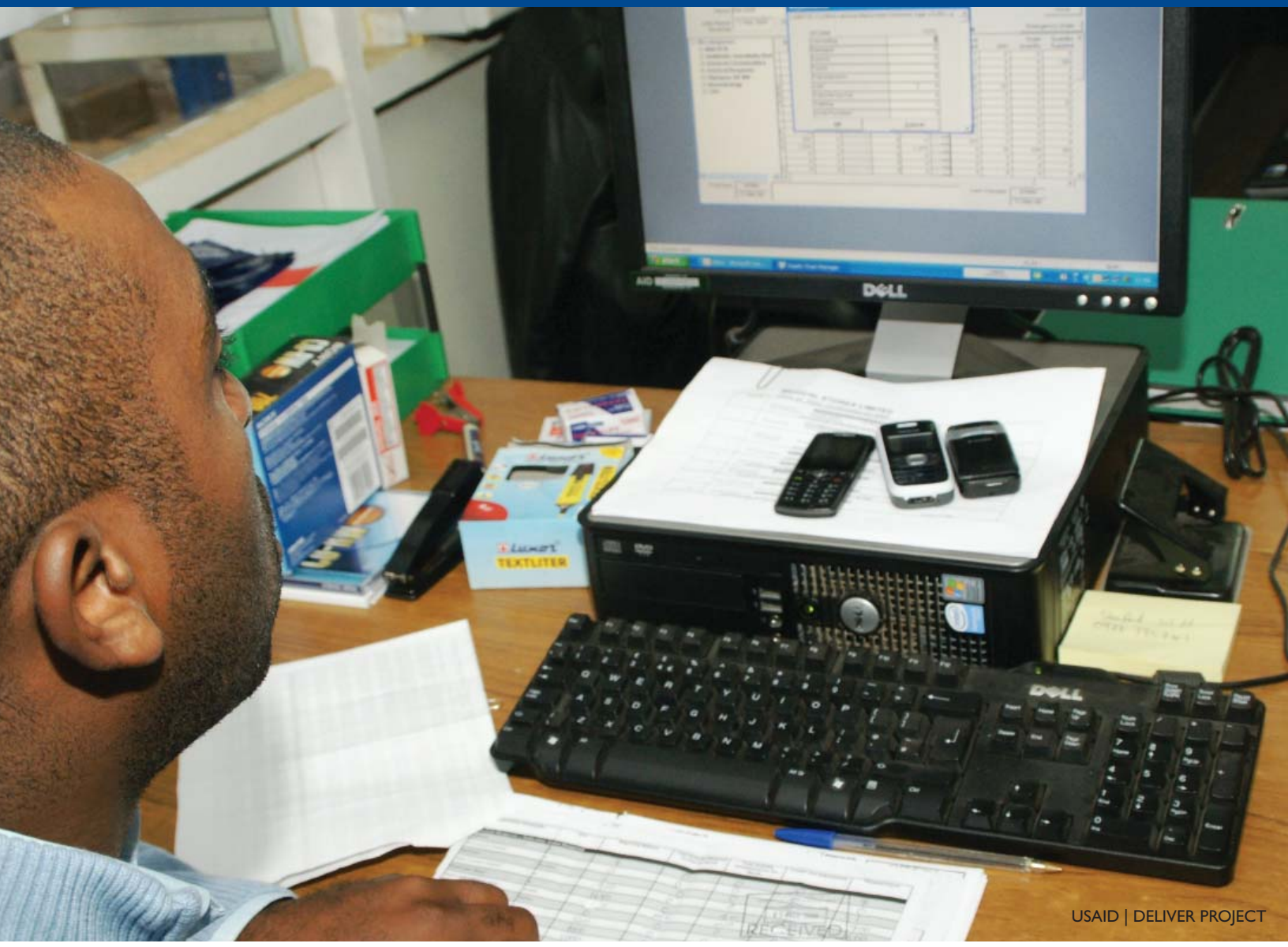
USAID | DELIVER PROJECT

Computerizing Logistics Management Information Systems

A Program Manager's Guide



USAID | DELIVER PROJECT

OCTOBER 2012

This publication was produced for review by the U.S. Agency for International Development. It was prepared by the USAID | DELIVER PROJECT, Task Order 4.

Computerizing Logistics Management Information Systems

A Program Manager's Guide

USAID | DELIVER PROJECT, Task Order 4
The USAID | DELIVER PROJECT, Task Order 4, is funded by the U.S. Agency for International Development (USAID) under contract number GPO-I-00-06-00007-00, order number AID-OAA-TO-10-00064, beginning September 30, 2010. Task Order 4 is implemented by John Snow, Inc., in collaboration with Asociación Benéfica PRISMA; Cargo Management Logistics; Crown Agents USA, Inc.; Eastern and Southern African Management Institute; FHI 360; Futures Institute for Development, LLC; LLamasoft, Inc; The Manoff Group, Inc.; OPS MEND, LLC; PATH; PHD International (a division of the RTT Group); and VillageReach. The project improves essential health commodity supply chains by strengthening logistics management information systems, streamlining distribution systems, identifying financial resources for procurement and supply chain operation, and enhancing forecasting and procurement planning. The project encourages policymakers and donors to support logistics as a critical factor in the overall success of their healthcare mandates.

Recommended Citation
USAID | DELIVER PROJECT, Task Order 4. 2012. *Computerizing Logistics Management Information Systems: A Program Manager's Guide*. Arlington, Va.: USAID | DELIVER PROJECT, Task Order 4.

Abstract
As in-country public health logistics systems become more integrated and sophisticated, many countries are looking to automate their logistics management information systems (LMIS) in order to improve the quantity, quality, and timeliness of logistics data throughout the country. These guidelines were written for managers in the Ministry of Health, program managers, donors, and management information system (MIS) program officers as a reference when considering starting an LMIS automation project, planning for one, and executing that plan. Organized around the phases of a software development project, these guidelines provide practical information and resources to ensure proper program management of a complex technology planning process.

Photo credit: Health logistics staff person enters data into a computer. USAID | DELIVER PROJECT 2010.

USAID | DELIVER PROJECT
John Snow, Inc.
1616 Fort Myer Drive, 16th Floor
Arlington, VA 22209 USA
Phone: 703-528-7474
Fax: 703-528-7480
Email: askdeliver@jsi.com
Internet: deliver.jsi.com

Contents

Figures

Tables

Acronyms

CMS	central medical store
COTS	commercial, off-the-shelf (solution)
CRDM	Collaborative Requirements Development Methodology
HIS	health information systems
HMIS	health management information system
ICT	information and communication technology
IT	information technology
IV&V	independent verification and validation
JAD	joint application design
LMIS	logistics management information system
MIS	management information system
MOH	Ministry of Health
RFI	request for information
RFP	request for proposal
SDLC	software development life cycle
SDP	service delivery point
TCO	total cost of ownership
USAID	U.S. Agency for International Development
WMS	warehouse management system

Acknowledgments

The primary authors of this guide – Emily Bancroft, Jessica Crawford, Marasi Mwencha, Mimi Whitehouse, Naomi Printz, Joy Kamunyori, and Kate Waldman – would like to acknowledge the numerous people involved in the framing, research, drafting, and review of this document, including Ashraf Islam, Ron Pankiewicz, Wendy Bomett-Dodie, Greg Roche, Ashley Smith, and Abdourahmane Diallo. The authors would also like to acknowledge all the USAID | DELIVER PROJECT team members who have assisted in the implementation of computerized logistics management information systems during the course of the project. The expertise and lessons learned that resulted from these experiences all contributed to the content of this guide.

Introduction

As countries continue to expand health programs and strengthen the supply chains that support them, there is an increased need for user-friendly tools and software packages to support the timely and accurate collection and reporting of logistics management information. This information can be used for operational decisionmaking, advocacy, and resource mobilization. Automation of a logistics management information system (LMIS) can greatly facilitate the work of supply chain managers by enabling faster collection, transmission, and aggregation of data; by reducing human error in calculations; and by allowing for visibility of data up and down the supply chain. Reducing the time required for data collection, transmission, and aggregation results in data being available more quickly for timely decisions and actions to help ensure products are available where and when needed.

Software development and, more specifically, the automation of an LMIS can and should follow project management and information technology (IT) best practices. These guidelines were written for managers in the Ministry of Health, program managers, donors, and MIS program officers as a reference when considering starting an LMIS automation project, planning for one, and executing that plan. By having a common reference to draw on, the authors hope the different managers who are involved in critical decisions of an automation project will understand the steps required to develop a robust IT solution and their role in supporting it.

How to Use This Guide

These guidelines are intended for program managers who understand how logistics systems operate and who are planning to automate all or part of their LMIS. The creation of an automated, or computerized, system is a process and not an event. The purpose of this guide is to ensure that program managers working on in-country supply chain improvements have the tools and knowledge they need to guide the complex process of overseeing a software design and implementation process. This guide should provide program managers with enough information to understand the process of automating an LMIS; it is not, however, intended to make a program manager an IT specialist or a business analyst. The authors suggest that, whenever possible, the program manager work with a skilled IT technical project manager who has experience in successfully managing the development of computerized information systems.

Throughout the guide, the document will refer to various outputs that can be expected during the life of the computerization project. Each time an output is referenced in the text, the icon at left will identify it so the reader is alerted to the purpose of the output and any guidance for its creation.

This guide and multiple templates that are referenced throughout the guide are available electronically at http://deliver.jsi.com. The outputs noted in the electronic version will also link to templates or examples of documents that have been created for other projects that program managers can use or reference in their own automation project.

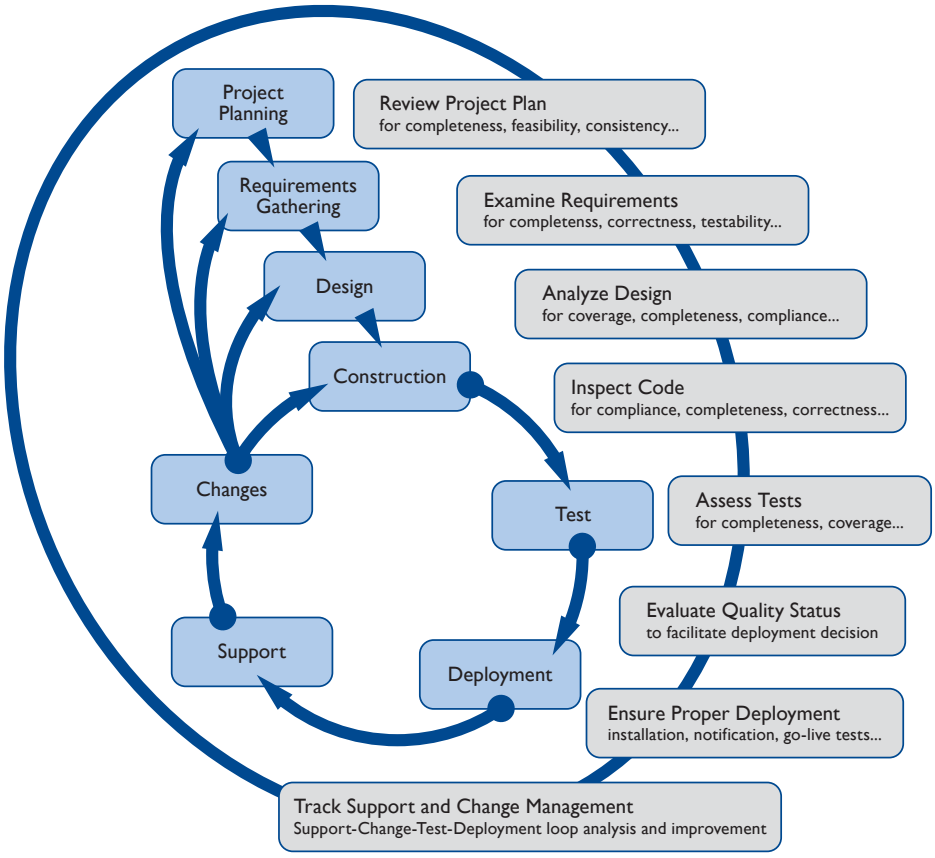
Program Manager Responsibilities

For the purposes of this guide, program managers are distinguished from other types of managers, such as project managers or technical managers. A program manager might be the manager of a vertical program within the Ministry of Health (MOH) who is responsible for multiple projects, with each project having its own project manager. For an automated LMIS, an IT manager who has technical skills may also provide an important level of management. Throughout the process, committed and skilled staff will be required to design, manage, develop, and use the automated system. However, someone will need to be responsible for pulling everything together, and that responsibility will lie with program manager.

In order to ensure rigor is brought to the project management process, the program manager needs to ensure the automation process follows the software development life cycle (SDLC) methodology, a best practice approach toward articulating and constructing software solutions. The program manager should understand the SDLC process so you can use the same vocabulary as your LMIS project manager. The SDLC process guides the development of software systems and the process that people generally use to develop these systems. In any such lifecycle, people, process, and technology all play a role in success.

Figure 1 shows one example of the software development lifecycle.

Figure 1. Software Development Life Cycle (SDLC)



Source: USAID | DELIVER PROJECT, Task Order 1. 2009. *Turning the Digital Corner.*

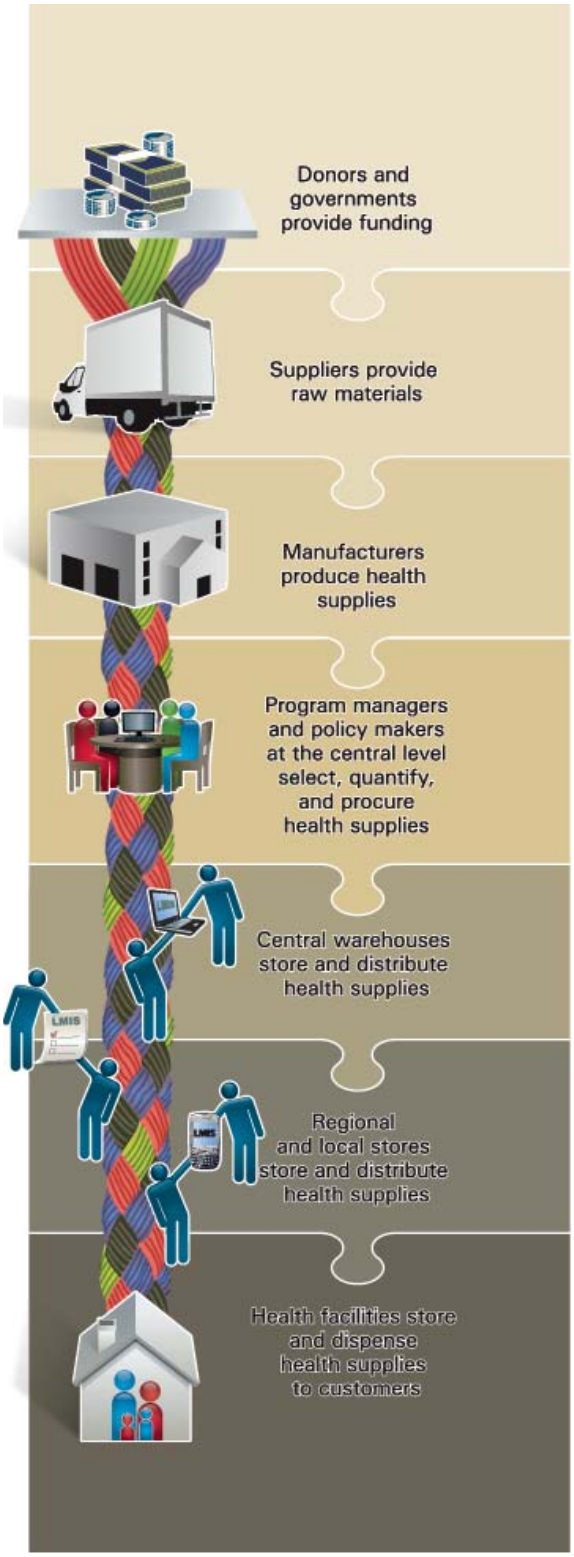
Each donor may also have specific guidelines that you need to follow as part of the software development process, and the program manager is responsible for making sure the project complies with all donor requirements. For example, if you are using U.S. Government funds to support the project, you may be required to show compliance with the SDLC process by providing copies of all documentation to your contract management team or by adhering to specific approval guidelines such as the ADS 548 Independent Verification and Validation (IV&V) Reviews. Although this guide does not go into great detail on specific requirements such as ADS 548, it does provide the framework for compliance with the SDLC process.

The program manager will need to manage expectations, generate the appropriate documentation, manage risk, and effectively communicate the status of the project to various key stakeholders. Everyone who comes into contact with an automated LMIS will have expectations about how the system will work, what information it will provide, how the information will be provided, when the system will be completed, and how much it will cost. In today's global economy, implementing partners, donors, and program managers within the MOH are under pressure to provide positive results in a short amount of time and within budget. Managing these expectations requires strong leadership during the implementation of an automated logistics management system. In addition to that, there will be a need, especially in the initiation and acquisition phase, to ensure the requirements (i.e., what the automated system needs to do) are appropriately specified and the timeline for the project and budget are adhered to.

What Is an LMIS?

An LMIS is a system of records and reports – whether paper based or electronic – used to aggregate, analyze, validate, and display data (from all levels of the logistics system) that can be used to make logistics decisions and manage the supply chain. A well-functioning LMIS provides decision-makers throughout a supply chain with accurate,

Figure 2. The Integrated Public Health Supply Chain



Source: John Snow, Inc.. 2011. *Getting Products to People: The JSI Framework for Integrated Supply Chain Management in Public Health.*

timely, and appropriate data, such as stock on hand, losses and adjustments, consumption, demand, issues, shipment status, and information about the cost of commodities managed in the system.

Figure 2 shows a framework for understanding a public health supply chain system. In most systems, the data that describe the logistics system and track movement of products through the system come from periodic reports prepared by the personnel in the facilities who manage the products. Using these data, a computerized LMIS calculates essential stock status indicators and provides information in the form of reports and graphs to decisionmakers to assist with the monitoring of logistics system performance and the planning of future product and logistics system requirements. A computerized LMIS provides several important benefits over a manual LMIS, such as ensuring mathematical accuracy, rapid aggregations, calculations, and productions of reports and graphs. A computerized LMIS also provides functionalities such as alert mechanisms to assist in decisionmaking.

An automated LMIS is just one component of the systems that may be in place to manage supply chain operations in your country. There are likely also warehouse management systems (WMS), transportation management systems, as well as billing and accounting systems or any of the myriad other types of systems that support a logistics operation in place already. The difference between an LMIS and the other kinds of systems is the type of decision the system helps to inform. A WMS, for example, helps manage and control products within a warehouse. It also helps manage warehouse processes, such as receiving and storage, order processing picking and packing, and shipping products out of the warehouse.

In a public health setting, the collection of data for logistics operations has been, at times, confounded with the collection of data for other purposes, particularly with health information systems (HIS) that track incidence of disease and health service delivery. Logistics information and health information management are designed to facilitate very different decisions, often by different stakeholders. The types of questions that can be answered using logistics data are:

- What is the demand for a product through this system over a certain period of time? Does demand change throughout that period?
- How long will current supplies last? When do we need to procure supplies?
- Where are our supplies in the pipeline? Do we need to move supplies between tiers in our system?
- Where is demand the highest? What are the implications for positioning inventory?
- Are we experiencing losses from the system that require us to take action? Where are losses occurring and what are the causes of these losses?
- Are supplies flowing smoothly through the pipeline? Do we need to adjust our inventory to account for bottlenecks in the system?

Are You Ready for Automation?

Before moving forward, it is important to make sure certain factors are in place to ensure the project's success. There are four main success factors that should be in place before moving forward on an automation project. As program manager, you will want to discuss these success factors with stakeholders involved in the process of automating the LMIS, so make sure these critical criteria are in place before you start expending extensive resources on an automation project.

Success Factor 1: Strong Existing Logistics Business Processes, or a Commitment to the Time and Resources Needed to Improve Business Processes Before or During Automation

While automation enables faster collection, transmission, and aggregation of data, and helps to eliminate human error in calculations, it will not solve every supply chain challenge. Often other interventions are needed prior to automation. As a case in point, automating bad management practices can waste money and make poor processes operate faster while achieving no performance improvement. As another example, if there are data quality issues at one level of the supply chain, automation will not improve the data quality without appropriate efforts in quality assurance, management, supervision, and data collection training. If strong business practices are not present prior to automation, the commitment of time and resources for process improvement will be necessary to ensure that the automation will successfully address the identified supply chain challenges and not exacerbate them.

Success Factor 2: A Strong Multidisciplinary Team

The implementation of automation initiatives requires thoughtful planning and management to achieve optimal outcomes. Automating any part of a logistics system involves input from a variety of people, including end users, developers, logistics personnel, funders, IT experts, and project managers. No single person will be able to grasp and explain what is needed from every level and dimension of the supply chain. Developing a strong automated system is dependent on having all perspectives involved during the design phase. If you are going to move forward on automating any aspect of your LMIS, you need to make sure that adequate personnel from key areas are available to participate in the design process. In addition, strong and capable leadership is an essential element for overall project success. This leadership should be at all levels, including the project manager, program manager, and executive sponsor levels. This will be discussed in more detail in Section One: Getting Started.

Success Factor 3: Long-Term Political and Institutional Support

It takes time to develop necessary user requirements for a system, select an appropriate solution, build or adapt solutions to meet the specific requirements, and then implement and roll out the solution. Depending on the solution, the levels at which you want to automate, and complexity of the system, the planning, design, and implementation of an automation project can easily take two to five years. Before moving forward, make sure political and institutional support is available to keep the key stakeholders engaged throughout the whole timeframe. In order to keep momentum during this long process, make sure to have interim milestones and deliverables to implement and test. For example, the automated LMIS could be rolled out in phases over time, starting at the least complex level of the system. Throughout the course of creating a plan, making the process transparent to the MOH and effectively communicating with it will help ensure stakeholders' continuous engagement.

Success Factor 4: The Resources to Go the Distance

Developing an automated system requires significant resources, both initially and throughout the life of the system. This includes not only hardware and software resources, which are required to develop and maintain the LMIS, but also human resources, which are required for sustainability of the system. It is important to consider the anticipated commitment from the individuals working on

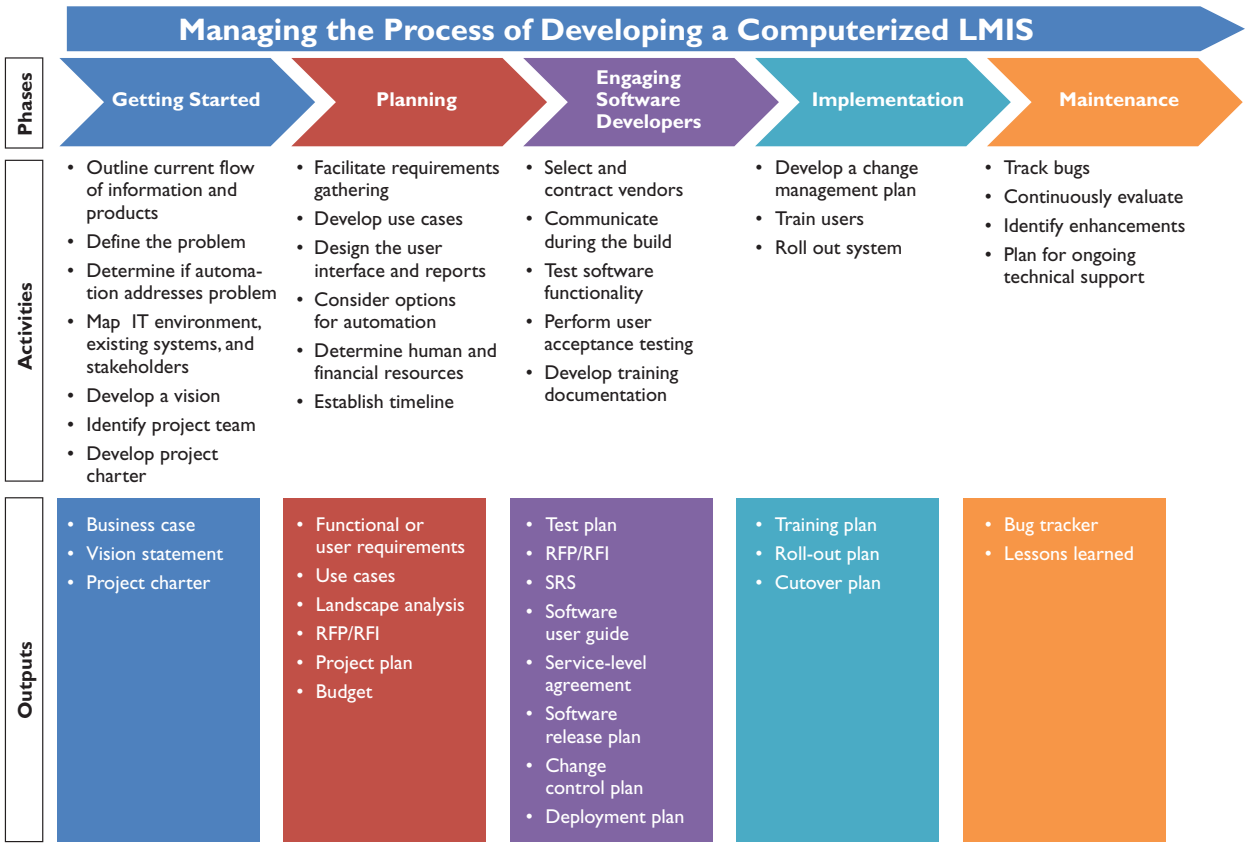
the system and to establish that adequate protocols are in place to ensure knowledge if key individuals leave the project. The resources needed include both an initial investment for developing and deploying the system as well as continued financial commitment to maintain the system (including future upgrades of software and equipment.) The amount of financial resources required depends on the solution selected.

Before starting a project, make sure the total cost of ownership of the system is well understood by those who are supporting the development of the system. Those who are going to be responsible for maintaining it also need to be made aware of the financial commitments. These funds need to be available and committed before you move forward on an automation effort. Securing funding for only the development and hoping that the funds for maintenance, ongoing support, and upgrades will be available in the future will put the project's success at risk. More information about the resources needed for both development and ongoing maintenance of a system is discussed in Section Two: Planning.

Structure of the Guidelines

These guidelines are structured around the SDLC, which progresses from project initiation to planning, execution, implementation and, finally, closure.

Figure 3. Managing the Development of a Computerized LMIS



For the purposes of these guidelines, we renamed the SDLC phases to more meaningful descriptions for a program manager, namely: getting started, planning, engaging software developers, implementation, and maintenance. Each of these phases is made up of a number of key activities, leading to outputs that will be important to the overall management of an automation project. The diagram below shows the phases, activities, and outputs. Descriptions of each phase, the activities necessary to carry out the phase, and templates and guidelines for the outputs are included in each chapter of this guide.

The **Getting Started** section of this guide will walk a program manager through the necessary steps to determine if an automation project is appropriate and feasible in the current environment, inlcuding outlining the current supply chain flow of information and products, identifying bottlenecks and other problems, and mapping the existing IT environment. In addition, this section will guide the program manager through the process of putting together a team and gaining the support of critical stakeholders for the automation project. The section will conclude with the steps needed to create a vision and project charter.

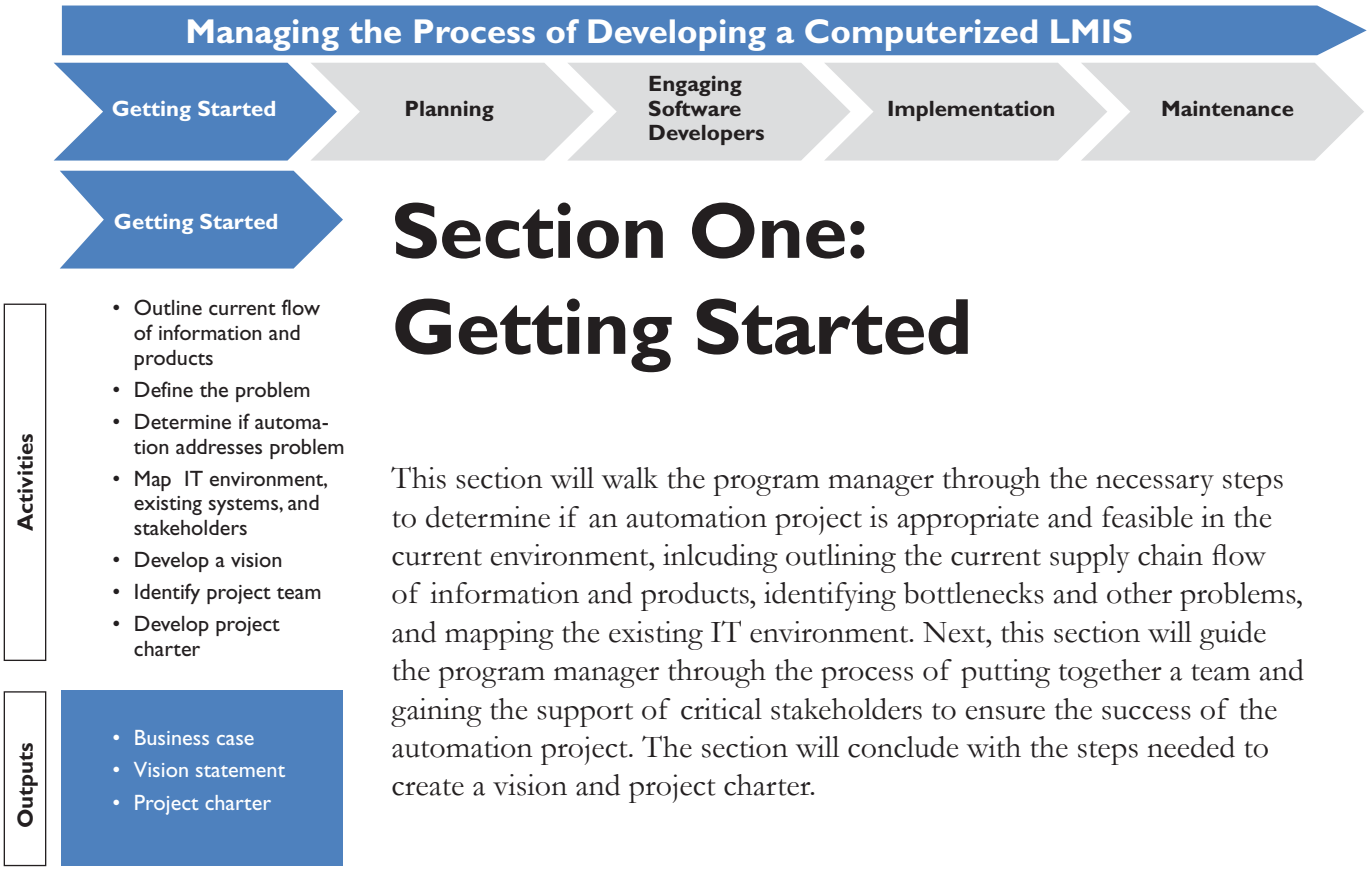
Once the project team, vision, and charter are in place, it is time to begin planning for the development of the automated LMIS. This **Planning** section will guide the program manager through the requirements gathering processes, the development of use cases, and the beginning stages of user interface design. Depending on the size of the project, the program manaager may articulate the project's needs through a consultative process, or coordinate this articulation by making use of IT professionals, typically called business analysts. The program manager will conduct or oversee a landscape analysis of existing solutions and consider the options for automation to determine the best way forward given the stated needs. At that time, more definite numbers can be put toward the resources and timeline, and this section will provide guidance for understanding the total cost of ownership (TCO) of the solution. Just as an architect will draw blueprints before construction begins, the end of this phase is marked by similar "blueprints," which are called the system architecture and technical design documents. They will guide the developers who will construct the system.

Regardless of the chosen solution, **Engaging Software Developers** will be necessary to construct or tailor the automated LMIS. Expectations of what IT specialists will do, the titles that are typically used, and the responsibilities they have are covered in this section. This section will guide the program manager through the selection and contracting process for developers. It will also explain the steps of the development process that the program manager will be responsible for, including communication with developers and users, overseeing the functionality testing, and the overall approval of the system. It is critical that the program manager treat this phase as a partnership, not an outsourcing relationship, to ensure that the IT solution will meet the needs of the users.

When the automated LMIS has been tested and works to satisfaction, a program manager may ask: "Now that I have a system, what next?" **Implementation** begins and introduction of the system to users takes place. In some instances, a pilot helps inform managers what other enhancements are required for the LMIS, and after they are made to the software, a larger roll-out plan is executed. If the automated LMIS is replacing another system (whether paper based or legacy system), a cutover plan will articulate the steps needed to make a smooth transition, including training and deployment

options. This section will guide the program manager through the steps toward ensuring a successful implementation of the automated LMIS.

Regardless of how well designed the automated LMIS is, users will identify problems and suggest enhancements regarding the performance of the new automated LMIS. Since the resources needed to maintain and support the system are identified and discussed in the Planning phase, the **Maintenance** section of the guide focuses on the role of the program manager after the system has been implemented and deployed. It is important to continue tracking the success of the system over time, addressing bugs or problems that arise and tracking user needs for future updates or upgrades to the system. A well supported and successful system will be one that is regularly improved based on user experience and evolving stakeholder needs.



Section One: Getting Started

This section will walk the program manager through the necessary steps to determine if an automation project is appropriate and feasible in the current environment, inlcuding outlining the current supply chain flow of information and products, identifying bottlenecks and other problems, and mapping the existing IT environment. Next, this section will guide the program manager through the process of putting together a team and gaining the support of critical stakeholders to ensure the success of the automation project. The section will conclude with the steps needed to create a vision and project charter.

Outline Current Flow of Information and Products

To help determine which process, decision, or action makes sense to automate, it is important to outline the flow of products and information in the existing logistics system. This can be done by working with the logistics team to develop a simple diagram that shows both the flow of data and the flow of products throughout the logistics system. The purpose of this diagram is to help illustrate the following:

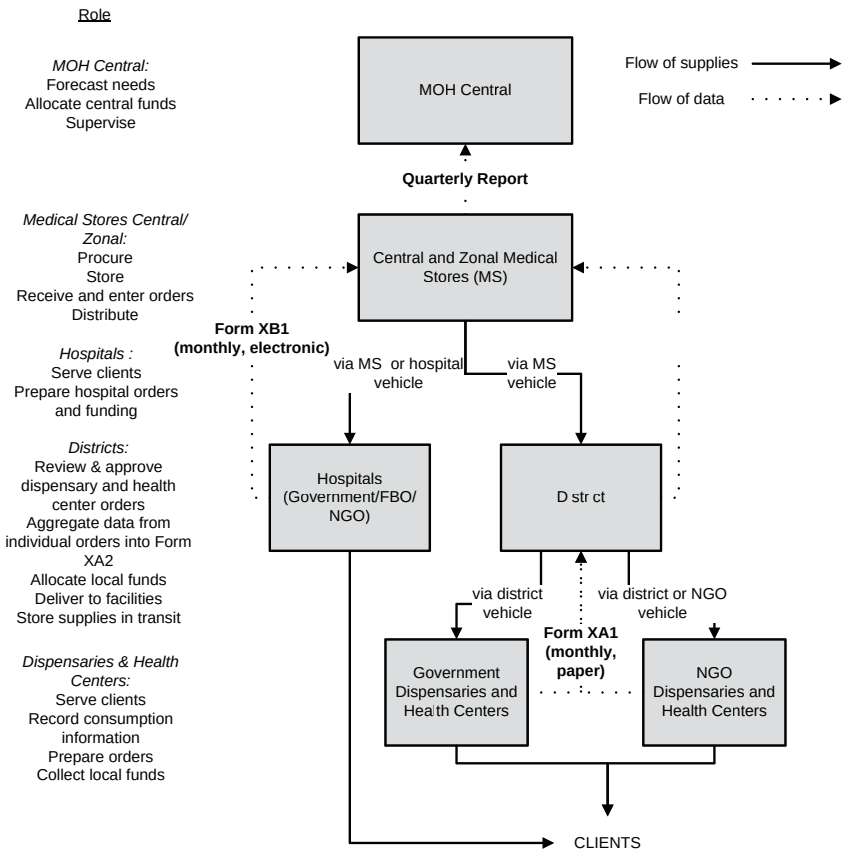
- The data collected at each level of the health system
- The forms/tools used to collect these data
- Points of data aggregation within the system
- The means by which data move from each level

Define the Problem

Using the diagram showing the flow of products and data, identify where decisions are made at each level of the system and the data that are needed to inform these decisions. This exercise will help highlight gaps in the information flow or current bottlenecks, where there may be opportunities for improving the system through automation. It may also be helpful to discuss this flow of information with stakeholders to identify their biggest problems in the logistics system. Some of the issues that are often raised include:

- Lack of data and inability to monitor the functioning of the logistics system at all levels of the health system

Figure 4. Sample LMIS Information and Supply Flow Diagram



- Poor quality of logistics data
- Delays in receiving reports and orders from health facilities
- Lack of information regarding current stock on hand and resupply quantities needed at service delivery points (SDPs)
- Delays in shipment of medicines from national level to SDPs
- Inaccurate forecasts and plans for future investments in medicines and supplies at the national level
- Limitations in addressing demand-driven need for medicines and supplies, such as seasonal fluctuations of disease burden, outbreaks, or changes in accessibility of SDPs due to weather or geographical barriers.

Table 1 shows a sample description of the decisions, data needs, and bottlenecks at various levels of the health system.

Table 1. Sample Description of Data Needs and Bottlenecks of Current System

Level	Decisions	Data Needed	Level of Data Visibility Required in LMIS	Current Bottlenecks
Health Facility	• Medicines and supplies needed for next month or next quarter	• Usage data from previous months and years • Population data • HMIS data • Budget availability	• Historic data on previous usage (at facility level) • Previous orders for facility • Budget remaining for facility	Orders and consumption data stored on paper forms; not aggregated for easy view of historical information
District	• Medicine supplies needed for each facility in district • Repositioning of stock between facilities to address shortages • Forecasting to region/national level for future stock needs	• Usage data from previous months and years • Real time stock information at facility level (if possible) • Population data • HMIS data • Budget availability	• Historic data on previous usage (at all facilities • Previous orders (across district, by facility) • Budget remaining (across district, by facility)	Orders and consumption data from facilities stored on paper forms; not aggregated for easy view of historical information or for data across the district; currently no real time visibility into stock levels at facility
Regional / National Medical Stores	• Timing of deliveries for new procurements • Immediate national stock needs for next six to nine months	• Order data from previous months and years • Plans of MOH, partners and donors for procurement	• Historic data on previous usage (at all facilities) • Previous orders at all levels • Credit line remaining for all customers	Only order data automated, making other data hard to analyze or use for decisionmaking; LMIS not easily connecting with MOH procurement records to record upcoming national orders or donor plans for procurement
National Level - MOH	• National stock levels required for three- to five-year forecast	• Usage data from previous months and years • Order data from previous months and years • Population data • HMIS data/disease incidence survey data • Budget availability • Plans of partners and donors for procurement	• Historic data on previous usage (at all facilities) • Previous orders at all levels • Budget data	No access to logistics data stored at medical stores; paper-based records of logistics data at district and regional levels making it difficult to aggregate and analyze in timely fashion; limited access to donor procurement plans to feed overall procurement plan

Determine If Automation Addresses Problem

Once the problem(s) has been defined, it will be necessary to determine if automation will address it. Automation can facilitate a range of actions, from the collection of data at the facility level, the transmission of these data from the facility to a higher level in the health system, the aggregation of these data at the district, provincial, or central level, to the analysis of the data and the development and transmission of reports back down the system.

A few example actions that automation could help with include:

- Moving data more quickly between levels of the supply chain
- Calculating accurate resupply quantities for facilities based on previous data
- Monitoring the quality, completeness, and timeliness of reports and providing that information quickly and accurately
- Assessing national stock status and providing quick feedback on pipeline issues based on current, past, and future usage patterns
- Tracking discrepancies of stock (for example, quantities issued vs. quantities received) within the system

Investing in any automation effort is a fundamental change to a logistics system, as in most cases it will seek to enhance the performance of the logistics system by introducing new efficiencies in information management and decision-making. As mentioned above, automation will not necessarily solve the logistics system's current challenges. The desired improvements in efficiency and/or effectiveness should justify the cost of an IT investment. Using the list of bottlenecks or barriers identified by the team, talk through each one to determine whether automation will help address the identified problem. and the increased efficiency or effectiveness that could be realized by investing in automation.

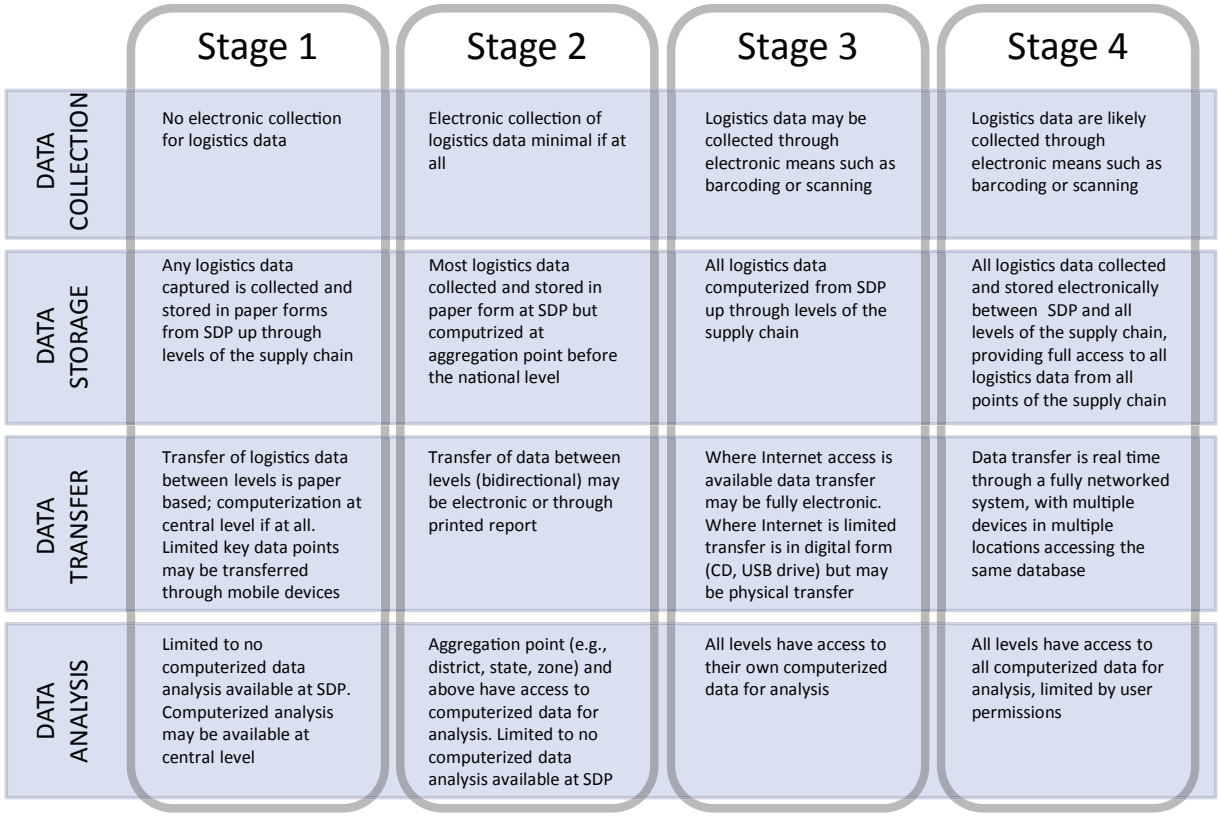
Automation of an LMIS can be very helpful to improving logistics system performance. For example, in Bangladesh, implementation of a web-based LMIS cut the time for LMIS reports from the field to the central level by more than half, from two months to twenty days. However, automation will not help solve all problems in a logistics system. Sometimes investing in a simple solution – such as the redesign of a complicated stock card or form to make it simpler for staff to understand and complete accurately – can also make a difference in overall system performance.

Another intervention, such as a system design, may be necessary before automating any part or level of the existing logistics system. Before embarking on any software development or automation effort, one should have in place business processes that support the flow of technically sound information.

Every level in the logistics system could benefit from automation, from the service delivery points, intermediary levels, and the central level. However, given the problem(s) that should be addressed, every level may not need automation. Or given the IT environment and the financial and human resources available, it may not be realistic to try to automate the entire system at the same time. Prioritize the levels that would most benefit from an automation intervention.

As shown in the stages of automation diagram below (Figure 5), there are different processes – data collection, data storage, data transfer, and data analysis – that can be automated, but not all processes need to be done at once. The supply chain in a country comprises several different levels, such as the central level, districts, hubs, and facilities. In some situations, it may make sense to only automate some levels of the supply chain instead of trying to automate the entire supply chain. For example, in a country where there are Internet and electricity challenges in rural areas but reliable Internet and electricity in larger cities, the best option may be to continue to have a paper-based LMIS at the facility level whose data would feed into an automated LMIS at the central and district levels. As mobile technologies become more accessible to people in rural areas, a mobile-based LMIS could be instituted at the facility level to feed into an automated LMIS at higher levels of the system. Data collection, transfer, and analysis could remain paper based, while data storage is automated. Or certain levels of the supply chain could automate while others stayed paper-based.

Figure 5. Stages of Automation



One common pitfall that often happens in software development projects is that the team has a solution in mind before analyzing and understanding the problem. For example, some members of the team may be convinced that a solution using mobile phones or a certain software package is the solution before determining and analyzing the problem at hand. Following the process in this guide should help team members avoid moving too quickly to arrive at a solution before determining the problems to be solved, the specific user requirements for the system, and the infrastructure and technology resources available. Spending time on the planning of the system as a group will lead to better long-term decisions about the technology or the solution needed.

Map IT Environment, Existing Systems, and Stakeholders

It is important to understand the environment and context in which an automated LMIS will be introduced. Mapping the current IT environment, systems, and stakeholders will help ensure that the automated LMIS will be accepted, sustained, and integrated into current and future IT initiatives in the country.

Review and Understand the Existing IT Strategy for the Country

Over the last decade, electronic HIS have become a critical part of the monitoring and evaluation and planning strategies of the health sector in most low-income countries. As a result, ministries of health have been actively building their capacity to plan and manage these investments to make sure they can support and maintain the systems over time. Most ministries of health now have Information and Communication Technology (ICT) departments, which are tasked with doing both long-term planning and ongoing support of electronic HIS. Many countries have written IT strategies specifically for the health sector.

Countries often write their eHealth strategies, which are then approved at the national level. A few examples of countries with national eHealth strategies include Kenya, Tanzania, and Ghana.

An overview of the current ICT strategy for health in the country and a vision are generally included in eHealth strategies. A more specific road map, guidance on standards and interoperability, costs and benefits, and implementation guidelines also are often included. These documents can help guide a program manager in automation projects.

Before getting started with any significant system design, make sure to review and understand the existing IT plans of the health sector. Many of these plans include requirements for coordination with other development efforts, preferences on software development languages and/or platforms, and considerations for support, training, and implementation of any new systems. There may also be requirements about how new systems interact with core existing systems, which may have an impact on the type of system you can implement. Key strategic decisions from the IT plan, such as the need to prioritize the strengthening of existing systems over development or implementation of new systems, could impact the timeline or support for your proposed automation efforts.

Understand the Current IT Environment and the Status of Reliable Internet Access, Electricity, Data, and Cellular Networks throughout the Country

It is important that the system is designed to work with the infrastructure currently available, with options for growth as infrastructure improves or changes. As described in the introduction, automation can be interpreted in several ways. Automation can happen at different levels of the LMIS – for example, only at the central level, where orders are entered, or all the way to the facility level. Depending on the system design, internet access or specialized hardware may be required. If the system design requires facilities to send in order forms via mobile phones, appropriate phones may be necessary, as would strong cellular networks with data capability. Simple SMS systems require just a mobile network. No matter which option is pursued, access to electricity is a key consideration. Once an automated system is in place, users will begin to depend on it, and electricity outages can cause the

logistics system itself to break down. Understanding the current IT environment will be essential to the successful deployment of an automated LMIS.

Identify Existing Automated Systems

An LMIS does not operate in a vacuum, but in a dynamic environment, where other automated systems are already being invested in, planned for, and implemented. In recognition of these linkages, automation of the entire supply chain will involve more than just the LMIS. Examples might include a warehouse information system, an enterprise resource planning (ERP) system, a transportation management system, or a vendor management system. There may be SMS systems already operating in-country that track stock levels of key tracer commodities and could therefore integrate with a future automated LMIS. As you start the planning process, one of the first key steps is to understand the other automated systems already in place in the health sector, both within and outside of logistics.

Given the need to collect data in areas with unreliable electricity and Internet access, more governments are turning to innovative solutions to ensure the availability of datafrom rural and remote areas. For example, cell phones and tablets are regularly used to collect and send data across cellular networks, reducing the need for expensive Internet connections at health facilities and hospitals. Partners in mHealth are working with governments to set up low- power computers and servers with battery or solar backup to maintain crucial software and databases, even in places where power is intermittent or non-existent. These innovations are expanding the possibilities for automation of health data in many low- and middle-income countries.

The LMIS may also need to interact with other information systems serving other health domains. For example, the routine health management information system (HMIS) may already be automated at a subregional level and could rely on data from the LMIS for key logistics indicators. Or an electronic medical records system could feed data about pharmacy dispensing into an LMIS. Mapping these existing systems is very important during the pre-planning stages. All the stakeholders involved in the automated LMIS process should understand interdependencies that may influence system architecture and design. Any possible areas of overlap with existing systems should be identified and understood before any software development begins.

Work with government officials, partners, and health staff to identify the existing systems that operate at the levels of the health system where you will be implementing the LMIS. For each system, try to answer the following questions:

- What is the purpose of this system? What data does it collect and at what levels of the health system?
- Does this system have any of the same goals of the proposed automated LMIS? Does it collect any of the same data?
- Does the automated LMIS need to share any data with this system?

Even though you have yet to design your system, answering these questions will help to clearly define the role of the automated LMIS in relation to other existing systems. This will help define the scope of the project and will contribute to the requirements when the team reaches that stage. More information on developing requirements is available in Section Two: Planning.

Identify the Key Stakeholder Group for Automated LMIS Design and Implementation

Designing and implementing an automated LMIS require input from a broad range of stakeholders. Committed and skilled staff members need to design, manage, develop, and use the automated system for it to be successful. In addition to the staff using the system, other stakeholders should be involved in the automation process. Everyone who comes into contact with an automated system will have expectations about how the system should work, what information it should provide, how the information should be provided, and when the system will be completed. This includes logisticians; medical stores department staff; national government staff involved in clinical planning, procurement, and monitoring and evaluation; ICT staff; donors; and implementing partners involved in supply planning or supply chain strengthening. Engagement with these stakeholders early in the process will help manage their expectations as time goes on. As mentioned previously, maintaining transparency during the automation project will help ensure that stakeholders are engaged and understand what is happening.

Because this is an automation process, the initial reaction of stakeholders may be to include in the process primarily technical staff who understand software development. It is very important to have a broad stakeholder group that includes logisticians, planners, managers, and IT staff, especially during the early stages of planning. Levels of knowledge, both about logistics processes and about software or technology design, will vary greatly among the participants. One way to help ensure that all stakeholders – regardless of technology knowledge or expertise – are able to contribute is to involve a business analyst in the planning process. A business analyst is someone who is trained to understand programmatic needs and processes and translate them into language understood by software developers. Identifying and involving someone with this skill set as a facilitator early in the planning process can improve communication among stakeholders and improve the quality and outcome of the planning process.

With all the above information in mind, a program manager should develop a business case document. A business case provides decisionmakers with a tool to analyze and assess options to inform and justify the impact of these project decisions. The business case document generally includes sections on background, current state, objectives, future state, needs, financial analysis, and conclusions and recommendations.

Business Case

Develop a Vision

A clear vision for the automated LMIS, agreed upon by all key stakeholders, is crucial to the success of the automation process. Being able to articulate a shared vision for an LMIS will help stakeholders define requirements for the future system, prioritize needs when resources are limited, and stay focused on the desired impact of the system.

In order to create the vision, it is valuable for a program manager to have an executive sponsor and steering committee already organized to assist in the process.

The **executive sponsor** is a manager with demonstrable interest in the outcome of the project. The executive sponsor acts as a vocal and visible champion, legitimizes the project's goals and objectives,

keeps abreast of major project activities, has final approval of all scope changes, and signs off on approvals needed to proceed to each succeeding project phase. The executive sponsor may elect to delegate some of the above responsibilities to the program manager.

The steering committee generally includes management representatives from the key organizations involved in the project. Steering committee members act individually and collectively as vocal and visible project champions throughout their representative organizations; generally they approve project deliverables, help resolve issues, make policy decisions, approve scope changes, and provide direction and guidance to the project. Depending on how the project is organized, the steering committee can be involved in providing resources, assist in securing funding, act as liaisons to executive groups and sponsors, and fill other roles as defined by the project. The steering committee will also help address any risks that are escalated to them by the project team. They are instrumental in guiding the project to successful completion and establishing the high level road map and vision that the project team should follow. Some of the benefits of developing a vision include:

- Determining and clarifying expectations of policymakers and users of the system on what the system will eventually be able to do as well as how long it might take to implement
- Reaching consensus on what functions the system will support
- Cultivating buy-in from all relevant stakeholders
- Building ownership of the system
- Defining the objectives and desired results of computerization
- Identifying the resources that need to be mobilized for the development and ongoing maintenance of the system

Vision

There are various ways to develop a vision statement. One option is to have a facilitated discussion with stakeholders, where small groups discuss how the work of key supply chain management staff and health service delivery could change in five to ten years as a result of a stronger logistics system. By looking forward five to ten years, participants are more likely to imagine technologies or scenarios they may not think exist or are feasible at the current point in time. The descriptions of the changes expected as a result of a strong future logistics system could then be distilled into key concepts that should be reflected in the vision. Interviews with stakeholders to understand how the changes they expect to see from the implementation of an automated LMIS, as well as to understand how they see the new LMIS interacting with other systems and processes already in place, could also provide valuable information for developing the overall vision statement for the project.

More information on developing a vision is available in *Turning the Digital Corner: Essential Questions for Planning for a Computerized Logistics Management Information System.*

Identify the Project Team

As the program manager, you will be responsible for ensuring that the appropriate team is in place to successfully carry out the project. As mentioned previously, the implementation of an automation initiative requires thoughtful planning and management to achieve optimal outcomes. Automating any part of a logistics system involves input from a variety of people, including end users, developers, logistics personnel, IT experts, and project managers. As the person ultimately responsible for the success of the project, you may not understand all aspects of the software development process,

but you are responsible for making sure you have people on your team who do understand these details. You are also responsible for knowing enough about the SDLC to provide guidance and leadership throughout the process. There are many groups of people involved in both the project and project management lifecycles. In addition to working with the executive sponsor and steering committee, program managers will need to ensure that a comprehensive project team is responsible for planning and executing the project. The project team consists of a project manager and project team members who are brought in to deliver their tasks according to the project schedule. All projects will require a different composition of individuals, but at a minimum, a program manager should expect to have a team that comprises the following staff, whose roles are described below.

The **project manager** is the person responsible for ensuring that the project team completes the project. The project manager develops the project plan with the team and manages the team's performance of project tasks. It is also the obligation of the project manager to secure acceptance and approval of deliverables from the executive sponsor and steering committee. In general, the project manager is responsible for making sure the project is delivered within the budget, on schedule, and within the work scope. The project manager is also responsible for communication, including status reporting, risk management, and the escalation of issues that cannot be resolved in the team. The project manager should be someone who has been trained in software project management or has prior experience managing successful software projects.

The **project staff team members** are responsible for executing tasks and producing deliverables as outlined in the project plan and directed by the project manager, at whatever level of effort or participation that has been defined for them. On larger projects, some project team members may serve as team leads, providing task and technical leadership, and on occasion maintaining a portion of the project plan. Users are individuals who identified the need for the system and/or those who will be primarily responsible for operating and managing the LMIS once it is deployed. Their feedback and engagement during the automation project will ensure that the end system will meet their needs and be a useful and appropriate solution.

Depending on the solution identified, **vendors and/or software developers** may be part of the project team. Vendors provide maintenance and ongoing support for off-the-shelf products that may be used as part of the LMIS. Software developers, either contracted or in-house, develop or refine existing products to meet the specific needs of the LMIS. Engaging with vendors and software developers is discussed in more detail in Section Three: Engaging Software Developers.

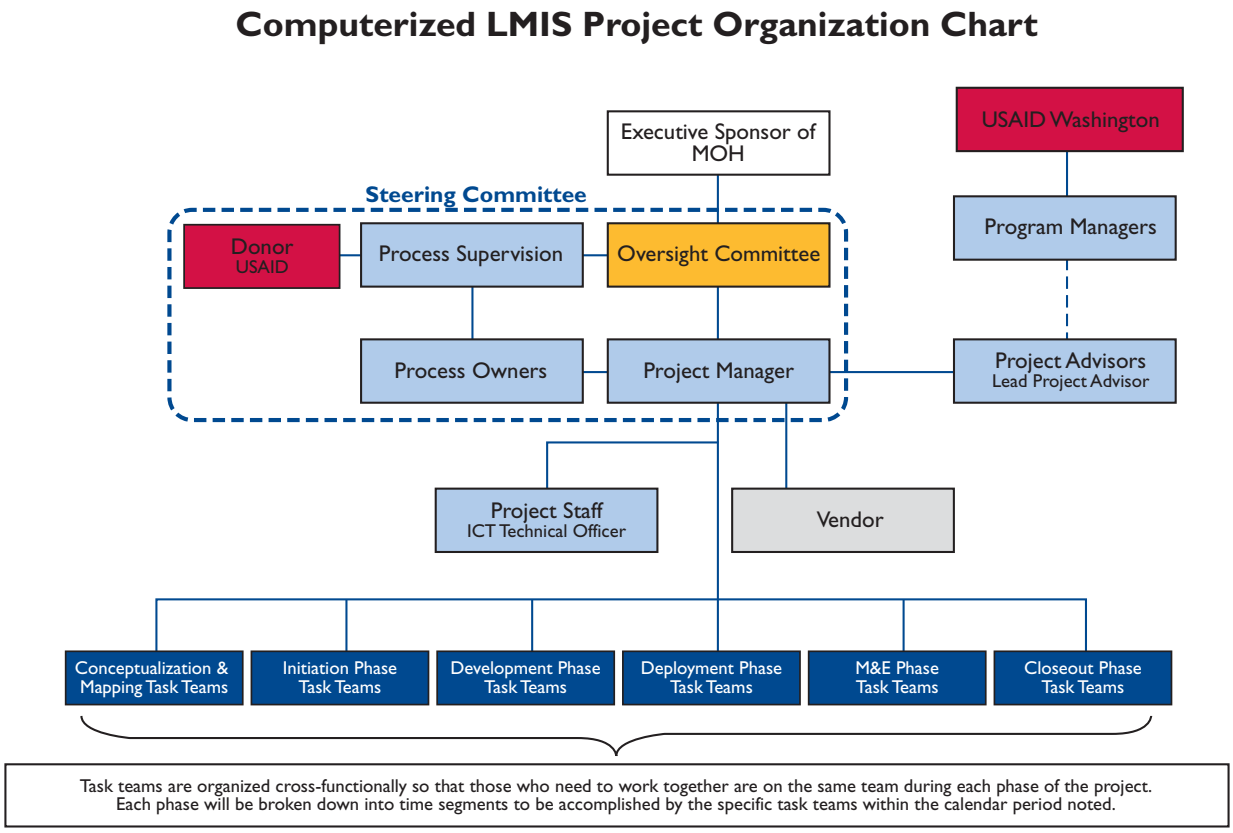
The following table demonstrates the key members of the project team as well as their roles and responsibilities.

Each project will have its own variation on the project roles listed above – just be sure all the roles are covered by someone on the team, and there is clarity on team member responsibilities and overall project ownership. The following organization chart illustrates one example of how an automation project can be structured.

Table 2. Project Team Members

Role	Responsibility
Executive Sponsor	• Provide policy guidance and final project sign-off • Approve final project deliverables
Steering Committee	• Review and approve the project vision and objectives • Help resolve issues and make policy decisions • Approve major scope changes • Provide direction and guidance to the project
Program Manager	• Assist the project manager with the implementation processes • Raise all risks and changes as they become apparent • Maintain all the documentation required throughout the project – changes, risk, issues, communication with donors and MOH, and acceptance of the project • Undertake all tasks necessary to produce the required deliverable
Project Manager	• Create a project plan and ensure that project follows the defined project plan • Produce deliverables on time, within budget and to specification • Implement management processes – time, cost, quality, change, risk issue, communication, and acceptance of project • Monitor and report project performance • Ensure timelines are met and adhered to • Monitor quality of the project
Project Staff	• Undertake all tasks necessary to produce the required deliverables on time • Work with the task teams to reach consensus on decisions • Report back to project manager on the progress of the project
Users	• Primary system testers throughout the course of development or customization • Give final user acceptance following testing
Vendors/Software Developers	• Respond to all feedback from user testing in a timely manner • Provide guidelines for testing and acceptance • Fix bugs discovered throughout testing

Figure 6. Organization Chart



Develop a Project Charter

A project charter is a critical document that will help in articulating the vision of the project, its intended scope, governance model, timeline, and budget. Before looking at business requirements or spending much time on a project, it will be vital for all stakeholders involved in developing the automated LMIS to have agreement on the scope of the project, the responsibility of the various parties, and the decision-making process for the development process. This information should be written in a project charter, approved by the executive sponsor and signed by all partners involved in the planning, development, and deployment of the automated LMIS. This document could also be referred to as a Memorandum of Understanding between the various project partners.

The project charter should include the following sections:

- Introduction and objectives of the automated LMIS project
- Project scope – what the automated LMIS will be and what it will not be
- Who the customer and end users are for the project
- Expected project success factors and risk factors
- Strategies for addressing and mitigating project risks
- Expected project budget and how expenditures are approved
- Governance plan for the project, including roles and responsibilities of all individuals involved in the project

Project Charter

- A communication plan to inform sponsors, customers, and other impacted parties of progress on the project
- Constraints and assumptions
- Timeline expected for project completion
- Milestones and deliverables

Once all this information has been agreed upon, it should be put in writing. Once the aforementioned information has been documented and has received sign-off from the steering committee members – including those from implementing partners, ministries of health, and donors – must sign off on the document, indicating they are all in agreement with the conceptualization of the project. This initial buy-in is critical to avoid any potential future disputes where one of the key parties involved in the implementation of the project states he or she never agreed to some portion of the documented information in the project start-up documentation. This is especially important for automated LMIS implementation projects. Often, a year or two passes between the time the project gets under way and the time the final LMIS is implemented and users are trained.

The project charter can also be used by the project manager when creating a project plan. Having a clear timeline for tasks and deliverables that the project manager shares with the program manager can help ensure that the project meets deadlines and expectations.

Section Two: Planning

Activities

- Facilitate requirements gathering
- Develop use cases
- Design the user interface and reports
- Consider options for automation
- Determine human and financial resources
- Establish timeline

Outputs

- Functional or user requirements
- Use cases
- Landscape analysis
- RFP/RFI
- Project plan
- Budget

Once the project team, vision, and charter are in place, it is time to begin planning for the development of the automated LMIS. This Planning section will guide the program manager through the requirements gathering processes, the development of use cases, and the beginning stages of user interface design. Depending on the size of the project, the program manager may articulate the project's needs through a consultative process, or coordinate this articulation by making use of IT professionals, typically called business analysts. The project manager will conduct or oversee a landscape analysis of existing solutions and consider the options for automation to determine the best way forward given the stated needs. At that time, more definite numbers can be put toward the resources and timeline, and this section will provide guidance for understanding the TCO of the solution. Just as an architect will draw blueprints before construction begins, the end of this phase is marked by similar "blueprints," which are called the system architecture and technical design documents. They will guide the developers who will construct the system.

Facilitate Requirements Gathering

After completing the visioning process discussed in Section One: Getting Started, the next step is to gather the requirements for the system. Clear and well explained requirements will serve as a primary reference for the rest of the software development process. Two valuable components of the requirements-gathering process are the "user" or "functional" requirements and the "system" or "technical" requirements. The user requirements are what a user would like the automated system to do. The system requirements define the hardware and software components necessary for the system to operate, which are often not visible to the users.

The following are examples of user requirements for mobile phones. Users must be able to initiate a connection by entering a phone number. Users must also be able to hear the person speaking on the other end, to speak and be heard themselves, and then to hang up the phone when finished with the call. In addition, users may need to save numbers so they can be retrieved easily for subsequent calls. The technical requirements that a mobile phone needs to satisfy in order to operate as specified include the ability to connect to a cellular network to initiate the call. It also must have a certain amount of memory available in order to save phone numbers, since the user has requested that as a functional requirement. System requirements are important to the success of making a phone call, but the user may not be aware of the technical aspects of a cell phone call or be able to define why they are important.

As you can see from the example above, a program manager needs people with different types of expertise to define requirements. The actual users of the system – in this case, the logistics experts and the program staff – are the ones who need to define the user requirements. The technology and software experts are needed to help define the system requirements, which are based on what the users want the system to do.

Documenting user requirements should, at a minimum, involve going through the steps mentioned below. Some of these steps may have been started early in the process, but they should be reviewed by the group in charge of gathering requirements.

Requirements

- Identify the subject matter experts and project representatives from each part of the logistics system who will serve as the users of the automated LMIS.
- Document current activities or business processes you expect the new automated LMIS will impact. For logistics, this will likely include activities, such as submitting reports and orders, processing orders, demand forecasting and procurement.
- Discuss what is not working about the process to identify areas of need.
- Document the requirements for the future automated LMIS based on the activities and the priority area of needs. The convention for capturing written requirements is for each statement to begin with "The system must . . ." with a description of what the system needs to do to support user and stakeholder needs.
- Review the results with users and stakeholders to ensure they are complete.In a collaborative forum that consists of all key project stakeholders and a user group, document user requirements discussed with all customer groups and subject matter experts. Specify the problems and needs, e.g., how the problems are affecting the health system and how the solution will help improve logistics management in the country.

In determining requirements, think about how the system will be used and what reporting will be required. The key to any successful automated logistics system will be in ensuring decisionmakers have access to actionable data for supply chain decisions. If the project manager does not pay close attention during the initial requirements phase, the automated system could fail to meet expectations. Once the requirements have been determined, you will then match the business needs to the scope you created in the project charter – or change the scope, which would require an amendment to the project charter document and new signatures.

Once the right set of requirements is documented and it matches the project scope, ensure that project sponsors, implementing partners, the MOH, and donors acknowledge the requirements, that the project is active and sponsored, and that they are in agreement with moving forward to the next project phases. There are many ways to gather requirements, including the technique described in the following paragraphs.

The Collaborative Requirements Development Methodology (CRDM) offers one way to facilitate requirements gathering for information systems. CRDM is a framework for organizing a shared approach to information systems requirements development. Developed by the Public Health Informatics Institute, CRDM is a process that can generate high-level user and system requirements that are understandable, adaptable, and useful to stakeholders and managers. This enables greater clarity

Using the Global Common Requirements for LMIS and the CRDM

Experience from using the Global Common Requirements for LMIS and the CRDM has found that using these resources has been effective in generating user and system requirements that are understandable, adaptable, and useful to stakeholders and managers for acquiring, enhancing, or developing an LMIS for any health product. A key aspect of the effectiveness of the methodology is the use of nontechnical language that is familiar to users and subject matter experts. This allows a large and diverse group of stakeholders to quickly create a set of starting requirements by reviewing and adapting the Global Common Requirements in a workshop setting.

The Global Common Requirements for LMIS includes a set of 12 standard business processes designed to improve the performance of logistics management in countries. It also includes a set of associated global, common requirements that were developed through extensive review with public health logistics experts from around the world. After being adapted to a country setting, these common requirements can be used to make informed decisions when acquiring commercial or open source LMIS solutions, evaluating capabilities of an existing software product, or enhancing existing implemented solutions. For implementing partners, the use of these requirements can be applied across diverse projects to encourage the development of common, reusable LMIS solutions. Well designed, robust, and proven global applications that are reusable can lead to lower total costs and reduce the time associated with implementing health information solutions. For donors, resources can be applied in more consistent, focused development efforts, which increase the value of these investments by producing stronger, better designed, reusable solutions.

The Global Common Requirements for LMIS have been adopted and customized in Kenya, Rwanda, Senegal, Tanzania, Vietnam, and Zambia. The report and the full list of Global Common Requirements are available at http://openlmis.org/download/tech_res/TS_lmis_crdm_ENG.pdf.

and accuracy when communicating to software and system engineers and vendors the needs of users. In 2010, PATH and the Public Health Informatics Institute used this methodology to develop a set of global common requirements for LMIS. These requirements are available at www.openlmis.org and for use and adaptation at the country level.

In addition to using the CRDM as a process to develop requirements, multiple techniques can be used to get buy-in and input into what the stakeholders expect from the automated LMIS.

They include the following:

- CRDM is a specific application of what is more commonly known as a *joint application design (JAD) session*. JAD sessions are generally facilitated sessions where various parties collaborate on the documentation of requirements. These sessions are most effective with the use of two third party consultants, or analysts, who can act as a facilitator and a scribe to generate and document the requirements.
- Conducting *interviews*, either one-on-one or in groups, is a common way to gather requirements. Sitting down with stakeholders and asking them what they do and what they want often generate the information needed for LMIS requirements. It is important to plan the interview questions in advance, using mainly open-ended questions that will allow the interviewer to probe where necessary. It is also important to recognize the perspectives of all interviewees and appropriately weigh and address their suggestions.

- *Brainstorming* in a formal or informal setting with appropriate stakeholders can effectively cast a wide net and identify many different possibilities for the LMIS. The program manager will need to work with the stakeholders to prioritize ideas that come out of a brainstorming session; the resulting consensus of best ideas can be used for initial requirements.
- *Direct observation and process mapping* can be helpful when gathering information on current processes. A program manager or a staff member may shadow individuals, such as health workers or district pharmacists, to gather information on the implicit requirements that may be overlooked in other methods of requirements gathering.
- *Prototyping* uses preliminary requirements to build an initial version of the solution. With a prototype, stakeholders have the ability to react and provide additional requirements for the LMIS. This process can be repeated until the LMIS meets the critical mass of user needs.
- Implementing a *survey or questionnaire* can be an informal and inexpensive way to generate requirements from various stakeholders, especially those in remote locations or who would likely have minor suggestions for overall LMIS requirements. Questionnaires may also be used to gather information from large numbers of people.

Requirements should be expressed in terms that are easily understood by the stakeholders, using the vocabulary and concepts of the supply chain domain. Requirements should also be neutral to the specific system implementation, and should neither discuss nor depend on the implementation. Requirements include not only the behavioral and quality of service expectations of the users, but also statutory constraints and commercial standards. In addition to developing functional requirements, focus should be placed on developing non-functional or system requirements that focus on the performance, usability, reliability, data integrity, security, upgradeability, and maintainability of the automated LMIS.

When the requirements have been written or updated, they should be reviewed by the appropriate stakeholders to ensure they adequately describe all user interactions with the product. Key stakeholders may include a supply chain expert, a business analyst, and a user experience architect. If any problems are spotted, e.g., requirements are omitted, they must be addressed so the requirements are valid at the conclusion of this activity.

Develop Use Cases

Use cases are an important part of the requirements process. They represent the basic functions of the automated LMIS, such as receiving or issuing products. In some cases, teams may want to start with use cases and then use them to develop requirements. In other cases, teams may start with other requirement-gathering methodologies and then develop the use cases as a final step to validate the requirements with stakeholders.

Use cases are written in a clear and concise manner to represent user's goals and the actions the computerized system performs to reach these goals. Simple, action-oriented, verb-object phrases are used to name use cases. (Examples of use cases' names for LMIS and WMS include "Receive Products" and "Issue Products.") A sample use case is depicted in Table 3 for reference.

Use Cases

The following guidelines can help focus the use cases solely on user and system actions and, at the same time, eliminate wordiness.

- Use simple grammar. The use of short, action-oriented, subject-verb-object sentences is the clearest way to present the story.
- Show clearly who is responsible for the action taking place. Include a subject in all sentences. Otherwise, people may have a hard time deciphering who does what to whom.
- Show the process moving forward. Do not include minute actions (e.g., "User hits tab key.") that slow down the use case.
- Show the user's intent in interacting with the system. Do not describe the user's movements in operating the user interface.
- Provide positive validation. The main use case is a success scenario, so use words such as "validate," not "check whether." Do not include tasks that lead to use case failure. Instead, include failure points in the extensions section.
- Use literal language. "User has System A alert System B." Many system requirements include the need to interact with other computerized systems. This phrase is the simplest way to convey that the user has control over the interaction, without specifying how the interaction is initiated.
- Use literal language. In many use cases, one or more steps can be repeated. List this repetition at the end of the step or steps being repeated by saying, "Do steps X-Y until condition." To make the use case easier to read, do not number the repetition.

Table 3. Sample Use Case

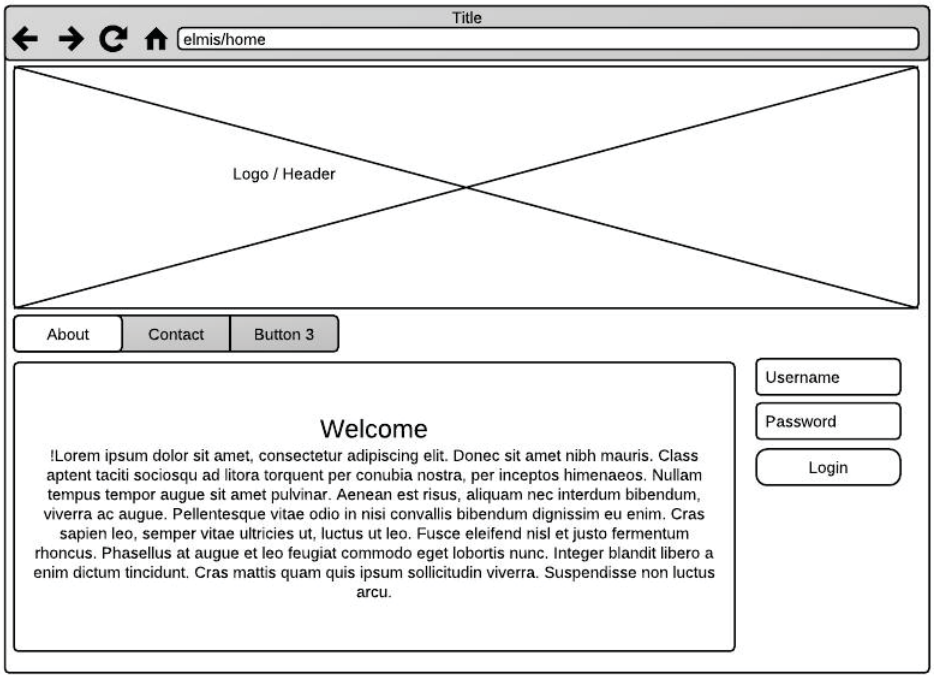
Use Case Example: View Warehouse Stock Information	
Use Case ID	UC-REQ09
Description	View warehouse stock information for a specific product.
Primary Actor	District Pharmacist
Stakeholders & Interests	MOH, Program managers
Pre-conditions	1. User must have appropriate permission to view warehouse stock information. 2. Stock status has been sent from the warehouse to the eLMIS.
Triggers	User logs on to system.
Basic course of events	1. System validates users and displays the eLMIS dashboard. 2. The user selects the "View Warehouse Stock" tab on the dashboard 3. The system prompts for search criteria including product code, generic name, or program. 4. The user enters one or multiple search criteria. 5. System validates search criteria entered. 6. System displays list of products matching search criteria, grouped by program. 7. User chooses an item from the list 8. System provides additional information on the selected item: • System displays stock status, in the form of a number, next to each product. • System displays stock status number in different colors to indicate if product will or will not be available for order. • System displays cost per unit of stock • System displays last time data was updated from warehouse
Post Conditions	
• Success end conditions	System displays WH products stock status.
• Failure end conditions	System does not display WH products stock status.
• Minimal guarantee	System displays notification for why stock status is not available for the specific search criteria.

One of the artifacts of the use cases for a software developer are the business rules that guide how the system needs to operate. Business rules are the critical behind-the-scenes rules the computerized LMIS must follow, e.g., adherence to the product list managed by the central medical store (CMS). In many cases, business rules are incorporated into the use cases, but it may be helpful to have a software developer – or someone very experienced in software project management – review the use cases and determine the business rules. Some business rules will need to be documented separately since they do not directly relate to the user interactions described in the use cases.

Design the User Interface and Reports

After the project participants agree on the requirements for the new computerized system, the next step is to create more detailed descriptions on how the new system will look and act – descrip-tions or template images of all user screens and reports, as well the database design. Some of these details around system design – including the architecture and database design – will likely be completed once you have your software developers or software vendor on board. More information about finding and engaging with software developers or software vendors is included in Section Three: Engaging Software Developers of this guide.

Figure 7. Sample Wireframe of a Webpage



User Interface Design

User interface design is very important to the success of the end product. It also provides a great opportunity for getting feedback from future users on the overall system design. Keeping the users in mind while developing the user interface will help ensure the system is easy to understand and incorporated into the appropriate workflows. To help users envision what the system will look like, the program manager can work with the software developers – or other team members who are good at

user interface and understand the functioning of the system – to develop sample screen layouts or "mockups" of the system's look and feel.. Software developers will often refer to these mockups as "wireframes." There are simple tools – some free and some for a fee – that developers use to create sample screens. PowerPoint or any graphics program can be used to demonstrate how the automated LMIS should look.

Report Design

Reports are a crucial part of the automated LMIS, as access to data is likely one of the problem areas driving your program to automate its LMIS. Additionally, users have more of an incentive to enter data into any system if they are confident they will be able to retrieve the data they need. Reports are the main outputs of a computerized LMIS. Logistics managers use them for evidence-based decisionmaking, and high-level managers may rely on them to implement policies affecting the national supply chain. But the process of implementing a computerized LMIS often focuses on data entry operations – in other words, how to get data into the computer rather than how to get data out of it. An approach to LMIS implementation that focuses on reports as well as data entry stands a better chance of meeting the needs of higher-level decisionmakers.

During the requirements development, the team should have spent time discussing the types of reports that will be needed from the new system. As a next step, it is a good idea to also develop sample reports to ensure that all the detail behind the report calculations is clear to the software developers and to ensure that all the important reports and their expected outputs are included in the system design. -level logistics managers and decisionmakers will only see the system's outputs and may never see the software itself. Showing these key users sample reports during the design phase can bolster high-level support for the development process. It will also help ensure that you have not missed any important reports in the requirements.

Reports are designed to answer the following key questions about the stock status of a particular facility or product, and about the overall performance of the logistics system.

What is the stock status for a particular product at different geographic levels (national, regional, etc.)?

- What is the stock status of a particular facility?
- How much of a particular product is distributed to each facility?
- How much of a particular product should be resupplied to a facility?
- What is the status of reporting for a particular period?
- Are there any errors in reporting by facilities?
- Are there any errors in data entry?
- Are any facilities not stocked according to plan, i.e., overstocked, understocked, or stocked out?
- Are there any unusual patterns of losses or adjustments for a particular product or facility?

Consider Options for Automation

Once the requirements have been developed, the next step is to consider how you get from the requirements to an actual software solution. A major decision will likely be whether you buy an existing solution from a vendor, often called "commercial, off-the-shelf" (COTS) software, or build a new solution that meets your unique needs, starting from scratch or building off a similar solution.

This is often referred to as the "buy or build" decision. In reality, there are many options that fall between buying a pre-developed solution and building a new one.

Conduct a Landscape Analysis

Landscape Analysis

The requirements document will be the basis for evaluating your "buy or build" decision. Searching for existing software that may serve the needs of your project before making the "buy or build" decision is a valuable enterprise. Given that many countries have begun implementing automated LMIS, a natural starting point would be conducting a landscape analysis to determine if there is any system that matches most of your requirements. If there is one that does, then an acquisition strategy can be developed that allows for the procurement of that system and its further customization to meet the country-specific needs. If, however, there is no system that remotely matches your requirements, a decision can be made to procure software from a vendor that specializes in automated LMIS, or build a new one with internal development personnel. Do not extend scope or budget without going back through the sign-off process.

Off-the-Shelf Software

Commercial solutions for automating systems can be purchased commercially or obtained through open source. While there are many advantages to implementing a previously developed, packaged solution, it should be noted that such a solution would require some customization in order to fit the functional needs of a specific country or organization. At times, customizing off-the-shelf software can be as costly and time-consuming as custom building software.

Buying Commercial Off-the-Shelf (COTS) Software Solutions

Buying packaged software is an attractive alternative for implementing automated systems that have common features across multiple organizations and environments. There are several advantages to buying packaged software, including:

- Lower maintenance costs over the long term. While the initial cost for licensing and technical support may appear daunting, the long-term costs for maintaining packaged software are lower than the costs for custom-developed software. The cost for maintaining packaged software is spread across a large number of clients, so each client's share of that cost is small compared to the cost for maintaining software developed specifically for one client.
- Faster implementations. Commercial software has already been tested by the vendor as well as by numerous clients, so unlike custom-developed software, it does not need to be tested thoroughly before implementation. It was also designed to be installed in a variety of client environments and includes utilities to facilitate installation.
- Access to upgraded versions of the software. Vendors continuously develop new versions of packaged software to fix identified defects and to include suggested enhancements from clients. Clients can then choose to upgrade to the latest version of the packaged software rather than spend time and resources developing custom modifications.
- Access to product and user support. When clients buy a software package, they also have the option of purchasing support from the vendor for some period of time, usually one year. This support helps address any technical problems clients experience using the software. In addition to support agreements with the vendor, clients can find answers to their questions from other users of the software, often via Internet-based discussion forums set up for that purpose.

Obtaining Open Source Software

Open source software is software that is available in source code form and, depending on the licensing terms, permits users to study, change, and improve the software for their needs. Obtaining an open source solution may be an attractive option for an organization that does not want to custom build a solution, but would prefer not to pay the sometimes substantial licensing fees associated with COTS software. It is worth noting, however, that open source software requires customization and installation by someone knowledgeable in order to make it applicable to a given context. In addition, maintenance and support services may have to be purchased from external organizations, if the client does not have sufficient in-house IT capabilities. The savings obtained from not having to pay licensing fees, therefore, may be offset by the costs of paying for installation and support.

Purpose-Built Software

The automated LMIS software can be developed specially for a specific client and country of implementation. This has the advantage of building the software to suit the particular needs and idiosyncrasies of the system in the country, taking into account all nuances and possible issues that may arise during development. Purpose-built software can be developed by an in-house development team or be commissioned from an external IT consultant or organization.

Using In-House Developers

Using in-house resources is helpful since developers are nearby and often already familiar with many of the LMIS needs. However, pursuing this route requires human resources or IT skills the client may not have.

Buying the Services of an IT Consultant or Organization

Hiring an external consultant or organization to custom build a solution may be a suitable approach for public health organizations that have limited or no IT skills and limited human resources. It also allows these organizations to benefit from the skills of outside consultants or companies that have experience implementing automated systems for a number of clients in a variety of settings.

JSI has used the approach of hiring an external consultant to custom build solutions in several countries, including Bangladesh, Honduras, Mozambique, Nepal, Philippines, Tanzania, and Zimbabwe.

However, there are challenges associated with hiring external IT support. Organizations that hire external consultants have to manage them to ensure that targets are met on time and on budget. This management role may sometimes take more time and money than anticipated, and it is important to ensure that adequate resources are available for this role before selecting this option.

Table 4 illustrates a few advantages and disadvantages of off-the-shelf and purpose-built software.

Determine Human and Financial Resources

The final step in the planning phase is to determine the TCO – the human and financial resources needed to complete the development of the automated LMIS, implement it, and support and maintain it over its lifespan. Software development projects often fail because program managers look at these steps as individual budget items and therefore do not consider the TCO in advance. One of the most important things a program manager of an automated LMIS project can do is to make sure

Table 4. Advantages and Disadvantages of Software Development Methods

	Advantages	Disadvantages
Off-the-shelf	• Lower maintenance costs over the long term • Faster implementations • Access to upgraded versions of the software • Access to product and user support	• May not meet all the needs of the system • Risk of support loss if vendor no longer sells product • More expensive up-front costs • May require advance training for users
Purpose-built	• Designed with specific requirements of system in mind • Lower up-front costs • Training may be easier because system designed with user input • Maintenance resources can be developed within the organization	• Longer development and implementation timeline • Burden of maintenance costs on the organization • Additional costs for continuous improvements and upgrades • Small user and support base

the project team and leadership fully understand the TCO of the automated LMIS. It is important to do this early in the process once your requirements and needs are defined, but before you start any software development since understanding the TCO will help you better clarify future decisions around software development and purchasing.

Budget

A budget created using the TCO as a guide should include not just the initial, up-front costs needed to develop or purchase the automated LMIS, but also the ongoing costs of training, deployment, support, and maintenance for at least five years after development.

A sample TCO model is depicted in Table 5. Your automated LMIS may not include budget line items for everything on the list in the sample model. But the team involved in developing, implementing, and supporting the LMIS should ensure that all these items have been discussed during the TCO process to ensure that no anticipated costs are left off the model. In some cases, it may be very difficult to estimate all costs accurately. Collect as much information as you can from as many key informants as possible. Other Ministries that are supporting large software deployments may be able to provide sample costs for some categories based on their experiences. Your vendor or software developers may be able to provide cost information on other categories. Your budget office may be able to provide estimates for certain categories based on other similar projects.

When developing your TCO model, make sure all the necessary project team members are included in the budget. The TCO model above integrates the various staff into the different budget categories since some may be short-term contracts, some may be part of the fees you pay for software development, deployment, and support, while others may be longer-term staff who will be employed through the project. If any of the staff on your team do not fit into the categories above, be sure to add new lines for them in the TCO budget before finalizing.

Table 5. Calculating Total Cost of Ownership (TCO)

	Cost Category	One Time Costs			Recurring Costs					
		Unit Cost	Quantity	Total Cost	Yr. 1	Yr. 2	Yr. 3	Yr. 4	Yr. 5	Total
1	**Computing, Telecommunication and Network Hardware**									
	Servers + OS + storage rack mounted includes rack and cabinet									
	Fixed PC Workstations									
	Mobile computers (laptops, netbooks)									
	Networked Scanners + OCR Software									
	Printers									
	Network and data communication electronics (routers, switches)									
	Data communication device (phones, tablets, etc)									
	Solar chargers									
	Protective cases (ruggedizing, security)									
2	**Packaged Software**									
	Application Software Licenses									
	Databases (Server or Processor Licensing)									
	Utilities (antivirus, system management, remote management, etc)									
	Reporting & Analytics									
3	**Custom Software Development**									
	LMIS application development									
	Software interface development (HMIS, WMS, other interfaces between systems)									
	Software Hosting									
	Mobile Application Development									
	Localization/Translation									
	Travel for Software Development Team									
4	**Implementation**									
	Hardware installation									
	Hardware acceptance testing									
	Cable and network installation									
	Software installation									
	Software user acceptance testing									
	Overall System Testing (ensuring all hardware, software, interfaces, networks perform to specification)									
	Data migration and conversion									
	Technical Training									
	User Training									
	Travel expenses									
	Documentation development and translation									
5	**Support & Maintenance**									
	Application Support									
	Hardware Support									
	Software maintenance and upgrades (updates, upgrades, etc)									
	Hardware maintenance									
	Field support and operations costs (help desk support, field support, province or district IT support)									
6	**Network and Communication Services**									
	Central Data Center Install									
	Central Data Center Services									
	Data Center Support related staffing costs									
	Data services (GPRS or other data connections)									
	Voice services									
	design and optimization services									
7	**Program Management and Support**									
	Program Manager / Project Manager									
	Hardware and software engineering									
	Training and implementation support staff									
8	**Technical Assistance**									
	International (include time and travel)									
	Local (include time and travel)									
	Totals									

Some of the human resources you will likely need to include in your TCO are:

- Program Manager
- Project Manager
- MIS Advisor
- Business Analyst
- Coder(s), Programmer(s), or Developer(s)
- System Architect
- Testers
- Trainers
- Application Administrator
- Database Administrator
- Server/OS/System Administrator
- Deployment Team

Engaging Software Developers

Section Three: Engaging Software Developers

Activities

- Select and contract vendors
- Communicate during the build
- Test functionality of software
- User acceptance testing
- Develop training documentation

Outputs

- Test plan
- RFP/RFI
- SRS
- Software user guide
- Service-level agreement
- Software release plan
- Change control plan
- Deployment plan

Regardless of the chosen solution, Engaging Software Developers will be necessary to construct or tailor the automated LMIS. Expectations of what IT specialists will do, the titles that are typically used, and the responsibilities they have are covered in this section. This section will guide the program manager through the selection and contracting process for developers. It will also cover the steps of the development process that the program manager will be responsible for, including communication with developers and users, overseeing the functionality testing, and the overall approval of the system. It is critical that the program manager treat this phase as a partnership, not an outsourcing relationship, to ensure that the IT solution will meet the needs of the users.

Select and Contract Vendors

Once the development plan is in place, the next step will be to find a vendor to help meet the requirements and develop a product. Choosing and working with software developers can be a challenge, but having good requirements in place can improve communication between the developers, the end users, and the managers involved in the project. There are a number of questions about using software developers, and this guide attempts to answer a few of them.

What can I expect a software developer to do?

The short answer is – it depends. What follows is a comprehensive list of what IT specialists can do, and, depending on a number of factors, including – but not limited to – funding, timeframe, and ownership goals, the project manager can decide what the IT specialists should be responsible for. Ultimately, however, the project manager and/or a knowledge expert should plan on taking an active part in all aspects of the work because while IT specialists will know "how" to create the desired application, it is the project manager's role to ensure that input is given throughout the process on "what" the application does. Also, as the program manager, you will be responsible for ensuring that all aspects of the project are successful.

IT specialists can and should be involved in all stages of the project, but they do not necessarily need to be the ones guiding or leading the process in the early stages of getting started and planning.

Being able to speak the language of the users while also understanding how to translate user needs into language that will make sense to software developers is very important during the planning stages of your automated LMIS project. However, not all IT specialists are able to communicate with both users and software developers in clear and understandable language, so do not assume that just because someone is an IT specialist they are the right person to work with users on requirements gathering or other design issues. IT specialists may need to be paired with subject matter experts in order to provide the most value during the design stages of the process.

IT specialists (either in-house or outsourced) are most commonly used to:

Software User Guide

- Facilitate the requirement gathering to create functional requirements.
- Develop the technical design for the solution.
- Develop the test plan for the solution, which would include system testing, user acceptance testing, and any applicable stress tests.
- Code the solution.
- Create and disseminate the operations and administration manual.
- Create the support and maintenance plan.
- Create a cutover plan for migrating data and planning the change from the old system to the new system.
- Create a software user guide.

As seen in the listing above, IT specialists can do much more than just the traditionally conceived part of "writing the code." The advantages of involving IT specialists in initiation and planning include their ability to understand the bigger picture, context, purpose, and goals of the initiative, as well as having continuity in the design of the system. A disadvantage is the cost.

Whatever the role for IT specialists, it is important to communicate expectations early. (If you are outsourcing this role, the IT specialists' involvement would start at the request for proposals [RFP] level.) In addition to the expectations setting, it is critical to clearly lay out expected deliverables, with the level of detail required in each deliverable clearly outlined from the onset. Without this level of specificity, the contractor can misunderstand the level of effort required, and when bidding, may not properly budget for the work.

How do I find a vendor or software solution to meet my requirements?

Depending on the funding source for your project, you may have specific obligations regarding how you choose a software solution, software developer, or the other resources needed to implement the requirements you have developed for your automated LMIS. This guide cannot address all these different obligations and policies, so the authors encourage you to be sure you understand your contractual requirements.

RFP/RFI

This guide will focus on the steps to take in a competitive bidding process since that is a common method for finding a software vendor or developer in many countries. The process often begins with a request for information (RFI) being issued to collect material on relevant topics before bidding starts. The request for proposals (RFP) is the document issued to solicit competitive bids, and includes all the appropriate materials required for vendors to adequately respond to the project.

Preparation for Competitive Bidding

If your organization has determined that the best way to proceed is through a competitive bidding process, the next step is to create the documentation that will be needed to notify potential vendors about your project and to describe the work needed. There are a number of steps that need to be taken to prepare for a competitive bid process.

1. *Draft a statement of the work to be performed.* It is crucial that your work description is as detailed as possible, drawing on the requirements gathering process and the use cases developed by your team. Items to be included in the statement of work include:
   - Functions the software must perform
   - Outputs (reports and graphs) that the software must produce
   - Hardware, operating systems, other software, and network that the software must work on or with
   - Number of users to be supported, including the number of concurrent users
   - Deployment plan required (installation, setup, testing, data migration, and training)
   - Special requirements (language, accessibility, and interface with other software)

2. *Determine selection criteria.* To be transparent, it is important to list the criteria by which bidders will be evaluated in the RFP. On the basis of these criteria, a more detailed list should be prepared for evaluation and should include the following categories:
   - Ability of the vendor to meet the user and system requirements specified in the RFP
   - Size and qualifications of the team members who will be working on the project, including how up-to-date their training is on new technologies or new approaches to software development
   - Vendor's experience with developing and implementing systems with similar requirements, scale, and in similar environments
   - Vendor's experience in the programming language proposed for the system
   - Vendor's process for working with the client on development, testing, and deployment This may include the vendor's chosen software development methodology, which is described in more detail in the Communicate During the Build section of this guide.
   - Total cost for purchase and installation
   - Post sales/installation support (service, response time, and cost)
   - Direct references from other clients about their experience working with the vendor

For each category, the selection committee for the RFP should specify the minimum acceptable criteria, additional desirable criteria, and scores for both the minimum and desirable criteria. In addition, the committee should identify which criteria are essential, so any package that fails to meet them will not be a candidate for selection.

3. *Select a bidding process.* Select how the bidding process will take place: open local competition, open regional competition, or short-listed competition (local or regional). The selected process must be consistent with the procurement rules in effect. If companies are short-listed, this can either be through a formal expression of interest and evaluation process or through a more informal market survey. The documentation needed to justify the selection of companies will be determined by the procurement rules, but a large pool should be selected to guarantee a reasonable level of competition.

4. *Determine dates.* Select the dates for the various steps along the award process, including, but not limited to, the deadline for questions, when a bidders' conference will be held (if applicable), when proposals are due, and when decisions will be made or a short list will be selected.

5. *Develop a submittal process.* Describe where, how, and when you would like proposals to be submitted. For example, some agencies require several copies of the proposals to be submitted along with separate technical and cost proposals, CDs of the proposals, and do not allow email submissions.

6. *Form a selection committee.* This committee should include the program manager, at least one person with enough IT experience to judge the technical quality of the vendors' proposals, and a member of the funding agency, if funding is not coming from the program itself. Others can be added, as needed, but avoid adding too many people or anyone who will not be able to technically or programmatically judge the quality of the proposals. Committee members will identify and evaluate candidate packages and select one package for implementation.

7. *Observe standard clauses.* As mentioned previously, depending on one's funding source and organi¬zational requirements, certain standard clauses and procurement processes must be followed. This should be coordinated with your organization's procurement and contracts officer.

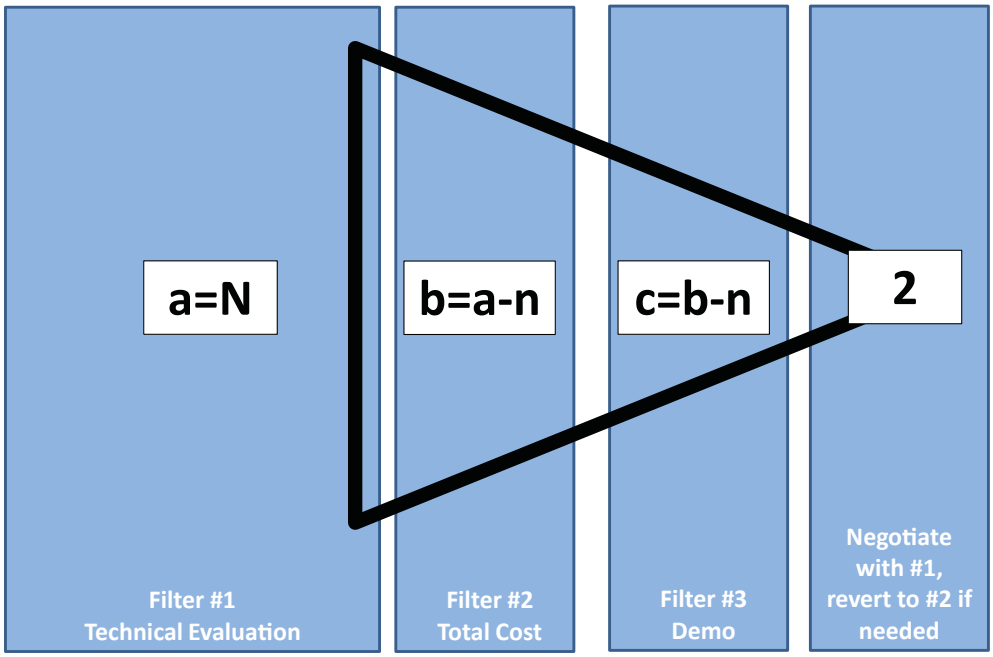
8. *Host a bidders' conference.* Holding a bidders' conference has many advantages. It allows the man¬ager to determine whether potential vendors understand the requirements, allows for clarifica¬tion of any unclear portions of the RFP, and can contribute to higher quality proposals. A bidders' conference should generally be organized within a week or two of issuing the RFP to allow the vendors time to review the proposal and formulate questions but not so long after the issuance that vendors will have invested a lot of time in preparing a response.

9. *Plan for responding to questions.* In general, the RFP process should include a period during which vendors can submit written questions. The manager or committee should then provide a copy of all questions and responses in writing to all those who collected the RFP. The only questions that should be answered are those submitted in writing, and the answers should go to all vendors to avoid any issues of procurement integrity.

Process of Evaluating Offers

In a competitive bid process, any offers received by the deadline are evaluated. The evaluation is usually a formal process that includes the steps described on the next page. The goal of the evaluation is usually to select the most technically sound proposal that presents the best value to the program. Price is important but not the most important criterion.

Figure 8. Vendor Evaluation Protocol



1. *Verify that proposals have met the requirements listed in the request.* If the proposal is not complete, it should be rejected.
2. *Convene the review committee and review the proposals that are complete.* The committee normally reviews the technical proposals first and scores the proposals with the evaluation criteria agreed on during the RFP process. Typically, the committee members meet and agree on a consensus score for each vendor and then rank the vendors by these scores. If the committee cannot meet together in one place, an average score made up of the individual scores of each reviewer can be used instead. During the evaluation process, the committee can choose to submit written questions for clarification to one or more vendors. All communication with the vendors should be done in writing or via email and the documentation saved in case questions of interpretation arise.
3. *Score the cost proposals.* If the cost proposals are submitted separately, they should be opened and scored separately from the technical proposals. This may be done by the same committee or be limited to a subset of its members with financial expertise. The same consensus or averaging process as described above should be used. The cost scores are added to the technical scores to derive composite scores and a final ranking.
4. *Request demonstrations of vendor products.* Some vendors may offer a custom-built solution, while others may offer a COTS package that can be customized. For the COTS packages, if desired, the committee can request demonstrations to better understand what functionality already exists and what modifica¬tions will be needed to meet the program's needs. Prior to the demonstration(s), the committee should develop a list of questions for the vendor(s) to answer and the features they would like the vendor(s) to display. The committee may want to document each vendor's responsiveness and suit¬ability after the demonstrations to use as part of the final technical evaluation.

5. *Interview references.* References can be a very important part of the proposal review process, especially for software developers who do not have a product similar to what you are looking for available for a demonstration. Look for references that have undertaken projects of a similar size and scope with the vendor, and have a list of questions to ask each one. Topics to cover should include technical quality of the product produced, the vendor's communication style and process, any problems or challenges, the quality of the implementation, the vendor's responsiveness, and the reference's overall satisfaction with the vendor. Reference checks are also typically undertaken by your final vendors.

If possible, it is also a good idea to perform a trial phase with two top vendors to see how the requirements match up to the product. After the trial phase, it is good practice to follow up with customers to demonstrate the products against the requirements before proceeding with the software acquisition.

SRS

The customers should make the final selection. At this point, the project manager and the IT specialists will have to write a technical specifications document – often called a System Requirements Specification or SRS – that matches up against the business requirements. The purpose of the technical specification document is to demonstrate the IT requirements for the product, how the product will meet business requirements, and what business requirements cannot be met or can only be met partially. Before beginning a major development phase, be sure to go back to the sponsors, MOH, donors, or other representative IT parties for agreement on the specifications and plan for moving forward. This phase will also require an updated project schedule outlining the full development schedule, resource requirements, and commitment from involved parties.

What happens once I have chosen my preferred vendor or developer?

To avoid problems at a later time and to ensure that they are getting exactly what they paid for, program managers should request a Service-Level Agreement (SLA) before actually signing any contracts. Only when a vendor is happy with the terms in its Service Level Agreement should it continue with the automated LMIS development process. Once you have chosen your preferred vendor, you will need to develop a service contract that both you and the vendor sign. This is an important agreement document because it clearly defines what the vendor is offering and what consequences it will face if it fails to deliver these services as required.

There are several stages in the development of software that you and the builder of the automated LMIS should agree on that deal with where you are in the process and what you expect to happen next. These stages are marked by a deliverable, which should be reviewed thoroughly by someone who understands what is desired in the end. Attention to details at every stage is important, as the next stage builds upon the information contained in the previous stage. In addition to development, these deliverables often are linked to payment schedules in the contract.

Both parties should also agree on the vendor's work location. If the vendor outsources the work, the IT specialist can work at the vendor's office, the project office, or the.MOH/CMS (user of the system). When working with a vendor that is located in a different geographical area from the client, e.g., in another country, it is important to keep a few things in mind. There are a number of conse-

quences when dealing with availability agreements or other types of agreement clauses. For example, some vendors may have a clause stating that they guarantee the availability of software development support during business hours. However, a vendor should be clear about what exactly is meant by the phrase "business hours." For example, does it mean that the vendor's services will be available during the client's business hours, or does it mean that its services will be available during the vendor's business hours? Additionally, if the vendor has any vaguely written agreement clauses, program managers should clarify what these clauses mean. If program managers are not happy with the clauses, they should negotiate their own agreement clauses that better match their needs.
The contract should also include a description of ongoing support costs and the process for quickly addressing bugs or system problems that may appear during or after deployment of the system.

Be sure to understand the issues surrounding software licensing. Many software vendors have an annual license fee. If the fee is not paid, the software ceases to operate. Other software companies also offer country-specific software licenses that do not allow the software application that is being developed to be shared outside the country. Other vendors also develop proprietary systems whose accompanying licenses do not allow software code to be owned by the country. You should understand the licensing rules before choosing a vendor, since they will impact country ownership of the automated LMIS.

After awarding the contract, the LMIS development must be managed proactively to avoid problems. The next section of this guide talks about how to ensure good communication with your software development or customization team during the development and testing process. Again, please note that this should not replace any contractual guidance or regulations required by your organization or donor.

Communicate During the Build

Communication during the build is very important. A project plan and schedule can include the frequency with which this communication will occur. There are two different communications the program manager needs to focus on. The first is the communication with stakeholders and project sponsors, and the second is the communication with the vendor or software developers. Both of these communication channels are equally important to the success of the project and should continue through testing and deployment of the automated LMIS.

Communicating with Stakeholders

Throughout the development phase, it will be critical to continuously check in with the customers and sponsors at many points throughout the software development life cycle – updating them on progress as well as any unforeseen challenges. Some methodologies of software development, such as agile software development, which will be described later, include frequent user testing as a way to elicit user feedback throughout the development cycle. This will ensure that customers are not surprised by the end result or that the project has not gone in the wrong direction or was developed incorrectly. It is much better to catch these things while development is still going on – the impact on the timeline will not be as bad and the perception of project success by customers and sponsors will be much higher. Ultimately, it is best not to have any such challenges during the development process; however, it is probably not realistic to expect that you will not have any problems. That is why the program manager is on the team; your job is to work through such issues and ensure that project is still completed on time.

Communicating with the Software Developers

The amount of communication and the methods of communication will likely be defined by the method of software development of your chosen vendor. Although there are many software development processes with different names, there are two main categories of development that are covered in this guide: waterfall software development and agile software development. Each of these methods and their impact on communication needs during the build process are described in more detail below.

Waterfall Model. In the waterfall model of software development, the development process begins with requirement analysis and then flows through other phases of development: design, implementation, testing, integration, and maintenance. With this model, the communication between the users and the software developments may be concentrated into a few phases. There is heavy communication at the beginning of the development process to ensure that requirements are well defined and clear to the developers. The developers then move forward on development through a specific phase, interacting with the users again once a full deliverable has been reached. Successful waterfall development depends on the users being very clear about their requirements at the start of the project, as well as on the software developers' confidence that they understand the requirements given to them at the beginning of the development process. Waterfall development is often criticized as not being flexible enough to adapt to the real world environment, where full requirements may not be known at the beginning.

In the waterfall model, the program manager must be able to spend a lot of time at the onset of the project ensuring that all requirements have been clearly communicated to the developers before software development starts. A challenge with the waterfall method is that it may be difficult to keep stakeholders and users updated on the project's progress during the development process since there will be few chances for them to view the product until a large development milestone has been reached.

Agile Development. The agile development approach focuses on breaking the larger project down into smaller deliverables that can be achieved within a compressed timeframe – usually between one and four weeks – with frequent communication within the team and with users on each deliverable. At the end of an iteration, a working product (not necessarily fully featured) is presented to end users for testing. The product is not considered ready for market release, but it should be functional and have minimal defects that need to be fixed. Following multiple iterations, a portion of the system is considered suitable for release, and developers move on to the development of the next portion. Although having your requirements developed in advance is still important in an agile development process, this process allows for user feedback to guide and refine requirements throughout the software development process instead of waiting until a major milestone has been reached.

Agile development, by definition, involves regular and consistent communication between the software developers and the end users. As the program manager, managing both sides of this communication to ensure that feedback is given on time – in a consistent manner, and in a language that will make sense to the software developers – will be a large part of your responsibility during the building process.

Software Development Project Management Tools

Regardless of the methodology chosen, it is important to use a web-based project management software to track the progress of software development, issues and bugs, and resolution of any issues within the core project team. There are a variety of uses for this type of software, including allowing multiple programmers to work from the same source code, communicating the reason for code changes and additions to code, providing a space to evaluate and approve code, and having a central repository of all approved code. In addition, most software development systems include a method for tracking and reporting bugs in the software or functions that work differently than expected.

The three major functions of this system (or systems) are to store source code, track bugs, and communicate about any software changes or bugs.

1. Source code control system
   A source code control system stores the code base in a central repository. It allows individual developers to check out source code files to work on. Developers can lock files in the repository to prevent others from working on the same file at the same time. When multiple developers are working on unlocked files, the repository applies rules to track which changes were made by which developer. When a source code system is used, a typical development scenario includes:
   - Development manager creates a module in the source code repository.
   - Programmer writes initial code and checks in the file. The original programmer or other programmers can check out that code, do further work, and check back the code. The repository handles conflicts related to the same code being updated by more than one person.
2. Bug tracking system
   Throughout initial development and into software maintenance, bugs should be identified by programmers, testers, and other users. It is important that all participants report bugs in a central location so there is little duplication of effort, documentation of the steps taken to address defects, and prioritization of bug fixes.
3. Communications
   One of the benefits of formalizing a source code control system and a bug tracking system is so that communication among software developers and between developers and clients is documented. Some of these systems allow for users to add comments and questions so when new code is released, the developers can insert release notes and so clarification of bugs and communication of the schedule to address them are clearly documented.

There are a variety of options for project management software, including free and fee-based solutions. A sample, but non-exhaustive, list of options is included in Table 6. It is likely your chosen vendor will have a preference for a specific project management tool software, and it is best to let the vendor use the tool it is most comfortable with. As the program manager, you should ask to be included on the list of users for the project management software, so you can also track the progress of the software development and any issues that come up. Project management tool software tends to be more technical in nature and likely will not be appropriate for all stakeholders to have access to the details.

Table 6. Tools for Managing Software Development

Tool	Source code repository	Communication tool	Bug tracker	Cost
Unfuddle	X	X	X	$
Github	X	X	X	Free for open source, otherwise $
SourceForge	X	X	X	Free
Bugzilla			X	Free
Subversion	X			Free

Test Software Functionality

Testing the LMIS

Testing software before using it is a critical activity in developing and implementing a computerized LMIS, but it often is not given the full resources needed. In many cases, testing is done informally by the developer as each function of the LMIS is developed or enhanced. This informal approach is not adequate to ensure that the LMIS meets the client's objectives. To ensure software quality, a more formal and rigorous approach to testing, ideally with users of the system, is required. Involving a group of testers is another way to continue to engage stakeholders throughout the development process.

Managers should plan on providing the resources needed to conduct testing. At a minimum, it is recommended that the manager have users test the system using specific testing "scripts" built off the original use cases and requirements developed for the system. In an LMIS, this would likely include entering and submitting an order, running a report, reviewing and approving an order, and making changes to an order that has already been entered into the system. It would also include testing administrative functions such as adding new users, changing user data, adding or editing a facility and changing user permissions.

A test plan and detailed testing scripts should be developed by the program manager or project manager and the software developers to ensure that all key functionality is being tested and to ensure that the data in reports match the expected values and outputs.

Test Plan

A formal test plan can be developed for each module, form, report, or functional area. This plan can outline all sorts of tests to be performed. It can contain test cases, test data to be used, and expected outcomes. A testing script provides instructions on how to test a particular system function and provides detailed steps. A test case also ensures that the test is repeatable and reproducible.
By systematically going through testing scenarios that mimic daily use and exceptions together with the software developers, managers and users can better articulate and review the way the system should act.

During testing, testers focus on verifying that the software does what it is designed to do. Other major goals of testing include:

- Uncovering defects (bugs) in the software.
- Ensuring that the software does not do what it is not supposed to do. Often this is called "negative testing." An example would be to verify that the software does not let a user record a negative value in a field for existing stock.
- Creating confidence that the software performs adequately.
- Understanding how far a system can be pushed before it fails. Examples of this include the number of users that can use the system simultaneously, the number of products/facilities/records that can be entered or stored, etc.
- Understanding the risks involved in releasing a system to its users, such as hardware constraints or potential misuse of the system.

There are some tests that can be done by users and other tests that need to done by the software developers. As program manager, it is your responsibility to ensure that all relevant tests are being done and that results are documented and available for review. Before the automated LMIS is released for use, it is vital that someone other than the developer performs the tests. Not only does this serve as a quality check on the developer's work, but it also enables testing from the users' perspective, with the testers as surrogate users. In large software projects, a dedicated team of testers may fill this role. In smaller projects, the tester may be another developer, the program manager, or the users themselves.

Frequent Communication During Testing

As mentioned in the *Communicate During the Build* section, the testing process requires frequent communication between users, testers, and developers to report and fix bugs and to verify fixes. The tester identifies bugs or issues and reports them to the product manager or developer. The need to track individual bugs through a sometimes lengthy testing process makes a computerized incident tracking tool – for example, a database – especially useful. Such a tool enables testers, developers, and the product manager to monitor the status of individual bugs as well as the testing process itself. The software development system used by the developer to track software development progress and to track and address bugs will likely also be used to track test results.

The project manager, with input from the development team, prioritizes defects that are most important to address. The developer then fixes or addresses the defects and reports to the tester to test the fixed version. Often, there is additional back-and-forth communication between the tester and developer before a defect is completely addressed.

Perform User Acceptance Testing

The last step in the testing process is user acceptance testing, which also leads to overall product approval. Acceptance testing is performed by one or more user representatives to confirm that the software works correctly and is usable before it is formally delivered to the end users. During user acceptance testing, users try the system by performing typical tasks that will be carried out during the course of their workday. Wherever possible, to simulate real usage, actual data should be used for testing. User acceptance testing should also include reviews of any associated documentation, such as user manuals. This stage in the process is also a good time to check in with stakeholders to involve them in user acceptance testing to ensure their continued engagement and support.

User Acceptance Document

As the program manager, you have the responsibility to ensure that the software undergoes user acceptance testing before it is considered approved and ready to implement. The program manager also has the right to review detailed information on the overall testing process: the testers, test plans, and test results. If the development team cannot provide this information, it is an indication that a systematic testing process is not in place. In that case, they may need to plan for more extensive user acceptance testing before the software can be delivered to end users. In order to ensure that testing has been completed by appropriate stakeholders, a user acceptance document should be created and signed off on when testing is completed and the product is considered satisfactory.

A final note on testing: All newly developed automated LMIS – even the most well designed and thoroughly tested – contain bugs that users will uncover during operations. After the automated LMIS has been implemented, the mechanism for tracking bugs should remain in place; this will allow both users and developers to report and fix any bugs that are uncovered.

Prior to the implementation phase, the project leadership team needs to make decisions on the deployment plan for the software. In most cases, a phased roll-out is necessary due to distributed users, limited resources, and the need to ensure that the system is working effectively before a full deployment takes place. In some countries, it may make sense to first roll out the new LMIS at the central level with a small number of users at the national and warehouse level first. Once these users have adapted to the system, further deployment to users at other levels of the health system could be implemented. Decisions on deployment plans should be made collaboratively with the project leadership team, and with the sign-off of the project sponsor.

Develop Training Documentation

When the development phase is complete, it is important to document not only how to use the product, but also how the new automated LMIS will impact the functioning of the in-country logistics system.

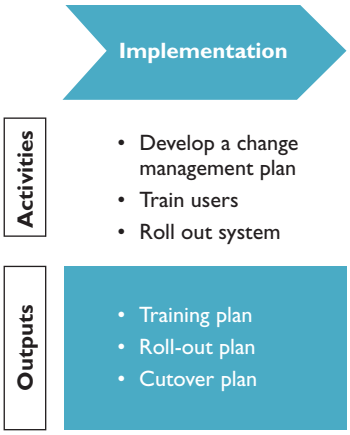
Outputs from Software Developers
- **User guide**
- **Technology guide**
- **Implementation and deployment manual**

The software development team will likely be responsible for large portions of the technical documentation regarding the new automated LMIS. Ensuring that a user guide, technology guide, and implementation and deployment manual are specified outputs in the software developer's contract is key to ensuring that these products are ready for implementation.

Documenting how the new LMIS will impact the function of the in-country logistics system is a responsibility of the program manager. This will require discussion with users, donors, the MOH, and implementing partner representatives about what the system will now do and what the new process should look like. It is important to have these changes documented in writing and to be in agreement with all stakeholders prior to any product rollout occurring. With expectations set appropriately, a much smoother training and rollout phase of the product can be expected. Without a carefully planned training and rollout phase that includes considerations not just for software

training but also for training on the changes to the overall system, the success of the project can be jeopardized.

As the program manager, this is also a good time to ensure that you have documented all lessons learned during the process of software development. This includes lessons learned from the requirements process, vendor selection, and the development process. Although lessons learned is an output from the final phase of this SDLC process, it is important to be tracking the lessons learned throughout the process.



Section Four: Implementation

When the automated LMIS has been tested and works to satisfaction, a program manager may ask, "Now that I have a system, what next?" Implementation begins and introduction of the system to users takes place. In some instances, a pilot helps inform managers what other enhancements are required to the LMIS, and after they are made to the software, a larger roll-out plan will be executed. If the automated LMIS is replacing another system (whether paper based or legacy system), the cutover plan will articulate the steps needed to make a smooth transition, including change management, training, and deployment options. This section will guide the program manager through the steps toward ensuring a successful implementation of the automated LMIS.

Develop a Change Management Plan

Change management is a systematic approach that a program manager or project manager will need to oversee to deal with the change the automated LMIS will bring, both from the perspective of an organization (e.g., the MOH) and on the individual-user level. Change management has at least three different aspects, including adapting to change, controlling change, and affecting change. A proactive approach to dealing with change is at the core of all three aspects. For an organization such as an MOH, change management means defining and implementing standard operating procedures and technologies, such as the automated LMIS, to deal with changes in the supply chain environment and to benefit from changing technologies.

As a program manager, planning for change management is an important part of your role. This includes aligning expectations of all team members and stakeholders throughout the software development and implementation processes, effectively communicating changes, and designing strategies to address potential challenges that the implementation of a new system may bring if not effectively managed. The active change management process may begin at implementation, when a new system is being rolled out, but as a program manager, you will be thinking about change management far before implementation begins. The type of change the implementation of the new system will bring to users and stakeholders needs to be considered early in the planning process, so training and implementation plans are prepared to respond to the operational and technological changes that will come as a result of the new system. Training is a key part of change management, but it is not the sole

solution. An effective change management strategy will combine early expectation setting with good communication and strong training and support resources that anticipate user and stakeholder needs.

Successful adaptation to change is a critical success factor to any implementation. The more effectively you and the stakeholders deal with change, the more likely the project is to be successful. Adaptation may involve establishing a structured methodology for responding to changes in the logistics environment, such as the use of data for supply chain decisionmaking or establishing coping mechanisms for responding to changes at various levels of the health system, such as automated technologies.

Train Users

Shortly before a new or enhanced software application is delivered, the intended users must be trained on how to operate it. The training plan will be highly dependent on the deployment strategy. Training can either be carried out on the job in structured sessions with users in their work environments or through classroom-based trainings with groups of users.

Training Plan

If configuration of computers in the workplaces is needed to run the software and if the number of people to be trained in each site is small, on-the-job training by a member of the development team or by a person specially trained by the development team may be the most suitable type of training. If on-the-job training is used, it is important to ensure that competency-based training principles are included, for example, create exercises that test the learner's ability to navigate the application, enter typical data, and find data that answer a particular question.

An important part of the training process is to ensure that program managers, officials, donors, and policymakers are also trained on how the software's data can be used to guide and support key decisions on product selection, procurement, resource allocation, and financing. Not taking this step may result in the software failing to influence major logistics decisions.

In many cases, users who will use the data to inform regional- or national-level decisionmaking do not have the time or inclination to attend a traditional, competency-based training session in a classroom. A viable training approach for these users is to present outputs of the software's data – reports, charts, and graphs – at regular meetings or forums in which key logistics decisions are being made. These meetings will vary by country but may include quarterly (or annual) meetings of the logistics coordinating committee, donor coordination meetings, and gatherings of high-level officials to review or change national policies governing health care.

During the roll-out and training phase, it is extremely important to communicate what users need to do if they need help with the product. During the aforementioned phase, a good project manager will have this plan in place and be ready to put the support process into motion. It is also important that you have customer groups agree on the support process.

Roll Out System

As the program manager, your responsibility is to ensure that the project team has produced a well thought out deployment plan. The checklist outlines some deployment considerations. The program

manager is not responsible for completing all the tasks but is in charge of ensuring that all the appropriate steps are being taken for a successful roll-out. One of the tasks during the roll-out is to move data from an existing system, whether paper based or electronic, to the new system. The plan for doing this data migration is called a cutover plan and is an important part of the process of rolling out a new system.

Rollout Plan
Cutover Plan

Table 7. Deployment Checklist

Deployment Type Activity	
Quality assurance	Ensure that the application is ready for deployment: • Review all test results and verify that all agreed-upon bugs have been fixed. • Verify that the application was tested under a configuration representative of the actual production environment. • Ensure that installation and configuration steps are documented by project team
Data	Review data migration needs for existing data: • Review back-up plan for existing data • Ensure team is ready to convert existing data to new system, as needed
Roll-out	Plan for actual roll-out: • Review list of sites and PCs where the application will be installed. • Review installation instructions for users and trainers. • Inform all concerned and give adequate notice about date and time and any potential loss of service. • Oversee creation of the roll-out package: release notes, brochure, user's guide. • Ensure that managers understand in advance the changes that will take place and the benefits of these changes. • Ensure that resources are in place to install, test, and conduct data migration at each site, as required.
Training	Plan for training needs: • Develop a detailed plan for training. • Determine training objectives. • Identify groups of people to train and suitable level of training. • Develop training materials. • Conduct training.
User support	Determine what type of user support should be available: • Develop a support plan. • Consider vendor-provided support during post-production period. • Ensure that users are aware of support options and have contract information. • Put in place a mechanism to record, respond, and track all support incidents. • Ensure that qualified and trained personnel are available to respond to support calls. • Put in place a mechanism to track support call volume, peak levels, average response time, incidence of emergency calls.

Maintenance

Section Five: Maintenance

Activities

- Track bugs
- Continuously evaluate
- Identify enhancements
- Plan for ongoing technical support

Outputs

- Bug tracker
- Lessons learned

Regardless of how well designed the automated LMIS is, there will be problems and enhancements identified by users regarding the performance of the new automated LMIS. Since the resources needed to maintain and support the system were already identified and discussed, the Maintenance section of the guide focuses on the role of the program manager after the system has been implemented and deployed. It is important to continue tracking the success of the system over time, addressing bugs or problems that arise and tracking user needs for future updates or upgrades to the system. A well supported and successful system will be one that is regularly improved based on user experience and evolving stakeholder needs.

As stated in the earlier sections of this guide, it is important that the TCO of the system is well understood by those who are supporting the development of the system, as well as by those who are going to be responsible for maintaining it. These funds need to be available and committed before you move forward on an automation effort. Securing funding only for the development and then hoping that the funds for maintenance, ongoing support, and upgrades will be available in the future will put the project's success at risk. Through the TCO model in Section 3 of this guide, you have already led the group through an exercise to determine the ongoing maintenance and support needs of the project.

Track Bugs

Regardless of how well designed the new automated LMIS is, there will be problems identified by users regarding its performance. These problems will likely fall into one of three categories:

Bug Tracker

- User errors that can be addressed through software support or additional training
- Bugs, which are faults in the software that prevent or hinder users from efficiently performing their tasks
- Enhancements and new features suggested by users or stakeholders to improve or expand software performance

As the program manager, you will want to make sure all three of these problems are resolved. Section 4 of these guidelines outlined a number of tools for tracking bugs or incorrect functionality that are discovered during software development and testing. These same tools can be used to track issues or bugs raised by users, the efforts that have been made to address the bug, and the eventual resolution. Using a bug tracking system will be very important to the ongoing maintenance of the system, and, as discussed in Section 4, there are a number of free or low-cost tools that can be used to maintain a running list of issues.

Different types of identified problems will require very different responses from the developers, depending on the severity of the issue reported. Some defects may need to be fixed as soon as they are reported – especially any critical defects that prevent users from performing an essential task; others can be grouped with the enhancements awaiting development of the next version of the software.

During deployment, make sure all users are aware of where to go for support when they run into problems using the software. As program manager, you should be keeping track of the issues users are running into and whether they are user errors that can be addressed by technical support and additional training, bugs that need to be addressed by software engineers, or new feature requests for future enhancements.

Continuously Evaluate

Once the new automated LMIS is deployed, a stage of continuous evaluation begins. The team should decide on regular points for review and evaluation of the system by users. It can be difficult to get input from dispersed users regularly, but some users will not be open about issues they are having unless they have a formal mechanism for providing feedback. Suggested timeframes for evaluation of the new automated LMIS are one to three months after deployment, six months after deployment, one year after deployment, and then every six months to one year after that.

You can solicit feedback from users through a survey, focus groups, or regular meetings. If users are dispersed, you can work through supervisors to obtain feedback on the system while doing site visits or during other meetings. Be sure to be very clear about the questions you want to ask of users so you get concrete, actionable feedback back from the field.

Formal feedback does not take the place of regular communications regarding user support needs or bug identification, but it can be a way to get more comprehensive information about the overall performance of the new automated LMIS.

The team that supported the development and deployment of the automated LMIS is probably the same team that will want to review the progress of deployment and the feedback received from users. This can be done through steering committee meetings or other venues where the team will come together, and a formal presentation on user response to the new automated LMIS, including user suggestions for improvement, can be discussed.

Identify Enhancements

As with any software development project, most automated LMIS implementation projects will always need to address the ongoing and never-ending tasks of fixing and enhancing the existing software application. A change control plan can help guide how the enhancements will be identified throughout the implementation. If you are using a COTS solution, these enhancements will be managed by the vendor and will be pushed out at various intervals under your support contract. If your country created its own automated LMIS or heavily customized existing software, you will be responsible for also managing the enhancement process with the help of software developers. At some point during the evaluation period, the project team will have developed a long enough list of items that need enhancement or changes that it will be time to consider combining these changes into a new release of the software. Regardless of the method of software development (COTS or custom built), the enhancements and changes must be prioritized by a key group of stakeholders.

Remember, when faced with enhancing the application with a significant upgrade, the system is really undergoing a new cycle of development. All steps described in these guidelines regarding development of the LMIS also apply to upgrades and should be followed accordingly. All the steps described in this document should be followed during subsequent releases to make good use of project resources and ensure that future enhancements are well designed, meet user requirements, and perform in the environments in which they are deployed.

Plan for Ongoing Technical Support

In addition to providing support for bug tracking and modifications, technical support is necessary for backing up the database to ensure smooth operation of the computerized LMIS and to manage user access. A program manager should ensure that technical support is in place to provide these services to the users for the life of the software use. For example, if the hard drive containing the database for an automated LMIS crashes, the data may be lost forever. One of the most important elements of technical support is ensuring that LMIS data are backed up regularly to avoid losing data as a result of a hard drive failure. Technical support providers should plan to regularly back up LMIS data on an external storage medium. In addition, like other information systems, automated LMIS may assign roles and privileges to different users as a way of protecting the integrity of LMIS data. Some users may only enter and update data, others may only view reports, and one or two users may have rights to change privileges of other users. Technical support personnel should put procedures in place for adding or inactivating users and assigning them roles.

Conclusion

Planning the design and implementation of a computerized LMIS is a complex and challenging project. These guidelines are meant to provide a framework to help program managers complete successful automation projects that meet user and stakeholder needs, are managed to budget and timeline, and are effective at improving logistic system performance.

Over the course of the USAID | DELIVER PROJECT, our team has had the opportunity to work with many ministries of health on designing and implementing successful computerized LMIS. As is discussed throughout this guide, automation is just one piece of a complex network of factors that make an LMIS effective and efficient. Over the course of the project and a number of LMIS implementations that range in automation, we have learned a number of important lessons that may be helpful to program managers as they embark on their own supply chain strengthening endeavors. Although these lessons learned are not all unique to automation projects, they are all important to creating the environment in which your automation project will be successful. Some of our key lessons learned include:

Create an Information Culture: A robust supply chain relies on accurate and timely data. An information culture fosters the collection of information and ensures better decisionmaking at all levels. An information culture requires reliable forecasting of product needs, use of data in decision-making, a well-functioning LMIS and efficient service delivery networks, and analytical support in procurement and supply chain areas.

Use a Systems Approach: It is essential to continuously analyze the supply chain system and improve the processes; programs can obtain the best results when they work with partners to correctly diagnose problems and determine the best way to improve the efficiency and effectiveness of the program. Successful programs also continue to introduce, in phases, appropriate innovative technologies, including web-based technologies; computerization of inventory management; and bar coding, to improve supply chain efficiency.

Be Careful of the Quick Fix: Software development projects that are designed to create solutions that will meet long-term requirements of a country's logistics systems take extensive time and resources. Often stakeholders want to see a solution happen quickly, but adapting existing solutions that were not designed for the specific requirements, scale, or scope of a national LMIS may actually take more time and effort. Although the quickest solution may seem attractive, make sure your team understands the actual level of effort that will be required to create a product that meets all requirements and can be scaled up to meet national needs. Often the quickest solution ends up taking longer and consuming more resources to get to the desired result than the well planned, but seemingly longer, alternative.

Keeping these lessons in mind and using the tools and resources available in this guide will provide you and your team with a strong foundation on which to build your automation project. As the program manager, you have the challenging but rewarding job of providing guidance to the project team, keeping the momentum going throughout the process, and managing the resources needed to successfully deliver the final product. We hope these guidelines will provide you with the support you need to lead your project forward.

References

USAID | DELIVER PROJECT. 2006. *Guidelines for Implementing Computerized Logistics Management Information Systems (LMIS). Second Edition.* Arlington, Va.: USAID | DELIVER PROJECT, for the U.S. Agency for International Development.

USAID | DELIVER PROJECT, Task Order 1. 2009. *Turning the Digital Corner: Essential Questions for Planning for a Computerized Logistics Management Information System.* Arlington, Va.: USAID | DELIVER PROJECT, Task Order 1.

For more information, please visit deliver.jsi.com.

USAID | DELIVER PROJECT
John Snow, Inc.
1616 Fort Myer Drive, 16th Floor
Arlington, VA 22209 USA
Phone: 703-528-7474
Fax: 703-528-7480
Email: askdeliver@jsi.com
Internet: deliver.jsi.com