

# Management Information Systems for Microfinance

## An Evaluation Framework



Widening the circle, moving ahead

### **MICROENTERPRISE BEST PRACTICES**

Development Alternatives, Inc., 7259 Woodview Avenue, Suite 200, Bethesda, MD 20814, USA



A USAID-funded project, implemented by DEVELOPMENT ALTERNATIVES, INC. in collaboration with ACCION International, Foundation for International Community Assistance, Harvard Institute for International Development, International Management Center, International Cooperation, Ohio State University Rural Finance Program, Community International, and the Small Enterprise Education and Promotion Network.

# Management Information Systems for Microfinance: An Evaluation Framework

by

Andrew Mainhart  
Development Alternatives, Inc.

November 1999

This work was supported by the U.S. Agency for International Development, Bureau for Global Programs, Center for Economic Growth and Agricultural Development, Office of Microenterprise Development, through funding to the Microenterprise Best Practices (MBP) Project, contract number PCE-C-00-96-90004-00.

*Andrew Mainhart, a Senior Development Specialist, has worked at Development Alternatives, Inc. (DAI) since 1998. He has provided technical advisory support to the CARD Bank in the Philippines, the Bank for Agriculture & Agricultural Cooperatives in Thailand, and ACLEDA in Cambodia. Mr. Mainhart has supported several USAID projects around the world including the Program for the Recovery of the Economy in Transition in Haiti, the Newbiznet project in the Ukraine, and the TFED project in Tanzania. In addition, he has spent time in Indonesia working with the information technology team at the Bank Rakyat Indonesia studying the bank's information systems.*

*Mr. Mainhart holds a B.S. degree in industrial management from Carnegie Mellon University. Before entering the microfinance field, he worked as a consultant with Andersen Consulting, implementing large-scale software projects for Fortune 500 companies and as a manager in the Information Technology Division of Bell Atlantic, a large U.S.-based telecommunications company.*

## ACKNOWLEDGMENTS

The author has benefited greatly from the comments of Robin Young, Matthew Buzby, Nhu-An Tran, John Magill, and Zan Northrip of Development Alternatives, Inc.; Anicca Jansen of the USAID Microenterprise Development Office; David Ferrand of the Department for International Development; Xavier Reille of Catholic Relief Services; Brigit Helms of CGAP; and Tony Sheldon, Todd Girvin, Nick Ramsing, and Ruth Goodwin. The author would also like to acknowledge the assistance of all those who participated in the MBP Information Systems in Microfinance seminar held in Washington, D.C., on March 8, 1999. To all those who provided comments, guidance, and other assistance, the author expresses profound thanks and gratitude. Of course, all views, omissions, and errors are attributable solely to the author.

## TABLE OF CONTENTS

<b>CHAPTER ONE</b>	
<b>INTRODUCTION</b>	<b>1</b>
BACKGROUND.....	1
PURPOSE.....	1
METHODOLOGY.....	2
USAGE .....	2
DOCUMENT STRUCTURE.....	4
<b>CHAPTER TWO</b>	
<b>THE FRAMEWORK</b>	<b>5</b>
CATEGORY HIERARCHY.....	5
RESEARCH MATRIX.....	6
RATING METHOD .....	6
<b>CHAPTER THREE</b>	
<b>CATEGORY 1: FUNCTIONALITY AND EXPANDABILITY</b>	<b>9</b>
<b>CHAPTER FOUR</b>	
<b>CATEGORY 2: USABILITY</b>	<b>19</b>
<b>CHAPTER FIVE</b>	
<b>CATEGORY 3: REPORTING</b>	<b>23</b>
<b>CHAPTER SIX</b>	
<b>CATEGORY 4: STANDARDS AND COMPLIANCE</b>	<b>27</b>
<b>CHAPTER SEVEN</b>	
<b>CATEGORY 5: ADMINISTRATION AND SUPPORT</b>	<b>31</b>
<b>CHAPTER EIGHT</b>	
<b>CATEGORY 6: TECHNICAL SPECIFICATIONS AND CORRECTNESS</b>	<b>37</b>

<b>CHAPTER NINE</b> <b>CATEGORY 7: COST</b>	<b>41</b>
--	-----------

<b>ANNEX A: RATING SCORE SHEET</b>	<b>A-1</b>
------------------------------------	------------

# CHAPTER ONE INTRODUCTION

## BACKGROUND

Over the past 5 to 10 years, microfinance institutions (MFIs) have been paying increasing attention to information systems, particularly management information systems (MIS). As both practitioners and donors have become aware of the great need for formal and informal financial institutions to manage large amounts of data, the drive to improve the manipulation and understanding of these data has grown.

Information lies at the very heart of microfinance. Whether by hand or by computer, microfinance institutions maintain large amounts of critical business data, from basic client information to detailed analyses of portfolio statistics. These data must be stored, manipulated, and, most important, presented coherently to system users so that they can make sound management decisions.

A good information system should do just that: It should act as a conduit through which raw data becomes useful and useable information. A good information system is a necessary tool for managing an institution successfully. This assertion, however, begs two questions: What is a good information system? And how does one determine whether a system is good?

The easy answer to both questions is any system that meets an organization's needs cost-effectively and allows the organization to grow without creating problems or inefficiencies is a good system. Obviously, this simple answer is not terribly helpful. For this reason, the need for an evaluation framework has become imperative, especially considering the microfinance field's relative lack of knowledge about information systems and software development.

## PURPOSE

The primary purpose of this paper is to present a mechanism for analyzing information systems, both those bought off-the-shelf and those developed internally. This MIS Evaluation Framework offers the industry a tool to determine the quality of an information system. The framework is very flexible and can be used by MFIs, donors, and other external stakeholders, as well as systems developers, to address different objectives:

- MFIs can use it to evaluate off-the-shelf systems in their search for an appropriate solution.
- MFIs can use it to appraise the quality of their existing system (off-the-shelf or internally developed) to help identify improvements.

- External entities can use it to evaluate off-the-shelf or internally developed systems either to assist an MFI, identify alternatives, or as part of an institutional appraisal.
- Software developers and information system planners can use it to build better systems.

A certain level of generality was used in constructing the framework to meet the needs of a diverse audience. Consequently, factors specific to the organization or vendor in question (for example, stage of development, growth prospects, and number and complexity of products) should be added on a case-by-case basis. Because the framework is a tool, the evaluator needs to understand the situation and apply the framework in the most logical and effective way.

## METHODOLOGY

Drawing from computer industry standards, such as DataPro, GartnerGroup, and Patricia Seybold Group,<sup>1</sup> as well as the experience of software professionals, this paper outlines and defines a set of categories that can and should be used to evaluate any information system used in microfinance. To ensure this framework incorporates issues specific to microfinance—particularly the variety of microfinance institutional models, products, and services—the author consulted the Consultative Group to Assist the Poorest (CGAP) *Handbook for Management Information Systems for Microfinance Institutions* and spoke with numerous microfinance professionals.

## USAGE

Any organization that decides to use this framework to perform a systems appraisal needs to be aware of several important factors:

**Team membership.** The appraisal team should consist of at least two people. The size of the team will depend on the depth of the review and the experience and skills of the team members. The following factors should be considered when formulating such a team:

- # *Information technology expertise.* The framework is written in an attempt to simplify the often-complicated world of computer systems; however, as with any technical field, background knowledge of the subject matter is a necessity for the reviewer. At least one member of the appraisal team should have knowledge of computers and software, especially software development and support. A sound understanding of the standard software development process (such as the one outlined in the CGAP *Handbook for Management Information Systems for Microfinance Institutions*) and information architectures is crucial to the performance of this type of review.

---

<sup>1</sup> These organizations publish detailed reports on various kinds of computer software, from operating systems to database management systems to banking software. Information technology specialists typically refer to these sources when selecting appropriate software for a given technical and/or functional environment. Generally, their reports offer in-depth, qualitative and quantitative analyses across a broad range of industry requirements and specifications.

- # *Functional expertise.* Since these reviews will be conducted on microfinance institutions, at least one member of the appraisal team should have expertise in microfinance. In addition, a general knowledge of both accounting and financial principles is a minimal requirement for performing any appraisal of a financial system.
- # *Institutional expertise.* When reviewing systems for use in a specific institution, there should be internal representation on the team. This team member would know the MFI's operations intimately and could replace the team member providing functional expertise. If the institution has an experienced systems person, the team could consist of solely internal personnel.

**Access to key personnel.** Reviewers will need to meet with both the users and developers of the system.

- # In the case of a study of an internally developed system, the reviewers should have ample time to discuss the system with management and the technical support staff, as well as field operators, such as tellers and loan officers.
- # When reviewing off-the-shelf packages, the appraisal team should meet with the vendor, including vendor management and technical support. Site visits are a must when evaluating vendor software. The review team should visit at least one organization using the system and question the users and technical staff about the system.

**Detailed business information.** Before undertaking a review of information systems, the appraisal team must have a clear understanding of the MFI's business processes. This requirement is equally valid for an MFI conducting a review of off-the-shelf packages or evaluating its own systems, as well as for an external review team evaluating a particular MFI's system. The appraisal team (internal or external) will need to work closely with the MFI's operations staff to understand the processes and methodology, as well as personnel, geographic locations, and, most important, future goals and growth plans. The review team will need to obtain documentation about the institution, including organizational charts, strategic and business plans, growth projections, operational structures and procedures, product descriptions, and personnel descriptions.

**System documentation.** The appraisal team should obtain copies of all manuals (such as user handbooks and administration guides), reports, screens (windows), and operations schedules (version schedules, backup schedules, and so on).

**Detailed system information.** The appraisal team will need additional information that is typically not available in the system documentation. The following information, which is more fundamentally technical, will provide the reviewers with a deeper level of understanding of the system:

- # *Database information.* Whenever possible, the reviewers should obtain a copy of the database structure (that is, how the data are stored in the form of tables, column, rows, and so on), usually found in a data model. If a formal data model is not available, then a

printout of the database structure will suffice. (The only major difference is that a data model shows the relationships between different pieces of information, which are especially important in terms of performance and flexibility.)

- # *Demonstration.* Whenever possible a demonstration version (or demo) of the software should be procured. Regardless, the reviewer should always have access to a fully functional version of the system.
- # *Source code.* Access to the source code<sup>2</sup> and the database (as well as the data stored within) is not necessary but highly recommended, especially for more in-depth analysis. Many vendors and even some institutions will balk at this request, so beware. As stated, source code is not a requirement, but will greatly assist the appraisal team.
- # *Testing.* To determine if the calculations performed by the system are accurate, it is valuable to enter actual data from the MFI and compare the calculations with expected results. In this case, access to the database would be crucial.

**Site visit.** To perform a detailed review, a two-person team will need at least three to five days on site, depending on the complexity of the system and the experience level of the team members. Allow an additional one to two days for off-the-shelf software when site visits are required. Of course, the reviewer will also need time to write the actual report.

Since the team will need to review a large amount of information in the course of the appraisal, it is best to request the above documentation well in advance. How quickly and efficiently the reviewers are provided with much of this information (especially system-generated or related information, such as reports and user guides) may indicate the system's and user's competence or completeness.

## DOCUMENT STRUCTURE

The remaining portion of this document describes the framework itself. The framework has been broken into distinct *categories*. Each category has been further broken into one or more *topics*, which are defined in a *research matrix*. The research matrix describes each category with all corresponding topics, including any subtopics, with a category and topic (or subtopic) definition, measurement criteria, and a rating method.

---

<sup>2</sup> When creating a software application, computer programmers use special instructions written in computer languages to tell the computer what to do and how to function. There are many different computer languages on the market today, including C, C++, Visual Basic, and Delphi. The collection of all the instructions for a given application is called "source code." Having access to the source code allows the appraiser to review in-depth the logic and soundness of a given application.

## CHAPTER TWO THE FRAMEWORK

A hierarchical listing of the framework's categories and their associate topics and subtopics is provided below for quick reference. A full description of each category is given in the subsequent research matrix, in which each category is defined and supported with in-depth information about each topic and subtopic. The research matrix is the basic operational tool contained in the framework.

**Table 1: Category Hierarchy**

<b>Functionality and Expandability</b>
<ul style="list-style-type: none"> <li>▪ Functional completeness, appropriateness, and integration               <ul style="list-style-type: none"> <li>- Accounting package</li> <li>- Portfolio tracking</li> <li>- Deposit monitoring</li> <li>- Customer information system</li> </ul> </li> <li>▪ Expandability and institutional growth</li> <li>▪ Flexibility               <ul style="list-style-type: none"> <li>- Customer-centric vs. account-centric</li> <li>- Institutional types</li> <li>- Lending methodologies</li> <li>- Loan interest types</li> <li>- Savings and deposit account types</li> <li>- Deposit interest types</li> <li>- Payment types</li> <li>- Payment frequencies</li> <li>- Multiple branches or regions</li> <li>- Multiple languages</li> <li>- Multiple currencies</li> </ul> </li> </ul>
<b>Usability</b>
<ul style="list-style-type: none"> <li>▪ Ease of use and user-friendliness</li> <li>▪ User interface</li> </ul>
<b>Reporting</b>
<ul style="list-style-type: none"> <li>▪ Reports</li> <li>▪ Report generation</li> </ul>
<b>Standards and Compliance</b>
<ul style="list-style-type: none"> <li>▪ Accounting soundness and standards</li> <li>▪ Governmental and supervisory adherence</li> </ul>
<b>Administration and Support</b>
<ul style="list-style-type: none"> <li>▪ Security</li> <li>▪ Backup and recovery</li> <li>▪ Fault tolerance and robustness</li> <li>▪ End-of-period processing</li> <li>▪ Support infrastructure and maintenance</li> <li>▪ Version control and upgrade strategy</li> </ul>
<b>Technical Specifications and Correctness</b>
<ul style="list-style-type: none"> <li>▪ Technology and architecture</li> <li>▪ Performance</li> <li>▪ Number and date handling</li> </ul>
<b>Cost</b>
<ul style="list-style-type: none"> <li>▪ Pricing and costs</li> </ul>

## RESEARCH MATRIX

The matrix that begins in Chapter Three is organized by category. The category describes, at a high level, the areas in which an information system should be evaluated. Categories represent logical groupings of different subject areas that indicate the fundamental quality of a system. Each category, along with a listing of the corresponding topics and subtopics, is given in Table 1.

The matrix outlines each topic within a category using three columns: topic (or subtopic), definition, and measurement criteria. An additional column is provided for the evaluator's rating or comments:

- *Topic (or subtopic)*: name of the topic or subtopic. A subtopic is denoted by a → before the name and the name of the parent topic in parenthesis underneath the subtopic name.
- *Definition*: brief definition of the topic or subtopic. The definition is descriptive in nature and should provide enough information for the general user of this matrix to understand what the topic is and what issues or concerns it addresses.
- *Measurement criteria*: type of items or issues that should be looked at when assessing a system for the particular topic. The measurement criteria for any given topic are merely suggested. Some effort has been made to delineate the primary criteria for assessing a topic; however, given the varied nature of both computer systems and microfinance, users of this evaluation tool should determine from their surroundings whether more or less criteria would be useful in assessing the topic.
- *Rating/comments*: a blank column providing the evaluation team with a space to write comments and determine ratings for the individual topics and subtopics according to the guidelines given below.

## RATING METHOD

Each topic or subtopic will receive a rating of 1 to 5 with five being the highest. The rating for each topic or subtopic is determined by using the corresponding definition and measurement criteria as guidelines. It is important to note that the framework provides limited detail on how to evaluate each point, since it assumes a knowledgeable team with sufficient expertise to use judgement in determining ratings. A rating score sheet is provided in Annex A to assist the evaluation team in determining weights and tallying ratings.

For any topic with subtopics, the overall topic rating should be an average of the scores for the corresponding subtopics. For example, the “Functional Completeness, Appropriateness, and Integration” topic has four subtopics. The overall rating for this topic would consist of the average of these four scores. Of course, these ratings are subjective, and evaluators may want to adjust the overall rating for any given topic beyond the straight subtopic average.

In these cases, evaluators should submit a comment that explains the reason for the adjustment.<sup>3</sup> A slightly different technique should be applied for homegrown systems versus off-the-shelf systems. For instance, an evaluation of an internal system needs to take into consideration the appropriateness of that system in light of the microfinance institution's stage of development, strategic direction, scale, complexity, and projected growth. In many ways, off-the-shelf systems should be held to a higher standard because they should be applicable to a wider range of organizations and environments.

Because this framework is an analytical tool, the evaluators are responsible for determining the best method for applying the framework to any given situation. For instance, in some cases it may not be necessary to rate the software on certain topics or sub-topics because they may not be relevant to the MFI (i.e. multiple currencies, different languages, etc.).

Before completing the rating, the appraisal team should prioritize the criteria and then assign weights accordingly. For instance, many MFIs feel that the availability and quality of technical support is a critical issue. Many MFIs have fallen into the trap of buying a feature-rich system that has poor technical support, or no local technical support. This situation can be worse than having no system at all.

---

<sup>3</sup> This framework does not provide a mechanism for overall rating of the system, but rather rates the software by topic only. The evaluator could attempt to determine a rating for each category by taking an average, weighted or otherwise, of its corresponding topics, but this is not necessarily recommended or required.

## CHAPTER THREE

### CATEGORY 1: FUNCTIONALITY AND EXPANDABILITY

**Description:** Measures the extent to which a software product meets the requirements of different types of microfinance programs and whether it has the capacity to grow and expand with a microfinance institution as the organization evolves and adds functions and clients. On a deeper level, this category deals with how functionally rich a program is—including the types of institutions and methodologies that the software supports. The evaluation of this category would include both front-office and back-office functions.

The most intriguing complexity that microfinance imposes on the software developer is its diverse nature. Financial systems by themselves are difficult enough to develop. Microfinance adds several new layers. For example, within the field of microfinance there are numerous different lending methodologies, such as village banking, solidarity groups, and individual lending. There are also many institutional models, such as credit unions, cooperatives, non-governmental organizations (NGOs), and banks (from informal to formal and from small scale to large scale). In addition, the methods for interest and payment calculation are as varied as the number of loan products offered in the field today. To exacerbate the problem, many of these variations can occur inside the same organization. Of course, all of these organizations operate within diverse social, political, economic, regulatory, and legal environments around the world. Add in the different currencies, languages, and reporting requirements and one can begin to fathom the difficulties in creating a quality application that supports one organization, let alone one that supports many. The appraisal team should consider these complexities as the appraisal team completes this section of the Framework.

**Topics:** Functional Completeness, Appropriateness, and Integration; Expandability and Institutional Growth; Flexibility

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Functional Completeness, Appropriateness, and Integration</b>	<p>The features of the system meet the needs of the business in an appropriate fashion.</p> <p>Integration refers to how well the different components of the system can communicate with each other, thus allowing data sharing and reducing the need for multiple entry of data (the need to input the same data into different parts of the system).</p>	<p>(* see subtopics for more details)</p> <ul style="list-style-type: none"> <li>▪ Accounting functionality*</li> <li>▪ Portfolio tracking capabilities*</li> <li>▪ Savings/deposit tracking facilities*</li> <li>▪ Client information tracking facilities*</li> <li>▪ Systems integration</li> <li>▪ Tracking of nonfinancial activities (impact, business development services, etc.)</li> </ul>	
<b>→ Accounting Package (Functional Completeness ...)</b>	<p>Ability of the software to perform a full range of accounting activities.</p>	<ul style="list-style-type: none"> <li>▪ Integrated with the savings and portfolio tracking system, or stand-alone<sup>4</sup></li> <li>▪ Level of integration—direct (changes made in savings and/or portfolio tracking system immediately affect the proper accounts) or indirect (accounting package is separate, necessitating a periodic update of accounting data)</li> <li>▪ Complete, consistent, flexible and user-definable chart of accounts (number of digits, levels, and formats)</li> <li>▪ Tracking of cash flow, revenues, and expenses by several sources or profit/cost centers (donor, account, branch, product, etc.) in addition to consolidated tracking of this information</li> <li>▪ Ability to perform cost/profitability analysis by product, branch/region, client, etc.</li> <li>▪ Cash vs. accrual—if accrual, proper provisioning of receivables</li> <li>▪ Loan loss provisioning and reserves</li> <li>▪ General ledger</li> <li>▪ Trial balances</li> <li>▪ Permits entry of nonportfolio or deposit related income and expenses</li> </ul>	

<sup>4</sup> Whether or not an integrated system is preferable is a hotly debated point within the microfinance industry. The appraisal team should decide beforehand whether system integration is important in the context of the specific review.

Topic	Definition	Measurement Criteria	Rating/Comments
		<ul style="list-style-type: none"> <li>▪ Full range of standard financial reports (balance sheet, income statement, cash flow, etc.)</li> <li>▪ Track and apply overhead expenses</li> <li>▪ Asset and liability management facilities</li> <li>▪ Payroll module</li> <li>▪ Fixed assets module</li> <li>▪ Treasury functions</li> </ul>	
<p><b>→ Portfolio Tracking (Functional Completeness ...)</b></p>	<p>Ability of the software to monitor and manage the loan portfolio.</p>	<ul style="list-style-type: none"> <li>▪ Integrated with accounting system, deposit monitoring, and/or customer information system</li> <li>▪ Permits the addition and modification of loan products</li> <li>▪ Historic data on products</li> <li>▪ Forced deposits (linked to deposit monitoring) with ability to block access to forced savings (where appropriate)</li> <li>▪ Linking of forced savings to loans or “membership”</li> <li>▪ Collateral (cash and noncash) tracking</li> <li>▪ Guarantor tracking and number of guarantors per loan</li> <li>▪ Identifying and cross-referencing of group guarantees</li> <li>▪ Loan officer specific information (active portfolio, delinquency, number of clients, etc.)</li> <li>▪ Correct portfolio aging mechanisms</li> <li>▪ Proactively informs the users of potential problems (delinquency, cash standing, productivity, etc.)</li> <li>▪ Delinquency management facilities</li> <li>▪ Delinquency calculation methodology</li> <li>▪ Handling of early, late, partial, and extra payments</li> <li>▪ Credit scoring capabilities</li> <li>▪ Advanced functionality such as credit cards or smart cards</li> </ul>	
<p><b>→ Deposit Monitoring (Functional Completeness ...)</b></p>	<p>Ability of the software to handle deposit accounts.</p>	<ul style="list-style-type: none"> <li>▪ Integrated with accounting system, portfolio tracking, and/or customer information system</li> <li>▪ Permits the addition and modification of deposit types</li> <li>▪ Historical data on products</li> <li>▪ Voluntary deposits (with some tracking of or information about forced savings—linked to portfolio tracking)</li> <li>▪ Permits different account types</li> <li>▪ Advanced functionality such as ATMs, wire transfers, and smart cards</li> <li>▪ Tax withholding functionality</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
		<ul style="list-style-type: none"> <li>▪ Dormancy for inactive accounts</li> <li>▪ Identification of beneficiaries in case of death or incapacitation</li> <li>▪ Option for jointly held accounts</li> </ul>	
<p><b>→ Client Information System (Functional Completeness...)</b></p>	<p>Ability of the software to maintain information about clients.</p>	<ul style="list-style-type: none"> <li>▪ Strong search capabilities</li> <li>▪ Maintains customer information such as name, family information, age, gender, address (home and business), and type of business, as well as impact information</li> <li>▪ Tracks clients at different levels, from individual to group to center to village bank and so on</li> <li>▪ Able to maintain group and/or village bank information</li> <li>▪ Facilities to check customer behavior—i.e., credit and deposit status and history (either from external or internal sources)</li> <li>▪ Historical data on customers</li> <li>▪ Aggregation of customer data (by region, area, economic activity, loan officer, etc.)</li> <li>▪ Able to track clients at different stages of the process</li> <li>▪ Tracks nonclient information, especially guarantors</li> <li>▪ Performs account transfer functionality (from forced to voluntary or vice versa, one geographic location to another, etc.)</li> <li>▪ Identifies potential duplicates (i.e., double entry of clients)</li> </ul>	
<p><b>Expandability and Institutional Growth</b></p>	<p>Ability of the system to support horizontal and vertical institutional growth. In essence, the scalability of the system is in question.</p>	<ul style="list-style-type: none"> <li>▪ Modules available to support new products and services (e.g., demand deposits, credit cards, mortgage loans, lines of credit, and money transfers, in addition to standard microfinance services)</li> <li>▪ Can move with the organization from informal (NGO) to formal financial institution (appropriate reports, treasury functions, etc.)</li> <li>▪ Number of terminals it can support efficiently—with reasonable response times</li> <li>▪ Number of clients it can handle with reasonable response times</li> </ul>	
<p><b>Flexibility</b></p>	<p>Software is considered flexible when the system is easily altered to meet a new or different business requirement. Flexibility also refers to how adaptable an application is to</p>	<p>(* see subtopics for more details)</p> <ul style="list-style-type: none"> <li>▪ Support customer-centric or account-centric view*</li> <li>▪ Supports different institutional types*</li> <li>▪ Supports multiple methodologies*</li> <li>▪ Supports different loan interest calculations*</li> <li>▪ Supports different deposit interest calculations*</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
	<p>different organizational or business situations. The adaptations or alterations can be in the form of changes to the program itself (e.g., code) or parameters used to set up the information system. In addition, the flexibility of a system is reflected in how easily important business-level items (i.e., products, tracking information such as donor, sex, or business type, etc.) can be added, modified, or deleted.</p> <p>Flexibility basically asks the question of how extensible or configurable a system is.</p>	<ul style="list-style-type: none"> <li>▪ Supports multiple payment types*</li> <li>▪ Supports different payment frequencies*</li> <li>▪ Supports multiple branches and regions*</li> <li>▪ Supports multiple languages*</li> <li>▪ Supports multiple currencies*</li> </ul>	
<p>→ <b>Customer-centric vs. Account-centric (Flexibility)</b></p>	<p>Software of this nature can view the world as starting either from a customer or from an account. This simple nuance can determine how flexible a system is, because a customer-centric system is usually more flexible than an account-centric system.</p> <p>In the customer-centric view, accounts are associated to a customer. In an account-centric world, customers are associated to an account.</p>	<ul style="list-style-type: none"> <li>▪ Allows a customer to have more than one account and account type (deposit, credit, etc.)</li> <li>▪ Allows the tracking and maintenance of customer data such as contact information, gender, marital status, business activity, etc.</li> <li>▪ Allows detailed information about each account to be stored, such as account type, usage of funds, amount, etc.</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
→ <b>Institutional Types (Flexibility)</b>	Types of institutions the system is designed to support or could support given minimal modifications.	<ul style="list-style-type: none"> <li>▪ Full-service banks</li> <li>▪ Limited-service banks</li> <li>▪ Cooperative savings and credit societies/unions</li> <li>▪ Microfinance institutions</li> <li>▪ Limited liability companies</li> <li>▪ Foundations or trusts</li> <li>▪ Other</li> </ul>	
→ <b>Lending Methodologies (Flexibility)</b>	The different types of microfinance loan approaches the system can support or could support given minimal modifications.	<ul style="list-style-type: none"> <li>▪ Handles only one lending methodology or can handle multiple methodologies simultaneously</li> </ul> <p>Typical lending methodologies include:</p> <ul style="list-style-type: none"> <li>- Individual clients</li> <li>- Solidarity groups with individual loans</li> <li>- Solidarity groups with group loans</li> <li>- Village banks with individual loans</li> <li>- Village banks with group loans</li> <li>- Other</li> </ul>	
→ <b>Loan Interest Types (Flexibility)</b>	Different forms of interest rate calculations the system supports or could support given minimal modifications.	<ul style="list-style-type: none"> <li>▪ Flat</li> <li>▪ Declining balance</li> <li>▪ Discounted from the loan</li> <li>▪ Capitalized</li> <li>▪ Variable rate</li> <li>▪ Stepped rate</li> <li>▪ Commissions and fees</li> <li>▪ Penalty fees for late payments</li> <li>▪ Other (user definable, etc.)</li> </ul>	
→ <b>Savings and Deposit Account Types (Flexibility)</b>	The different types of savings accounts supported by the deposit programs or could be supported given minimal modifications.	<ul style="list-style-type: none"> <li>▪ Passbook (with or without passbook)</li> <li>▪ Term deposits (i.e. certificates of deposit)</li> <li>▪ Group savings</li> <li>▪ Group insurance fees</li> <li>▪ Off-book group savings</li> <li>▪ Demand deposits</li> <li>▪ Overdraft accounts</li> <li>▪ Current accounts</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
→ <b>Deposit Interest Types (Flexibility)</b>	The different types of interest supported by the deposit programs or could be supported given minimal modifications.	<ul style="list-style-type: none"> <li>▪ Day of deposit to day of withdrawal</li> <li>▪ Minimum daily balance</li> <li>▪ Minimum monthly balance</li> <li>▪ Minimum quarterly balance</li> <li>▪ Average daily balance</li> <li>▪ Average monthly balance</li> <li>▪ Other (user definable, etc.)</li> </ul>	
→ <b>Payment Types (Flexibility)</b>	The ability of the software to handle different common loan payment schedules as well as updates or changes to a payment schedule. How easily the system could be modified to handle new payment types should also be reviewed.	<ul style="list-style-type: none"> <li>▪ Term loans with constant payments</li> <li>▪ Term loans with constant principal</li> <li>▪ Irregular payments</li> <li>▪ Single payment</li> <li>▪ Balloon</li> <li>▪ Selection of initial and subsequent payment dates</li> <li>▪ Other (user definable, etc.)</li> </ul> <p>Payment methods:</p> <ul style="list-style-type: none"> <li>▪ Cash (different currencies where appropriate)</li> <li>▪ Check</li> <li>▪ Credit card</li> <li>▪ Smart card</li> <li>▪ Money order</li> <li>▪ Other</li> </ul> <p>Special payment modifications:</p> <ul style="list-style-type: none"> <li>▪ Permits the suspension of penalty fees</li> <li>▪ Permits deferment of loan payment</li> <li>▪ Permits grace periods</li> <li>▪ Permits refinancing (recalculation of payment amounts) of loan</li> </ul>	
→ <b>Payment Frequencies (Flexibility)</b>	The ability of the software to handle frequent payments as well as standard payments; the ease with which the system can be modified to handle different payment frequencies.	<p>Payment frequencies supported:</p> <ul style="list-style-type: none"> <li>▪ Daily</li> <li>▪ Weekly</li> <li>▪ Biweekly</li> <li>▪ Semimonthly</li> <li>▪ Once every four weeks</li> <li>▪ Monthly</li> <li>▪ Other (user definable, etc.)</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
		Days (or weeks) of the year supported: <ul style="list-style-type: none"> <li>▪ 365</li> <li>▪ 360</li> <li>▪ 332</li> <li>▪ 50 weeks</li> <li>▪ Other</li> </ul> Payment aberrations support: <ul style="list-style-type: none"> <li>▪ Prepayments</li> <li>▪ Late payments</li> <li>▪ Underpayments</li> <li>▪ Overpayments</li> </ul>	
<b>→ Multiple Branches and/or Regions (Flexibility)</b>	Ability of the software to handle multiple offices, either on a branch or regional level. In addition, how quickly the system can be updated to handle multiple branches or regions or new locations.	<ul style="list-style-type: none"> <li>▪ Mechanisms for separating information on an office basis</li> <li>▪ Mechanisms for aggregating office level data (on-line, store and forward, etc.)</li> <li>▪ Reporting on an office or area basis</li> <li>▪ Frequency of updates to head office or area office</li> <li>▪ Bank and/or account transfers (intra- or inter-)</li> </ul>	
<b>→ Multiple Languages (Flexibility)</b>	Can the software handle multiple languages or could it handle them given minimal modifications?	<ul style="list-style-type: none"> <li>▪ Supports local language (if written)</li> <li>▪ Can support languages on a user basis—multiple languages simultaneously</li> <li>▪ All messages are in the language of choice</li> <li>▪ All screen information is presented in language of choice</li> <li>▪ Multiple language support requires recoding (language is hardcode) or is intrinsic to the system (language is parameterized)</li> </ul>	
<b>→ Multiple Currencies (Flexibility)</b>	Can the software handle multiple currencies or could it handle them given minimal modifications?	<ul style="list-style-type: none"> <li>▪ Supports local currencies and foreign currencies</li> <li>▪ Supports payments and disbursements in different currencies</li> <li>▪ Supports foreign exchange exposure calculation facilities</li> <li>▪ Handles maintenance of value accounts and other inflationary risk mitigation functions</li> </ul>	

## CHAPTER FOUR

### CATEGORY 2: USABILITY

**Description:** Measures the degree to which the user interface of the software helps users to perform their tasks effectively and efficiently, with little or no chance of errors. Includes the availability and usefulness of user documentation, screen layouts and navigational aids, on-line help, and prompts. Evaluation would consist of a checklist of key aspects concerning the user interface and a qualitative assessment of the ease of use.

One of the best ways of determining the usability and user satisfaction of any software product is a user survey. Therefore, a user survey should be performed to determine how different users rate the software. A user survey is an essential tool that should be used to elicit the answers to the following questions:

- Who is the user? (i.e., job title, responsibilities, age, education and training, how long and how much have they interacted with the system, etc.)
- How does the software help the users do their work?
- What do users like about the system? Why?
- Do users think the software is easy to use? Why or why not?
- What do users like about the system? Why not?
- What else could the system do to improve overall efficiency and job satisfaction?
- Overall rating of the system by users

**Topics:** Ease of Use and User-Friendliness; User Interface

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Ease of Use and User-Friendliness</b>	<p>Ability of the user to perform needed tasks easily and efficiently without errors.</p> <p>A major factor when implementing a new information system is how easily both the user community and the administrative personnel can learn and understand the system. The ease of use of a system will determine many things, including the training needs of the organization, the number of user mistakes, and how much the user will like using the system.</p>	<ul style="list-style-type: none"> <li>▪ Amount of training required for users, who provides it? Where and how do people get trained? Is it convenient or disruptive to operations?</li> <li>▪ Quality of user documentation</li> <li>▪ Usefulness of on-line help (detail, ability to drill-down through topics, hyperlinks, etc.)</li> <li>▪ Straight-forwardness of operations</li> <li>▪ Easy error correction</li> <li>▪ Program does not bomb out or crash when the user does something unexpected</li> <li>▪ Prompts or guides user to correct actions</li> </ul>	
<b>User Interface</b>	<p>For the most part, ease of use can be determined through the user interface. The user interface consists of screen and forms through which users interact with the system.</p>	<ul style="list-style-type: none"> <li>▪ Graphical vs. text-based user interface (e.g., Windows vs. DOS)</li> <li>▪ Physical appearance, layout, and logic of screens and menus</li> <li>▪ Screens flow logically and consistently</li> <li>▪ Appropriate information displayed for each level or type of user</li> <li>▪ Provides a mechanism for mass data entry</li> <li>▪ Consistent language and verbiage throughout</li> <li>▪ Error messages in “user-ese” not “tech speak”</li> <li>▪ Consistent and logical use of color to assist user where possible</li> <li>▪ Keyboard and mouse access to all major functions</li> <li>▪ Follows general platform standards (Microsoft for PCs, etc.)</li> </ul>	

## CHAPTER FIVE

### CATEGORY 3: REPORTING

**Description:** Examines the extent to which built-in reports cover management and operations requirements, the accuracy of the information presented, and the visual effectiveness of the reports. Examines whether users can modify existing reports or create new reports on their own. Investigates whether the system allows institutions to export data to spreadsheets for further manipulation or whether the data must be re-entered by hand. Evaluation would compare the list of available reports and report customization facilities against a standardized list of required reports, and include a qualitative assessment of the visual layout and effectiveness of the reports.

**Topics:** Reports; Report Generation

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Reports</b>	Adequacy and accuracy of the standard reports produced by the system. Includes how easy and useable the different reports are.	<p>General:</p> <ul style="list-style-type: none"> <li>▪ CGAP suggested reports</li> <li>▪ Micro-Banking Bulletin standard reports and report formats</li> <li>▪ Consolidated, as well as separated, reporting capabilities</li> <li>▪ User modifiable report formats</li> <li>▪ Reports according to audience (manager, operations, supervisory, etc.)</li> <li>▪ Report formats are clear and readable</li> <li>▪ Budget vs. actual reporting is possible</li> </ul> <p>Management reports:</p> <ul style="list-style-type: none"> <li>▪ Key statistical summaries</li> <li>▪ Cash-flow projections</li> <li>▪ Branch office and loan officer performance</li> </ul> <p>Operational reports:</p> <ul style="list-style-type: none"> <li>▪ Daily listings</li> <li>▪ Daily delinquency reports</li> <li>▪ Portfolio quality reports</li> </ul> <p>Financial reports:</p> <ul style="list-style-type: none"> <li>▪ Trial balance</li> <li>▪ Daily sub-ledger reports</li> <li>▪ Daily transaction reports</li> <li>▪ Monthly, quarterly, and annual financial statements</li> <li>▪ Ratios and trends</li> <li>▪ Clear, correct calculations of ratios and indicators</li> <li>▪ Audit trail</li> <li>▪ Accounts for inflation and subsidy adjustments</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
		Customer reports: <ul style="list-style-type: none"> <li>▪ Statements</li> <li>▪ Balances</li> <li>▪ Queries</li> </ul>	
<b>Report Generation</b>	The mechanism through which reports are created. Does the system provide standard ("canned") reports and does it allow the user to generate additional reports without requiring assistance from the software vendor?	<ul style="list-style-type: none"> <li>▪ Batch report generation</li> <li>▪ Ad-hoc report generation</li> <li>▪ Canned reports</li> <li>▪ User-defined reports</li> <li>▪ Report generation tool (crystal reports, etc.)</li> <li>▪ Report information can be dumped into a file readable by standard word processing or spreadsheet software</li> <li>▪ Printer and paper size requirements</li> <li>▪ Reports are generated frequently and are useful to intended audience</li> </ul>	

## CHAPTER SIX

### CATEGORY 4: STANDARDS AND COMPLIANCE

**Description:** Studies the extent to which the software product adheres to industry, government, and supervisory standards and regulations. Any system providing accounting support should be assessed to determine whether the system is in accordance with Generally Accepted Accounting Principles (GAAP) and International Accounting Standards (IAS). In addition, every system should meet the regulations set by local, regional, and national supervisory organizations. As the microfinance industry grows and becomes increasingly commercialized, adherence to regulatory mandates will become increasingly important.

**Topics:** Accounting Soundness and Standards; Governmental and Supervisory Adherence

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Accounting Soundness and Standards</b>	The topic area asks whether the accounting portion of the software product meets generally held standards and process accounts in a sound and consistent way.	<ul style="list-style-type: none"> <li>▪ Accounting package can be modified to meet local legal requirements</li> <li>▪ Adheres to GAAP and/or IAS provisions</li> <li>▪ Adheres to French accounting principles (where appropriate)</li> <li>▪ On-line or batch ledger updates</li> <li>▪ Posting order for partial or late payments</li> <li>▪ Categorizing current vs. delinquent loans</li> <li>▪ Accrual vs. cash</li> <li>▪ Existence of accounts to handle interest and principal due but not received separate from “accrued”</li> <li>▪ Software ceases to accrue interest on late loans</li> <li>▪ When savings interest is accrued, posted, and compounded</li> </ul>	
<b>Governmental and Supervisory Adherence</b>	The activities of many microfinance institutions are under the purview of governmental and supervisory regulators. Whether a software package meets these requirement and regulations is the focus of this topic.	<ul style="list-style-type: none"> <li>▪ Meets government/supervisory regulations</li> <li>▪ System can be easily modified to meet changes or additions to regulatory body requirements</li> <li>▪ Supports reporting requirements of central bank</li> <li>▪ Integrated into the national payment system</li> </ul>	

## CHAPTER SEVEN

### CATEGORY 5: ADMINISTRATION AND SUPPORT

**Description:** A broad range of activities and features are covered by this category. For example, to assess this category accurately, the software provider's ability to provide support and maintenance to the organization should be determined. Support includes installation, conversion, error correction, user queries, and periodic software improvements. Points to be covered would include years in business, number of staff, location of support offices, number of installed systems, existence of system documentation and operations manuals, and training materials and plans.

The category also covers security, which refers to provisions in the software to prevent or detect unauthorized entry, accidental or intentional damage to data, and falsification of records.

Other factors to be assessed are the ability of the software to recover accurately from crashes (power outages and others); provisions within the software to ensure the reliability, validity, and safety of the data; and mechanisms for accurately backing up and restoring data. The evaluation would also entail making a checklist of key recovery and backup features and assessing the accuracy and performance of these features.

Lastly, this category examines the ability of the vendor or internal staff to provide initial and ongoing support to the software, including the ability of the software provider to provide support and maintenance. Such support includes installation, data conversion, error correction, user queries, and period software improvements.

**Topics:** Security; Backup and Recovery; Fault Tolerance and Robustness; End-of-Period Processing; Support Infrastructure and Maintenance; Version Control and Upgrade Strategy

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Security</b>	Built-in safeguards of the system to prevent illegal or accidental alteration of the data files.	Look for presence of major security procedures: <ul style="list-style-type: none"> <li>▪ Different levels of user access with functions reserved for specific user levels (integrated into the user interface)</li> <li>▪ Supports different user views based on user permissions and types</li> <li>▪ User passwords</li> <li>▪ Does the system prompt users to change their passwords on a regular basis?</li> <li>▪ Users restricted to specific activities</li> <li>▪ Inherent security of the database and system architecture, protecting the system from both direct attacks and “back door” entry, the isolation of the database, and proper firewalling</li> <li>▪ Encryption of the database and passwords</li> <li>▪ Notification of file violations</li> <li>▪ Audit trail on transactions that identifies user</li> <li>▪ System violation log</li> <li>▪ Time-of-day or terminal access restrictions</li> <li>▪ Self-auditing program</li> <li>▪ Off-site data storage of records</li> </ul>	
<b>Backup and Recovery</b>	Provisions to store completed transactions, balances, and statements safely, and to restore this information if necessary.  Ability of the software to accurately recover from an accidental or improper shutdown.	<ul style="list-style-type: none"> <li>▪ Automatic end-of-day processing, with built-in recovery mechanisms</li> <li>▪ Length of time required for backup</li> <li>▪ Full vs. incremental backups</li> <li>▪ Does system provide for (or require) full backups on a regular basis—such as weekly or monthly</li> <li>▪ When restarting the system, correctly completes or restarts transactions or activities to avoid duplication of entries or lost data</li> <li>▪ How difficult and timely is the recovery process</li> <li>▪ System keeps track of current status and activity for both the central processing and each user</li> <li>▪ Does the system have archival facilities for off-loading old, unused data (to keep the database from growing exponentially)</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Fault Tolerance and Robustness</b>	Ability of the system to either recover after a crash (either software or externally induced) or remain on line during such an incident. Since information is the primary resource for managing any financial organization, it is imperative that the software maintains the information integrity of the data.	<ul style="list-style-type: none"> <li>▪ For a networked system, the system remains on-line if the network goes down</li> <li>▪ Notification to user of transactions not completed</li> <li>▪ System continues to function properly despite database or operating system errors</li> <li>▪ System handles major errors “gracefully” by providing users adequate time and information to react correctly</li> </ul>	
<b>End-of-Period Processing</b>	Most systems require some sort of administrative intervention to perform end-of-period processing such as interest calculation on deposits and reporting. This topic covers how well the system handles the different end of period procedures.	<ul style="list-style-type: none"> <li>▪ Proper interest posting and compounding</li> <li>▪ Late fees and penalties are calculated correctly</li> <li>▪ System correctly “closes the books” and prepares for end of period reporting</li> <li>▪ Transactions are moved from the journal to the general ledger</li> <li>▪ Report generation cycle is kicked off</li> </ul> <p>Period duration:</p> <ul style="list-style-type: none"> <li>▪ Daily</li> <li>▪ Weekly</li> <li>▪ Monthly</li> <li>▪ Quarterly</li> <li>▪ Yearly</li> <li>▪ Other (user-definable)</li> </ul>	
<b>Support Infrastructure and Maintenance</b>	This topic describes a wide range of issues related to how well the system in question is supported either by internal or external resources.	<p>Off the shelf:</p> <ul style="list-style-type: none"> <li>▪ Years in business</li> <li>▪ Financial strength of vendor</li> <li>▪ Number, size, and length of installed systems</li> <li>▪ Location of nearest support office</li> <li>▪ Response times/response levels</li> <li>▪ Support hotline</li> <li>▪ Number and technical ability of support staff</li> <li>▪ Access to source code (where appropriate or necessary)</li> <li>▪ Support of code once changed by the institution</li> <li>▪ Upgrade support and timeline (regional support offices)</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
		<ul style="list-style-type: none"> <li>▪ Change request procedure</li> <li>▪ Support capabilities (installation, data conversion, customization, training, day-to-day technical support)</li> <li>▪ System manuals and training materials</li> </ul> <p>Homegrown information system staff:</p> <ul style="list-style-type: none"> <li>▪ Response time</li> <li>▪ Internal help desk</li> <li>▪ Support hotline</li> <li>▪ Technical ability of support staff (number and experience)</li> <li>▪ Change request procedure</li> <li>▪ Support capabilities (installation, data conversion, customization, training, day-to-day technical support)</li> <li>▪ System manuals and training materials</li> </ul>	
<b>Version Control and Upgrade Strategy</b>	<p>Version control is how a software developer bundles and packages functionality into a product. To do this bundling properly, the developer needs to manage the program files (code, etc.) in a coherent and controlled way. Good version control alleviates the problems of having different copies of the software running simultaneously. Usually versions are numbered in the format 1.0, 1.1, 1.1.1, 2.0, etc.</p> <p>New versions usually translate into upgrades to the different users of the software product. Since upgrades sometimes involve major changes to the program, it is important for the software developer to have a clear plan for implementing these updates. This plan is called an upgrade strategy.</p>	<ul style="list-style-type: none"> <li>▪ Source code maintenance</li> <li>▪ Clear versioning of software (developer can tell immediately what versions of the software are currently in production and what functionality is supported by each version)</li> <li>▪ Clear upgrade strategy (developer can explain how to move from one version to another and what the effort make such an update will be)</li> </ul> <p>Rollout mechanisms used:</p> <ul style="list-style-type: none"> <li>▪ “Big Bang”—completely stop the current (old) system and start new system at the same time (<b>dangerous</b> and not recommended but commonly attempted)</li> <li>▪ Parallel—continue to run the current system and the new systems simultaneously for a certain period of time, in order to check the accuracy and performance of the new system</li> </ul>	

## CHAPTER EIGHT

### CATEGORY 6: TECHNICAL SPECIFICATIONS AND CORRECTNESS

**Description:** Analyzes the programs and programming language of the software, the type of network and hardware it is designed to work on, and the implications of these for future performance. In addition, the overall performance of the system in terms of speed and storage requirements should be assessed. In addition, it should be determined how well the system handles large numbers (typically, currency amounts) and dates (i.e., the year 2000).

**Topics:** Technology and Architecture; Performance; Number and Date Handling

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Technology and Architecture</b>	When analyzing computerized systems, an important facet is the technological platform upon which they are implemented. The platform consists of a set of technologies put together in a systematic or architectural way. This topic refers to the type of hardware and software used to create, execute, and maintain the system. The infrastructure and environment of the organization should drive technology and the technical architecture, both now and in the future.	<p>Have the following system areas been appropriately applied to the local technical environment:</p> <ul style="list-style-type: none"> <li>▪ Architecture (networked vs. stand-alone; client/server; etc.)</li> <li>▪ Database (Dbase, Oracle, Paradox, etc.)</li> <li>▪ Hardware (PC, Macintosh, Unix, etc.)</li> <li>▪ Minimum hardware configuration</li> <li>▪ Operating system (DOS, Windows 3.x, Windows 95, Windows NT, UNIX, Macintosh)</li> <li>▪ Network (Novell, Banyan, TCP/IP, etc.)</li> <li>▪ Development methodology (none, traditional, object-oriented, etc.)</li> <li>▪ Programming language (Clipper, FoxPro, Delphi, COBOL, C/C++, Java, etc.)</li> <li>▪ Infrastructure (electricity, telephone, etc.)</li> <li>▪ Source code control (none, PVCS, source safe, etc.)</li> <li>▪ Client/member-centric vs. loan/product-centric</li> </ul>	
<b>Performance</b>	<p>This topic determines how fast the software system executes any given task and how efficiently it stores and maintains its data.</p> <p>Whenever possible use a standard hardware configuration to test the system (not just for performance). Suggested configuration is: Pentium 133 or better and 16 megabytes of RAM or more.</p>	<p>Speed:</p> <ul style="list-style-type: none"> <li>▪ User interface (how fast is client data displayed, how quickly are long lists shown to the screen, etc.)</li> <li>▪ Report generation (how fast are weekly, daily, and yearly reports generated)</li> <li>▪ Multiple users (does the system slow down with more than one user, at what point does speed start to degrade, etc.)</li> <li>▪ Significant performance degradation as the size of the database grows</li> </ul> <p>Storage:</p> <ul style="list-style-type: none"> <li>▪ Empty system (how much data space does the software require after initial installation, i.e. before entry of any client data?)</li> <li>▪ Client data (how much space is required per client?)</li> <li>▪ Product data (how much space is required per</li> </ul>	

Topic	Definition	Measurement Criteria	Rating/Comments
		loan or saving product?) <ul style="list-style-type: none"> <li>▪ Maximum limits (database specific)</li> </ul>	
<b>Number and Date Handling</b>	This topic refers to all number or date related problems, of which the best known is the Year 2000 problem. Additional problems include the ability to handle large numbers (for inflated currencies) and year problems (1999, 2000, and 2001).	<ul style="list-style-type: none"> <li>▪ Year 1999, 2000, 2001</li> <li>▪ Large numbers (&gt;16 digits)</li> </ul>	

## CHAPTER NINE

### CATEGORY 7: COST

**Description:** Considers all costs associated with purchasing, installing, and operating the system. Cost information should include the base price of the software (as well as an assessment of the pricing structure), maintenance agreements, installation and training, conversion, upgrades, and maintenance releases.

**Topics:** Pricing and Costs

Topic	Definition	Measurement Criteria	Rating/Comments
<b>Pricing and Costs</b>	A consideration of all costs associated with purchasing, installing, modifying, updating, and operating the system.	<ul style="list-style-type: none"> <li>▪ Base price and pricing structure (licensing policies, etc.)</li> <li>▪ Customization and/or development costs</li> <li>▪ Additional costs for computer and network hardware</li> <li>▪ Additional costs for source code (where appropriate)</li> <li>▪ Maintenance and technical support fees and charges</li> <li>▪ Administrative costs (internal support)</li> <li>▪ Installation and training costs</li> <li>▪ Documentation costs</li> <li>▪ Conversion costs (moving from an old to a new system)</li> <li>▪ Costs of future upgrades and new releases</li> <li>▪ Overall costs per user and costs per information system staff</li> <li>▪ Price appropriate to level/complexity of functionality</li> <li>▪ Overall value proposition (functionality as a function of cost)</li> </ul>	

**ANNEX A**  
**RATING SCORE SHEET**

<b>Functionality and Expandability</b>	<b>Weight</b>	<b>Rating</b>
<ul style="list-style-type: none"> <li>▪ Functional completeness, appropriateness, and integration               <ul style="list-style-type: none"> <li>- Accounting package</li> <li>- Portfolio tracking</li> <li>- Deposit monitoring</li> <li>- Customer information system</li> </ul> </li> <li>▪ Expandability and institutional growth</li> <li>▪ Flexibility               <ul style="list-style-type: none"> <li>- Customer-centric vs. account-centric</li> <li>- Institutional types</li> <li>- Lending methodologies</li> <li>- Loan interest types</li> <li>- Savings and deposit account types</li> <li>- Deposit interest types</li> <li>- Payment types</li> <li>- Payment frequencies</li> <li>- Multiple branches or regions</li> <li>- Multiple languages</li> <li>- Multiple currencies</li> </ul> </li> </ul>		
<b>Usability</b>		
<ul style="list-style-type: none"> <li>▪ Ease of use and user-friendliness</li> <li>▪ User interface</li> </ul>		
<b>Reporting</b>		
<ul style="list-style-type: none"> <li>▪ Reports</li> <li>▪ Report generation</li> </ul>		
<b>Standards and Compliance</b>		
<ul style="list-style-type: none"> <li>▪ Accounting soundness and standards</li> <li>▪ Governmental and supervisory adherence</li> </ul>		
<b>Administration and Support</b>		
<ul style="list-style-type: none"> <li>▪ Security</li> <li>▪ Backup and recovery</li> <li>▪ Fault tolerance and robustness</li> <li>▪ End-of-period processing</li> <li>▪ Support infrastructure and maintenance</li> <li>▪ Version control and upgrade strategy</li> </ul>		
<b>Technical Specifications and Correctness</b>		
<ul style="list-style-type: none"> <li>▪ Technology and architecture</li> <li>▪ Performance</li> <li>▪ Number and date handling</li> </ul>		
<b>Cost</b>		
<ul style="list-style-type: none"> <li>▪ Pricing and costs</li> </ul>		