

# IDENTIFYING QUALITY-BASED REQUIREMENTS

Ledis Chirinos, Francisca Losavio, and Alfredo Matteo

**This article proposes a model for identifying quality requirements based on three different views of quality. By specifying quality requirements in this manner, the framework establishes quality requirements that can be quantified and measured as part of an overall quality assurance program. This results in two levels of quality: one at the system level that ensures a quality software product and another at the process level that mitigates project risk.**

**T**HE IMPORTANCE OF A PRECISE AND consistent requirements specification drastically increases with the complexity of software systems, particularly if the stakeholders (e.g., users, clients, engineers, and developers) require a product corresponding to their expectations. Such distributed communication-based or component-based systems as E-commerce applications, accomplishing properties such as interoperability, adaptability, scalability, flexibility, and configurability, involve the offering of services that must respond to precise measurable quality properties or attributes, such as rates of time delay and response time. These aspects must be seriously considered by software practitioners and drive research into requirements engineering. In particular, the interest of the software community in the engineering of nonfunctional requirements, which have a direct effect on achieving services or functionality offered by software components, has greatly increased. Requirements engineering is a complex set of iterative processes, the goal of which is to establish a clear common understanding of the requirements driving the development of the software product.<sup>1,2</sup>

There is no general agreement in the literature on the precise meaning of a *software requirement* or on what a requirement represents.<sup>1-5</sup> Intuitively, it is a property that is required by the software system. This article uses the definition from Sommerville and Sawyer,<sup>2</sup>

and defines requirements as descriptions of software's behavior and its properties or attributes (e.g., scalability or mean-time-to-failure), as well as the constraints on the process, such as the response time rate. This definition is interesting for classification purposes because it considers different kinds of requirements.

There is consensus on the fact that the output of the requirements specification process is a document describing what a system does under specific conditions. This document must be produced early in the software development process to mitigate risks. Moreover, it must be pointed out that a software system is structured or articulated on the basis of its architecture or "baseline," which takes into account quality requirements.<sup>6,7</sup>

Engineering quality requirements starts at the beginning of a software project. It can be the basis of quality-based software development, which involves the control and management of software quality throughout the entire software development cycle.<sup>1</sup> In the literature, there is no consensus on the terminology used to describe a general requirements engineering process.<sup>5,8-10</sup> However, several standardization organizations are working toward this aim.<sup>3,4,11</sup> The use of different terminology leads to different taxonomies.

In Cristel,<sup>12</sup> Oberg,<sup>10</sup> Sawyer et al.,<sup>13</sup> Sommerville et al.,<sup>2</sup> and Wiegers,<sup>5</sup> the requirements engineering discipline is considered part of the

*LEDIS CHIRINOS holds a Master's degree in computer science from the Central University of Venezuela, where she is enrolled in the doctoral program.*

*FRANCISCA LOSAVIO is a professor at the School of Computer Science, Venezuela Central University, where she coordinates the Software Technology Laboratory.*

*ALFREDO MATTEO is a professor at the School of Computer Science, Venezuela Central University, where he coordinates the TOOLS Laboratory.*

**R**equirements are derived from an organization's business needs.

software engineering process. Its activities — *elicitation, analysis, specification, validation, and management* — are clearly differentiated. They are repeated until an agreed-to *requirements specification document* (RD) is achieved. Once a final RD is produced, any further change is considered within the scope of the requirements management activity. The elicitation activity focuses mostly on the capture of functional and nonfunctional requirements. In particular, those aspects related to the *software product quality* (represented by the quality of its services) appear only in informal and mostly optional software documentation. This lack of formalization unbalances requirements specifications and implies the delivery of applications that possibly do not comply with stakeholders' expectations, further implying increased project risks.

Several efforts have been made toward the explicit specification of quality requirements (i.e., software measurable properties) based on the ISO/IEC 9126-1<sup>14</sup> quality model. The SQUID (Software QUality In the Development process)<sup>15</sup> approach and the ISO/IEC 14598-3<sup>16</sup> standard are some examples. The underlying idea of ISO/IEC 14598-3 is the definition of a quality model<sup>16</sup> and its use in a software evaluation framework, where the quality requirements are the basis for control and evaluation of the quality of the final software product. The SQUID approach based on ISO/IEC 9126-1 and ISO/IEC 14598-3 aims to quantitatively manage (i.e., control and evaluate) the quality of a software product during its development process. On the other hand, Bosch<sup>17</sup> and Clements et al.<sup>6</sup> suggest that the choice of software architecture is greatly influenced by the quality requirements. These approaches, however, do not precisely define the activity performed to identify quality requirements, which are the input to the architecture evaluation process. They are also input for monitoring and controlling the quality of software built according to the selected architecture. In typical software development methods,<sup>18</sup> guidelines are not explicitly considered for quantifying goals and measuring the actual values of quality requirements.

This article presents a general requirements classification model based on a quality perspective. This model is called RECLAMO (REquirements CLAssification MOdel) and one of its aims is to facilitate the work of requirements engineers. Another aim is to facilitate identifying different kinds of requirements involved in defining a software system, in particular, quality requirements. The motivation for establishing this

model can be found in several previous works,<sup>7,19,20</sup> where quality requirements management has been applied to quality-based software development. A case study illustrates the application of this taxonomy to specifying quality requirements in the domain of enterprise application portals (EAPs).

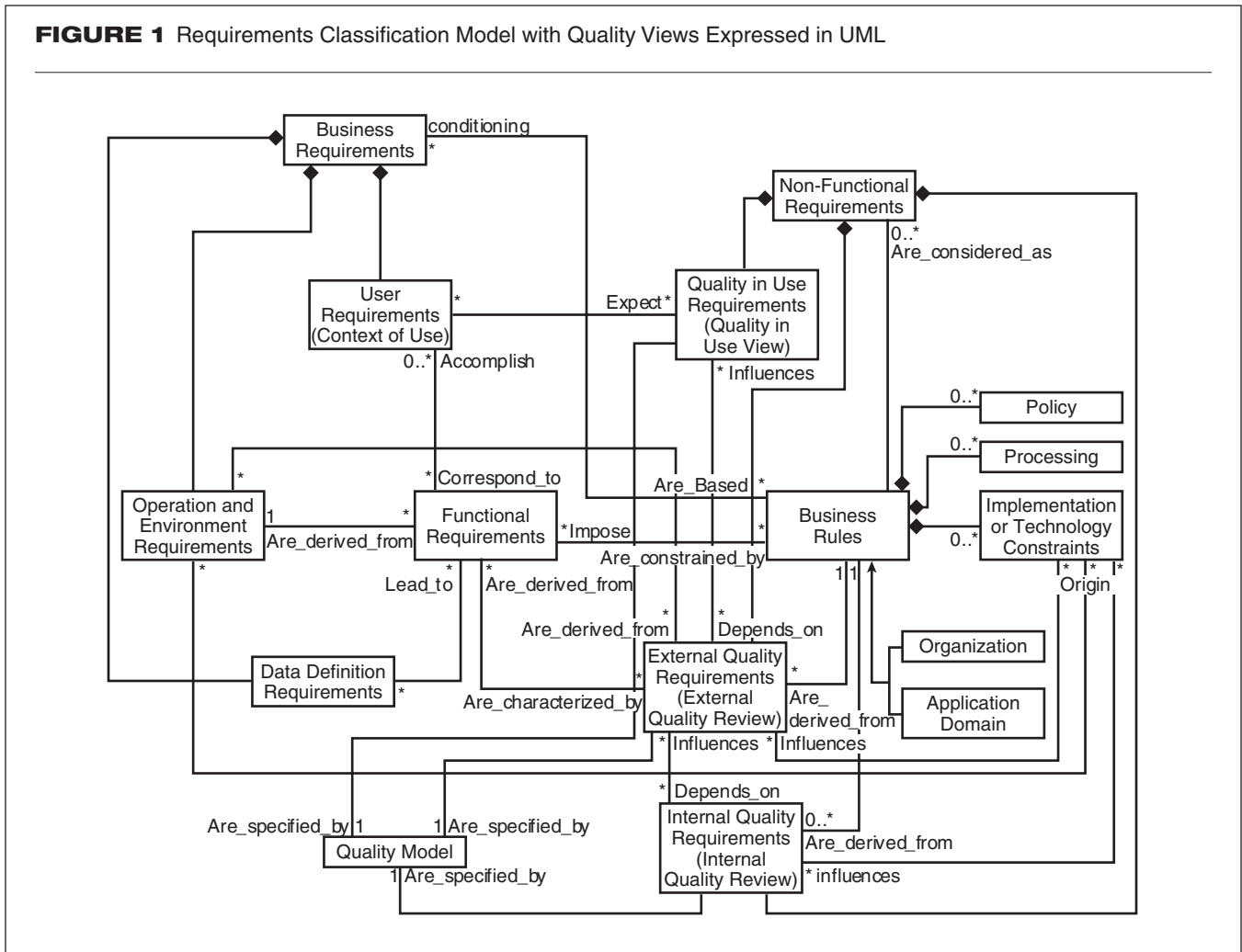
## TYPES OF REQUIREMENTS

### Requirements and Business Rules

Requirements vary according to the purpose and properties they represent. In the context of software system development, requirements are derived from an organization's business needs. Requirements describe goals that clients, organizations, and other stakeholders want to achieve.<sup>5</sup> A software project is chartered to achieve these goals or requirements. Implicit is that all software characteristics and product specifications must be defined to meet these goals. Business requirements express a set of user requirements, operational and environmental requirements, and data definition requirements. Business requirements are based on business rules. User requirements group all the activities that users must perform with the final system under certain desired quality conditions and in different contexts (i.e., contexts of use). User requirements correspond to functional requirements. In turn, functional requirements specify the functionality that must be built into the software product to accomplish user requirements. Desired quality conditions imposed on these requirements express the quality-in-use requirements and are considered part of nonfunctional requirements. They constitute the quality-in-use view of requirements. Operational and environmental requirements define requirements of other systems (i.e., external devices, hardware, or software) functioning within the environment of the software system. They may require or provide a service or functionality to or from a system or act to constrain its implementation or the behavior of its functionality, which could be considered part of the external quality view.

Business rules include corporative policy, government regulations, industrial standards, specific computational algorithms, and application domain standards. For example, EAPs must guarantee the integration of business processes and fulfill such quality requirements of this domain as availability and efficiency. Business rules are related to both the organization and the application domain. As such, business rules are classified as rules related to policy

**FIGURE 1** Requirements Classification Model with Quality Views Expressed in UML



(see also Johnatone et al.<sup>21</sup>). Policy is a set of rules and constraints on organizational processes and the treatment of objects (e.g., clients and invoices). Processing rules describe constraints on the execution of organizational processes, and implementation rules describe constraints on the implementation of organizational processes and technology.

Business rules are not software requirements per se because they concern aspects external to software. A business rule often establishes a new functionality that software must incorporate to meet a particular business rule. Business rules can also require that software must comply with specific external quality requirements. To meet business requirements, new functions or mechanisms are often implemented in software. New functions originated by a business rule are often called *implicit functionality*. Sometimes, business rules can originate internal quality requirements that must be taken into account during development. These internal quality requirements constitute

the internal quality view of requirements and they may influence external quality requirements. Other kinds of implicit functionality can be derived from the operational environment requirements and data definition requirements that describe data structures, data types, and permitted ranges of data values.

### Functional and Nonfunctional Requirements

Requirements that must be incorporated into software to satisfy business requirements and guarantee that end users can accomplish their goals are known as *functional requirements* or *system behavior*. These requirements are expressed through services or functions. As shown in Figure 1, functional requirements correspond to user requirements. Implicit functions are part of functional requirements and are derived from operational and environmental requirements, data definitions, and business rules. The requirements that specify the conditions that a system must satisfy and

**A**chieving required system quality often involves achieving internal quality; that is, satisfying quality requirements of the intermediate artifacts produced during the development process.

control the specification, implementation, and execution of functional requirements are known as nonfunctional requirements. Both functional and nonfunctional requirements are used in designing software that implements the required functionality within the desired quality goals and imposed constraints.

In the literature, nonfunctional requirements are often treated as quality requirements because they are, in general, expressed as some kind of quality requirements. For example, if a business requirement is to use the ADA language because the organization is doing business with the U.S. DoD, this business requirement is a constraint on the development process, imposed as a business rule. It implies a policy of avoiding errors because of ADA's exception handling capability. As a result of this business requirement, more robust software can be built, and the effect of the constraint imposed can be measured internally by a maturity attribute (i.e., reliability). Such decisions could affect the overall achievement of the required quality goals of the final system. However, identifying quality requirements is not an easy task, and the model proposed in this article is only a step toward achieving this goal. The next section of this article examines how to identify quality requirements using RECLAMO.

### IDENTIFYING QUALITY REQUIREMENTS USING RECLAMO

The term "quality" refers to an entity's set of attributes that are characteristic of its ability to satisfy established and implied needs.<sup>16</sup> Quality requirements specify properties of the functions or services that software must provide.<sup>5,14</sup> Achieving required system quality often involves achieving internal quality; that is, satisfying quality requirements of the intermediate artifacts produced during the development process. The level of quality expected by end users of a system (i.e., quality in use) is affected by the quality of these intermediate artifacts.

Quality requirements in RECLAMO are grouped into the three following views or levels:<sup>11</sup>

1. External
2. Internal
3. In-use or system

These views are based on the quality views found in the ISO 9126-1 standard.<sup>14</sup>

Quality-in-use requirements or system-view requirements correspond to the needs of particular users performing tasks in specific software and hardware environments. They are identified when actual users of a system state the tasks that the software system must do. The quality-in-use view applies to the quality requirements specified using RECLAMO, which are identified according to the use scenario established by stakeholders.

External quality requirements (or the external view of a system) are derived from the behavioral requirements of a system (i.e., functional requirements). They are also identified when operational environment requirements and business rules are established. External quality requirements are the goals for validating the different stages of development. They are identified when business rules are established or are generated by external quality requirements. The external quality view applies to the following situations:

- When the properties describing the services or functionality of the system are defined
- When the operational environment requirements are defined, considering other software systems and the hardware interacting with the system, and the business rules.

Internal quality requirements (or the internal quality view of a system) are related to the internal properties of artifacts generated during development and to the development process itself. The internal quality view considers those requirements that are relevant to the artifacts produced during the development process.

Quality requirements are, in general, specified according to a quality model,<sup>14,22,23</sup> as illustrated in Figure 1. They are defined as a set of quality characteristics and their relations.<sup>14</sup> High-level characteristics are refined into subcharacteristics that form a multilevel hierarchy. These, in turn, are described by a set of attributes at the lower level of the hierarchy. An attribute is a measurable property of a software product created during any stage of development. They are expressions involving characteristics, subcharacteristics, or attributes of a model. To evaluate these attributes, a measure must be identified and defined. Moreover, a range over a measurement scale must be defined to represent the degree of satisfaction with respect to the attribute's goal value.

In particular, the ISO 9126-1 quality model<sup>14</sup> decomposes external quality into six, high-level quality characteristics:

**U**sing  
**COMERX**,  
customers  
should be able  
to order  
products,  
browse order  
status, access  
updated news,  
and request  
customer  
service and  
technical  
assistance.

1. Functionality
2. Reliability
3. Usability
4. Efficiency
5. Maintainability
6. Portability

The quality-in-use view<sup>14</sup> consists of four, high-level characteristics:

1. Satisfaction
2. Productivity
3. Safety
4. Effectiveness.

A software quality model must reflect the correspondence between the definition of each quality characteristic in the software's particular domain to guarantee proper management of the quality of the software project and product within a particular organization.<sup>24</sup> According to RECLAMO (see [Figure 1](#)), identifying quality requirements should take into consideration the different types of requirements.

This article assumes that quality requirements are identified because requirements are elicited during requirements engineering. Identifying quality requirements is assumed to be part of the analysis activity in classical requirements engineering. RECLAMO explicitly deals with quality requirements to avoid semantic ambiguity and inconsistency. This favors formal and precise quantification of software quality. RECLAMO is a useful tool for classifying and identifying different types of elicited requirements according to the quality views introduced by the ISO 9126-1 standard.<sup>14</sup>

The software development process, as an engineering practice, is a complex job involving different stakeholders performing different roles. Often, each stakeholder uses its own vocabulary and has its own view of what a final system should be. Therefore, each stakeholder has its own perspective of the same software requirements. This creates synergies and conflicts as well as increasing risk. One way to mitigate such risk is to identify and classify the quality requirements according to the quality-in-use, external, and internal quality views used by RECLAMO. Within and among these perspectives, conflicts must be resolved and synergies identified, which may involve possible trade-offs among stakeholders. Quality aspects from ISO/IEC WD 25029,<sup>11</sup> ISO/IEC 9126-1,<sup>14</sup> ISO/IEC 14598-3,<sup>16</sup> Maguire,<sup>25</sup> and Robertson<sup>26</sup> have been considered in this sense, which is reflected in RECLAMO.

[Figure 2](#) shows, in detail, the activity of the quality requirements identification, with the respective tasks, artifacts, or deliverables, based on the UP (Unified Process) framework of process definition.<sup>18</sup> Notice that the worker involved in these tasks is mainly the Requirements Engineer. This activity is considered an extension of the Requirements Engineering Process.<sup>27</sup> The main deliverable is the requirements specification document (RD), as discussed in the first section of this article. Accompanying the RD are corresponding documents containing the quality requirements list, classified according to each quality view. These accompanying documents are written in the language of the stakeholders involved in each particular view.

## CASE STUDY

In this case study, RECLAMO is used to identify quality requirements to develop an enterprise application portal (EAP) system. These requirements are determined according to the tasks specified in [Figure 2](#).

## System Description

An EAP is a platform used to centralize commerce, collaboration, and information management functions. It enables business processes by seamlessly integrating back-end applications and systems (e.g., ERP and legacy systems) within the enterprise and with front-end applications and systems. It provides a personalized, single point of access to multiple heterogeneous information sources and applications. X is interested in building an EAP to articulate its main business activities supported by front-end B2C, SCM, and CRM systems. X's major goal is to be a broker of petroleum by-products such as plastic resins. Multiple companies located in different geographical areas manufacture the plastic resins and have an established commerce relationship with X.

X has named its EAP COMERX. With COMERX, X aims to integrate organizational processes and services and offer them through the Web. X's customers are both domestic and international. Using COMERX, customers should be able to order products, browse order status, access updated news, and request customer service and technical assistance. As shown in [Figure 3](#), a local database (COMERX DB) is part of the portal. It registers information about orders and services requested by customers. This information is input into an ERP system's databases (RMAN DBs), which is hosted

**FIGURE 2** Tasks and Artifacts Involved in Identifying Quality Requirements

Activities and Subactivities	Tasks	Artifacts
<p>1. Identification of quality requirements</p> <p>Identify quality-in-use, external quality, and internal quality requirements from the global list of elicited requirements, using the classification established in RECLAMO</p>	<p>a. Identification of quality-in-use requirements related with:</p> <ul style="list-style-type: none"> <li>• The user's context (considering tasks, physical, cultural, organizational, and technical aspects). The context of use must be identified and specified in the requirements elicitation, according to ISO 9241-11<sup>a</sup> and Maguire<sup>b</sup></li> </ul>	<p>1.1 Document describing the list of quality in use requirements.</p>
	<p>b. Identification of external quality requirements related to:</p> <ul style="list-style-type: none"> <li>• Functional requirements</li> <li>• Business rules (policy, processing, constraints imposed by implementation or technology)</li> <li>• Quality-in-use requirements</li> <li>• Operation and environment requirements</li> </ul>	<p>1.2 Document describing the list of external quality requirements. In some cases, the different components or product portions with the functionality and corresponding quality requirements could be part of this document.</p>
	<p>c. Identification of internal quality requirements related to:</p> <ul style="list-style-type: none"> <li>• The artifacts generated through the development process, as a function of the external quality requirements. This task could be supported by a model of the development process<sup>c</sup></li> <li>• The business rules (policy, processing, implementation or technological constraints)</li> </ul>	<p>1.3 Document describing the list of artifacts generated through the development process, according to the internal quality view.</p>
	<p>d. Identify and reduce conflicts, inconsistencies, or ambiguities between quality requirements for each quality view and between the quality views</p>	<p>1.4 Document describing the list of inconsistencies eliminated, for each quality view</p>
	<p>e. Combine related requirements and identify missing requirements for each quality view</p>	<p>1.5 Document describing the list of combined requirements and missing requirements, for each quality view.</p>
	<p>f. Identify and analyze risks associated with each requirement</p>	<p>1.6 Document describing the list of risks associated with each requirements.</p>
	<p>Observations: tasks d, e, and f apply to each quality view</p>	<p>Observations:</p> <ul style="list-style-type: none"> <li>• Links of influences must be explicitly maintained for each quality view.</li> <li>• External and internal quality requirements must be identified for each component of the system, if applicable<sup>c</sup></li> </ul>

<sup>a</sup> ISO 9241-11, "Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs). Part 11: Guidance on Usability," December 1998.

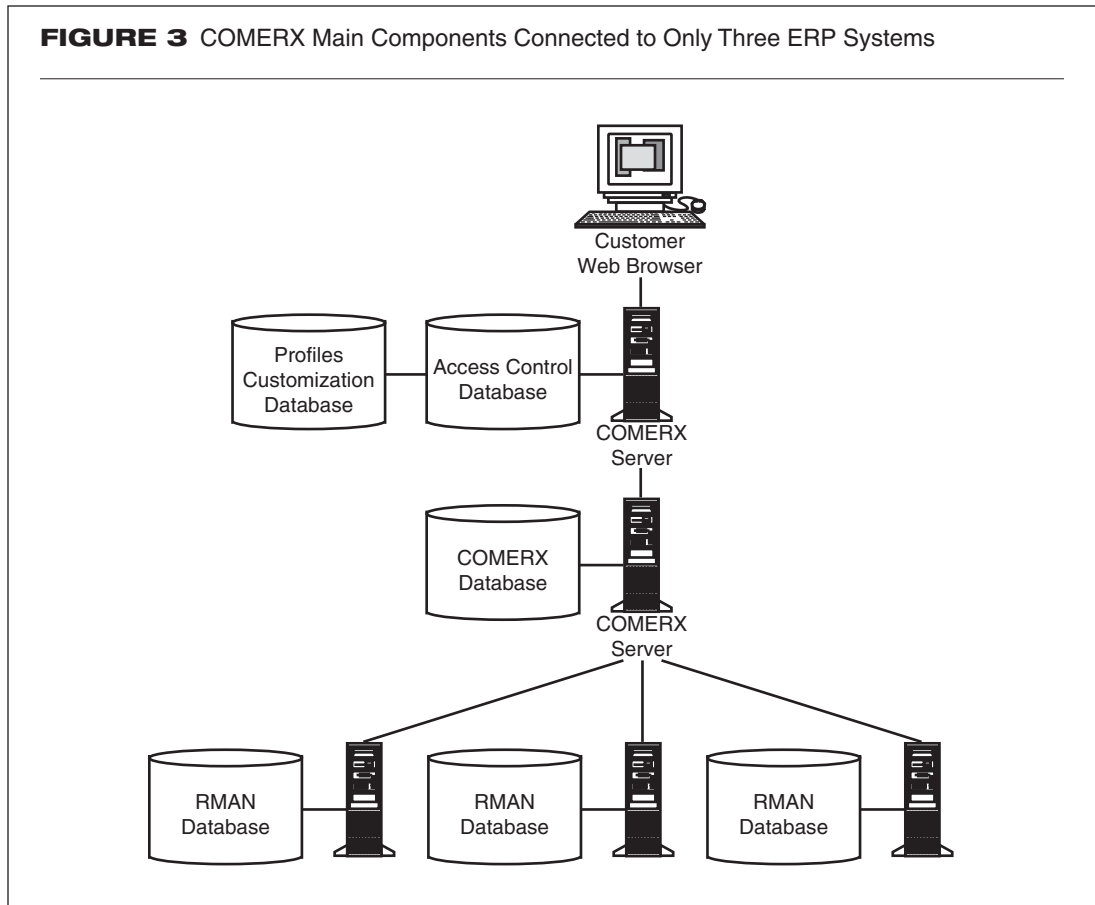
<sup>b</sup> Maguire, M., *User Requirements Framework Handbook: Respect*, 1997.

<sup>c</sup> Bøegh, J., Depanfilis, S., Kitchenham, B., and Pasquini, A., "A method for software quality, planning, control and evaluation," *IEEE Software*, March/April 1999, 69–77.

by manufacturers of the resin. Any of the manufacturers' ERP data relevant to X's customers is also stored in the COMERX DB. A schedule must be established for periodically exchanging information between the COMERX DB and the RMAN DBs. X administers the COMERX DB and is responsible for ensuring customer access. Through profile customization and access controls, COMERX allows customers to update information display by the portal. Customers

access COMERX through a commercial browser. Information about customer orders is stored in the COMERX DB. This data is sent to the respective RMAN DB within an established time period. The manufacturers are responsible for sending customers the resins ordered. They are also responsible for updating information about order status stored in the COMERX DB so it is accessible by customers. This implies that X's management, including sales executives

**FIGURE 3** COMERX Main Components Connected to Only Three ERP Systems



and logistics managers, should also be able to access the portal.

X's strategic objectives are to guarantee the continuous availability of the services offered by COMERX and to efficiently provide answers to customer needs.

**Applying RECLAMO**

The following sections detail how RECLAMO is used to identify the following requirements for COMERX:

- Users
- Functional
- Business rules
- Quality requirements

**Identifying Business Requirements.** X's business requirements are assumed to have already been elicited. They include:

- Integrating organization processes and sell resins, offering them through the Web
- Guaranteeing continuous availability of services
- Efficiently responding to customer needs
- Maintaining customer satisfaction with the services offered by X

These requirements are very general and expressed in business terms; hence, they are difficult to measure. The purpose of RECLAMO is to identify more specific requirements that are used to derive quality requirements that can be expressed in measurable terms.

**Identifying User Requirements.** COMERX users are classified into the following groups:

- Customers (national customers, international customers, or branch customers)
- Web master
- Transportation manager
- Sales executive
- Technical and logistics manager

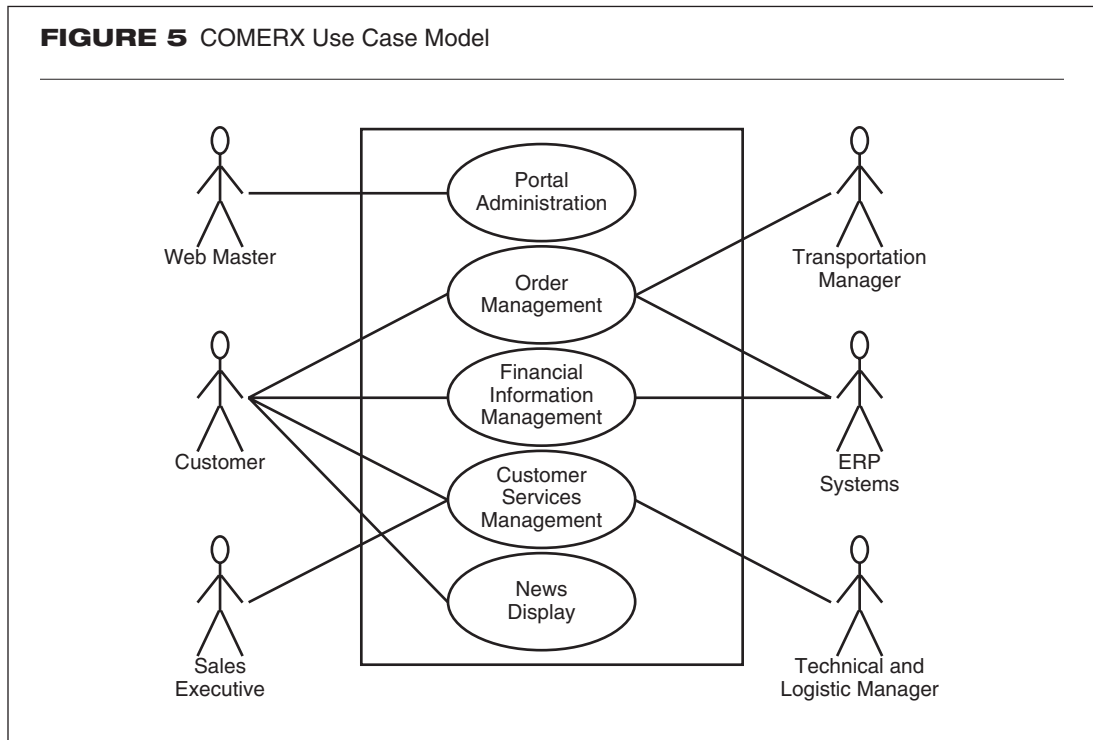
The requirements of these users are shown in Figure 4. Notice that each group of users belongs to a particular COMERX context of use.

**Functional Requirements.** COMERX must possess the functions allowing users to perform the requirements specified in Figure 4. Figure 5 presents the use case model, showing users (i.e., actors) and functionality (i.e., use cases).

**FIGURE 4** Users Requirements

Kind of Users	Users Requirements
Customers (national customers, international customers or branches)	Perform resin purchases (orders processing) Consult financial information related to resin transactions (orders, invoices, etc.) Formulate service requests (complaints with the purchased product, technical assistance) with documents supporting the request
Web master	Establish the levels of authorized access to the services Consult or update COMERX information
Transportation manager Sales executive	Register information related to resin transport Process service requests (verify, review, approve, distribute, add information)
Technical and logistic manager	Monitor service requests Assign responsibility to attend to service requests

**FIGURE 5** COMERX Use Case Model



**Identification of Operations and Environment Requirements.** The COMERX database must exchange information with the ERP databases hosted by the resin producers.

**Business Rules.** Business rules are specialized for the application domain and the organization (see Figure 1). These, in turn, could deal with policy, processing, and implementation or technological constraints. The application domain is that of Web-based EAP systems and all the functionality, standards, policy, and regulations related to this domain are considered.

- **Processing.** E-commerce functionality must be guaranteed. The system must have backward integration to ERP and legacy systems; forward integration to B2C, SCM, and CRM systems; collaborative work; and centralized information management. Simple and personalized access must be provided to multiple and heterogeneous applications and heterogeneous information sources.
- **Implementation.** Users must be able to access COMERX through a commercial browser.
- **Organizational policy.** The exchange of information between the COMERX DB and the ERP RMAN DBs of other resin producers

**FIGURE 6** Part of a Document Describing Quality-in-Use Requirements

System: COMERX	
Kind of Users	Quality-in-Use Requirements
Customers (national customers, international customers, or branches)	At least 95 percent of the functionality corresponding to the customers must be executed with complete satisfaction of the customer Less than 10 percent of the customers must complain in relation to the use of COMERX to satisfy the requested services

must be performed in strict time periods. Each type of user is allowed certain options. When a complaint form has not been opened in 24 hours, a note to the technical and logistic manager is sent accordingly. If a complaint form has been opened, the customer is notified about the processing of the complaint. After consulting a sales manager about a customer request, customers are notified by e-mail about confirmation or rejection of their orders. Customers are e-mailed about changes to their orders, if any.

- *Processing.* A password must be enforced to gain access to the portal. Orders must be loaded online or from a file. The customer must open only one order for each resin. Once a customer sends a resin order, it cannot be modified.
- *Implementation.* Mercurio and Visual Studio.net tools are to be used to develop the COMERX system.

**The Identification of Quality Requirements**

- *Identification of quality-in-use requirements.* These requirements express the requirements that the system must satisfy in different contexts of use. In each context, measurable properties must be identified (i.e., quality-in-use of the software product). A context of use is defined by the characteristics of its users, user goals, and the environment in which the system operates. A partial listing of the quality-in-use requirements for the customers are presented in Figure 6.
- *Identification of external quality requirements.* For purposes of simplification, only some external quality requirements are presented in Figure 7. Some of these requirements are characterized by the functional requirements. Others are derived from the business rules, problem domain, and implementation and technological constraints.

Figure 8 shows the external requirements after applying tasks described in Figure 7.

- *Identification of internal quality requirements.* To identify these requirements in RECLAMO, business rules (i.e., policy and best practice guidelines) and external quality requirements are considered. In this article, the architecture artifact is identified, because it is relevant to achieve the requirements imposed by external quality (see Figure 9).

From the requirements document, the formalization (i.e., specification and quantification) of quality requirements must be established to express the quality requirements in measurable terms. This specification could be done using a standard quality model such as ISO 9126-1,<sup>14</sup> customized to the application domain. This customization can achieve consistency between the standard and the domain terminology. The guidelines to construct an ontology could be used in this sense to avoid ambiguities in the interpretation of the terms used.<sup>28</sup> The quality model approach is used to validate the completeness of quality requirements. Quantification can be achieved using a measurement model to specify the elements and relations involved in the measure definition.

**CONCLUSION**

This article presented RECLAMO, a model to classify requirements elicited in the early phases of software development. Specifically, it identified quality requirements and the different views of quality according to the ISO 9126-1 standard. Because different types of requirements are clearly distinguished, the task of requirements analysis is made more effective, which is considered critical for requirements engineering and for mitigating risk in the entire software development process. Moreover, RECLAMO is a useful tool for formalizing non-functional requirements, which benefits quality-driven software development. ▲

**FIGURE 7** Preliminary 1.2 b Artifact: Document Describing the External Quality Requirements Derived from Functional Requirements, Business Rules, and Quality-in-Use Requirements

**COMERX system. External quality requirements derived from:**

Functional requirements	Financial information management	1. Guarantee accurate processing of business information
	Order management	2. Allow secure access to information 3. Guarantee continuity of functionality (high availability) 4. Guarantee reliable and efficient processing of orders 5. Guarantee continuity of functionality (high availability) 6. Guarantee accurate processing of business information
	Customer services management	7. Guarantee reliable and efficient processing of customer services 8. Guarantee continuity of functionality (high availability) 9. Provide efficient response to customer services
Business rules related to the application domain (processing)	Portal administration	10. Update COMERX information
	E-commerce must be guaranteed	11. Systems that interact with COMERX must be interoperable 12. Allow rapid and secure access to information 13. Internet demands that the systems using this technology should operate under different environments and platforms 14. Reliable and efficient support for an increasing number of customers
	Backward integration (ERP, legacy systems, etc) and forward integration (business-to-consumer (B2C), supply chain management (SCM), customer relationship management (CRM), etc.), collaborative work and centralized information management must be provided	15. Systems that interact with COMERX must be interoperable 16. Allow rapid and secure access to information 17. Internet demands that the systems using this technology should operate under different environments and platforms 18. Reliable and efficient support for an increasing number of customers
Operations and environment requirements Quality-in-use requirements	A simple and personalized access must be provided for multiple and heterogeneous applications and heterogeneous information sources	19. Allow for easy usage
	COMERX database must exchange information with the ERP database of other resins enterprises	20. Data formats must be exchangeable 21. Reliable and efficient support for an increasing number of transactions
	At least 95 percent of the functionality corresponding to the customers must be executed with complete satisfaction of the customer	22. The effort to use the portal must be minimal, meaning that the learning curve must be small 23. Guarantee continuity of COMERX functionality (high availability) 24. Provide appropriate functionality to satisfy implied and established needs
	Less than 10 percent of the customers must complain in relation to the use of COMERX to satisfy the requested services	25. The effort to use the portal must be minimal, meaning that the learning curve must be small 26. Guarantee continuity of COMERX functionality (high availability) 27. Provide appropriate functionality to satisfy implied and established needs

**FIGURE 8** Final Artifact 1.2.b: Document Listing External Quality Requirements

**COMERX System: External quality requirements. Final 1.2 b Artifact**

1. Guarantee accurate processing of business information (combines 1 and 6)
2. Allow rapid and reliable access to information (combines 2, 12, and 16)
3. Guarantee continuity of COMERX functionality (high availability) (combines 3, 5, 8, 23, and 26)
4. Guarantee reliable and efficient processing of orders
5. Guarantee reliable and efficient processing of customer services
6. Provide efficient response to customer services
7. Update COMERX information
8. Systems that interact with COMERX must be interoperable (combines 11 and 15)
9. Internet demands that the systems using this technology should operate under different environments and platforms (combines 13 and 17)
10. Reliable and efficient support to an increasing number of customers (combines 14 and 18)
11. The effort to use the portal must be minimal, meaning that the learning curve must be small and allow for easy usage (combines 19, 22, and 25)
12. Data formats must be exchangeable
13. Reliable and efficient support for an increasing number of transactions
14. Provide appropriate functionality to satisfy implied and established needs (combines 24 and 27)

**FIGURE 9** Artifact 1.3 c: Document Describing the Internal Quality Requirements for Systems Architecture

**System: COMERX**

**Internal quality requirements related to the architecture**

Artifact	Description
Architecture	<p>The architecture for Web applications to integrate business processes must be flexible (support to changes) (7), portable (9, 10, 13), interoperable (8), efficient (good performance) (4, 5), and reliable (2).<sup>a</sup> With respect to COMERX, accuracy is required (1). It implies accurate computations in some of the provided functions. Usability is also required (11); however, it is not relevant to the architecture because it depends mostly on the browser used. The architecture must guarantee the separation of concerns between the user's interface and the logic of the application (decoupling). The portal server must guarantee this aspect. Requirement 12 involves consistency. To achieve this, an additional mechanism must be introduced (implicit functionality). The same happens with requirement 1 for accuracy. Functional requirements must be then revised to add the new functionality.</p> <p>In particular, this architecture must offer:</p> <ul style="list-style-type: none"> <li><i>Robustness</i>: error handling, identification, and recovery</li> <li><i>Capacity to handle changes</i>: addition and removal of components without affecting the application (flexibility), scalability</li> <li><i>Adaptation to different environments</i>: reliable data exchange</li> </ul> <p><sup>a</sup> Losavio, F., Dinarle, O., and Perez, M., "Modeling EAI," <i>XII Encuentro Chileno de Computación, International Conference</i>, Copiapó, Chile, <i>Proceedings IEEE</i>, November 2002, 195–203.</p>

**References**

1. Thayer, R.H. and Dorfman, M., *Software Requirements Engineering*, 2nd edition, IEEE Computer Society Press, Los Alamitos, CA, 1997, 7–22.
2. Sommerville, I. and Sawyer, P., *Requirements Engineering. A Good Practice Guide*, John Wiley & Sons, New York, July 1997.
3. IEEE STD-829, "Recommended Practice for Software Requirements Specifications," 1998.
4. IEEE Standard 610, 1990.
5. Wiegers, K., *Software Requirements*, 2nd edition, Microsoft Press, 2003; ISBN 0-7356-1879-8.
6. Clements, P., Kazman, R., and Klein, M., *Evaluating Software Architectures*, Addison-Wesley, Reading, MA, 2002.
7. Losavio, F., Chirinos, L., Levy, N., and Ramdane-Cherif A., "Quality characteristics for software architecture," *Journal of Object Technology*, 2(2), 133–150, 2003, [http://www.jot.fm/issues/issue\\_2003\\_03/article2](http://www.jot.fm/issues/issue_2003_03/article2).

8. Jitnah, D., Han, J., and Steele, P., "Software Requirements Engineering: An Overview," Technical Report 95-04, Peninsula School of Computing and Information Technology, Monash University, Australia, September 1995.
9. Leffingwell, D. and Widrig D., *Managing Software Requirements. A Unified Approach*, Addison-Wesley, Reading, MA, 2000.
10. Oberg, R., Probaco, L., and Ericsson, M., "Applying Requirements Management with Use Cases," Rational Software Corporation, Technical Paper TP505, 1998.
11. ISO/IEC WD 25029, Software Engineering — Software Quality Requirements and Evaluation (SquaRE), May 2002.
12. Christel, M.G. and Kang, K.C., "Issues in Requirements Elicitation," Technical Report CMU/SEI-92-TR-12, ESC-TR-92-012 Software Engineering Institute, Pittsburgh, September 1992.
13. Sawyer, P. and Kotonya, G., "SWEBOK: Software Requirements Engineering Knowledge Area Description," Version 0.6 URL: [www.swebok.org/documents/stoneman06/Know...ersion\\_0\\_6\).PDF](http://www.swebok.org/documents/stoneman06/Know...ersion_0_6).PDF)
14. ISO/IEC 9126-1, "Software Engineering — Product Quality. Part 1: Quality Model," 2001.
15. Bøegh, J., Depanfilis, S., Kitchenham, B., and Pasquini, A., "A method for software quality, planning, control and evaluation," *IEEE Software*, March/April 1999, 69-77.
16. ISO/IEC 14598-3, "Information Technology — Software Product Evaluation — Part 3: Process for Developers. Software Engineering," June 1998.
17. Bosch, J., *Design and Use of Software Architecture*, Addison-Wesley, Reading, MA, 2002.
18. Krutchen, P., *The Rational Unified Process*, Addison-Wesley, Reading, MA, 1999.
19. Chirinos, L. and Losavio, F., "Comparison of approaches for software measurement." *Proceedings of II Workshop de Ingeniería de Software*, Jornadas Chilenas de Computación 2002 — Copiapó, Chile. November 2002.
20. Chirinos, L. and Losavio, F., "Control cuantitativo de la calidad del software en un proceso de desarrollo orientado a objetos." *XXVII Conferencia Latinoamericana de Informática CLEI'2001*, Mérida, Venezuela, September 2001.
21. Johnstone, M. and McDermid, D., "Extending and Validating the Business Rules Diagram Method." [www.vuw.ac.nz/acis99/Papers/PaperJohnstoneMcDermid-065.pdf](http://www.vuw.ac.nz/acis99/Papers/PaperJohnstoneMcDermid-065.pdf).
22. Boehm, B.W., *Characteristics of Software Quality*, TRW Inc., 1978.
23. McCall, J.A., Richards, P.K., and Walter, G.T., "Factors in Software Quality," Vols. 1, 2, 3, ADLA-049-015/055, National Technical Information Service, Springfield, MD, 1977.
24. Kitchenham, B., Linkman, S., Pasquini, A., and Nanni, V., "The SQUID approach to defining a quality model." *Software Quality Journal*, 1997.
25. Maguire, M., *User Requirements Framework Handbook: Respect*, 1997.
26. Robertson, S., "Requirements Specification Template: Volere," Edition 8, 1995.
27. Chirinos, L., Losavio, F., and Matteo, A., "Ingeniería de Requisitos de Calidad." *Proceedings of 6th Iberoamerican Workshop on Requirements Engineering and Software Environments*, IDEAS'2003, La Asunción, Paraguay, 30 Abril-2 Mayo 2003, pp. 328-335.
28. Uschold, M. and Jasper, R., "A framework for understanding and classifying ontology applications," *Proceeding of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5)*, Stockholm, Sweden, August 1999.