Assessing the Quality and the Complexity of OMT Models[1]

Marcela Genero ✠, Mª Esperanza Manso✦, Mario Piattini ✠, Francisco Jose García✣

✠ Departamento de Informática - Universidad de Castilla-La Mancha
Tel. (+34) 926 29 53 00 Ext. 3715 – Fax (+34) 926 29 53 54
Ciudad Real – Spain
{mgenero, mpiattin}@inf-cr.uclm.es
✦Departamento de Informática - Universidad de Valladolid
Tel. (+34) 983 42 30 00 – Fax 983 42 36 71
Valladolid – Spain
manso@infor.uva.es
✣ Departamento de Informática y Automática - Universidad de Salamanca
Tel (+34) 923 29 44 00 – Fax (+34) 923 29 45 14
Salamanca – Spain
fgarcia@gugu.usal.es

Abstract

Object oriented-information systems are becoming increasingly popular in industrial software development environments. Software reuse is one of the solutions for the software crisis. The reuse level was the code level, but the situation was progressing to a systematic reuse, all products got for the software development cycle were reusable products. This is the origin of the generic reusable element idea: the *asset.* Reuse will only really be a success if we can assure the quality of the assets. The first step towards improving the quality of the assets is to define quantitative and objective measures. There is no doubt that there is currently an intense interest in and demand for good OO metrics for both process and product management. We propose, in this work, two suites of closed-ended metrics for assessing the complexity an the correctness of OMT models.

Keywords: OO metrics, conceptual models quality, software reuse

1. Introduction

Object oriented-information systems are becoming increasingly popular in industrial software development environments. Software reuse is one of the solution for the software crisis. It is specially related to the object oriented paradigm as a likelihood solution which satisfies a market requiring more and more higher reliability, lower costs and faster time-to-market in services and products (Jacobso.: et al., 1997; McClure, 1997; Karlsson, 1995). First of all, the reuse level was the code level, but the situation was progressing to a systematic reuse, all products got for the software development cycle were reusable products. This is the origin of the generic reusable element idea: the *asset.* Reuse will be really a success only if we can assure the quality of the assets, therefore it is necessary to measure their quality.

The first step towards improving the quality of the assets is to define quantitative and objective measures. There is an extensive literature of software metrics, including much that pertains to object-oriented development (Meyer B., 1998). Among others, proposal for metrics for OO systems are reported in Fenton et al. (1997), Chidamber et al. (1994), Churcher et al. (1995). But surprisingly most of the proposed metrics are open-ended and subjective.

This work is based on the idea of closed-ended metrics presented by Lethbridge (1998*). Closed-ended metrics* (specially the ratio) are very useful, because they are bounded and their meaning is more relevant.

[1] *This research is part of the TIC97-0593-C05-05 subproject of the MENHIR project financed by CICYT TIC97-0593-C01 project, and also part of the MANTICA project, partially supported by CICYT and the European Union (1FD97-0168).*

The proposed metrics are intended to allow early measurement of the quality of the system under development.

We propose, two suites of closed-ended metrics:

1) The aim of the first is to quantify the quality of OMT (Object Modeling Technique, Rumbaugh (1991)) diagrams such as object diagrams and DFDs, taking into account quality factors as *correctness*.
2) And the second one, allows us to measure the *complexity* of OMT object diagram. It is common understanding that the greatest complexity is strongly correlated with the development efforts and the overall quality of the system.

This work is structured as follows. In Section 2 we show the GIRO repository and how is evaluated its quality. In Section 3 we present a set of metrics to evaluate the complexity of OMT object diagrams and we apply them to an OMT object diagram that belongs to GIRO repository. The conclusions and future work come in the last section.

2. A proposal to assess the quality of GIRO Repository assets

A research project about reuse is currently developing by GIRO Group (*Object-Oriented and Reuse Research Group* from Valladolid University, Spain); this project is directed to build a reuse model centred in a special coarse-grained reusable software element called Mecano (García et al., 1998) giving the name to the research project. Mecanos are composed of fine-grained elements classified in different abstraction levels. These elements have to be stored in an adequate repository. Mecano Project is part of the MENHIR project.

The GIRO Reuse Model (GRM) is constructed over the definition of a coarse-grained reusable element, with simultaneous support of different abstraction levels, named Mecano. This reuse model is articulated over three edges, the technical model, the process model and the qualification model (García et al., 1998).

Assets are products obtained from any life cycle phase, and they will be developed following different paradigms, structured or object oriented. According to the quality plan of the GRM (Manso et al, 1999), when an asset arrives to the repository must be audited in order to control its documentation. We know that, at the moment, in most cases the assets were developed with object-oriented methodology using OMT.

In order to develop the audit we have followed the IEEE standards. Reviews will be the principal mechanism for the audit that we describe here. According to the standard IEEE (IEEE, std 1028, 1994) an audit is defined as: *"An independent evaluation of software products or processes to ascertain compliance to standards, guidelines, specifications, and procedures based on objective criteria that include documents that specify:*

1) The form or content of the products to be produced
2) The process by which the products shall be produced
3) How compliance to standards or guidelines shall be measured"

The objective of this audit is to obtain a certified documentation in order to guarantee its correctness and completeness. First of all we inquire into the asset completeness in the sense that all mandatory documents for the used method are present, in addition we look for the presence of verification and validation documents. Second, we inquire into asset syntax or representation correctness. In any case, we are going to conclude, using measures, if the asset is qualified approval or not. The audited asset is qualitatively different from the input asset; quality increase is the result of the audit effort.

The *Checklist*, an input element, is the kernel of the audit. Depending on the asset's development phase and used method, we try to list in detail the general objectives specified before. After that, the way in which measure compliance to standards or guidelines shall be specified. We have constructed the checklist for the OMT classes and functional model. Structured model DFD was also taken into account. The next scheme was followed when constructing the mentioned checklist:

1) Identify the model that had been used in the asset documentation and the paradigm it belongs to
2) Represent in a list the model and language standard elements from the syntactic point of view
3) Associate the attribute metrics that we are going to use to each of the elements collected in the list

We have selected *attributes* from the standard model or language elements in order to make the checklists, and have collected information about the *attribute* measures. The word *attribute* is used in the sense of asset aspects that we are going to measure in contrast to the attribute that is class part.

For example, for class methods are two measures aspects: name (X_{1k}) and visibility (X_{2k}).We denote by X_{ik} the metric corresponding to the attribute i into the asset k. This metric is a part of the distance measure from the asset to an "ideal-model". A distance equal zero will be associated with assets according to all standards and guidelines (Fenton and Pfleeger, 1997) and the highest value will be associated with assets that haven't any compliance with the standards.

Following the previous example, for each asset we must consider its metric values and their frequency distribution. In the case of method name, when the name is unique and expressive the measurement will be zero, when it is unique but ambiguous will be 1, and 2 in other case. For the method visibility, if the visibility followed the standard the value will be 0 in other case 1.

All measures for this checklist take one of these ranges. It is obvious that each one must be interpreted in different way. We use an intermediary value between the best (when the standard is compliant) and the worst (when there isn't any standard) in order to asset qualification.

The *Output* contains two kinds of information, one of them with the asset situation, and the other about the audit. The asset situation includes the qualification in approval, contingent approval or disapproval, depending on the observed metrics, i.e.:

$F(X_{ik}(j)) \rightarrow$ the absolute frequency of value j corresponding to the asset k attribute i

If d(asset_k) = 0 then qualification is approval

Else if F_k(maximum) = 0 then qualification is contingent approval

Else qualification is disapproval

We define d(asset_k) and $F_k(j)$

$$d(asset\_k) = 1 - \left(\frac{F_k(0)}{\sum_j F_k(j)} \right) \qquad \sum_i F(x_{ik}(j)) = F_k(j)$$

The audit result for an asset will be disapproval if some metric have the highest value (the maximum). The reason for this classification is that highest values are consequence of the usage of incoherent terminology, and this will prevent us from a correct comprehension (or will cause misunderstanding).

Each asset will have an audit report which must include audit qualification, the error reports and conclusions summary. The audit report will include a summary about the effort, errors classification and audited asset conclusions. Furthermore, it will include recommendations about the audit process and asset documentation guidelines.

2.1 Audit Results

OMT assets	14 assets	14,29% with errors	0% with v&v documents	85,71% approval 5,17% contingent approval
DFD assets	28 assets	57,14% with errors	0% with v&v documents	42,86% approval 28,57% contingent approval 28,57% disapproval

Table 1. Results obtained in the audit process

Table 1 shows the summary of 42 audited assets. Note that OMT assets have a better qualification than the DFD assets, 85,71% approval against 42,86%. Within the DFD assets the 28,57% was disapproval. These results could indicate that has more difficulty understand/follow the DFD standards than OMT.

When we consider all the audited assets, the 21,43% were asset disapproval. There is a remarkable result, the total omission of verification and validation documentation. This will constitute an explicit recommendation for all asset suppliers.

3. A Proposal of Metrics for OMT Object Diagrams

In this section we propose a set of closed-ended metrics to evaluate the complexity of OMT object diagrams.

3.1 Metrics to measure OMT Relationships

In this section we consider three kinds of OMT relationships: associations, aggregations and generalisations.

3.1.1 ASvsC metric

The Associations vs. Classes metric measures the relation that exists between the number of associations and the number of classes in an OMT object diagram. It is based on M_{RPROP} metric proposed by Lethbridge (1998). We define this metric as follows:

$ASvsC = \left(\frac{N^{AS}}{N^{AS} + N^{C}} \right)^2$	N^{AS} is the number of associations in an OMT object diagram. N^{C} is the number of classes in an OMT object diagram. Being $N^{AS} + N^{C} > 0$.

3.1.2 AGvsC metric

The Aggregations vs. Classes metric measures the relation that exists between the number of aggregations and the number of classes in an OMT object diagram. We define this metric as follows:

$AGvsC = \left(\frac{N^{AG}}{N^{AG} + N^{C}} \right)^2$	N^{AG} is the number of aggregations in an OMT object diagram. N^{C} is the number of classes in an OMT object diagram. Being $N^{AG} + N^{C} > 0$.

We consider as the number of aggregations each level of aggregation hierarchies , ie. each symbol ◇ in the object diagram.

3.1.3 GEvsC metric

The Generalisations vs. Classes metric measures the relation that exists between the number of generalisations and the number of classes in an OMT object diagram. We define this metric as follows:

$GEvsC = \left(\frac{N^{GE}}{N^{GE} + N^{C}} \right)^2$	N^{GE} is the number of generalisations in an OMT object diagram. N^{C} is the number of classes in an OMT object diagram. Being $N^{AG} + N^{C} > 0$.

We consider as the number of generalisations each level of generalisation hierarchies, ie. each symbol △ in the object diagram.

3.1.4 N-aryAss metric

The N-ary Associations metric measures the number of N-ary associations (not binary) compared with the number of associations in an OMT object diagram. We define this metric thus:

$N-aryAss = \frac{N^{N-aryAss}}{N^{Ass}}$	$N^{N-aryAss}$ is the number of N-ary associations in an OMT object diagram. N^{Ass} is the number of associations in an OMT object diagram. Being $N^{Ass} > 0$.

3.2. Metrics to measure the characteristics of OMT classes

In this section we consider two kinds of characteristics of OMT classes: attributes and methods.

3.2.1 AvsC metric

The Attributes vs. Classes metric measures the relation that exists between the number of attributes and the number of classes in an OMT object diagram. It is based on M_{RPROP} metric proposed by Lethbridge (1998). We define this metric as follows:

$AvsC = \left(\frac{N^{A}}{N^{A} + N^{C}} \right)^{2}$	N^{A} is the number of attributes in an OMT object diagram. N^{C} is the number of classes in the OMT object diagram. Being $N^{A} + N^{C} > 0$.

For this metric we consider not only the attributes of the classes but also the attributes of associations.

3.2.2 MEvsC metric

The Methods vs. Classes metric measures the relation that exists between the number of methods and the number of classes in an OMT object diagram. We define this metric as follows:

$MEvsC = \left(\frac{N^{ME}}{N^{ME} + N^{C}} \right)^{2}$	N^{ME} is the number of methods in an OMT object diagram. N^{C} is the number of classes in an OMT object diagram. Being $N^{ME} + N^{C} > 0$.

3.3 Metrics to measure Redundancy

Generally, derived attributes, derived classes and derived associations are a source of redundancy in OMT object diagrams. In this section we define metrics to evaluate the complexity introduced by such redundancy. It would be better to remove the redundancy as much as possible.

3.3.1 DA metric

We define the Derived Attributes metric as the number of derived attributes that there are within an OMT object diagram, divided by the maximum number of derived attributes that may exist in an OMT object diagram (all attributes in an OMT object diagram except one). We define this metric as follows:

$DA = \frac{N^{DA}}{N^{A} - 1}$	N^{DA} is the number of derived attributes in an OMT object diagram. N^{A} is the number of attributes in an OMT object diagram. Being $N^{A} > 1$.

3.3.2 DC metric

We define the Derived Classes metric as the number of derived classes that there are within an OMT object diagram, divided by the maximum number of derived classes that may exist in an OMT object diagram (all classes in an OMT object diagram except one). We define this metric as follows:

$DC = \frac{N^{DC}}{N^{C} - 1}$	N^{DC} is the number of derived classes in an OMT object diagram. N^{C} is the number of classes in an OMT object diagram. Being $N^{C} > 1$.

3.3.3 DAss metric

We define the Derived Associations metric as the number of derived associations that there are within an OMT object diagram, divided by the maximum number of derived associations that may exist in an OMT object diagram (all associations in an OMT object diagram except one). We define this metric as follows:

$DAss = \frac{N^{DAss}}{N^{Ass} - 1}$	N^{DAss} is the number of derived associations in an OMT object diagram. N^{Ass} is the number of associations in an OMT object diagram. Being $N^{Ass} > 1$.

3.4 Metrics to measure Hierarchies

3.4.1 GenHer metric

The Generalisation Hierarchies metric assesses the complexity introduced by generalisation hierarchies in an OMT object diagram. It is based on M_{ISA} metric defined by Lethbridge (1998). This metric combines two factors in order to measure the complexity of the inheritance hierarchy. The first factor is the fraction of classes that are leaves of the inheritance hierarchy. This measure, called Fleaf, is calculated thus:

$FLeaf = \frac{N^{Leaf}}{N^{C}}$	N^{Leaf} is the number of leaves in a generalisation hierarchy. N^{C} is the number of classes in a generalisation hierarchy. Being $N^{C} > 0$.

On its own, Fleaf has the undesirable property that for a very shallow hierarchy (e.g. just two or three levels) with a high branching factor it gives a measurement that is unreasonably high, from a subjective standpoint. To correct this problem with Fleaf, an additional factor is used in the calculation of GenHer metric: the average number of direct and indirect superclass per non-root class, ALLSup (the root class is not counted since it cannot have parents).

GenHer metric is thus calculated using the following formula:

$$GenHer = FLeaf - \frac{FLeaf}{ALLSup}$$

This metric assesses the complexity of each generalisation hierarchy. The overall GenHer complexity is the average of all GerHer complexity in an OMT object diagram.

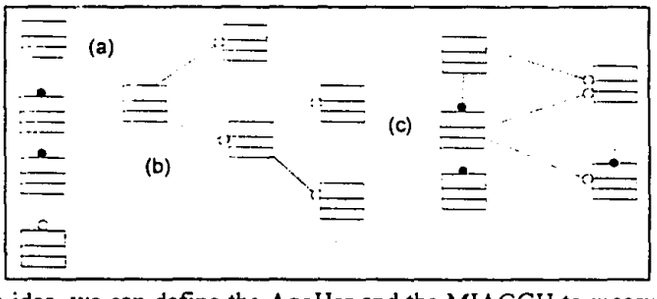
3.4.2 MIGH metric

The Multiple Inheritance for Generalisation Hierarchies metric measures the extent to which classes, in an generalisation hierarchy, have more than one parent, thus introducing complex issues associated with multiple inheritance such as the combination of values when two inherited values conflict. It just single inheritance is present, this metric is zero. It is based on M_{MI} metric defined by Lethbridge (1998). The MIGH metric measures the ratio of extra parents to each class.

We define this metric as follows:

$MIGH = \frac{N^{EX}}{N^{C}}$	N^{EX} is the number of extra parents per class in a generalisation hierarchy. N^{C} is the number of classes in a generalisation hierarchy. Being $N^{C} > 0$.

This metric assesses the complexity introduced by the multiple inheritance in each generalisation hierarchy. The overall MIGH complexity is the average of all MIGH complexities in an OMT object diagram.



Following the same idea, we can define the AggHer and the MIAGGH to measure the complexity of aggregation hierarchies.

3.5 Metrics to measure OMT Object Diagrams Cohesion

3.5.1 ODCO metric

With the Object Diagram Cohesion metric we want to assess situations like the one shown in Figure 1. It represents an object diagram composed of three unrelated subdiagrams. We are looking for a unique value, which lets us measure the cohesion degree of the different unrelated components, taking into account the number of classes in each component.

Figure 1. An Example of an OMT Diagram with three unrelated subdiagrams

We define the ODCO metric thus:

$ODCO = 1 - \frac{\sum_{i=1}^{\lvert U \rvert} (N_i^C)^2}{(N^C)^2}$	N^C is the number of classes in an OMT object diagram. $\lvert U \rvert$ is the number of unrelated subdiagrams. N_i^C is the number of classes in the subdiagram "i".

ODCO metric is zero when all the entities are in the same subdiagram, it is near to 1 when there is a high cohesion (many unrelated subdiagrams with very few classes in each one).

3.6 Interpreting the measurements of the proposed metrics

Table 2 summarises the meaning of various values of the proposed metrics Columns indicate the interpretation of measurements at the extremes of that range and in the middle.

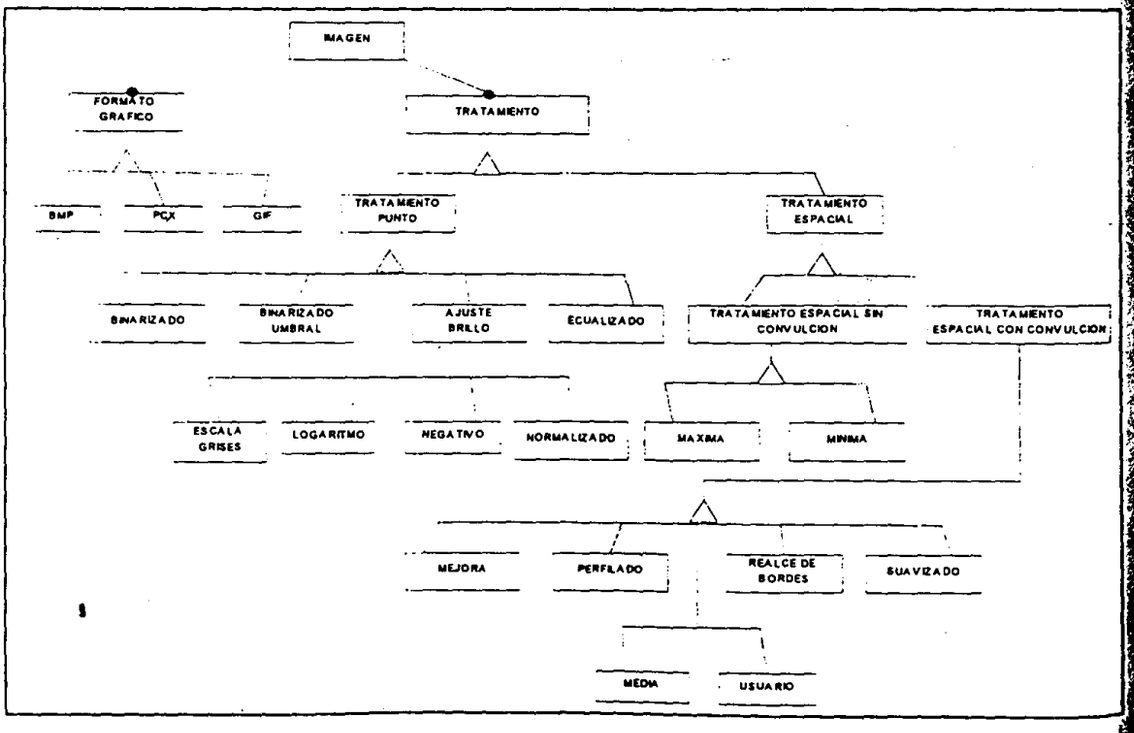
Metrics	Tends to 0 when...	tends to 0,5 when...	tends to 1 when...
AsvsC	No associations, or very few associations.	2,5 associations per class.	Very many associations per class.
AgvsC	No aggregations, or very few aggregations.	2,5 aggregations per class.	Very many aggregations per class.
GevsC	No generalisations, or very few generalisations.	2,5 generalisations per class.	Very many generalisations per class.
N-aryAass	No N-ary associations.	Half of associations are N-ary.	All of associations are N-ary
AvsC	No attributes or very few attributes.	2,5 attributes per class.	Very many attributes per class.
MEvsC	No methods or very few methods.	2,5 methods per class.	Very many methods per class.
DA	No derived attributes.	Half of attributes are derived.	All of attributes are derived.
DC	No derived classes.	Half of classes are derived.	All of classes are derived.
Dass	No derived associations.	Half of associations are derived.	All of associations are derived.
GenHer	Each subclass has about one superclass (parent).	All of generalisations hierarchies are binary trees.	Very bushy tree. A very complex hierarchy with multiple inheritance.
MIGH	No multiple inheritance.	Half of classes have an extra parents.	Very high degree of multiple inheritance.
AggHer	Each subclass has about one superclass (parent).	All of generalizations hierarchies are binary trees.	Very bushy tree. A very complex hierarchy with multiple inheritance.
MIAggH	No multiple inheritance.	Half of classes have an extra parents.	Very high degree of multiple inheritance.
ODCO	The object diagram is composed of only one diagram: $\|U\| = 1$.	The object diagram is composed of two unrelated subdiagrams and each one has the same number of classes.	A high percentage of unrelated subdiagrams, with very few classes each one.

Table 2. An Interpretation of measurements

3. 7 Applying metrics to GIRO repository assets

In this section we apply our metrics to an OMT object diagram taken from GIRO repository assets. This object diagram is shown in figure 2 The application domain of these assets is "Digital Image Processing". For more legibility we only show in figure 2 the class names.

Figure 2. An OMT Object diagram taken from GIRO repository



All the metrics values obtained for the OMT Object diagram shown in figure 2 are summarises in table 3.

Metrics	Values
ASvsC	0,0509
AGvsC	0
GEvsC	0,0351
N-aryAass	0
AvsC	0,5145
MEvsC	0,4017
DA	0
DC	0
DAss	0
GenHer	0,2153
MIGH	0
AggHer	0
MIAGGH	0
ODCO	0

Table 3. Values of the metrics applied to the example in figure 2

The Kiviat Diagram shown in figure 3, is a graphical representation of the values of the metrics shown in Table 3. This Diagram is useful because it allows designers to evaluate the overall complexity of an OMT object diagram at a glance. It also serves to compare different OMT object diagrams, and even to improve their quality.

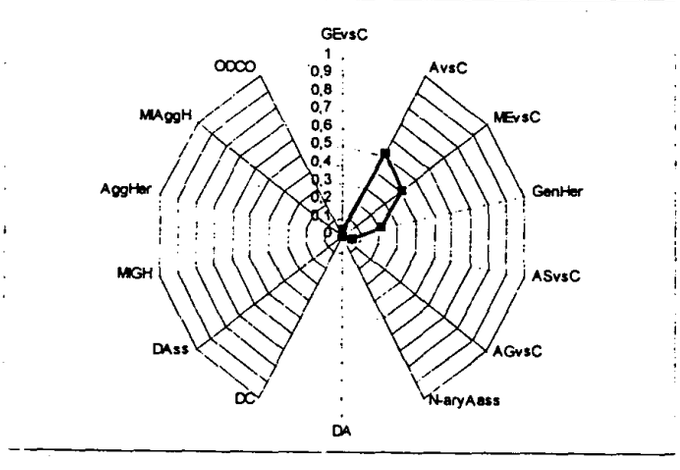


Figure 3. Complexity of the OMT object diagram shown in figure 2

5 Conclusions and future work

In the last years object oriented technology (OOT) has meant that software reuse has become a reality, but we believe that it is necessary to quantify the quality of the assets in order to assure the success of reuse. As expertise in managing object-oriented (OO) projects grows, such a body of OO metrics knowledge will become increasingly usable across the industry (Henderson-Sellers, 1996). There is no doubt that there is currently an intense interest in and demand for good OO metrics for both process and product management.

- Lethbridge, T. (1998). Metrics For Concept-Oriented Knowledge bases. *International Journal of Software Engineering and Knowledge Engineering* 8(2), 161-188.
- Manso, M., Romay M. and García, F. (1999). Repository asset audit. *Proceedings of the IV Jornadas de Trabajo MENHIR* Sedano(Burgos), Spain,11-15.
- McClure, C. (1997). *Software Reuse Techniques: Adding Reuse to the System Development Process*. PrenticeHall.
- Meyer, B. (1998). The Role of Object-Oriented Metrics. *IEEE Computer*, November, 123-125.
- Rumbaugh, J., Blaha M., Premerlani, W., Eddy, F., and Lorensen, W. (1991). *Object- Oriented Modeling and Design*. Prentice Hall. USA.
- Warmer J. and Kleppe A. (1999). *The Object Constraint Language*. Addison Wesley Longman.
- Zuse, H. (1998). *A Framework of Software Measurement*. Berlin, Walter de Gruyter.