Transport and Telecommunication, 2011, Volume 12, No 3, 41–53
Transport and Telecommunication Institute, Lomonosova 1, Riga, LV-1019, Latvia

SOFTWARE-ENGINEERING MEASUREMENT FOR LOGISTICS AND TRANSPORT SYSTEMS

Sergey Orlov, Andrei Vishnyakov

Transport and Telecommunication Institute
Lomonosova str.1, Riga, LV-1019, Latvia
E-mails: sorlov@tsi.lv, andrei.vishnyakov@gmail.com

Today logistics and transport systems are complex and expensive, so it is necessary to use modern metric suits for their development process, which takes into account the characteristics of such systems. It is very important to choose a metric suite on the early development stages and the choice should be based on objective facts and experience.

To choose the optimal metric suite it is necessary to use a technique that contains a number of steps. At each step, a specific criterion should be used to make a selection from the available metric suites. As long as the metric suite is chosen it is necessary to perform its improvement. To do so each metric in the suite should be considered separately and replaced if the analogous metrics provides a better criteria matching. In the end we obtain the modified metric suite, which is optimal according to the requirements.

Keywords: metric suite, object oriented design metrics, metrics selection algorithm, logistics and transport software, Chidamber & Kemerer's metrics suite, software package metrics, metrics for object oriented design, LCOM

1. Introduction

Modern logistics and transportation systems are complex and expensive, so it's necessary to use modern metric suits for their development process, which takes into account the characteristics of such kind of systems.

Among different groups of software metrics the suites of metrics for software design stage takes the essential part, including object-oriented and post-object-oriented design metrics. The total number of different metrics is extremely large, and they have different characteristics, as well as offer different approaches to measuring the quality of the systems. Metrics which are used to evaluate complex characteristics of systems and integrated into the groups called metric suites. As an example, one of the most commonly used metrics suite is Chidamber & Kemerer's metrics suite [1].

During the initial stage of development, it's necessary to have a technique that allows selecting a metric suite among of different suites. Also such technique should help to optimise the selected metric suite. In addition, such a choice should take into account the specific requirements for the logistics and transport systems.

2. Selection of a Metric Suite

To choose the optimal metric suite it's necessary to use a technique that contains a number of steps and based on some criteria. The criterion can be expressed as scalar or vector value and can vary in accordance with the required characteristics of logistics and transportation systems.

The algorithm for this technique consists of the following steps (see Figure 1):

1. Determination of metric suites;
2. Determination of criteria;
3. Obtaining values of the criteria;
4. Choice of the metric suite with the optimal values of the criteria.

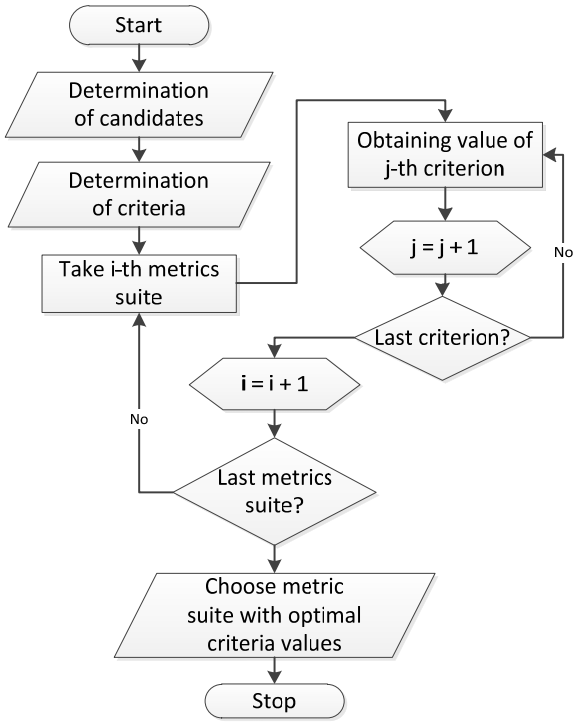

Figure 1. Flowchart representation of metric suite selection algorithm

2.1. Determination of metric suites

There are available several metric suites, which could be used at the software design stage:

- Chidamber & Kemerer's metrics suite [1, 2];
- Lorenz and Kidd metrics suite [1];
- Software Package Metrics proposed by Robert Cecil Martin [3];
- Metrics for Object Oriented Design proposed by Fernando Brito e Abreu [1, 4].

Let us take a close consideration of the Chidamber & Kemerer's metrics suite and the metrics suites proposed by Robert Cecil Martin and Fernando Brito e Abreu. These metrics suites are considered in many publications, and the calculations of values for their metrics are implemented in several commercial products.

2.1.1. Chidamber & Kemerer's metrics suite

This metric suite was proposed by Chidamber & Kemerer in 1994 [2]. The suite uses the class as a fundamental element of object-oriented system. The long usage experience of the suite has proved its efficiency and showed that it's performed well. Its widespread use just confirms it.

There are a number of works which examine and evaluate the proposed metric suite [1]. The results of such research have proved the relations between the metrics and such quality indicators as a project cost, reusability, maintenance complexity, etc. On the other hand, some of the metric in the suite are criticized and there are proposed various alternatives and modifications [5, 6, 7].

Chidamber & Kemerer proposed the following metrics [2]:

- Weighted Methods Per Class (WMC) – number of methods defined in class;
- Depth of Inheritance Tree (DIT) – maximum inheritance path from the class to the root class;
- Number of Children (NOC) – number of immediate sub-classes of a class;
- Coupling between Object Classes (CBO) – number of classes to which a class is coupled;
- Response for a Class (RFC) – set of all methods that can be invoked in response to a message to an object of the class;
- Lack of Cohesion of Methods (LCOM) – number of not connected method pairs in a class representing independent parts having no cohesion.

2.1.2. Software Package Metrics

This metric suite was proposed by Robert Cecil Martin for object-oriented software evaluation which is based on subsystem's relationship analysis [3]. Project which contains strong dependencies between the subsystems require more effort in developing and maintaining. On the other hand, it's impossible to avoid all dependencies. Robert Cecil Martin suggested the characteristics which can help to make system reusable, fault-tolerant and easy modifiable.

To measure these characteristics he proposed a number of easily applicable metrics:

- Afferent Couplings (Ca) – number of other packages that depend upon classes within the package;
- Efferent Couplings (Ce) – number of other packages that the classes in the package depend on;
- Abstractness (A) – ratio of the number of abstract classes in the analysed package to the total number of classes in the analysed package;
- Instability (I)– ratio of efferent coupling to total coupling;
- Distance from the Main Sequence (D) – perpendicular distance of a package from the idealized line A + I = 1;
- Package Dependency Cycles – packages participating in package dependency cycles.

These metrics measure the project correspondence to the ideal models of dependency and abstraction. Such measures are based on certain standards which might not be applicable for every system and as a result can only degrade the quality of the system.

The examination of this metric suite is important, since the suite is widely used for agile development, which is becoming more and more popular.

2.1.3. Metrics for Object Oriented Design

Metrics for Object Oriented Design was proposed by Fernando Brito e Abreu in 1994 [4]. The main objectives of the metric are: coverage of the basic structural mechanisms of the object-oriented paradigm, formal definition, size and language independence. All metrics are considering only the methods and attributes in the system. The results represent the average value for the entire system.

There are a number of papers devotes to this metric suite, though much smaller than for Chidamber & Kemerer's metrics suite. It is also used less frequently in practice than the previous two metric suites.

Fernando Brito e Abreu proposed the following metrics:

- Method Hiding Factor (MHF) – ratio of the sum of the invisibilities of all methods defined in all classes to the total number of methods defined in the system;
- Attribute Hiding Factor (AHF) – ratio of the sum of the invisibilities of all attributes defined in all classes to the total number of attributes defined in the system;
- Method Inheritance Factor (MIF) – ratio of the sum of inherited methods in all classes of the system to the total number of available methods for all classes;
- Attribute Inheritance Factor (AIF) – ratio of the sum of inherited attributes in all classes of the system to the total number of available attributes for all classes;
- Polymorphism Factor (POF) – ratio of the actual number of possible different polymorphic situations to the maximum number of possible distinct polymorphic situations;
- Coupling Factor (COF) – ratio of the maximum possible number of couplings in a system to the actual number of couplings not imputable to inheritance.

Each metrics in the suite reflects some basic structural mechanism of the object-oriented paradigm as encapsulation (MHF, AHF), inheritance (MIF, AIF), polymorphism (POF) and message passing (COF).

2.2. Determination of criteria

The evaluation and comparison of several metric suites isn't an easy task. Therefore the criteria, which take into consideration the requirements for the metric suite, should simplify the problem. The selection of the criteria depends on many factors, and every solution might have different criteria. In general, such criteria selection is made by software engineers and should be based on some expert evaluations.

For a criterion preference rule the maximization is used, where each criterion includes a combination of indicators. As long as not every indicator can be quantified or it might have different scales, we use a scale of correspondence for its evaluation. This scale has a range from 0 to 2, where 0 – no correspondence, 1 – average level of correspondence, 2 – full correspondence. The value of every indicator is chosen empirically.

The weighting coefficients could be used in addition to preference rule. Weight coefficients should be chosen according to the specified requirements for the system.

According to the preceding, our criterion preference rule is defined as:

$$K = \sum_{i=1}^{n} \alpha_i \cdot C_i \to \max ,$$

where:

- $\alpha_1, \alpha_2, \ldots, \alpha_n$, – weigh coefficient of preference indicators which usually meets the following conditions:

$$0 \le \alpha_i < 1, \sum_{i=1}^{n} \alpha_i = 1;$$

- C_i (i = 1, 2, …, n) – preference indicators which meet the conditions presented below:

$$\forall C_i \in R,\ R = \{0,\ 1,\ 2\}.$$

2.2.1. Determination of Indicators and Weight Coefficients

For the metric suits evaluation we use the following indicators:

- C_1 – Theory base and validation;
- C_2 – Usability;
- C_3 – Availability of tools;
- C_4 – Completeness;
- C_5 – Redundancy.

Such indicators are chosen based on the requirements for logistics and transport systems as well as metric suite requirements for such systems.

Weight coefficients are determined using the expert evaluation. Below there are listed the values of weight coefficients for our indicators:

- $\alpha_1 = 0.25$;
- $\alpha_2 = 0.20$;
- $\alpha_3 = 0.18$;
- $\alpha_4 = 0.22$;
- $\alpha_5 = 0.15$.

So, it's obvious that theory base and metric suite completeness have the greatest impact on the criterion. On the other hand, the redundancy has the smallest weight coefficient.

2.3. Obtaining values of the indicators

It's required to obtain values of the listed indicators for Chidamber & Kemerer's metrics suite and Software Package Metrics.

2.3.1. Theory base and validation

Theoretical base, empirical validation and analysis of metric suites characteristics allow us to make a decision regarding the metric suite quality.

Chidamber & Kemerer's metrics suite

In publication [2] Chidamber & Kemerer used the formal process of metric suite validation based on Weyuker's properties. The Weyuker proposed properties for metric suite evaluation which includes: noncoarseness, nonuniquencess, design details are important, monotony, non-equivalence of interaction,

interaction increases complexity. Chidamber & Kemerer conclude that all the metrics satisfy the majority of the properties prescribed by Weyuker. Only DIT and LCOM doesn't fully comply with it.

Since this metric suit almost completely corresponds to the indicator, the value of indicator $C1_{CK} = 2$.

Software Package Metrics

At the moment this metric suite hasn't any theoretical base, however, there are still a number of works devoted to the empirical validation of this metric suite [3, 8, 9].

Since there is a lack of sufficient research, the value of indicator $C1_{SPM} = 1$.

Metrics for Object Oriented Design

There are a number of works devoted to MOOD validation. For instance, in publications [4, 10] there are shown practical and theoretical validations of this metric suite. Also, this metric suite fulfills the following criteria: metrics determination should be formally defined, non-size metrics should be system size independent, metrics should be dimensionless or expressed in some consistent unit system, metrics should be obtainable early in the life-cycle, metrics should be down-scalable, metrics should be easily computable, metrics should be language independent.

As long as this metric suit has a reasonable amount of works denoted to its theoretical and practical validation, the value of indicator $C1_{MOOD} = 2$.

2.3.2. Usability

By usability we mean the convenience of metric usage, i.e. the simplicity of their calculations. It's obvious that the metrics that are easy to calculate could be understood easily. As a result they are used much wider.

Chidamber & Kemerer's metrics suite

The calculation of the metrics WMC, DIT, NOC and CBO are quite simple, so such calculations can be performed based on static models of a system. RFC calculating requires slightly bigger effort and could be done based on dynamic model of software system. LCOM metric is the most complex in the suite. This is because there are available a number of different modifications of the metric. Also the calculation requires some knowledge regarding the private data and some internal dependencies which may not be available at the design stage.

As long as there are no any significant weaknesses from the usability point of view, the value of indicator $C2_{C\&K} = 2$.

Software Package Metrics

The calculating of metrics Afferent Couplings, Efferent Couplings, Abstractness can be done using static models of a system. Such calculations aren't complex. Instability and Distance from the Main Sequence are kind of derivatives from the first three metrics. Such values obtaining shouldn't cause any difficulties as well. Package Dependency Cycles may require additional details in the static model of the system.

Since the calculation of metrics for a given suite doesn't cause major difficulties, the value of indicator $C2_{SPM} = 2$.

Metrics for Object Oriented Design

All the metrics in the given metric suite are relatively simple, so they do not require some complicated calculations. For their calculation we need to know the number of methods and attributes for every class, as well as the number of inherited methods and attributes. The problem of such calculation lies in the fact that they require to have some information regarding private methods and attributes. Such information might not be available at the design stage; hence you might need a reasonable effort to obtain them.

Since this metric suite requires additional data at the design stage, the value of indicator $C2_{MOOD} = 1$.

2.3.3. Availability of tools

Automated tools could significantly simplify the usage of metrics. This is especially useful for large projects where manual computation requires huge effort.

There are a number of tools available that allow obtaining the values of metrics automatically. In the Table 1 are listed the most popular metric tools and the level of coverage for each metrics suite.

Table 1. Metrics suite coverage level by the metric tools

Tool	Vendor	Languages	Chidamber & Kemerer's metrics suite coverage	Software Package Metrics coverage	Metrics for Object Oriented Design
Essintial Metrics	Power Software	C++, Java	Full suite	-	MHF, AHF, MIF, AIF, PF
SDMetrics	SDMetrics	C++, Java	WMC, DIT, NOC, CBO, RFC	Ca, Ce, A, I, D	-
Understand	Scientific Toolworks	C++, Java, C#, etc.	Full suite	Ce	-
JDepend	-	Java	-	Full suite	-
Ndepend	Ndepend	C#	LCOM	Ca, Ce, A, I, D	-
Eclipse metrics	-	Java	Full suite	Ca, Ce, A, I, D	-

Chidamber & Kemerer's metrics suite

Although the metric suite is supported by a large number of tools, there are some issues in the obtained results. In the publication [9] are listed the values obtained by different tools. These results indicate that different tools have a different interpretation for some of metrics, so the values are different. Based on that, the value of indicator $C3_{CK} = 1$.

Software Package Metrics

As long as there are enough tools which allow to obtain the metric suite's values, the value of indicator $C3_{SPM} = 2$.

Metrics for Object Oriented Design

Since this metrics suite is supported by a limited set of the metric tools, the value of indicator $C3_{MOOD} = 1$.

2.3.4. Completeness

This indicator evaluates how well a metric suite cover the characteristics of the system. We are interested mostly in evaluation of object-oriented systems, so completeness evaluation should be based on the characteristics of object-oriented systems.

Different researches show that the metric suites should cover the following features of the static UML models:

- Interfaces;
- Attributes;
- Methods;
- Generalization;
- Aggregation, Association, Dependencies.

The details regarding UML characteristic coverage by each metric suite are provided in table 2.

Table 2. UML features coverage by the metric suites

Scope / Suite	Interfaces	Attributes	Methods	Relationships	
				Generalization	Aggregation, Association, Dependencies
Chidamber & Kemerer's metrics suite	-	LCOM	WMC, LCOM	NOC, DIT	CBO, RFC
Software Package Metrics	A	-	-	-	Ca, Ce
Metrics for Object Oriented Design	-	AFH, AIF	MFH, MIF	POF, MIF, AIF	-

Chidamber & Kemerer's metrics suite

Chidamber & Kemerer's metrics suite reflects well the characteristics of object-oriented systems. The metric suite contains metrics which evaluate the number of methods (WMC), inheritance (NOC, DIT), external relations (CBO, RFC). In addition, the metric LCOM indirectly consider attributes and methods.

Since the most of UML features are covered by the suite, the value of indicator $C4_{CK} = 2$.

Software Package Metrics

Metrics from this suite have a fairly narrow focus. They evaluate mainly the relationships between objects/packages and the stability of the package by evaluating the number of abstract classes and interfaces.

As long as only few UML features are covered, the value of indicator $C4_{SPM} = 1$.

Metrics for Object Oriented Design

This metric suite reflects very well the characteristics of the object-oriented systems, since it was made for such types of system design evaluation. Each metric in this suite is related to one of object-oriented paradigm fundamental feature: encapsulation (MNF, ANF), inheritance (MIF, AIF), polymorphism (POF) or message passing (COF).

Based on the fact that most of the UML features are covered by this metric suite, the value of indicator $C4_{MOOD} = 2$.

2.3.5. Redundancy

In our case, the redundancy is the existence of several metrics which measure the same or similar characteristics of the system. Let's use the correlation coefficient to determine the redundancy, which determines the closeness of the linear relationship between variables (metrics).

The empirical data of cargo transport fleet management system were used to obtain correlation values.

Chidamber & Kemerer's metrics suite

Table 3 shows the correlation values for the metrics in the system.

Table 3. Correlation coefficient values for the metrics

	CBO	NOC	WMC	RFC	DIT	LCOM
CBO	1	-0.02144	**0.570007**	0.237647	0.126227	0.364266
NOC	-0.02144	1	0.155889	-0.03971	-0.16163	-0.05677
WMC	**0.570007**	0.155889	1	0.236418	0.091223	0.249562
RFC	0.237647	-0.03971	0.236418	1	**0.910849**	-0.19671
DIT	0.126227	-0.16163	0.091223	**0.910849**	1	-0.28036
LCOM	0.364266	-0.05677	0.249562	-0.19671	-0.28036	1

The values of correlation coefficient indicate that for this system there is a significant relation between CBO and WMC. Also there is a very close relation between the metrics RFC and DIT.

Since most of the metrics in the suite do not correlate with each other, the value of indicator $C5_{CK} = 2$.

Software Package Metrics

Table 4 shows the correlation values for the metrics in the system.

Table 4. Correlation coefficient values for the metrics

	Ca	Ce	A	I	D
Ca	1	-0.15545	**-0.80704**	0.370458	**0.767875**
Ce	-0.15545	1	0.285002	-0.18804	-0.22956
A	**-0.80704**	0.285002	1	**-0.65272**	**-0.87742**
I	0.370458	-0.18804	**-0.65272**	1	**0.652026**
D	**0.767875**	-0.22956	**-0.87742**	**0.652026**	1

The values of correlation coefficient show that the system has a significant number of metrics related to each other. For instance, metrics Ca and A, Ca and D, as well as A and D have a high correlation. Metrics A and I, as well as I and D have a significant relation. Part of such relations can be explained by the fact that couple of the metrics in this suite is derived from others.

As long as the correlation between the metrics are more noticeable, the value of indicator $C5_{SPM} = 1$.

Metrics for Object Oriented Design

Table 5 shows the correlation values for the metrics in the system.

Table 5. Correlation coefficient values for the metrics

	MHF	AHF	MIF	AIF	COF	POF
MHF	1	0.40704	-0.25262	-0.31742	0.04075	0.16284
AHF	0.40704	1	**-0.92571**	-0.27889	0.17063	**0.59182**
MIF	-0.25262	**-0.92571**	1	0.29818	-0.13308	**-0.52922**
AIF	-0.31742	-0.27889	0.29818	1	-0.46749	**-0.74875**
COF	0.04075	0.17063	-0.13308	-0.46749	1	**0.821254**
POF	0.16284	**0.59182**	**0.52922**	**-0.74875**	**0.821254**	1

The values of correlation coefficient shows that the system has a significant number of related metrics, especially almost every metric has significant or high correlation with POF. Partially such relations can be explained by the fact that most of the metrics take into account the inherited attributes and methods.

Since the correlation between the metrics is noticeable, the value of indicator $C5_{MOOD} = 1$.

2.4. Summary

The above described metric suites aimed of measuring the quality of the product. The main idea of all three metric suites is in evaluation of the individual classes or packages and the relationships between them. The difference is in the way how they make such evaluation. Software Package Metrics provides some generalized assessment. Chidamber & Kemerer's metrics suite provides statistics that help to a developer to make a decision regarding the product quality. Similar approach is used for Metrics for Object Oriented Design; however, instead of considering individual classes the metric suite obtains the values for whole system or package.

The total values of the indicators for the metric suites are presented in table 6.

Table 6. The values of the indicators for the metric suites

Indicator	Weight coefficients	Chidamber & Kemerer's metrics suite	Software Package Metrics	Metrics for Object Oriented Design
Theory base and validation	0.25	2	1	2
Usability	0.20	2	2	1
Availability of tools	0.18	1	2	1
Completeness	0.22	2	1	2
Redundancy	0.15	2	1	1

Hence the criteria for the considered metric suites are defined as follows:

$$K_{CK} = \sum_{i=1}^{n} \alpha_i \cdot C_{CKi} = 0.25 \cdot 2 + 0.2 \cdot 2 + 0.18 \cdot 1 + 0.22 \cdot 2 + 0.15 \cdot 2 = 1.82 .$$

$$K_{SPM} = \sum_{i=1}^{n} \alpha_i \cdot C_{SPMi} = 0.25 \cdot 1 + 0.2 \cdot 2 + 0.18 \cdot 2 + 0.22 \cdot 1 + 0.15 \cdot 1 = 1.38 .$$

$$K_{MOOD} = \sum_{i=1}^{n} \alpha_i \cdot C_{SPMi} = 0.25 \cdot 2 + 0.2 \cdot 1 + 0.18 \cdot 1 + 0.22 \cdot 2 + 0.15 \cdot 1 = 1.47 .$$

The values of the criteria for the metric suites lead us to the conclusion that Chidamber & Kemerer's metrics suite is more appropriate according to the requirements.

3. Metric Suite Optimisation

After the metric suite is chosen it's necessary to perform its improvement. To do so each metric in the suite should be considered separately, analogous metrics should be determined and compared against the original metric. After such comparisons, the decision should be made regarding the original metric replacement with the analogous one. After these steps we obtain the modified metric suite which is optimal according to the requirements.

The algorithm for this technique consists of the following steps (see Figure 2):

1. Determination of criteria;
2. Take *i*-th metric;
3. Determination of metric alternatives;
4. Obtaining criteria values for every alternative;
5. Replacing the metric with the alternative if alternative's criteria values are more optimal;
6. If it isn't the last metric in the suite then take the next metric (return to step 2).

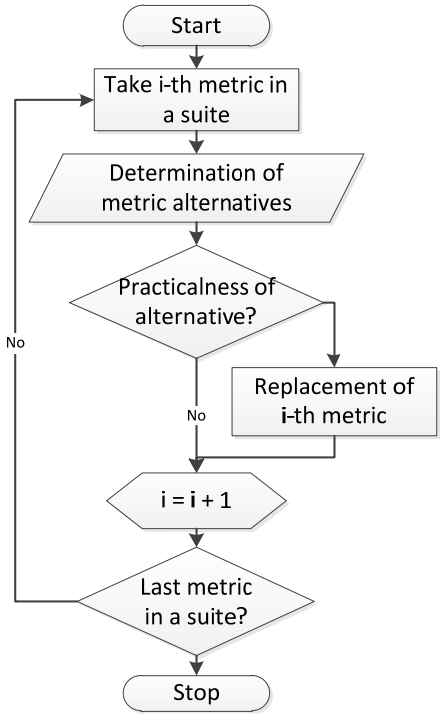

Figure 2. Flowchart representation of metric suite optimisation algorithm

3.1. LCOM metric

Let's make an improvement for one of the metric from Chidamber & Kemerer's metrics suite. Perhaps the most criticism in this suite was subjected to LCOM metric. Various studies have shown a lot of it weaknesses, so it should be used carefully. For instance, completely different classes can have the same value and often this metric very poorly reflects the cohesiveness of the class [7].

For these reasons there were proposed different variations and alternatives of this metric. In [1, 5, 6, 7, 11] are presented the following LCOM alternatives:

- LCOM1, LCOM2, LCOM3, LCOM4, LCOM5;
- Connectivity (Co);
- Unified Cohesion (Coh);
- Tight class cohesion (TCC);
- Loose class cohesion (LCC);
- Information-flow-based cohesion (ICH).

As long as there are so many alternatives, it's reasonable to determine the most appropriate metric which satisfy well the requirement of the system.

3.2. Determination of LCOM alternatives

Due to the fact that many of these metrics are computed in a similar way, let's select the most characteristic of them which are listed below:

LCOM1 – this metric was introduced by Chidamber & Kemerer in 1991 [12];

LCOM2 – this metric also known as LCOM and was introduced by Chidamber & Kemerer in 1994 [2];

LCOM4 – LCOM modification where value calculated as the number of connected vertices in undirected graph [6];

LCOM-HS – LCOM modification proposed by Henderson-Sellers [5, 11];

LCOM% – LCOM modification used in Understand Metrics tool and expressed in per cent.

3.3. Determination of Criteria

The criterion preference rule for metric suite optimization is similar to the criterion for metric suite selection and is defined as:

$$K = \sum_{i=1}^{n} \alpha_i \cdot C_i \to \max,$$

where:

- $\alpha_1, \alpha_2, \ldots, \alpha_n$, – weigh coefficient of preference indicators;
- C_i (i = 1, 2, …, n) – preference indicators.

For a metric suite optimization indicators should be chosen basing on the requirements for logistics and transport systems as well as the metric requirements for such systems. We are also using the same scale of correspondence for its evaluation.

3.3.1. Determination of Indicators and Weight Coefficients

According to our requirements for the considered systems, we use the following indicators for the metrics evaluation:

- C_1 – Usability;
- C_2 – Availability of tools;
- C_3 – Correlation with the cluster.

Weight coefficients are determined using the expert evaluation. Below are listed the values of weight coefficients for our indicators:

- $\alpha_1 = 0.35$;
- $\alpha_2 = 0.25$;
- $\alpha_3 = 0.4$.

We can see that indicator C_3 has the greatest impact on the criterion and C_2 – the lowest one.

3.4. Obtaining values of the indicators

It's required to obtain values of the listed above indicators for each metric.

3.4.1. Usability

Similarly as for metric suite selection, by usability we mean the convenience of metric usage, i.e. the simplicity of their calculations. Let's examine a calculation algorithm for each metric.

LCOM1

The initial metric defined by Chidamber & Kemerer, later it was replaced by LCOM2. The value of the metric is the number of pairs of methods in the class using no attribute in common.

LCOM2

Classical metric from Chidamber & Kemerer's metrics suite, it is also known as LCOM and has the following formula:

$$LCOM = \begin{cases} |P| - |Q|, if\ (P > Q) \\ 0, otherwise \end{cases},$$

where:

- P – number of pairs of methods without shared instance variables;
- Q – number of pairs of methods with shared instance variables.

LCOM4

For this metric calculation an undirected graph should be constructed, where the vertices are the methods of a class, and there is an edge between two vertices if the corresponding methods share at least one instance variable or one method explicitly uses another one. The result of the metric equals to the total number of edges between vertices.

LCOM-HS

Henderson-Sellers revise the LCOM metric to normalize it for the number of methods and variables that represent in the class, so they proposed the following definition [5]:

$$LCOM = \frac{\left(\frac{1}{a}\sum_{j=1}^{a}\mu(A_j)\right) - m}{1 - m},$$

where:

- m – the number of methods in the class;
- a – the number of variables;
- $\mu(A_j)$ – the number of methods of the class accessing a particular variable.

This metrics takes its values in the range [0–2].

LCOM%

It's a variation of LCOM metric where the value is expressed as a percentage. At first stage the metrics calculates what percentage of class methods use a given class instance variable. Next the average percentages for all of that class's instance variables is calculated and subtracted from 100%.

Results

Since the computational cost for all the metrics are similar (it is necessary to determine the relations between the methods or method and attributes) the indicators have the following values:

$C1_{LCOM1} = C1_{LCOM2} = C1_{LCOM4} = C1_{LCOM\ HS} = C1_{LCOM\%} = 1.$

3.4.2. Availability of tools

In the Table 7 are listed the information regarding the tools that provide the calculation of different LCOM variations.

Table 7. LCOM variations support by different tools

Tool	Vendor	Supported LCOM type
Essintial Metrics	Power Software	LCOM2
Understand	Scientific Toolworks	LCOM%
Ndepend	Ndepend	LCOM HS
Eclipse metrics	-	LCOM2, LCOM HS

We can see that there are no automated tools that calculate the metrics LCOM1 and LCOM4.
Thus, the values of indicator are the following:

- $C2_{LCOM1} = C2_{LCOM4} = 0$;
- $C2_{LCOM2} = C2_{LCOM\ HS} = C2_{LCOM\%} = 2$.

3.4.3. Correlation with the cluster

To determine this indicator the set of objects of the considered transport system was split into the three following clusters:

- Classes with low cohesion;
- Classes with mean cohesion;
- Classes with high cohesion.

Table 8 shows the correlation between the LCOM variations and the clusters variable for the considered system.

Table 8. The values of the correlation between the LCOM variations and the cluster variable

	Cluster
LCOM1	**0.5878015009**
LCOM2	**0.6375499653**
LCOM4	0.251728185
LCOM HS	**0.7276782418**
LCOM%	**0.558340097**

LCOM-HS has close relation with the cluster variable, thus $C4_{LCOM\ HS} = 2$.
Metrics LCOM1, LCOM2, LCOM% has significant relation with the cluster variable, so the value of indicators $C4_{LCOM1} = C4_{LCOM2} = C4_{LCOM\%} = 2$.
LCOM4 doesn't have a significant relation with the cluster variable, so $C4_{LCOM4} = 0$.

3.5. Summary

In table 9 are listed all values of the indicators for the metrics.

Table 9. The values of the indicators for the metrics

Indicator	**Weight coefficients**	**LCOM1**	**LCOM2**	**LCOM4**	**LCOM-HS**	**LCOM%**
Usability	0.35	1	1	1	1	1
Availability of tools	0.25	0	2	0	2	2
Correlation with the cluster	0.4	1	1	0	2	1

Hence the criteria for the considered metrics are defined as follows:

$$K_{LCOM1} = \sum_{i=1}^{n} \alpha_i \cdot C_{LCOM1i} = 0.35 \cdot 1 + 0.25 \cdot 0 + 0.4 \cdot 1 = 0.75.$$

$$K_{LCOM2} = \sum_{i=1}^{n} \alpha_i \cdot C_{LCOM2i} = 0.35 \cdot 1 + 0.25 \cdot 2 + 0.4 \cdot 1 = 1.25.$$

$$K_{LCOM4} = \sum_{i=1}^{n} \alpha_i \cdot C_{LCOM4i} = 0.35 \cdot 1 + 0.25 \cdot 0 + 0.4 \cdot 0 = 0.35.$$

$$K_{LCOM-HS} = \sum_{i=1}^{n} \alpha_i \cdot C_{LCOM-HSi} = 0.35 \cdot 1 + 0.25 \cdot 2 + 0.4 \cdot 2 = 1.65.$$

$$K_{LCOM\%} = \sum_{i=1}^{n} \alpha_i \cdot C_{LCOM\%i} = 0.35 \cdot 1 + 0.25 \cdot 2 + 0.4 \cdot 1 = 1.25.$$

The values of the criteria for the metrics lead us to the conclusion that metric LCOM-HS is the most appropriate according to the requirements.

4. Conclusions

In this paper the technique that allows making a decision regarding metric suite selection for logistics and transportation software is proposed. The technique consists of several steps where at each step a specific criterion should be used to make a selection from the available metric suites. There may be used several different criteria indicators for a metric suite like completeness, usability, availability of tools, etc. Such indicators should be changed in accordance with the required characteristics of logistics and transport systems.

As long as the metric suite is chosen it's necessary to perform its improvement. To do so each metric in the suite should be considered separately, analogous metrics should be determined and compared against the original metric. After the comparisons, the decision should be made regarding the original metric replacement with the analogous one. In the end we obtain the modified metric suite which is optimal according to the requirements.

For efficiency and usability evaluation of the proposed technique a series of experiments and empirical research were performed. During the experiments Chidamber & Kemerer's metrics suite, Software Package Metrics and Metrics for Object Oriented Design were used. The experiment showed that Chidamber & Kemerer's metrics suite is more suitable according to the defined requirements.

During metric suite improvement step LCOM metric were analysed, and several alternatives were proposed. The results of this experiment showed that LCOM-HS has the best fit according to the requirements.

The results indicate that the proposed technique is applicable for metric suite selection for logistics and transport systems development process.

References

1. Orlov, S. A. *Software Engineering: A Textbook for Universities, 3rd ed*. SPb.: Piter, 2004. 527 p. (In Russian)
2. Chidamber, S. R., Kemerer, C. F. A Metrics Suite for Object Oriented Design, *IEEE Transactions on Software Engineering,* Vol. 20, No 6, 1994, pp. 476–493.
3. Martin, R. C. *Agile Software Development: Principles, Patterns and Practices.* Pearson, 2003. 552 p.
4. Abreu, F. B., Carapuca, R. Object-Oriented Software Engineering: Measuring and Controlling the Development Process. *Proceedings of the 4th International Conference on Software Quality, McLean, Virginia, USA, Oct. 3–5, 1994*.
5. Henderson-Sellers, B., Constantine, L., Graham, I. Coupling and Cohesion: towards a valid metrics suite for object-oriented analysis and design, *Object-oriented Systems,* Vol. 3(3), 1996, pp. 143–158.
6. Hitz, M., Montazeri, B. Measuring Coupling and Cohesion in Object-Oriented Systems. *Proc. Int'l Symp. Applied Corporate Computing (ISACC '95), Monterrey, Mexico, Oct. 25–27, 1995*.
7. Gupta, B. S. *A Critique of Cohesion Measures in the Object-Oriented Paradigm: Master of Science Thesis.* Michigan Technological University, Department of Computer Science. 1997. 49 p.
8. Briand, L., Wüst, J., Lounis, H. Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs, *Empirical Software Engineering: An International Journal*, Vol. 6, No 1, 2001, pp. 11–58.
9. Lincke, R., Lundberg, J., and Löwe, W. Comparing Software Metrics Tools, *ISSTA'08, July 20–24, 2008, Seattle, Washington, USA,* 11 p.
10. Harrison, R., Counsell, S. J., Nithi, R. V. An Evaluation of the MOOD Set of Object-Oriented Software Metrics, *IEEE Transactions on Software Engineering, vol. 24, pp. 491–496,* June 1998.
11. Henderson-Sellers, B. *Object-oriented metrics: measures of complexity.* Prentice-Hall, 1996, pp. 142–147.
12. Chidamber, S. R., Kemerer, C. F. *Towards a Metrics Suite for Object Oriented Design.* OOPSLA'91, pp. 192–211.