MÉTRICAS ORIENTADAS A OBJETOS (OO)	conceptos, importancia y tipos

En [li1993object] y citando a otros varios autores, se resume claramente el propósito e importancia de las métricas y medición del software (SW) en la ingeniería de SW. Y aunque las métricas OO parecieran ser poco estudiadas, existen suficientes fuentes de referencia sobre investigación acerca de ellas y su validación para el desarrollo de sistemas de software bajo la perspectiva OO y que pueden ayudar a seleccionar un conjunto de métricas de acuerdo a las necesidades de los equipos de desarrollo de SW.

> *"[...]Software metrics can provide a quantitative means to control the software development process and the quality of software products […].*
>
> ***Software metrics***
>
> *Software metrics measure certain aspects of software. Software metrics can be generally divided into two categories: sofware product metrics and software process metrics. Software product metrics measure software products such as source code or design documents. Software process metrics measure the software develoments process such as the number of man hours charged to the development activitiess in the design and coding phases."*

Respecto a la calidad del producto, se suele observar las características internas y externas. Las internas están relacionadas a la manera en que se ha realizado la estructura interna del código y las externas, son las observables por el usuario final. Según [http://agile.csc.ncsu.edu/SEMaterials/OOMetrics.htm] citando a varios autores:

> *"**Object-Oriented Metrics***
>
> *Increasingly, object-oriented measurements are being used to evaluate and predict the quality of software [16]. A growing body of empirical results supports the theoretical validity of these metrics [3, 5, 14, 19]. The validation of these metrics requires convincingly demonstrating that (1) the metric measures what it purports to measure (for example, a coupling metric really measures coupling) and (2) the metric is associated with an important external metric, such as reliability, maintainability and fault-proneness [11]. Often these metrics have been used as an early indicator of these externally visible attributes, because the externally visible attributes could not be measures until too late in the software development process."*

Sobre los tipos y cantidad de métricas OO de acuerdo a [Bundschuh2008]:

> *"[…] More than 200 different object-oriented metrics have been propagated over the past two decades. Zuse (1997) has identified more than 130 of them in A Framework for Software Measurement (page 568) and has partially characterized them.[...]".*

Y también en [2006AggarwalSingh]:

> *"[...] the number of metrics available in literature is large, it becomes a tedious process to understand the computation of these metrics and draw inferences from their values. [...], the*

number of metrics proposed in literature are quite large as compared to number of features (e.g., coupling, cohesion, polymorphism, size and inheritance) captured by these metrics [...]".

Se deben repasar las peculiaridades del paradigma OO:

- Atributos y clases de objetos,
- Clases con atributos y métodos,
- Cohesión,
- Acoplamiento,
- Interfaces,
- Mensajes,
- Métodos,
- Objetos,
- Identidad de los objetos,
- Polimorfismo.

Es necesario esbozar los principios del Diseño OO:

- Abstracción de datos,
- Ocultamiento de la información,
- Modularidad por encapsulamiento de datos e interfaces bien definidas,
- Dinamismo y flexibilidad por instanciación,
- Reutilización de código por herencia y agregación.

Respecto a las categorías en que normalmente se organizan las métricas OO, nuevamente de acuerdo a [Bundschuh2008]:

"Generally, object-oriented metrics are characterized by unclear definitions, and they are nt based on extensive structures, contravening the prerequisite rule for a good metric. Object-oriented software metrics are often used to measure ***complexity, maintenance and clarity.*** *As such, object-oriented metrics are mostly quality metrics and can be categorized into the following three groups:*

- *system metrics, e.g., number of files, classes and inheritance trees*
- *Tree metrics, e.g., number of children (NOC) or classes*
- *Class metrics, e.g., number of methods."*

[…] the metrics suite could be reduced to five relevant and quantifiable measures for evaluating the size and complexity of object-oriented software:

- *The number of weighted methods per class (WMC) is an indicator of system size. The weight in this case is caused by complexity of the respective method*
- *The depth of the inheritance tree measures the complexity of the system design*
- *The NOC is a measure of the reusabilirty of a class*

- *The degree of coupling between classes indicates the degree to which the classes are independent. This is an important indicator for understanding classes and their division into subclasses. Strong coupling indicates a mal-practice of modular design*
- *The response behavior o a class."*

REFERENCIAS

[Bundschuh2008] Object-Oriented Metrics, The IT Measurement Compendium: Estimating and Benchmarking Success with Functional Size Measurement, p. 241—255, 2008, Springer Berlin Heidelberg. Disponible para lectura en https://goo.gl/xLJ1db

[li1993object] Li, W., & Henry, S. (1993). Object-oriented metrics that predict maintainability.*Journal of systems and software*, *23*(2), 111-122.

[2006AggarwalSingh] K.K.Aggarwal, Yogesh Singh, Arvinder Kaur, Ruchika Malhotra, Empirical Study of Object-Oriented Metrics, ETH Zurich, Chair of Software Engineering ©JOT, 2006 Vol. 5, No. 8, November-December 2006.