17 ноября 2018 г.

Reva P., Volobuieva O., Saukh D., Bahno O.

|  |
| --- |
|  |
| Achievement Quality Attributes |
|  |
| ProFinder |

Achievement Quality Attributes

ProFinder

Content

[Quality attributes types 2](#_Toc530225319)

[Performance 3](#_Toc530225321)

[Interoperability 4](#_Toc530225323)

[Usability 5](#_Toc530225325)

[Reliability 6](#_Toc530225327)

[Availability 7](#_Toc530225329)

[Security 8](#_Toc530225331)

[Maintainability 9](#_Toc530225333)

[Modifiability 10](#_Toc530225335)

[Testability 11](#_Toc530225337)

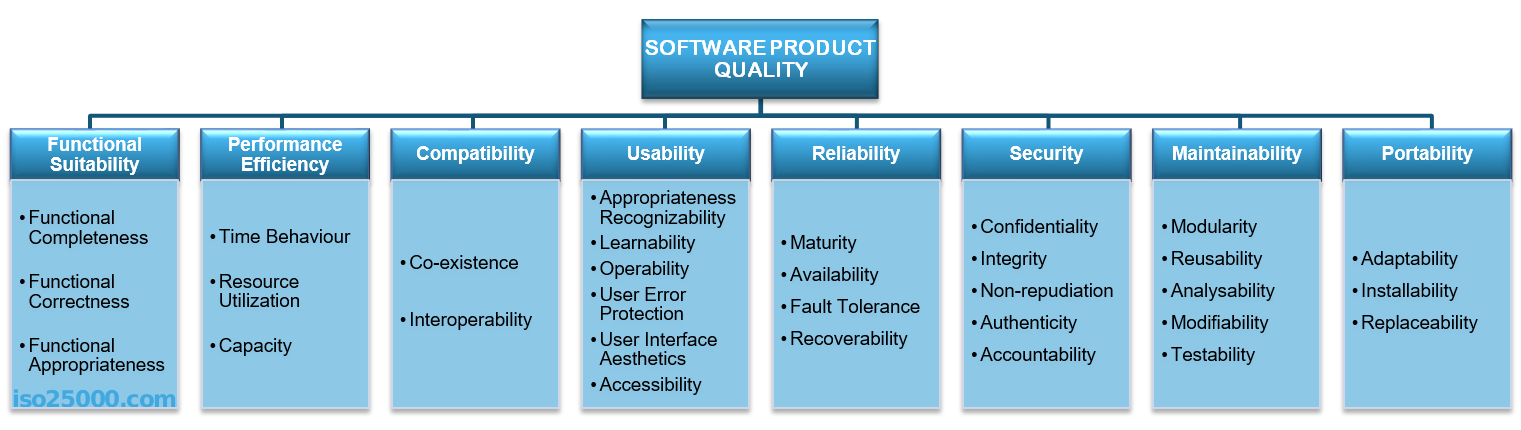
[Scalability 12](#_Toc530225339)

[Reusability 13](#_Toc530225341)

[Supportability 14](#_Toc530225343)

Quality attributes types

To consider the types of quality attributes, we can use a diagram from ISO 25010:



This standard describes the quality attributes for a software product. Next, we’ll look at what exactly each attribute means individually.

|  |  |
| --- | --- |
|  | Performance |

**Performance** shows the response of the system to performing certain actions for a certain period of time.

There **are two ways how to measure performance**:

* **Latency:** Time spent on responding to an event
* **Channel capacity**. The number of events that occur at a certain point in time.

In practice, **the possible performance indicators** include, for example:

* Average/maximum number of system users per time unit.
* Average page load time.
* Average method execution time.

Performance issues very often grow into problems that can affect everything, from the server’s capacity or the ways in which you develop your front-end to the efficiency of database queries or the capacity of communication channels.

Performance is almost always included in the list of key quality attributes that need to be considered by the architect, since it affects the entire system and can affect many parts of the architectural solution. Therefore, on the internet, you can find a large number of examples of how to deal with performance problems.

|  |  |
| --- | --- |
|  | Interoperability |

Interoperability is an attribute of the system or part of the system that is responsible for its operation and the transmission of data and its exchange with other external systems. A well-designed system facilitates integration with third-party systems. To improve the interoperability, you can use well-designed external interfaces, standardization systems, etc.

Naturally, there are a lot of problems for interaction:

* Outdated external systems.
* Different formats of data in similar external systems.
* Different versions of the API in external systems.
* Backwards compatibility of the API for integration.
* Poor quality and lack of standards of external systems.

Interoperability cannot be ignored. In the best case, you will have to create additional layers for the interaction API. At worst, it will be necessary to rebuild the entire system.

|  |  |
| --- | --- |
| C:\Users\Olha Bahno\Desktop\1_KsgJ5E7iSJtMT_jFH2Fb6g.png | Usability |

**Usability** is one of the most important attributes, because, unlike in cases with other attributes, users can see directly how well this attribute of the system is worked out. One of the key problems of usability is too much interaction or too many actions necessary to accomplish a task. Incorrect sequences of steps in multistage interfaces are also a problem of usability. Data elements and controls may be designed not according to the accepted patterns of user experience, which also complicates the interaction. For example, if you are developing an iOS application, then it is important to use [the guidelines from Apple](https://developer.apple.com/ios/human-interface-guidelines/overview/themes/), or [the guidelines from Microsoft](https://msdn.microsoft.com/ru-ru/library/windows/desktop/dn688964%28v=vs.85%29.aspx) — for Windows desktop applications.

Examples of important indicators for this attribute are:

* List of supported devices, OS versions, screen resolutions, and browsers and their versions.
* Elements that accelerate user interaction, such as “hot keys”, “lists of suggestions”, and so on.
* Average time a user needs to perform individual actions.
* Support of accessibility for people with disabilities.

|  |  |
| --- | --- |
|  | Reliability |

Reliability is an attribute of the system responsible for the ability to continue to operate under predefined conditions. Most often, the system fails due to the inaccessibility of external elements, such as databases, systems, and network connections.

|  |  |
| --- | --- |
|  | Availability |

Availability is part of reliability and is expressed as the ratio of the available system time to the total working time. Important indicators for this attribute are:

* Availability.
* Planned downtime.
* Time needed to update the software, and so on.

Availability is often expressed in the number of nines after the comma, that is nines of availability (hours / minutes / seconds):

* 2 9’s (99%) = up to 87.6h / 5256.0m / 315360.0 seconds of downtime per year.
* 3 9’s (99.9%) = up to 8.76h / 525.6m / 31536.0 seconds of downtime per year.
* 4 9’s (99.99%) = up to 0.876h / 52.559999999999995m / 3153.6 seconds of downtime per year.
* 5 9’s (99.999%) = up to 0.0876h / 5.256m / 315.36 seconds of downtime per year.
* 6 9’s (99.9999%) = up to 0.00876h / 0.5256000000000001m / 31.536 seconds of downtime per year.
* 7 9’s (99.99999%) = up to 8.76E-4h / 0.05256m / 3.1536 seconds of downtime per year.

|  |  |
| --- | --- |
|  | Security |

Security is responsible for the ability of the system to reduce the likelihood of malicious or accidental actions as well as the possibility of theft or loss of information. There are a number of measures that are used to protect systems: authentication, encryption, audit, and others.

Examples of this attribute in the work of the system are:

* The ability of the system to detect DDoS attacks and respond to them.
* Restrictions of user access in accordance with authentication/authorization.
* Prevention of SQL injection.
* Encryption of passwords and content.
* Secure connection.

|  |  |
| --- | --- |
| C:\Users\Olha Bahno\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1_dYKARf5sU8UN1K1zRcZ-Uw.png | Maintainability |

**Maintainability** is the ability of the system to support changes. Changes can be related to new business requirements or correction of old errors and affect system components or separate methods. Also, maintainability affects the time needed to restore the system after a failure. Excessive dependencies between components have a very negative effect on maintainability. In programming, there is a notion of anti-pattern spaghetti code which means excessive coherence in the code. In architecture, there is no such thing, but architecture is very close to programming in this sense. It is because of the maintainability attribute that such concepts as separation of responsibility, microservice architectures, and modularity have appeared. At the same time, this attribute affects not only development processes, but also management processes (for example, splitting teams into product-related parts).

|  |  |
| --- | --- |
| C:\Users\Olha Bahno\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1_WhUlcWKXu7H7f2UVVWxoRw.png | Modifiability |

Modifiability determines how many common changes need to be made to the system to make changes to each individual item. Ideal is the case where each change affects only one element.

|  |  |
| --- | --- |
|  | Testability |

Testability shows how well the system allows performing tests, according to predefined criteria. In addition to testing performance, testability makes it possible to effectively divide the system into subsystems.

The main indicators for this attribute are:

* Percentage of coverage with modular, integration, or unit tests.
* The final list of required test environments as well as the final list of used approaches to testing (manual/automatic, regression, integration, etc.).

|  |  |
| --- | --- |
| C:\Users\Olha Bahno\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1_Gm-x491hP7SFY5mCZo9dXw.png | Scalability |

Scalability is the ability of the system to handle load increases without decreasing performance, or the possibility to rapidly increase the load.

There are two ways to improve scalability:

* Vertical: To increase, we add more resources, such as memory, disks or processors into one system.
* Horizontal: We increase the number of computing units and divide the load.

The key indicators for measuring this attribute are:

* If the system allows for horizontal scaling.
* The time needed to increase scaling, in seconds.
* Scaling limitations: the number of servers or the network capacity.
* Possibility to scale: the increase in the number of transactions or the amount of content.

And this is only a small part of the indicators which you need to follow when designing. Scalability is one of the most important attributes, no matter what stage the project is at.

|  |  |
| --- | --- |
|  | Reusability |

Reusability is a chance of using a component or system in other components/systems with small or no change. Allocation of responsibilities, modularization, decreasing of copy-paste are all about reusability. Copying code, or worse, using different components for the same result in different places, is one of the biggest problems of reusability.

|  |  |
| --- | --- |
|  | Supportability |

Supportability is the ability of the system to provide useful information for identifying and solving problems. The main problems in ensuring supportability can be addressed with the following means:

* No diagnosis: How the activity and performance of the system are controlled. This includes various types of logging.
* No tools for troubleshooting: This includes backups, various systems for creating snapshots of the system, and tools for auditing the system. When the system fails, it is always more pleasant to wait for an automatic restart than to solve the issue manually.
* No health checking: This includes a variety of systems for measuring compilation time, deployment time, database size, or mobile application size.

Most often these are not considered in start-ups or small projects initially. The cost of maintaining the supportability attribute is high, and the result is only visible on a large scale. However, with the growth of the team and the product, this attribute becomes one of the key ones.