Software Architecture System
Quality Attributes

Revision History

Generated version Brenton Ross Sept. 2011

Copyright

Contents

1 Introduction

The driving force for the architecture of a system are the quality requirements of the desired system. This document lists the quality attributes that might have an impact on the design.

This document was generated from the Software Architecture Support System (SASSY) Quality Attribute ontology.

1.1 Scope

This set of quality attributes is defined by the Ontological Quality Model.

1.2 Audience

This document is designed to be used by analysts who are tasked with determining the non-functional requirements for a system. This list of quality attributes can be used as a guide to ensure that no important quality requirements are ommitted.

2 Quality Attributes

2.1 Computational Attributes

Access Audit The ability to determine who accessed the system or data or components therein.

Access Control The ability of the system to control who can access the system.

Accessibility Accessibility is a general term used to describe the degree to which a product, device, service, or environment is accessible by as many people as possible. Accessibility can be viewed as the "ability to access" and possible benefit of some system or entity. Accessibility is often used to focus on people with disabilities and their right of access to entities, often through use of assistive technology.

Accountability Describes how easy it is to determine if the system is liable for any changes made to the data.

Accuracy The state of being accurate; freedom from mistakes, this exemption arising from carefulness; exact conformity to truth, or to a rule or model.

Attractivity This is a subjective quality describing how attractive the application is to its users. It could be measured by asking a sample of users to rate the application on how pleasing it is to look at. An attractive application might not necessarily be more useable since it may take more resources to render and the UI components might look nice, but they might also be less useable.

Auditability Concerns the transparency of a system with regards to external audits.

Availability The degree to which a system, subsystem, or equipment is operable and in a committable state at the start of a mission, when the mission is called for at an unknown, i.e., a random, time. Simply put, availability is the proportion of time a system is in a functioning condition.

Capacity How much work will the system be required to handle?

Communicativeness Refers to how well the system communicates with its users.

Compliance Refers to how well the system matches various requirements that were placed on it.

Confidentiality How well the system prevents access to sensitive data by unauthorised people.

Correctness In theoretical computer science, correctness of an algorithm is asserted when it is said that the algorithm is correct with respect to a specification. Functional correctness refers to the input-output behaviour of the algorithm (i.e., for each input it produces the correct output).

Credibility Refers to the objective and subjective components of the believability of a source or message.

Degradability Refers to how well the system copes with reduced quality on its inputs. It can either be a graceful degradation where the system loses some functionality, or catastrophic where service is halted completely.

Demonstrability How easy it is to demonstrate the system.

Dependability The trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers

Determinisability Is the system deterministic? Will it always produce the same result for the same input?

Distributability Refers to how easy it is to spread the system across multiple computers.

Durability Refers to the the ACID property which guarantees that transaction's that have committed will survive permanently.

Effectiveness The accuracy and completeness of users' tasks while using a system.

Efficiency The extent to which a resource, is used for the intended purpose.

Execution Efficiency The extent to which the CPU resources are used for the intended purpose.

Fault Tolerance How well the system copes when things start to go wrong.

Fidelity Refers to the degree to which a model or simulation reproduces the state and behaviour of a real world object, feature or condition. Fidelity is therefore a measure of the realism of a model or simulation

Flexibility The ease with which the system can respond to uncertainty in a manner to sustain or increase its value delivery.

Functionality The proportion of the required functions that the system has implemented and are useable.

Helpfulness Describes how easy it is for users to get information on how to use the system.

Integrity Ensuring that information is not altered by unauthorized persons in a way that is not detectable by authorized users.

Interoperability The capability of a product or system – whose interfaces are fully disclosed – to interact and function with other products or systems, without any access or implementation restrictions.

Latency How quickly the system responds to inputs.

Learnability The capability of a software product to enable the user to learn how to use it.

MTBF The average time between failures of the system that are noticed by its users.

Operability The ability of products, systems and business processes to work together.

Performance Computer performance is characterized by the amount of useful work accomplished by a computer system compared to the time and resources used.

Precision A measure of the detail in which a quantity is expressed.

Predictability The degree to which a correct prediction or forecast of a system's state can be made either qualitatively or quantitatively.

Processing Traceability Refers to the ability to trace the execution path through the running system.

Recoverability Refers to how easy it is to get the system going again after a crash.

Recovery Time The time taken for the system to recover from a failure.

Relevance Refers to how closely a system matches the needs of its potential users.

Reliability A reliable protocol is one that provides reliability properties with respect to the delivery of data to the intended recipient(s)

Repeatability The variation in measurements taken by a single person or instrument on the same item and under the same conditions

Resilience How well the system copes with the unexpected.

Resource Behaviour Refers to the profile of how the system uses the computers resources, such as CPU, disk, and communications.

Response Time The time taken for the system to respond to a stimulus.

Responsiveness Describes how quickly it responds to user input.

Robustness The quality of being able to withstand stresses, pressures, or changes in procedure or circumstance.

Safety The condition of being protected against physical, social, spiritual, financial, political, emotional, occupational, psychological, educational or other types or consequences of failure, damage, error, accidents, harm or any other event which could be considered non-desirable.

Security Refers to how well the system prevents unauthorised actions.

Self-Descriptiveness Refers to how well the system is able to describe itself.

Storage Efficiency The extent to which memory and disk are used for the intended purpose.

Suitability Refers to how well the system meets the user's requirements.

Throughput The amount of work that the system can handle in a given time interval.

Time Behaviour Refers to how the system responds to a step change in its inputs.

Useability The elegance and clarity with which the interaction with a computer program or a web site is designed.

Userfriendliness Refers to how easy it is for a user to use the system without feelings of fear about what the system might unexpectedly do.

Utilization How well the system uses the system resources.

Variability Refers to how consistent the system is when presented with similar inputs.

2.2 Deployability Attributes

Access Audit The ability to determine who accessed the system or data or components therein.

Administrability How easy the system is to administer or control.

Availability The degree to which a system, subsystem, or equipment is operable and in a committable state at the start of a mission, when the mission is called for at an unknown, i.e., a random, time. Simply put, availability is the proportion of time a system is in a functioning condition.

Backup What is required in terms of backing up the software and data?

Communication Commonality Refers to how common the communication protocols are. A system that uses standard protocols like TCP/IP are preferred over less universal protocols such as ???

Composability A system design principle that deals with the inter-relationships of components. A highly composable system provides recombinant components that can be selected and assembled in various combinations to satisfy specific user requirements.

Configurability Refers to how easy it is to set the configuration data that the system relies upon, and how easy it is to maintain that data in a way that correctly controls the system.

Configuration Management How will the configuration of the system be managed?

Data Commonality Refers to the use of common data formats, such as XML.

Deployability Refers to how easy it is to put the system into production, or install on to a user's machine so that it can be readily used.

Documentation What documents should be supplied with the system.

Expandability Refers to how easy it is to increase the amount of processing required of the system.

Installability Refers to how easy it is to install the software.

Machine Independence Refers to how closely the system is tied to a particular hardeware environment. A program running on a JVM is much less dependent than a program written to use a particular CPU and GPU.

Manageability The ease with which the system can be managed so that it performs as required.

Marketability The use of the system with respect to the market competition.

Mobility Access to information or applications from occasionally-connected, portable, networked computing devices

Platform What environment is the system to run on?

Seamlessness Refers to the degree to which the technologies present a consistent structure and paradigm in interfaces and operations, so that the transition from one technology to another is not disruptive or confusing either in usage or integration.

Software-System Independence Refers to how closely the system is tied to a particular software environment. A system that includes most of its dependent libraries is more independent than one that expects them to be already installed.

Training Refers to the training materials available for the user to learn about the system.

2.3 Software Text Attributes

Adaptability Able to be modified to suit new requirements.

Changeability How easy the system is to change. For example, is the source code available or just the compiled binaries? How well is the system structured? Will making a small change break other parts of the system?

Complexity How easy it is to gain an understanding of the code.

Conciseness Refers to the amount of unnecessary noise in the source code.

Consistency Refers to how well things are named and used in a consistent manner.

Customisability Refers to how easy it is to adapt the system to the requirements of each user. It should include how well the individual customisations cope with new versions of the base software.

Generality Refers to how tightly the system is matched to the specific requirements for which it was constructed.

Instrumentation The source code devoted monitoring the activities of the system.

Localizability The ease with which the system can be adapted to a new user locale.

Maintainability The ease with which a software product can be modified

Modifiability Refers to how easy it is to change the system for slightly different requirements or circumstances.

Portability A general characteristic of being readily transportable to multiple platforms.

Replaceability Refers to how easy it will be to replace the system at some future point. A system which uses proprietry format data files might prove to be nearly impossible to replace without significant loss of data.

Reuseability The likelihood a segment of source code that can be used again to add new functionalities with slight or no modification.

Self-Descriptiveness Refers to how well the system is able to describe itself.

Stability Many of the objects will be stable over time and will not need changes.

Standards Compliance The goals of standardization can be to help with independence of single suppliers (commoditization), compatibility, interoperability, safety, repeatability, or quality.

Supportability The ability of technical support personnel to install, configure, and monitor computer products, identify exceptions or faults, debug or isolate faults to root cause analysis, and provide hardware or software maintenance in pursuit of solving a problem and restoring the product into service.

Testability Refers to the capability of an equipment or system to be tested.

Understandability How easy it is to understand the system. This involves comprehending the function, data flow, control flow, operations and state of a program.

2.4 Specification Attributes

Adaptability Able to be modified to suit new requirements.

Analyzability Able to be understood.

Buildability This refers to the ability to build the system in a timely manner.

Changeability How easy the system is to change. For example, is the source code available or just the compiled binaries? How well is the system structured? Will making a small change break other parts of the system?

Clarity Refers to the ease with which the design of the system or the usage of the system can be comprehended.

Compatibility Software compatibility can refer to the compatibility that a particular software has running on a particular CPU architecture such as Intel or PowerPC. Software compatibility can also refer to ability for the software to run on a particular operating system. Very rarely is a compiled software compatible with multiple different CPU architectures. Normally, an application is compiled for different CPU architectures and operating systems to allow it to be compatible with the different system. Interpreted software, on the other hand, can normally run on many different CPU architectures and operating systems if the interpreter is available for the architecture or operating system. Software incompatibility occurs many times for new software released for a newer version of an operating system which is incompatible with the older version of the operating system because it may miss some of the features and functionality that the software depends on. Software that works on older versions of an operating system is said to be backwards compatible.

Completeness The amount of the required system functionality that has been implemented.

Conceptual Integrity How well the components support the overall system concept.

Conformance Refers to how well the system meets specific industry standards.

Consistency Refers to how well things are named and used in a consistent manner.

Customisability Refers to how easy it is to adapt the system to the requirements of each user. It should include how well the individual customisations cope with new versions of the base software.

Evolvability How easy it is to modify the system to match changing usage patterns.

Explicitness The amount of the design that is stated specifically, and not left as an implied requirement.

Extensibility A system design principle where the implementation takes into consideration future growth. It is a systemic measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be through the addition of new functionality or through modification of existing functionality. The central theme is to provide for change while minimizing impact to existing system functions.

Functionality The proportion of the required functions that the system has implemented and are useable.

Generality Refers to how tightly the system is matched to the specific requirements for which it was constructed.

Instrumentation The source code devoted monitoring the activities of the system.

Interchangeability The ability that an object can be replaced by another object without affecting code using the object.

Maturity Refers to how much use a software component has had in other similar systems. A mature component will have had most of its bugs found and repaired.

Modularity A software design technique that increases the extent to which software is composed from separate parts.

Orthogonality Guarantees that modifying the technical effect produced by a component of a system neither creates nor propagates side effects to other components of the system.

Requirement Traceability Refers to the ability to track a requirement through the design to the source code and documentation.

Scalability Its ability to either handle growing amounts of work in a graceful manner or to be readily enlarged.

Simplicity Relates to the burden which a thing puts on someone trying to explain or understand it.

Suitability Refers to how well the system meets the user's requirements.

Traceability Refers to the completeness of the information about every step in a process chain.

Upgradeability The degree to which a computer may have its specifications improved by the addition or replacement of components

2.5 Architectural Process Attributes

Affordability Measured by its cost relative to the amount that the purchaser is able to pay.

Timeliness Refers to the delivery schedule for the system. Can it be delivered when it is needed?