# TOWARDS REFERENCE MODELS FOR REQUIREMENTS TRACEABILITY

***Author correspondence:***

S. Rohini
*Assistant Professor*
*AITS i*
*Tirupati-517502*
*srohini512@gmail.com*

Dr D. Vivekananda Reddy
*Assistant Professor*
*Sri Venkateswara University*
*Tirupati-517502*
*svuvivek@gmail.com*

## Abstract

A Reference model in systems, enterprise and software engineering is abstract framework or domain specific ontology consisting of an interlinked set of clearly defined concepts produced by an expert or body of experts in order to encourage clear communication. There are many uses of reference models. Main use of reference model is to create standards for both the objects that inhibit the model and their relationships. By creating the standards, the work of engineers and developers who need to create objects that behave according to the standard is made easier. In this paper we are proposing S-cube model of Quality reference model. Reference models are therefore an abstraction of best practice, condensed from many case studies over an extended period of time, followed by more case studies to refine and evaluate the proposed reference model. There are challenges towards the service-based systems. The network of excellence on software services and systems (S-cube) performs cross –discipline research to develop solutions for those challenges. In this paper we outlined the S-cube reference model life cycle and how is used to know how a service is developed. S-cube knowledge model which is a continuously updated on-line encyclopedia and a reference library. In this paper we are presenting about S-cube framework which is used to align and bring together the involved disciplines and co-existing of services.

***Keywords:***

Requirement Traceability; Reference Model;
S-Cube Reference Model;
Service Based Application;
S-Cube Life Cycle;
Service Life Cycle;
S-Cube Framework.
.

## 1. Introduction

Software testing is one of the important process in Software Development Life Cycle. Testing is a process of executing a program with the intent of finding an error". Requirement traceability is intended to ensure continued alignment between stakeholder requirement and system evolution. To be useful traces must be organized according to modeling some framework. This paper follows an factual approach and final observations demonstrates wide range of traceability models. Traceability inside a model is sometimes called vertical traceability and between different models called as horizontal traceability. There are 4 key processes in product development practices requirement-, change-, characteristic-, and decision management. These are considered as more basis for the development of traceability.

Reference models are prototypical models of some application domain, usually organized according to some underlying basic meta model. The purpose of reference models is to reduce significantly the task of creating application-specific models and systems: the user selects related parts of the reference model, adapts them to their problem, and configures an overall solution from these adapted parts. Since the analysis of a domain can take an enormous effort when started from scratch, the use of reference models has been reported to save up to 80% in development costs for systems in standardized domain. Not surprisingly, reference models have become highly successful in many industries, the best-known example being the SAP approach.

Not every domain is sufficiently standardized to allow for a reference model of the final product – the system to be built. Moreover, approaches like the one followed by SAP are not necessarily supportive of change, especially when this change goes beyond the initially covered domain. However, at least experience on how to get to the product should be reused. This leads to the idea of reference models for capturing the development process itself, not to be confused with prescriptive software process models.

The process of developing reference models is effortful. The traditional normative computer science approach of imposing such models on developers is long known to have failed in most cases. Reference models are therefore an abstraction of best practice, condensed from many case studies over an extended period of time, followed by more case studies to refine and evaluate the proposed reference model. There is nothing provably correct about reference models; they derive their relevance from the slice of practice they cover. Nevertheless, by formalizing a reference model in an appropriate mathematical framework, at least a number of elementary desirable properties can be ensured.

## 1.1 Role of Reference Models

Our interest is the development of reference models for requirements traceability. We define requirements traceability as "the ability to describe and follow the life of a requirement, in both a forward and backward direction, i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases". The importance of requirements traceability

is highlighted by the fact that, for example, the US Department of Defense spends about 4% of its IT costs on traceability – often without getting an adequate value for this money, as traceability in many organizations is random, the standards provide little reference, and the models and mechanisms vary to a large degree and are often poorly understood. Not surprisingly, the market for requirements traceability tools is booming, even though current tools support only rather simple traceability models and services.

Earlier models of requirements traceability focus on different aspects of requirements traceability. Version and configuration management systems focus on the SOURCE aspect (i.e physical artifacts where traceability information is maintained), emphasizing the document management role of traceability. The source aspect remains important: only trace objects available in *persistent* sources actually constitute long-term traceability, so traceability methods and tools have to make traces persistent reliably and at acceptable cost. Studies at the management level usually focus on the STAKEHOLDER aspect (i.e. agents involved in the management of traceability). Summarizing, the stakeholder aspect becomes more important as the duration and complexity of projects grow.

Despite the importance of considering physical trace management and stakeholder roles, current practice shows that the efficiency and effectiveness of traceability support is largely determined by the system of OBJECT types and traceability link types offered.

We therefore focus on the analysis of object types and stakeholders and organizing our reference models around them.

## 2. Research Overview

According to the organizations the advancements of structured information standards a reference model. Reference model is an abstract framework for understanding significant relationships between entities of some environment, and for development of consistent standards or specifications supporting that environment. Reference model is not directly tied to any standards and technologies or other concrete implementation details but does seek to provide a common semantics that can be used unambiguously and between different implementations. A testing reference model must not attempt to describe all things. It is only a defined environment to be useful. It should include a clear explanation of the problem and concerns of the stakeholders who need it to be solved.

But such a reference model for testing cannot exist alone, just as testing cannot (and should not) operate alone. The entities within the model have essential relationships with other entities outside the model. Therefore we need further models including those other entities. However we are only interested in environments that are in direct contact with the testing environment, and the relationships between their entities and the testing entities. We should not concern ourselves with relationships between the external entities, within or between their models: that is interference.

So what external models, in direct contact with the testing model, do we need? Here is my suggested list.

Quality reference model:

Quality is not a goal, it's a reason. Because I have quality, I want to test to ensure that the products and services I deliver also will have that same quality. This reference model can be used to ensure we all define quality in the same way. Work product reference model: software houses, suppliers, consultancies etc deliver products and services to clients. In the support of the process of delivering these products and services, other products are made that are not necessarily delivered to the client. This reference model can be used to ensure we all define products in the same way.

Process reference model:

No matter what we do or whether or not it is documented, we follow processes with activities and tasks. This reference model can be used to ensure we all define processes the same way.

Metric reference model:

To control things better, we must measure them better by making them more measurable. This reference model can be used to ensure we all define metrics in the same way.

Stakeholder reference model:

Have you ever listed all the parties involved in your testing project? From shareholders to management to teams and individuals to clients, customers, suppliers, partners and users, they all have a certain importance. This reference model can be used to ensure that we all define stakeholders in the same way.

Assessment reference model:

To know where you are going you need to know where to start from. You need to know what your capability and maturity is in order to know what your next steps will be. An assessment of your current situation will give you your baseline. That will be your starting point. This reference model can be used to ensure we all define assessments in the same way.

Continuous improvement reference model:

By improving your capabilities and maturity will not happen by accident. Only if you commit yourself to improvement through change will you be able to grow. Whether this growth is continuous or continual, this reference model can be used to ensure we all define improvement in the same way.

Lifecycle reference model:

Everything has a beginning, a middle and an end. We plan what we want to achieve. We prepare to improve our chances of achieving what we want. We execute our plan with the help of our preparation and finally we assess, we

verify, we validate what we have achieved. This reference model can be used to ensure we all define lifecycles in the same way.

## 3. Creating a Reference Model

We have already seen that a reference model contains entities and relationships within an environment and is abstract and technology-agnostic. The abstractness is what makes creating it hard. When thinking about such things I feel as I imagine does a child with attention difficulties; my mind tries to wander off in any direction other than the task. I need practical steps I can follow. For those I will turn to cognitive psychology. Dr Donald Meichenbaum suggested a way to teach children a structure to help them get tasks done in four simple steps often illustrated with a well-known graphic of four bears. Based on these, here are my suggested steps for creating a reference model.

1. Context: The scope, the plan, the outlines, the context of what the reference model should cover

2. Concept: The outlines, the wireframe, the policies, the rules, the semantics, for solving problems within the reference model

3. Physical: The objects that are requested to implement the reference model in a hands-on fashion

4. Verification: The ways how to verify that the reference model is used correctly to achieve the goals set out.

## 4. Proposed Reference Model

In this paper we are presenting about the quality reference model. Quality reference model is used to check whether the software or product is maintained with high quality or not. Quality is defined by satisfying the user requirements, having capability to rectify the faults easily and also supporting the reuse of a software modules. All this collectively forms high quality software by using suitable quality reference model based on the context.

1. Context: The quality plan • the quality environment: organizational unit(s) for which the quality plan is valid, ex: company, department, project, team, individual

• The quality categories: quality areas the quality plan describes, ex: process quality, people quality, product quality, project quality, price quality

•The quality levels of detail: aspects of the quality categories described by the quality plan, ex: context, concept, physical, verification.

2. Concept: The quality breakdown • the quality requirements: high level/ management quality requirements for a certain quality environment/ category/level of detail. They can be composite requirements, meaning that they cannot be implemented in one go so have to be decomposed into smaller requirements which are easier to implement and control. For example: “each member of the test team (quality environment = test team) needs to understand testing principles well enough and needs to be mature enough (quality category = people) to understand (quality level = context), explain (quality level = concept), implement (quality level = physical) and analyze (quality level = verification) testing practices”

•The quality factors: lower level quality requirements to detail quality requirements; for example holding a testing certification. A quality requirement will be covered by a number of quality factors. A quality factor is easier to control , measure and implement.

3. Physical: The quality blueprint, the quality sub-factors. The technical implementation of a quality factor. For example, the test certification can only be achieved if the candidate knows the generic testing process, is able to apply the technique state transition testing, etc. Exam marking schemes are a good example of the use of quality factors and quality sub-factors

• The quality attributes: a quality attribute makes a quality sub-factor measurable by describing how it can be measured. Each exam question can be considered as a quality attribute.

4. Verification: The quality measurements, the quality metric: used to make a quality attribute measurable. Using the exam analogy, this would be the allocation of marks to right answers (including negative marks for wrong answers)

## 5. S-Cube Reference Model

Innovative software systems increasingly emerge from composing independent third-party software services. Accordingly, the number and variety of software services increases permanently which goes along with rapid enhancements of their private and economic benefits. S-Cube, the European Network of Excellence coordinated by SSE, aims at developing innovative approaches for engineering adaptive software services and systems, modern techniques for the quality maintenance of services and forecasts, as well as new technologies for the realization of service-based systems.

To achieve the integration objectives, S-Cube is carrying out the following activities: bringing together European research associations, supporting the European exchange of scientists and doctoral candidates, establishing a European service-teaching program as well as organizing a large variety of conferences and industry workshops.

Aside from a vast number of scientific publications, important S-Cube results include:

- Concepts and techniques for the development and adaptation of service-based systems with a special focus on Cross-Layer Monitoring and Adaptation, Quality Prediction and Proactive Adaptation, Business Transactions Monitoring and Chemical Computing
- Target-audience orientated presentations of the project results (e.g. S-Cube Shopping List for Industry and S-Cube Use Case Repository)
- Integration of research associations by the definition and distribution of standardized service-engineering processes and views (see Integrated Research Framework, S-Cube Life-Cycle Model, Knowledge Model and Quality Reference Model)
- Sustainable development of research partnerships facilitated by the S-Cube Mobility Programme and the multilateral mentoring of doctoral candidates and final theses and dissertations as well as the organization of a Service-Engineering Summer School
- Integration and standardization of the European range of teaching through a Virtual Campus and the international Master program IMSE

## 5.1 Scope Of S-Cube Model

The rapid evolution of information and communication technology (ICT) means that the opportunities for new ways of computing andinteracting are growing. One such opportunity is developing innovative systems through the composition of soft ware services, available over widely distributed infrastructures. Those services have the power to provide utility to users in a much more dynamic and flexible way than is possible with traditional soft ware technology. However, service-oriented systems and their corresponding soft ware services require fundamental changes to the way soft ware is developed, deployed and maintained. Soft ware that constitutes a service-oriented system is no longer owned by on single organization but distributed and shared amongst many organizations. This distributed ownership opens up a whole range of research challenges, including the design, evolution, adaptation and quality assurance of service-oriented systems.

## 6. S-Cube Framework

To align and bring together the involved disciplines and co-existing, yet disparate, research Themes, S-Cube pursues dedicated research and integration activities within the network. Those activities have led to results which are expected to have a sustainable impact on Europe's research in service-oriented systems. Figure provides a high-level view on the framework.


The most important integration results of S-Cube include:

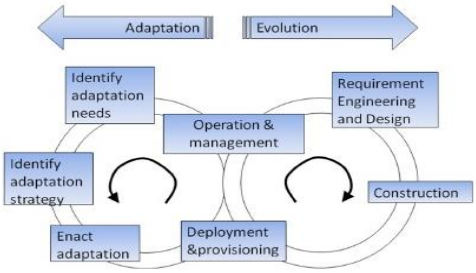
1. S-Cube Integrated Research Framework (IRF): The IRF guides S-Cube's research activities and ensures the integration of research results. The framework provides a clear separation of concerns and thus allows handling the complexity involved in aligning and integrating research activities of diverse disciplines.

2. S-Cube Knowledge Model (KM): The S-Cube KM is a continuously updated on-line encyclopedia and reference library for the Internet of Services, and is available via the S-Cube web portal .The network realized that the communities involved in S-Cube will not always be able to agree on a common terminology, so the KM is designed to provide interrelated and contextual definitions so researchers can translate between
the vocabularies of the various research communities involved in this research.

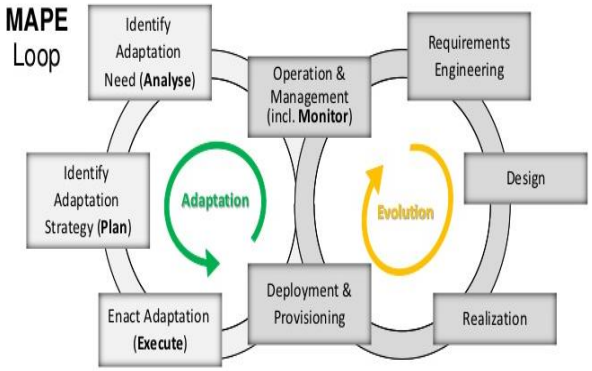
3. S-Cube Virtual Campus (VC): The S-Cube VC provides an online space facilitating the sharing of knowledge in the soft ware services and systems community. The VC provides a collection of learning modules, which provide teaching material based on the research themes of S-Cube. The material is intended to be used during lectures but can also be employed forself -study.
The most important research results of S-Cube include:

4. S-cube life cycle model: The two co-existing cycles support each other during application life time.
    1. Evolution: Design-Time iteration cycle, Explicit re-design of the application to address new requirements(i.e changing user preferences and cheating)

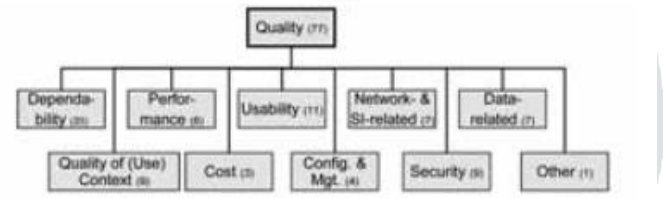
2. Adaption: Runtime cycle addressing adaption




**S-cube Service life cycle**: A life cycle model is a process model that covers the activities related to the entire life cycle of a service, a service based application , a software component or system




5. S-Cube Quality Reference Model (QRM): The S-Cube QRM provides a consolidated taxonomy of quality attributes for service oriented systems, resulting from a thorough analysis of quality models used in software engineering, grid computing, business process management and service-oriented computing. The quality reference model can serve as a central access point to understanding the relevant quality attributes of service-oriented systems.




6. Quality Prediction and Proactive Adaptation: Ideally, during run-time the service-oriented system predicts the degradation of quality and thus imminent failures and thus can apply counter-measures to prevent the actual occurrence of failures. Such proactive adaptation addresses key drawbacks of more conventional reactive adaptation approaches. This figure represents S-cube reference model.

This collection provides an excerpt of the KM including all quality attributes as defined as part of the S-Cube Quality Reference Model. This table presents the details of all the attributes of quality of a service or software or the product or else the component. Even the service modified time and author all the details of s-cube quality attributes are maintained for analysis or as for reference.

| Title | Author | Type | Modified |
|---|---|---|---|
| Accessibility | Benedikt Liegener | Page | Apr 27, 2012 11:11 |
| Accountability | Benedikt Liegener | Page | Apr 27, 2012 11:40 |
| Accuracy | Benedikt Liegener | Page | Apr 27, 2012 11:41 |
| Adaptability | Benedikt Liegener | Page | Apr 27, 2012 10:52 |
| Attractiveness | Dustin Hebgen | Page | Apr 27, 2012 11:42 |
| Auditability | Benedikt Liegener | Page | Apr 27, 2012 11:43 |
| Authentication | Benedikt Liegener | Page | Apr 27, 2012 11:41 |
| Authorization | Benedikt Liegener | Page | Apr 27, 2012 11:41 |
| Availability | Benedikt Liegener | Page | Apr 27, 2012 11:46 |
| Bandwidth | Benedikt Liegener | Page | Apr 27, 2012 11:45 |
| Capacity | Benedikt Liegener | Page | Apr 27, 2012 11:47 |
| Change Cycle | Benedikt Liegener | Page | Apr 27, 2012 11:49 |
| Compatibility | Dragan Ivanovic | Page | Apr 27, 2012 11:49 |
| Compensation | Dustin Hebgen | Page | Apr 27, 2012 11:50 |
| Completeness | Benedikt Liegener | Page | Apr 27, 2012 11:51 |
| Comprehensibility | Dustin Hebgen | Page | Apr 27, 2012 11:51 |
| Confidentiality | Benedikt Liegener | Page | Apr 27, 2012 11:51 |
| Configuration-Related Quality | Benedikt Liegener | Page | Apr 27, 2012 11:52 |
| Content Accessibility | Benedikt Liegener | Page | Apr 27, 2012 11:53 |
| Content Perceivability | Dustin Hebgen | Page | Apr 27, 2012 11:53 |
| Continuous Availability | Benedikt Liegener | Page | Apr 27, 2012 11:54 |
| Cost | Benedikt Liegener | Page | Apr 27, 2012 11:54 |
| Cost Model | Benedikt Liegener | Page | Apr 27, 2012 11:54 |
| Coverage | Benedikt Liegener | Page | Apr 27, 2012 11:55 |
| Data Accuracy | Benedikt Liegener | Page | Apr 27, 2012 11:56 |
| Data Completeness | Benedikt Liegener | Page | Apr 27, 2012 11:57 |
| Data Encryption | Benedikt Liegener | Page | Apr 27, 2012 11:57 |
| Data Integrity | Benedikt Liegener | Page | Apr 27, 2012 11:58 |
| Data Policy | Benedikt Liegener | Page | Apr 27, 2012 11:58 |
| Data Reliability | Benedikt Liegener | Page | Apr 27, 2012 11:58 |
| Data Timeliness | Benedikt Liegener | Page | Apr 27, 2012 11:58 |
| Data Validity | Benedikt Liegener | Page | Apr 27, 2012 11:59 |
| Data-Related Quality | Benedikt Liegener | Page | Apr 27, 2012 11:59 |
| Delay Variation | Benedikt Liegener | Page | Apr 27, 2012 11:59 |
| Dependability | Benedikt Liegener | Page | Apr 27, 2012 11:59 |

## 7. Conclusion

This paper presents the reference models that are used in testing. The results of research acquired in S-Cube are transferred into a consistent knowledge base that allows for their integration in a sustainable research environment. In particular we are presenting the Quality reference model. Under that S-cube reference model is giving the services to the Service-oriented systems. And also it can be used in both evolution and adaption stages of Software or system. There was an enormous usage of this S-cube reference model because of its feature called extensibility even at run-time also. In the vision of the Future Internet, soft ware services and service- oriented systems are expected to play a key role as an enabling technology and core building block. These services and systems will provide the correct level of abstraction from hardware and soft ware entities, extending from business functions to data storage, processing and networking, devices and content. Quality reference is used to ensure about the product and services are acceptable level. S-cube reference model was very accurate and easily extensible in later expansions also.

## References


[1] Organization of conferences and workshops, such As Service Wave, ICSOC, and BPSC (see http:// www.s-cube-network.eu/events).

[2] The organization of the annual SSAIE summer School on service-oriented computing (see http://www.summersoc.eu/).

[3] Setting up the Erasmus Mundus International Master on Service Engineering program, IMSE

[4]. www.slideshare.net/.../**scube**-lp-quality-of-service-**models**-for-service-org

[5] forestgrowth.unbc.ca/**scube**/ www.s-cube-network.eu › Knowledge Model

[6] cordis.europa.eu/fp7/ict/docs/.../**scube**.pdf

[7] www2.cs.uregina.ca/~dbd/.../d**cubes**/d**cubes**.html

[8] https://books.google.com/books?isbn=3642318754

[9] www.citilabs.com/.../vol-3-getting-started-with-**cube**-and-building-**you**r-...

[10] www.emeraldinsight.com/.../1463715031048...

[11] https://www.designsociety.org/.../integratingnew_product_development...

[12] www.westgard.com › Resources › Resources

[13] www.**s-cube**-network.eu/final-report.pdf