Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 47 (2016) 78 – 83

www.elsevier.com/locate/procedia

Product-Service Systems across Life Cycle

A System Quality Attributes Ontology for Product-Service Systems Functional Measurement based on a Holistic Approach

Arturo Estrada[1] and David Romero[1-2*]

[1]*Tecnológico de Monterrey, Del Puente 222, Col. Ejidos de Huipulco, Tlalpan, Mexico, 14380*
[2]*Griffith University, 170 Kessels Rd, Nathan QLD 4111, Australia*
*Corresponding author. Tel.: +52 (55) 5483-1605; Fax: +52 (55) 5483-1606. *E-mail address:* arturoestrod@hotmail.com, david.romero.diaz@gmail.com

Abstract

The importance of cost engineering within Product-Service Systems (PSS) discipline has increased in recent years. Literature reveals that there is a need for holistic PSS cost determination approaches dealing both with PSS cost engineering and PSS functional uncertainty - as a System of Systems (SoS), as PSS cost determination uncertainty arises from the limited understanding of PSS behavior as a SoS. This work proposes a System Quality Attributes (SQA) Ontology developed to measure the function of a PSS: How well it performs in light of its intended purpose? The PSS-SQA Ontology was constructed on the foundations of SoS Engineering and Reliability Engineering disciplines. It propose a holistic approach to PSS functional measurement, covering the most accepted PSS typology, including product-, use- and result-oriented PSSs. The work contributes to PSS engineering by offering an SQA Ontology that quantifies PSS functionality and reduces PSS functional uncertainty for further cost determination.

Peer-review under responsibility of the scientific committee of the 8th Product-Service Systems across Life Cycle

Keywords: Product-Service Systems; Holistic Modelling; System Quality Attributes; Ontology; System of Systems Engineering; Reliability Engineering.

1. Introduction

The importance of cost engineering within Product-Service Systems (PSS) discipline has increased in recent years [1-3]. Literature [1-4] reveals that there is a need for holistic PSS cost determination approaches dealing both with PSS cost engineering and PSS functional uncertainty - as a System of Systems (SoS). *PSS Cost Engineering* can be defined as: *"the set of activities in order to determine/predict the cost of a PSS functionality level as well as the awareness of the certainty degree of such determination/prediction"*. Hence, PSS cost determination *uncertainty* arises from the limited understanding of PSS's behavior as a SoS. Moreover, *uncertainty* is classified as *epistemic* and *aleatory* [4]. The first type arises from the lack of understanding of the analyzed entity's behavior (i.e. due to lack of relevant information), while the second type arises from an entity's random nature.

This work proposes a *System Quality Attributes (SQA) Ontology* developed to measure the function of a PSS: How well the PSS performs regarding its intended purpose? It uses *functionality* as the representation of PSS's behavior.

An *Ontology* describes the form and nature of the studied reality, while *Epistemology* is the way we acquire knowledge of reality. A lack of knowledge of a system and its emerging properties may mean that the ontology itself is *uncertain* [4].

Based on the above, we believe that PSS cost engineering requires a *holistic approach* to make the emergent properties of a PSS as a SoS visible through a *systemic ontology* thereby reducing the *epistemic uncertainty* in PSS cost determination.

The proposed *PSS-SQA Ontology* has been constructed on the foundations of *SoS Engineering (SoSE)* and *Reliability Engineering (RE)* disciplines.

A search in Elsevier, Springer and Taylor & Francis databases found no work connecting PSS, SQA, SoSE and RE.

Peer-review under responsibility of the scientific committee of the 8th Product-Service Systems across Life Cycle
doi:10.1016/j.procir.2016.03.215

This finding gave us the idea to develop a multi-disciplinary approach for PSS functional measurement, covering the most accepted PSS typology: product-, use- and result-oriented PSSs.

This paper presents the first part of an on-going research towards a *"Holistic Method for PSS Cost Engineering based on the combination of a SQA Ontology and a System Cost Uncertainty Analysis"* [see also 5]. Both *ontology* and *analysis*, are intended to mitigate the whole spectrum of PSS *uncertainty* (epistemic and aleatory respectively).

2. PSS Holistic Modelling and Ontology Development

2.1. Systems Science and Engineering

Systems Science and *Engineering* play an important role in understanding PSS complexity. A PSS is defined as: *"… products and services combined in a system to deliver required user functionality…"* [6], whereupon *"a system is an assemblage or combination of elements or parts forming a complex or unitary whole, with a functional relationship, and a useful purpose"* [6]. A system is composed of components with attributes and relationships. On a first level of abstraction components of a typical PSS are heterogeneous, and may be tangible (products) or intangible (services). On a second level of abstraction a product may be broken down into subassemblies (still products), and a service can be divided into processes, i.e. a PSS is a compound of products and processes. Some of PSS complexity arises from *interconnections* of heterogeneous components each with its own properties and behavior, while the system presents some characteristics or behaviors that cannot be attributed to any of its components [7]. This is expressed by the term *holism*, i.e. a system is more than the sum of its components. Properties only exhibited on the system level are *system attributes*, e.g. a PSS has attributes that cannot be found in any of its product or service subsystems. Not only are the components of a system interconnected, system attributes also have relationships [7]; this is called: *system attributes configuration*.

A system is *engineered* with a purpose in mind (what the system must generate as its output, how well it must perform to guarantee the output, and under what conditions it has to operate). PSS engineering, and in particular PSS cost engineering, must not only focus on the main system output, but also on the quality of the output, and on the circumstances in which such output quality must be achieved, as all parameters will impact on the overall PSS operational cost. The system attributes configuration depends on the nature of components, and their interconnections. Hence, system purpose depends on the system attributes configuration, which further depends on system components configuration. A PSS engineering team may try many system components configurations to attain the desired purpose. The problem is how to visualize the attributes configuration, and how to link it with the system's purpose?

Fig. 1 shows relationship between system components configuration, system attributes configuration, system purpose, and its relationships.

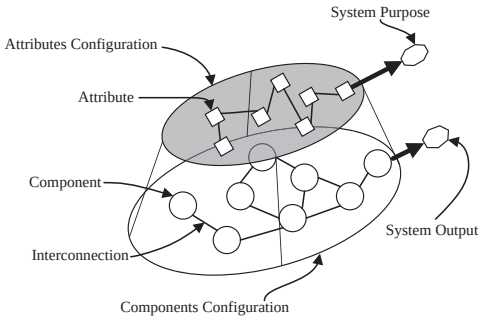

Fig. 1. System Components, Attributes and Purpose, and its Relationships

2.1.1. System of Systems (SoS) Engineering (SoSE)

A PSS as a system has components, and some components (often heterogeneous in nature) of this system may themselves be systems [6]. Hence, a PSS is a *System of Systems (SoS)*, and its component systems are called: *subsystems*.

"The main thrust behind the desire to view the systems as a SoS is to obtain higher capabilities and performance than would be possible with a traditional system view" [8].

The SoS view is a high-level perspective and explains the interconnections among independent systems. *"A SoS is a super system comprised of other elements, which themselves are independent complex operational systems and interact among themselves to achieve a common goal. Each element of a SoS achieves well-substantiated goals even if they are detached from the rest of the SoS"* [8]. A system is more than the sum of its components parts, however, the components of a system may themselves be systems [6]. Since a PSS is a compound of heterogeneous elements (products and processes) it is considered as a complex system. Moreover, some of its components can be described as a system itself; therefore a PSS is considered as a *System of Systems (SoS)*. In order to differentiate the compounding systems of the PSS from the PSS itself (since both are systems), the compounding systems are named: *subsystems*.

From the traditional systems view, the interconnections among components are commonly described by means of process-based thinking, in which a chain of inputs-outputs link the components in a certain configuration. This approach is useful when the components of the system are homogeneous; but for the case of a PSS this approach entails great complexity. Therefore, it is proposed to use a SoS point of view, in which every component of the PSS is a system itself. Hence, the PSS engineering team must define the boundaries of each system in order to architect a SoS comprised of (possibly heterogeneous) systems, each with its homogeneous components, while every subsystem must be independent. It is proposed that every product is defined as a system, and a service is divided into several independent sets of processes (if possible), each one representing a system; the level of granularity will depend on the PSS engineering team. For this case, interconnections among systems are proposed to be established by means of its intended purposes; in which a chain of functions link the systems in a certain configuration. This chain is defined as a *teleological* interconnection.

The main argument in order to use the SoS view instead of the traditional system view is that the traditional perspective represents a higher level of restriction when two entities are interconnected. From the process-based thinking approach, the interconnection between two components is described by means of a relationship (two-sided link); the output of component A is the input of component B, so it is an interconnection of dependency, where the operation of B depends on the output of A. The problem with this approach is how a PSS engineer can interconnect two independent components. For example, a manufacturing machine tool, which is periodically maintained, the components are the machine (product) and the maintenance activity (service). These components are independent from each other, but it is obvious that between both there is an interconnection. From the process-based thinking perspective there is no-input or no-output that links both components, there is no-dependency interconnection. The interconnection is in the purpose of the maintenance activity (functionality interconnection) aiming to extend the operational time of the machine tool. In order to increase the spectrum of components inter-connections, it is proposed that both types of interconnections must be used in a PSS design/engineering: (a) *Dependency interconnections* (among homogeneous components), and (b) *Functionality interconnections* (among heterogeneous components).

2.1.1.1. PSS Function Representation

The concept of PSS entails a focus on delivery of functions [9]. It is widely accepted that PSS brought a shift in the paradigm from selling products or services into an integrated value offer, where the customer looks for functionality instead of ownership [10], where function is defined as: *"the intended purpose of the system"* [11]. A PSS's function has a two possible dimensions of representation: (a) subjective, and (b) objective [9].

The subjective representation describes customer demands (what the customer wants the PSS to do); for the case of the proposed approach this subjective representation is called: *PSS functional requirements*. The objective representation of the function of the PSS corresponds to the set of attributes that the PSS must exhibit in order to comply with customer demands. This dimension is called: *PSS non-functional requirements*.

To understand how well the PSS function has to be performed, the concepts of *functional result* and *functional performance* are introduced. The *PSS functional result* is the standardized unit of function delivery (system output), while *PSS functional performance* expresses how many functional results are being delivered and how well [9].

Both functional result and functional performance enable the translation from *PSS functional requirements* into *PSS non-functional requirements;* a PSS engineering team must identify *functional results* that best represent the PSS's function.

If a customer wants to contract a PSS for certain manufacturing operation, the customer's demand is: the PSS must produce *Q* quantity of products, under *C* stated conditions, in *T* period of time: this is the *PSS functional requirement.* The PSS engineering team translates this into PSS attributes. The next step is to define the *PSS functional result* and *functional performance; functional result* can be: products under *C* stated conditions. The *functional performance* can be: *Q/T.* Now that it is known what the PSS has to do (functional requirement), what is the *function delivery unit* (functional result), and how well does the PSS have to perform (functional performance). The next step is to determine how the PSS will comply with customer requirements (non-functional requirements). The *PSS non-functional requirements* are expressed by means of PSS attributes, e.g. to comply with customer demands, the PSS must exhibit *R%* of reliability, *A%* of availability, and a *Re* value of responsiveness. This process is shown in Fig. 2

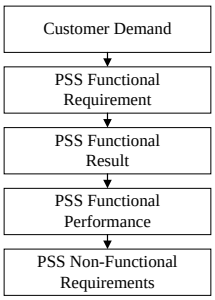

Fig. 2. PSS Function Determination Process

Although the customer requirement statement will vary depending on the nature of the demand the logic behind the objective representation of the *functional requirement* is generalized for every PSS case, where the PSS engineering team selects the most convenient PSS attributes.

The main contribution of our research is how to relate both aspects: *functional* and *non-functional requirements.*

2.2. PSS Functional Variables

From here on, the PSS component configuration is called: *PSS subsystems configuration*, since a PSS is considered to be a SoS. To relate the *PSS subsystems configuration* to the *PSS attributes configuration*, a set of variables are defined, characterizing *PSS functional performance* in time. When the PSS engineering team tests a particular *PSS subsystems configuration*, in each defined period of time the PSS will present a particular *functional performance*. After several time periods, the change in *PSS functional performance* can be analyzed. Below, a set of functional variables are proposed to carry out the *PSS functional performance behavior analysis:*

- MTBF - Mean Time Between Failures
- MTBeP - Mean Time Before a Perturbation
- MTBP - Mean Time Between Perturbations
- MTBeF - Mean Time Before Failure
- MFT - Mean Failure Time
- MTTR - Mean Time to Repair
- MTBM - Mean Time Between Preventive Maintenance
- MTTM - Mean Time to Monitor (the resolution time of the PSS performance update)
- MTTD - Mean Time to Detect (detect a perturbation)

A *perturbation* is any endogenous or exogenous event that modifies the stated PSS operational conditions. The presence of a perturbation may cause the PSS to fail if it is not solved. For the particular case of the PSS it is considered that a *failure* occurs when the PSS starts performing below the *minimum*

required performance level. Each functional variable expresses how the PSS is performing along the time. For the cases of MTTD, MTTM, MTBM and MTTR, it can be seen that all these variables express a particular PSS capability. For MTBP, and MTBeP, these variables represent how the PSS is operating against perturbations. Finally, for MTBF, MTBeF and MFT, these variables represent the PSS functionality behaviour. Fig. 3 shows each variable when a PSS is performing along the timeline.

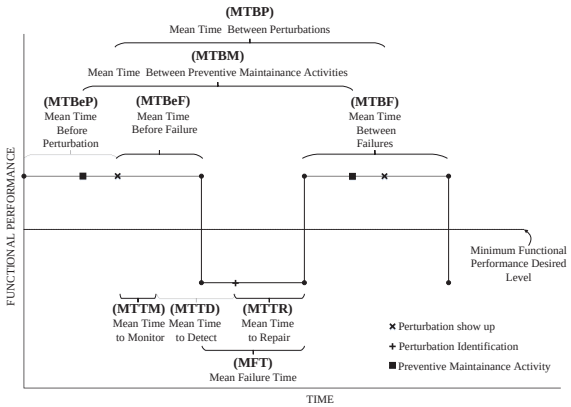

Fig. 3. PSS Functional Variables along the Time

Some interesting behaviors can be spotted in Fig. 3: e.g. If MTTM + MTTD + MTTR ≤ MTBeF, then MFT = 0. In the sequel the interconnections among PSS functional variables must be defined. Fig. 4 shows the interconnections among PSS functional variables: we can see that *PSS functional variables* can be (a) independent or (b) dependent variables. Independent variables, are the ones that are in control of the PSS engineering team: MTTD, MTTM, MTBM and MTTR, while dependent variables depends on the value of the first ones, and represent the status and the function of a PSS.

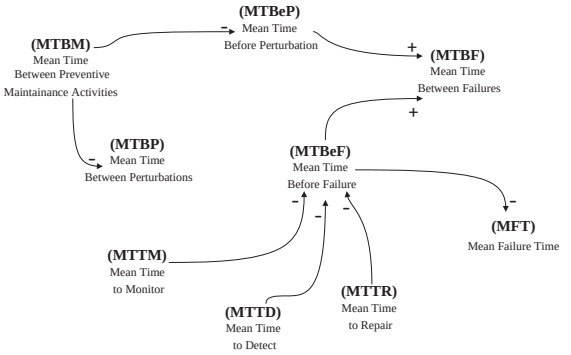

Fig. 4. PSS Functional Variables Configuration

2.2.1. PSS Functional Variables - Attributes Configuration Relationship

After the behavior of a *PSS functional performance* has been analyzed, every *PSS functional variable* will have a defined value. The *PSS attributes configuration* is defined as: *"the semantic representation of the PSS functional variables configuration"*. Fig. 5 presents the relationships between subsystems configuration, functional variables configuration and attributes configuration.

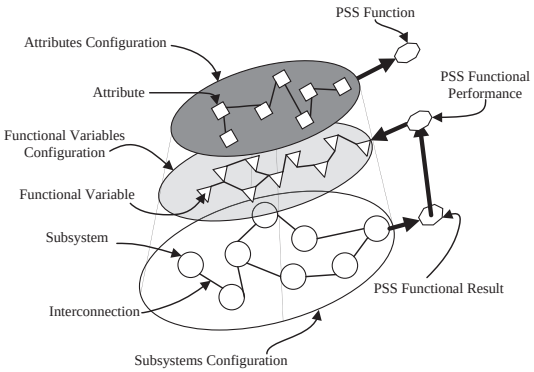

Fig. 5. PSS Subsystems: Functional Variables, Attributes and Function Relationship

Three types of effects a *PSS subsystems configuration* exerts on *PSS functional performance:* (a) change, (b) maintain, and (c) measure/monitor.

Depending on the value of the *PSS functional performance*, the PSS status can be "above minimum required performance level*"* or "below minimum required performance level".

Finally, the PSS function can be determined by means of the behavior of the PSS status along the time. This behavior is represented by the concept of *Reliability.*

2.3. Reliability Engineering (RE)

Reliability Engineering (RE) is an evolving discipline, whose main purposes are the evaluation and prediction of reliability of components, equipment, and systems [12]. For the case of the proposed approach, reliability engineering is applied to the PSS itself, and not to its components.

Whether the PSS will work for a particular period of time can be expressed as a *probability* [13]. *Reliability* is *"the probability that an item will perform a required function without failure under stated conditions for a stated period of time"* [14]. Therefore, the *reliability* for the proposed PSS is: *"the probability that the PSS will perform its intended purpose without failure under customer's stated conditions for a stated period of time"*.

One of the most important parameters in reliability engineering is *Failure Rate*, which is *"the number of failures in a given period of time"* [12]. A *failure* occurs when the system stops performing its required function [12], for the particular case of the PSS it is considered that a *failure* occurs when the PSS starts performing its intended purpose below the minimum required performance level. It can be seen that this definition does not exclude the one from [12]; if the PSS stops performing its intended purpose it is obvious that its performance is under desired minimum level. Since *failure rate* varies over time, it is useful for mathematical representation to define a single number [14], the Average Failure Rate (λ), used in the mathematical expressions for reliability (R(t)), which is expressed as [12]:

$$R(t)=e^{-\lambda t} \quad (1)$$

There are two types of systems within reliability engineering: *repairable* and *non-repairable systems.* For the *non-repairable systems,* reliability turns to be the survival probability over the system's expected life [13]. For this reason, the presented approach considers the PSS as a *repairable system.*

The proposed approach considers that the PSS is operating under a stable period. When a PSS is implemented, it passes through an early failure period, where *failure rate* has a decreasing trend [14]. The *failure rate* decreases because initial systematic failures are identified and eliminated. These types of failures are produced by errors or mistakes that may require a change in the PSS design [12]. From the reliability engineering *perspective,* a PSS is operating under a stable period when failures occur in a random fashion at a uniform or constant rate [11]. When a *repairable system* operates under the particular condition of a constant failure rate, average failure rate can be represented by means of the Mean Time Between Failures (MTBF) [13].

$$\mathrm{MTBF} = 1/\lambda \tag{2}$$

Therefore, considering 1 and 2, the *reliability* for a PSS is:

$$\mathrm{R(t)} = e^{-t/\mathrm{MTBF}} \tag{3}$$

Reliability engineering is used to describe the effectiveness of the PSS, expressing if the objective (intended purpose) is being attained, to satisfy *customer functional requirements* and *Reliability* is a *non-functional requirement* representation of these. It is believed that the PSS will always comply with its intended purpose if the desired reliability value is achieved.

2.4. System Quality Attributes (SQA)

System Quality Attributes (SQAs) are used in the software industry, but. are also found in literature as System Lifecycle Properties, or *Ilities* [14] *"… properties of engineering systems that often manifest and determine value after a system is put into initial use"…"rather than being primary functional requirements, these properties concern wider impacts with respect to time and stakeholder"* [16]. In the case of a PSS, SQA are both *customer non-functional requirements* and *PSS operational attributes. Customer functional requirements* are translated into *non-functional requirements* by means of SQA; therefore there is *no interface* between customer requirements and PSS operational attributes.

A main *non-functional requirement* is considered to be *reliability.* One of the main features of a PSS is that it is oriented towards selling functionality instead of pure products or pure services [9]. *Reliability* expresses the function of the PSS as means of the intended purpose of the system.

Using SQA, we can represent the main PSS attributes, their interactions, and their impact on the PSS function [17]. There has been an increasing concern regarding system properties in industry, government, and academia, since systems displaying certain SQA are more resilient to threats or value loss [15], therefore it is proposed that PSS should be engineered by considering SQA. It is the task of PSS engineering team to determine the most valuable attributes to maximize reliability.

2.4.1. PSS-SQA Semantic Field

Currently there is ambiguity in SQA: the challenges is to develop a standardized semantics of the field [17]. e.g. how to differentiate the meaning of *flexibility* and *adaptability?* Therefore, we shall propose a generalized *PSS-SQA ontology* and try to develop it in a Means-ends hierarchy.

2.4.1.1. PSS-SQA Means-ends Hierarchy (Ontology)

This hierarchy represents how the PSS attributes interact, and the impact of this on the *PSS function,* as measured by *Reliability, Availability and Responsiveness.*

Reliability expresses the effectiveness dimension of the PSS function, while *Availability* and *Responsiveness* represent the efficiency dimension. These attributes are the first tier of the hierarchy. The second tier is characterized by *Robustness, Stability* and *Recoverability.* This level represents the PSS operational status, which can be "above minimum required performance level", or "below minimum required performance level". Both *Robustness* and *Stability* belong to the first PSS status. The third tier is formed by *Traceability, Monitoring, Maintainability* and *Reparability.* This level represents the effects that PSS subsystems exert upon the *PSS functional performance.* As previously discussed, the effects that PSS subsystems have on the *PSS functional performance* can be: (a) change, (b) maintain, and (c) measure/monitor. *Reparability* belongs to the first effect, while *Maintainability* to the second one. Table 1 represents the *PSS-SQA Means-ends hierarchy.*

Table 1. PSS-SQA Means-ends Hierarchy (Ontology)

PSS SQA Means-ends Hierarchy				
FUNCTIONALITY	Tier 1	Effectiveness	Efficiency	
		Reliability	***Availability***	***Responsiveness***
STATUS	Tier 2	Above Minimum Requiered Performance Level		Below
		Stability	***Robustness***	***Recoverability***
EFFECT	Tier 3	Measure/Monitor	Maintain	Change
		Traceability	***Maintainability***	***Reparability***
		Monitoring		

2.4.1.2. PSS Attributes Interconnections

Since *PSS attributes* are the semantic representation of the *PSS functional variables;* now that the *PSS functional variables* have been defined, as well as their *interconnections, PSS attributes'* meaning can be established and their *inter-connections* determined. In the first tier of the *PSS-SQA hierarchy* (see Table 1) the attributes are discussed below:

- *Reliability:* probability that the PSS will perform its intended purpose without failure under customer's stated conditions for a stated period of time (see Equation 3).
- *Availability:* probability that the PSS will perform above the minimum required level at any point in time. Based on reliability engineering equation, Availability is expressed as [12]:

$$\mathrm{Availabiliy} = \mathrm{MTBF}/(\mathrm{MTBF}+\mathrm{MTTR}) \tag{4}$$

In [13] it is considered that MTTR=MFT, but for the proposed PSS approach this assumption is not correct. It can be seen from Fig. 3 that: if [(MTTM + MTTD + MTTR) ≤ MTBeF], then [MFT=0], else [MFT = (MTTM + MTTD + MTTR) - MTBeF]. Therefore *Availability* is:

$$\text{Availabiliy} = \text{MTBF}/(\text{MTBF}+\text{MFT}) \quad (5)$$

- *Responsiveness:* the average performance level above the minimum required performance level.

For the second tier of the *PSS-SQA* attributes are:

- *Stability:* PSS ability to operate without the presence of perturbations (MTBeP).
- *Robustness:* PSS ability to cope with perturbations and stay operational (MTBeF).
- *Recoverability:* PSS ability to recover from perturbations or failure (MTTM+MTTD+MTTR).

For the third tier of the *PSS-SQA* attributes are:

- *Traceability:* PSS capability to detect perturbations (MTTD).
- *Monitoring:* PSS capability to update the performance level values (MTTM).
- *Maintainability:* PSS capability to carry out preventive maintenance activities (MTBM).
- *Reparability:* PSS capability to solve the perturbation or failure (MTTR).

Now that the *PSS-SQA semantic field (ontology)* is established, every *PSS attribute* has its own meaning; the *PSS attributes configuration* is defined in Fig. 6.

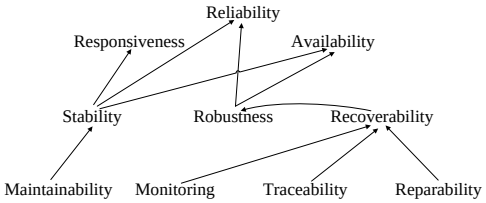

Fig. 6. PSS-SQA Configuration (Ontology)

3. First PSS-SQA Ontology Validation

The *PSS-SQA Ontology* was designed based on the most common accepted PSS typology: product-, use- and result-oriented PSSs. The purpose is to measure *PSS functionality* by means of SQA. For every PSS there is a *Functionality SQA* representing the PSS intended purpose. The *Ontology* does not change its structure when applied to different PSSs, it just focuses on a particular SQA, which does not imply that other *SQA* cannot be measured. e.g. in the case of a photocopier provider contracted under a *use-oriented PSS scheme*, the customer will be an enterprise that must print documents during working days, so the PSS intended purpose is: the customer requires certain *Availability*, and the provider must perform in >95% availability. For a *result-oriented* PSS, considered a medical ventilator. In this case, SQA Availability does not represent the intended purpose of the PSS since a 100% of Availability does not imply that a failure will not occur. *Reliability* is the main SQA, restricting the probability of a failure. *SQA Responsiveness* measures how well the function is being performed, thus is always a useful insight of the functionality regardless the PSS scheme. In case of a *product-oriented scheme*, ownership of the product remains with the customer, and the provider sells additional services to guarantee or optimize the functionality of the product. Considering the example of the medical ventilator from a product-oriented perspective, the purpose of the product is the same because it is given by customer needs. The difference is that the provider is not absorbing the risk of failure. It is important to understand that the SQA that best represents the functionality is given by customer requirements. For the cases of *use-oriented* and *result-oriented PSSs* it is straight forward to assign *Availability* and *Reliability SQAs* since these PSS types are already defined in terms of customer expectations, but *product-oriented* type has to be managed in a subtler manner: the intended purpose that the customer requires has to be first identified in order to offer the set of services that best guarantee or optimize the functionality of the product..

4. Conclusions and Further Work

The present work proposes a *PSS-SQA Ontology* developed for *measuring* the function of a PSS: How well it performs regarding its intended purpose? The quantification of PSS functional measurement enhances the understanding of a PSS behavior, therefore it reduces the *epistemic uncertainty* for its further PSS cost determination.

Further work is needed to validate this *ontology* to improve the level of detail in the PSS emergent properties models. The limitations in the present work is the current ambiguity in the use of *SQA*, as there is no standardized SQA ontology in literature. Hence, the accuracy of the *PSS-SQA Ontology* can only be validated by logical arguments, and a set of case studies should be carried out.

References

[1] Walker WE, Harremoës P, Rotmans J, van der Sluijs JP, van Asselt MBA, Janssen P, Krayer von Krauss MP. Defining Uncertainty: Conceptual Basis for Uncertainty Management in Model-Based Decision Support. Integrated Assessment 2003; 4(1):5-17.
[2] Erkoyuncu J, Roy R, Shehab E, Wardle P. Uncertainty Challenges in Service Cost Estimation for Product-Service Systems in the Aerospace and Deference Industries, CIRP 1st IPS2 Conf., 2009: 200-207.
[3] Erkoyuncu J, Roy R. Understanding Service Uncertainties in Industrial Production-Service System Cost Estimation. International Journal of Advanced Manufacturing Technology 2010; 52(9):1223-1238.
[4] Goh Y, Newnes L, Settanni E, Thenent N, Parry G. Addressing Uncertainity in Estimating Cost for a PSS Delivering Availability: Epistemology and Ontology. Springer 2015.
[5] Estrada A, Romero D. Towards a Cost Engineering Method for Product-Service Systems based on a System Cost Uncertainty Analysis. CIRP 8th IPS2 Conf., Product-Service Systems across Life Cycle 2016.
[6] Goedkoop M, van Halen C, te Riele H, Rommens P. Product Service Systems, Ecological & Economic Basics. 1999, www.pre.nl/pss/default.htm
[7] Blanchard BS, Fabrycky WJ. Systems Engineering and Analysis. 4th Ed., Prentice-Hall 2006.
[8] Jamshidi M. SoS Engineering: Principles & Applications. CRC Press 2008.
[9] Van Ostaeyen J, Van Horenbeek A, Pintelon L, Duflou JR. A Refined Typology of Product-Service Systems based on Functional Hierarchy Modeling. Journal of Cleaner Production 2013; 51(15):261-276.
[10] Baines T, Lightfoot H, Evans S, et al. SOTA in Product Service Systems. The Institution of Mechanical Engineers Part B 2007; 221(10):1543-52.
[11] Keuneke A. Device Representation – The Significance of Functional Knowledge. IEEE Expert 1991. pp. 22-25.
[12] Birolini A. Reliability Engineering: Theory and Practice. 3rd Ed., Berlin: Springer 1999.
[13] O'Connor PDT. Pratical Reliability Engineering. 4th Ed., Wiley 2002.
[14] Tobias PA, Trindade DC. Applied Reliability. NY: Van Nostrand Reinhold 1995.
[15] Ricci N, Fitzgerald M, Ross AM, Rhodes DH. Architecting Systems of Systems with Ilities: An Overview of the SAI Method. Conference on Systems Engineering Research 2014.
[16] OL W, Ross AM, Rhodes DH. Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (ilities). Conference on Engineering Systems 2012.
[17] Ross AM, Rhodes DH. Towards a Prescriptive Semantic Basis for Change-type Ilities.Conference on Systems Engineering Research 2015