A PSS Approach in Software Development

S. Brad, M. Fulea, E. Brad
Research Lab of Competitive Engineering in Design and Development
Technical University of Cluj-Napoca, Cluj-Napoca, 400020 Ctin Daicoviciu 15, Romania
stelian.brad@staff.utcluj.ro, mircea.fulea@staff.utcluj.ro, emilia.brad@muri.utcluj.ro

Abstract

Various studies reveal that over ninety percent of new software applications are unsuccessful in bringing the expected value added to their target markets. This is because software producers concentrate mainly on product innovation instead of seeing their software solutions as part of product-service systems. A methodology for concurrent planning and design of software-PSSs against multi-target functions is proposed in this paper. Borders between product and service are marked in a cost-effective way, keeping also a superior balance between customer and producer benefits. The concept is exemplified on a software-PSS related to business excellence assessment and analysis.

Keywords

Concurrent planning of PSS, new software development, PSS innovative design

1 INTRODUCTION

Software systems are characterized by flexibility, high complexity per unit of development time and "invisibility" [1], [2]. Therefore, development of software systems requires careful planning and continuous overpass of a multitude of challenges. In the common perception, a software project is seen successful if it meets the level of quality expected by the user, within a given development time and budget [3].

In this framework, various studies have revealed that meeting user expectations is the primary criterion for software system success [4], [5], [6]. Researches have shown that whether a software system is viewed as a success or failure depends on the perception of the person viewing the system [7]. Consequently, a software system may be considered successful by one user and a failure by another one [7]. Thus, user expectations must be correctly identified and constantly reinforced in order to avoid failures [4], [8], [9].

A critical aspect in this context is when users have improper assumptions about the value brought by the software system into the business framework, because then users may perceive that the objectives associated with the respective system have not been met [4], [5], [10]. Faulty expectations ("unrealistic" expectations) lead to lower levels of software system success as measured by perceived system quality, perceived information quality, or user satisfaction [4], [11]. Accordingly, software systems, after initial release, are subjected to continuous evolutions and modifications [12], [13], this representing the most costly part of the software system lifecycle [14].

Moreover, in the equation of success other stakeholders must be included (e.g. producer, integrators and consultants). For producers, commercial success is perceived as relevant profitability over system's life-cycle, meeting in the same time adequate rates of the financial indicators (e.g. IRR, NPV) [1], [15]. In the same context, application service providers, as well as consultants are changing the way a software system is sold, delivered and used, which might significantly influence user's perception on the value added which the respective system brings into its business.

Considering the mix of previously mentioned issues, the current trend in software product industry is "servicization" of software solutions; that is, provision of systemic solutions consisting of software products and services which jointly are capable of fulfilling final customer needs [16].

In other words, development of Product Service Systems (PSS) becomes an increasingly important strategy in achieving social, economic and, to certain extends, environmental sustainability in software industry, which advocates reducing resource consumption but delivering better, more widely available goods and services [17]. Actually, such models more adequately address the orientation of software production towards individual behaviours and highly personalised needs [18].

2 THE PROBLEM

The software literature provides many published papers and surveys that report alarming figures for software systems failures at various levels or discuss inconvenient software project situations [19]. Such failures take the form of either full collapse (e.g. project cancellation) or inability to fulfil partial but critical project goals, such as user satisfaction, project budget or time constraints [19].

For example, a survey on a representative sample of software projects, whose results are reported in the paperwork [20], indicates that 53% of them were unable to be delivered on schedule, within budget, and with the required functionalities, while 18% of the software projects were cancelled. According to [14], problems with cost and schedule estimations persist in spite of on-going research into effort and schedule estimation. Based on various sources, the same reference concludes that 26% of all software projects failed, and another 46% experienced cost and schedule overruns or significantly reduced functionality. Other reports reveal that software project implementations were often over budget and over time, and delivered less than the promised benefits [11].

Another reference, citing several sources, even indicates that over 90% of new software applications are unsuccessful in bringing the expected value added to their target markets [1].

In conclusion, the alignment of product, project and business decisions is a major problem in the software industry [2]. Mature design solutions based on the mix of

material and immaterial components, which satisfy the requirements of each stakeholder are therefore essential for the business success [18]. In this effort, product and service concepts cannot be clearly disconnected [17], [18], [21]. In the combination product-service, the relative weight between product and service differs from case to case [21]. This also determines that products in general and software products in particular, should have capabilities to support the delivery of product-related services [22].

The role of software designers in such partnership is therefore critical, although the definition of specific methodologies to manage some critical aspects of the design process of PSS has rarely been considered in software design-related disciplines. Design of services in the frame of products should be adequate integrated in order to maximize customers' value [23].

In this respect, the present paper introduces an approach that concurrently plans and designs extended software products by considering multiple dimensions of value-to-customer (e.g. quality, price, ergonomics, operational cost, easy to use, etc.) and value-to-producer (e.g. development time, development cost, integration cost, etc.). It also allows outlining adequate borders between those components of the PSS which should be incorporated into the product and which should be included into the service.

The next sections of the paper describe the methodology of concurrent planning and design of software-PSSs against multi-target functions, as well as a case study for exemplifying its use into practice.

3 THE METHODOLOGY

The methodology relies on the fact that at the ground of any product or service which is developed for commercialization there is a market need that has to be satisfied. From this perspective, in most of the cases there are several options in formulating a solution for meeting a targeted market need. It might be found under the form of a pure service or a pure product, or mixes of products and services in different weights and combinations. The challenge is to establish the best PSS combination, or at least one from the set of viable PSS combinations, under a group of given constrains.

According to this idea, the proposed methodology for developing software-PSSs consists of the following steps:

Step 1: Define market need under the form of a set of requirements and rank these need-related requirements. For ranking, tools like ANP method [24] or AHP method [25] might be used.

Step 2: Define a set of target-functions in accordance to the market need and deploy them against the set of need-related requirements. Thus, value weights of target-functions in relation to the market need are determined. Correlations between target-functions have to be also established. QFD-type relationship and correlation matrices could be used to fulfil this step [26].

Step 3: For each target-function, a set of specific key requirements are formulated and further ranked (e.g. AHP method).

Step 4: For each target-function, the set of key value characteristics should be afterwards determined. This process is fulfilled by deploying the corresponding key requirements formulated at step 3. In this respect, QFD-type relationship matrices are applied. Further, value weights of the key value characteristics are calculated. Correlations between the key value characteristics for each target-function are established, as well as targets to be achieved. Where negative correlations occur, innovative problem solving is required.

Step 5: Formulate vectors of innovation for each negative correlation and for each challenging target. TRIZ method is a powerful tool to fulfil this process [27]. The resulted vectors of innovation represent paths towards which creativity and skills must be directed when local solutions are elaborated.

Step 6: Define major local PSS-specifications (major functionalities) to be fulfilled in relation to each target-function. This is performed by deploying into PSS-specifications the corresponding key value characteristics for each target-function. QFD-type relationship matrices are used in this respect. Value weight of each major PSS-specification is further calculated. In principle, a significant part of the major PSS-specifications are met in the case of more or all target-functions. By comparing the results of local deployments, one is able to see the impact which each major PSS-specification has in the equation of each target-function.

Step 7: Define a set of criteria to analyse the opportunity of implementing major PSS-specifications either to the level of product or service, or to the both, in various quantities. Rank criteria by means of AHP method.

Step 8: For each local set of major PSS-specifications establish the relationship level of each PSS-specification against each criterion from two perspectives: product perspective and service perspective. Calculate the result for each major PSS-specification by multiplying the importance of each criterion with the corresponding relationship level in relation to each major PSS-specification and adding the local results. Thus, for each local major PSS-specification a weight in relation to product and a weight in relation to service are determined. These weights provide a means for deciding how to implement each PSS-specification in the mix product-service (the case of local solutions).

Step 9: Use information from steps 4 (key value characteristics, weights and targets), 5 (vectors of innovation), 6 (PSS-specifications and weights) and 8 (product-service mix of distribution) to formulate local solutions of PSSs in relation to each target-function. Various tools can be used to support this step (e.g. CSDT [28]). Local solutions are detailed only at modules and sub-modules levels.

Step 10: Aggregate local solutions into an overall solution. In principle, the overall solution should comprise to the maximum possible extend the strengths of all local solutions. An aggregation algorithm is further proposed.

The target-function with the highest value weight in the set from step 1 will be taken as the starting point in the aggregation algorithm. It is symbolized with *PTF*. The other target-functions are grouped relative to *PTF*. *PTF* is correlated with the other n–1 target-functions in various ways: positive, negative or not correlated, as well as at various strengths. The type of correlation between two target-functions is determined by the correlations between their constitutive requirements. In this respect, the rest of the n–1 target-functions can be sorted on three categories: the group of those target-functions that are positive correlated with *PTF*, the group of those target-functions that are not correlated with *PTF* and the group of those target-functions that are negative correlated with *PTF*.

For each target-function (excepting *PTF*) a priority index is calculated by multiplying its value weight (from step 2) with the correlation coefficient (also from step 2) between the respective target-function and *PTF*. In the group of target-functions that are positive correlated with *PTF,* the target-functions will be ordered starting with the one

having the highest priority index and ending with the one having the lowest priority index. The same rule is kept for the group of target-functions that are negative correlated with *PTF*. It is highlighted the fact that the correlation coefficients are <0 in the group of negative correlated target-functions, so the one with the highest priority index will have the lowest magnitude in absolute value. The target-functions that are not correlated with *PTF* will be ordered starting with the one having the highest value weight and ending with the one having the lowest value weight.

In the last stage, the aggregated solution is generated following an iterative approach. The aggregated solution will result as a "compromise & combination" of the set of *n* local solutions. In this respect, the following rule is applied:

(a) The solution corresponding to the *PTF* will be taken and analyzed together with the solution corresponding to the first target-function in the group of target-functions that are positive correlated with *PTF*. Because the two target-functions are positive correlated, the best ideas from the local solutions will be combined, resulting an improved hybrid solution.

(b) The hybrid solution from (a) will be then analyzed against the local solution corresponding to the second target-function in the group of the target-functions that are positive correlated with *PTF*. The new variant will result as a combination of the best ideas from the hybrid solution generated at phase (a) and from the current local variant.

(c) The process will go on in the manner above described until all target-functions from the group of target-functions that are positive correlated with *PTF* are consumed. After that, the group of no correlated target-functions is taken into account and the process is continued until all of these target-functions are consumed. At the end, the group of target-functions that are negative correlated with *PTF* will be taken into account. Because at this phase potential conflicts could occur, they have to be solved without compromises, if possible. In this respect, it is firstly required to identify pairs of conflicting problems between the compared variants. Afterwards, innovative solutions have to be formulated. Methods like TRIZ could offer a real support in this respect. At the end of this process, the complete overall solution will be defined.

Step 11: Further, by calculating the overall value weight of each major PSS-specification, one can establish cost targets for development, having as starting point a target budget. The overall value weight of each major PSS-specification is obtained by deploying all requirements belonging to all target-functions (see step 3) into PSS-specifications.

Step 12: Results from steps 10 and 11 are used for further detailing the software-PSS (detailed design and planning at component level). Use-cases, modelling languages (e.g. UML [1]) and other specific tools for software analysis and design could be used to support this step.

4 APPLICATION EXAMPLE

For exemplifying the practical application of the methodology, the case of a software-PSS related to business excellence assessment and analysis is here considered.

The market need, expressed as general requirements, is (step 1): RQ_1 - training on organizational excellence, RQ_2 - organizational excellence assessment, RQ_3 - consultancy on organizational excellence, and RQ_4 - formulation of improvement projects.

		TF_5				
		TF_4				0
		TF_3				-1	+1
		TF_2			+2	-1	+1
		TF_1		+2	0	-1	0
		TF:	TF_1	TF_2	TF_3	TF_4	TF_5
RQ:	Impor-tance	Relationship matrix (bottom) and Correlation matrix (up)					
RQ_1	3.5	9	1	1	3	
RQ_2	4.0	3	9	3		27
RQ_3	1.4	9	1		9	
RQ_4	1.1	9			27	3
Value weight [%]		23.0	14.3	5.4	18.4	38.9

Table 1: Target-functions deployed against the need-related requirements.

The target-functions are (step 2): TF_1 - increased quality, TF_2 - increased learnability, TF_3 - increased ergonomy, TF_4 - low cost, TF_5 - short assessment duration. Table 1 summarizes steps 1 and 2 of the methodology.

A scale of [-2, -1, 0, 1, 2] was used for showing correlations between target-functions (negative, no or positive-correlated) and a scale of [0, 1, 3, 9, 27] for showing relationships between requirements and target-functions (0: no; 1: weak; 3: medium; 9: strong; 27: very strong relationship).

For TF_1, the following specific key requirements were formulated (and ranked using the AHP method): $KR_{1.1}$ - the solution should be easy to work with (25%), $KR_{1.2}$ - the solution should be reliable; it should work exactly as it's supposed to (42%), $KR_{1.3}$ - the software-PSS should be tolerant to errors (16%), $KR_{1.4}$ - the solution should be scalable; it should behave decently no matter how much data is to be handled (9%), $KR_{1.5}$ - the solution should be portable; the user should not be nailed down to a specific operating system (8%).

For TF_2, the following specific key requirements were formulated: $KR_{2.1}$ - the solution should be well-documented (63%), $KR_{2.2}$ - the solution should be consistent with other existing solutions (12%), $KR_{2.3}$ - all solution's constitutive elements should be clear enough to maximize user productivity (25%).

For TF_3, the following specific key requirements were formulated: $KR_{3.1}$ - all constitutive elements of the solution should behave as expected (31%), $KR_{3.2}$ - the user should have complete control on the solution (18%), $KR_{3.3}$ - permanently clear solution state (23%), $KR_{3.4}$ -only relevant information should be available to the user (28%).

For TF_4, the following specific key requirements were formulated: $KR_{4.1}$ - the solution should be ROI-attractive (82%), $KR_{4.2}$ - there should be a minimal setup cost (10%), $KR_{4.3}$ - there should be a minimal maintenance and support cost (8%).

For TF_5, the following specific key requirements were formulated: $KR_{5.1}$ - for a user which is familiar with organizational excellence models, the assessment session should be fast (30%), $KR_{5.2}$ - for an inexperienced user, completing a session should not take long (39%), $KR_{5.3}$ - an inexperienced user should easily understand the information on organizational excellence provided by the solution (31%).

Percentages in the brackets represent the local relative importance of the key requirements, as they have been determined with the AHP method.

For TF_1, the following key value characteristics were formulated: $KV_{1.1}$ - installation time, $KV_{1.2}$ - configuration / customization time, $KV_{1.3}$ - critical errors, $KV_{1.4}$ - number of assessment sessions to be handled, $KV_{1.5}$ - number of environments the solution might work in.

For TF_2, the following key value characteristics were formulated: $KV_{2.1}$ - documentation comprehensiveness, $KV_{2.2}$ - documentation access time, $KV_{2.3}$ - documentation topic access time, $KV_{2.4}$ - number of available languages, $KV_{2.5}$ - task understanding time.

For TF_3, the following key value characteristics were formulated: $KV_{3.1}$ - number of sessions after which a user becomes skilled in using the solution, $KV_{3.2}$ - complexity/ operation seen as number of constitutive elements/ operation, $KV_{3.3}$ - time for documenting during an assessment session.

For TF_4, the following key value characteristics were formulated: $KV_{4.1}$ - the initial investment for the user, $KV_{4.2}$ - the return on investment for the user, $KV_{4.3}$ - the maintenance costs, $KV_{4.4}$ - future releases costs, $KV_{4.5}$ - update costs.

For TF_5, the following key value characteristics were formulated: $KV_{5.1}$ - the assessment session duration for an experienced user, $KV_{5.2}$ - the assessment session duration for a new user, $KV_{5.3}$ - number of assessments after which a user becomes enough skilled in using the solution, $KV_{5.4}$ - assessment reports generation time, $KV_{5.5}$ - number of assessment report formats.

Table 2 highlights, for each target-function, the value weights and targets associated to the key value characteristics after the QFD-deployment process against the key requirements (step 4).

Three of the targets from table 2 are seen challenging: target for $KV_{1.5}$ - at least 3 environments in which the solution has to work, $KV_{2.5}$ - time to understand the task less than 1 minute, and $KV_{5.2}$ - the assessment session duration for a new user less than 1 day. Two of these three challenging targets are of higher priority from competitiveness point of view: targets for $KV_{2.5}$ and $KV_{5.2}$.

Analysing $KV_{2.5}$, the following mini-problem could be formulated: how to decrease task understanding time while making it (a) as simple as possible and (b) with minimal costs. TRIZ application [27] leads to the following vectors of innovation: (1) arrange “objects” in a way they can go immediately into action when required (e.g. intuitive/user-centric interface); (2) replace “stationary fields” with “moving fields” (e.g. dynamic help, including tutoring); (3) apply asymmetry (e.g. single format in, multiple formats out); (4) discard and regenerate parts (e.g. keep a pool of complex items for further use); (5) alternating (e.g. alternate service use with product use to fit better each task); (6) replace a continuous action with an impulse (e.g. use tutoring only where and when necessary); (7) make actions more “fluid” (e.g. use flexible ways to assess an issue); (8) enrich the “environment” (e.g. implement expert modules).

Regarding $KV_{5.2}$, the following mini-problem could be formulated: how to minimize the assessment session duration for a new user without increasing training or self-documenting time. TRIZ reveals the following vectors of innovation: (1) use sharing to counterweight parts of the system (e.g. division in multiple frames); (2) automatic adjustments (e.g. self-adjust scale); (3) change the flexibility and density (e.g. merging of modules, distributed completion/use of information); (4) enrich the “environment” (e.g. implement expert modules).

TF_1	$KV_{1.1}$	$KV_{1.2}$	$KV_{1.3}$	$KV_{1.4}$	$KV_{1.5}$
Local value weight [%]	10.7	12.4	29.3	21.7	25.9
Global weight [%]	2.5	2.9	6.7	5.0	6.0
Target	< 3 min	< 3 min	0	>100	≥3
TF_2	$KV_{2.1}$	$KV_{2.2}$	$KV_{2.3}$	$KV_{2.4}$	$KV_{2.5}$
Local value weight [%]	42.3	15.0	14.4	13.0	15.2
Global weight [%]	6.1	2.2	2.1	1.9	2.2
Target	>60 pp	<10 s	<3 s	≥3	<1 min
TF_3	$KV_{3.1}$	$KV_{3.2}$	$KV_{3.3}$	-	-
Local value weight [%]	35.1	47.0	17.9	-	-
Global weight [%]	1.9	2.5	1.0	-	-
Target	<5	<30	<30 min	-	-
TF_4	$KV_{4.1}$	$KV_{4.2}$	$KV_{4.3}$	$KV_{4.4}$	$KV_{4.5}$
Local value weight [%]	30.8	30.8	14.5	10.9	13.0
Global weight [%]	5.7	5.7	2.7	2.0	2.4
Target	<100 €	>250%	$\rightarrow 0$	$\rightarrow 0$	$\rightarrow 0$
TF_5	$KV_{5.1}$	$KV_{5.2}$	$KV_{5.3}$	$KV_{5.4}$	$KV_{5.5}$
Local value weight [%]	9.0	36.2	20.3	12.7	21.9
Global weight [%]	3.5	14.1	7.9	4.9	8.5
Target	< 2h	< 1day	≤4	≤1min	≥2

Table 2: Value weights and targets for *KV*s.

Vectors of innovation above presented are further considered as guiding patterns within step 9 of the methodology. The next step requires the definition of the major software-PSS specifications for each target-function.

Thus, for TF_1 we should have the following major PSS specifications: a setup and customization feature ($PSS_{1.1}$), a security policy centre for deciding access control to various data ($PSS_{1.2}$), a feedback centre, to report errors or inconsistencies ($PSS_{1.3}$), a migration feature from one environment to another one ($PSS_{1.4}$).

For TF_2, we should have: a documentation centre ($PSS_{2.1}$), a feedback centre ($PSS_{2.2}$) to report errors or inconsistencies, a start-up centre ($PSS_{2.3}$) to allow newbies to easily familiarize with the solution, even before actually using it.

For TF_3, the following PSS specifications are identified: a documentation centre ($PSS_{3.1}$), an assessment session manager, to easily add, edit or remove sessions ($PSS_{3.2}$), a status centre, where all important parameters of the solution should be available to the user ($PSS_{3.3}$).

For TF_4, major PSS specifications comprise: a setup and customization feature ($PSS_{4.1}$), a feedback centre, to report errors or inconsistencies ($PSS_{4.2}$), a migration feature from one environment to another ($PSS_{4.3}$), a documentation centre ($PSS_{4.4}$).

For TF_5, we should have: a documentation centre ($PSS_{5.1}$), an assessment session manager, to easily add, edit or remove sessions ($PSS_{5.2}$), an assessment centre, where the actual assessment of organizational excellence is done ($PSS_{5.3}$), a report generator ($PSS_{5.4}$), and an

expert centre, where improvement projects (based on assessments) are suggested ($PSS_{5.5}$).

Deploying local *KV*s via QFD-type matrices (step 6) leads to the calculation of the local value weights of local PSS-specifications (see Table 3).

TF_1	$PSS_{1.1}$	$PSS_{1.2}$	$PSS_{1.3}$	$PSS_{1.4}$	-
Weight [%]	21.9	6.7	40.7	30.7	-
TF_2	$PSS_{2.1}$	$PSS_{2.2}$	$PSS_{2.3}$	-	-
Weight [%]	73.1	10.5	16.4	-	-
TF_3	$PSS_{3.1}$	$PSS_{3.2}$	$PSS_{3.3}$	-	-
Weight [%]	35.4	20.1	44.5	-	-
TF_4	$PSS_{4.1}$	$PSS_{4.2}$	$PSS_{4.3}$	$PSS_{4.4}$	-
Weight [%]	26.5	30.8	24.9	17.8	-
TF_5	$PSS_{5.1}$	$PSS_{5.2}$	$PSS_{5.3}$	$PSS_{5.4}$	$PSS_{5.5}$
Weight [%]	27.3	5.1	37.6	20.6	9.3

Table 3: Local value weights for PSS-specifications.

The following criteria set was defined and ranked in order to analyse the opportunity to implement major PSS specifications (step 7): adaptability (23.2%); development time (8.8%); development cost (16.5%); implementation time (5.5%); implementation cost (20.2%); quality of deliverables (25.7%).

Further on, the relationship level of each PSS specification against each criterion, from both product and service perspectives, for each local set of major PSS specifications are established (five local sets are elaborated). A scale of [0, 1, 3 and 9] is used in completing the relationship matrices. For space reasons, only the first matrix is shown entirely (see Table 4). For the other four cases, results are highlighted in Table 5.

TF_1: increased quality			$PSS_{1.1}$	$PSS_{1.2}$	$PSS_{1.3}$	$PSS_{1.4}$
Adaptability	23.2	p	9	1	3	3
		s	1	3	3	3
Development time	8.8	p	9	3	3	3
		s	3	1	1	0
Development cost	16.5	p	3	3	3	9
		s	1	1	1	3
Implementation time	5.5	p	9	9	9	9
		s	3	1	3	9
Implementation cost	20.2	p	9	9	9	9
		s	3	1	3	3
Quality of deliverables	25.7	p	1	1	1	1
		s	1	0	1	0
Product (p)	Mark		569	330	377	476
	Weight [%]		79.9	73.2	68.7	67.5
Service (s)	Mark		143	121	172	229
	Weight [%]		20.1	26.8	31.3	32.5

Table 4: Product-service distribution for TF_1.

	TF_2	$PSS_{2.1}$	$PSS_{2.2}$	$PSS_{2.3}$	-	-
Product (p)	%	60.9	76.6	72.6	-	-
Service (s)	%	39.1	23.4	27.4	-	-
	TF_3	$PSS_{3.1}$	$PSS_{3.2}$	$PSS_{3.3}$	-	-
Product (p)	%	64.5	85.3	86.1	-	-
Service (s)	%	35.5	14.7	13.9	-	-
	TF_4	$PSS_{4.1}$	$PSS_{4.2}$	$PSS_{4.3}$	$PSS_{4.4}$	-
Product (p)	%	74.8	83.6	83.6	83.5	-
Service (s)	%	25.2	16.4	16.4	16.5	-
	TF_5	$PSS_{5.1}$	$PSS_{5.2}$	$PSS_{5.3}$	$PSS_{5.4}$	$PSS_{5.5}$
Product (p)	%	88.5	88.1	82.5	77.5	65.6
Service (s)	%	11.5	11.9	17.5	22.5	34.4

Table 5: Product-service distribution for the target-functions TF_2, TF_3, TF_4 and TF_5.

Further on (step 9), the following five local solutions were formulated, one for each target function. The local solution for TF_1 - increased quality (detailed at generic module level) is: (a) there should be an internet site to assist the user in downloading and installing the product, (b) the site should also have a feedback module, to record user observations on the overall solution, (c) there should be a procedure for recording feedback from the user via a consultant, (d) solution customization should be carried out as a service (via expert/tutor), (e) there should be different versions of the product for each operating system or the product should be cross-platform, (f) there should be an export / import module for data in order to allow migration from one environment to another one (for example, from one operating system to another one, for the product component of the solution).

The local solution for TF_2 - increased learnability (detailed at generic module level) comprises the following: (a) training sessions on organizational excellence, both as integrated help in the software and as courses held by a consultant (expert), (b) automated feedback reports, provided by the software, (c) feedback procedures to be collected by a consultant / expert during solution implementation, (d) there should be a quick-guide on software usage and organizational excellence (for beginner users), both in the software and online, on the solution's site, (e) there should be a “demo” version to contain enough training material to make the solution attractive for SMEs (the intended target audience).

The local solution for TF_3 - increased ergonomy (at generic module level) is about: (a) assessment sessions should be managed by a dedicated module (to easily add, edit, and remove sessions), (b) an import / export feature should be built for backup purposes and / or switching to versions running on other platforms, (c) system status should be always visible (e.g. status & progress bars).

The local solution for TF_4 - cost (detailed at generic module level) is: (a) an offline setup wizard for installing the software, (b) a settings panel to allow the application customization, (c) there should be a procedure, maybe the first step of the implementation, to allow service customization / personalization, (d) an online feedback module (included in the site), (e) an error-reporting module, integrated in the software, (f) an import / export module for all user data, (g) an assessment session management module.

The local solution for TF_5 - assessment duration (detailed at module level) includes: (a) an integrated help module, (b) an integrated documentation module on organizational excellence, (c) an assessment session management module, (d) an assessment session module, where a session should be completed or revised, (e) a report

generation module, based on a completed assessment session, (f) an expert module to generate recommendations based on the previously generated reports, (g) a procedure to analyse reports and expert module recommendations (direct meeting between user and consultant).

$PTF = TF_5$	TF_1	TF_2	TF_3	TF_4
Value weight [%]	23.0	14.3	5.4	18.4
Correlation relative to *PTF*	0	+1	+1	0
Priority index	0	14.3	5.4	0
Group (+, not or -)	not	+	+	not
Order	3	1	2	4

Table 6: Grouping TFs relative to *PTF*.

The aggregation process for formulating the final solution is the next step of the methodology. According to the algorithm from step 10 of the methodology, in this case study TF_5 (short assessment duration) is *PTF*.

Analysing Table 1 from step 2, we have TF_2 and TF_3 positive correlated with *PTF* and TF_1 and TF_4 not correlated with *PTF*. There are no negative correlations between *PTF* and other target functions. The priority indexes for TF_1, …, TF_4 are shown in Table 6.

The first integration process consists of combining local solution 5 with local solution 2, and further the resulted hybrid combined with local solution 3. The following intermediate solution was obtained: (a) a software integrated help module, (b) training sessions on organizational excellence, both as integrated help in the software (an integrated documentation module) and as courses held by a consultant (expert), (c) feedback reports, provided by the software (feedback on PSS usage, problems encountered, errors, inconsistencies, etc.); both automated and collected by a consultant / expert during solution implementation, (d) a quick-guide on software usage and organizational excellence (for beginner users), both in the software and online, on the solution's site, (e) a user friendly “demo” version to contain enough training material to make the solution attractive for SMEs (the intended target audience), (f) assessment sessions should be managed by a dedicated module (to easily add, edit, remove sessions), (g) an assessment session module, where a session should be completed or revised, (h) a report generation module, based on a completed assessment session, (i) an expert module to generate recommendations based on the previously generated reports, (j) a procedure to analyse reports and expert module recommendations (direct meeting between the user and the consultant), (k) an import / export feature should be built for backup purposes and / or switching to versions running on other platforms, and (l) system status should be always visible (e.g. status & progress bars).

The result is further aggregated with the local solution of TF_1, and the combination is finally aggregated with the local solution of TF_4. This integration process produces the (final) solution listed below. All modules and sub-modules (noted below with $PSS_{f.x}$, where *x* ranges from 1 to 16) that build the final solution are deployed against the initial requirements (the key requirements for TF_1, …, TF_5); a value weight is computed for each (sub)module (as required by step 11). Thus, we have:

- $PSS_{f.1}$ - a software integrated help module: 6.45%.
- $PSS_{f.2}$ - training sessions on organizational excellence, both as integrated help in the software (an integrated documentation module) and as courses held by a consultant (expert): 7.43%.
- $PSS_{f.3}$ - feedback reports, provided by the software (feedback on PSS usage, problems encountered, errors, inconsistencies, etc.); both automated and collected by a consultant / expert during solution implementation: 3.05%.
- $PSS_{f.4}$ - a quick-guide on software usage and organizational excellence (for beginner users), both in the software and online, on the solution's site: 9.45%.
- $PSS_{f.5}$ - a “demo” version to contain enough training material to make the solution attractive for SMEs (the intended target audience): 13.79%.
- $PSS_{f.6}$ - an offline setup wizard should be available; there should also be a site module to assist the user in downloading and installing the product: 1.61%.
- $PSS_{f.7}$ - a settings panel to allow the application customization: 3.27%.
- $PSS_{f.8}$ - a procedure, maybe over the first step of the implementation, to allow service customization / personalization: 5.98%.
- $PSS_{f.9}$ - assessment sessions should be managed by a dedicated module (to easily add, edit, remove sessions): 4.84%.
- $PSS_{f.10}$ - an assessment session module, where a session should be completed or revised: 6.46%.
- $PSS_{f.11}$ - a report generation module, based on a completed assessment session: 6.24%.
- $PSS_{f.12}$ - an expert module to generate recommendations based on the previously generated reports: 7.86%.
- $PSS_{f.13}$ - a procedure to analyse reports and expert module recommendations (direct meeting between user and consultant): 10.67%.
- $PSS_{f.14}$ - different versions for the product for each operating system: 5.71%.
- $PSS_{f.15}$ - an import / export feature for backup purposes and / or switching to versions running on other platforms: 3.41%.
- $PSS_{f.16}$ - system status always visible (e.g. status & progress bars): 3.82%.

Apparently surprising, the analysis revealed that the most important features to focus on are training-related: the demo version, the training sessions on organizational excellence or the quick-guide.

A problem domain document, use cases, an architecture document and a technical specification document were elaborated in step 12, based on the data obtained in steps 10 and 11.

A software application for assessing organizational excellence, named *business eXXplorer*, was developed to fulfil the product side of the software-PSS. Programming was done in Delphi and is now being ported to FreePascal / Lazarus which supports cross-compiling (thus being able to produce executables for Linux, Mac and Windows). The menu of *business eXXplorer* consists of the following items:

- Organizational excellence: a tutorial on what organizational excellence is and how it is assessed.
- User guide: the help section, where all topics related to *business eXXplorer*'s usage are described.
- Business assessment: opens the assessment session manager and, further on, the assessment module. It includes an intuitive guiding interface, and consistent help for each step.
- Result interpretation: opens the assessment session manager, so that the user can choose a (completed) session (called a “project” in *business eXXplorer*) and,

further on, the report generation and visualization module (which includes the expert module that issues recommendations).

- Settings: allows the application customization.
- Exit: closes *business eXXplorer*.

The interface of version 1 is available in Romanian only, but the next versions will support multiple languages.

Apart from the actual software, the *business eXXplorer* package consists of an internet site with free access (containing technical information, detailed explanations about organizational excellence, detailed description of *business eXXplorer*'s main features, a demo version for download, and usage examples consisting of assessment reports), a user manual, a quick-start brochure, a video tour of the software, and an automated setup tool.

Consultants can easily integrate and adapt their training and tutoring activities with the *business eXXplorer*'s features. Highlights about the main modules of *business eXXplorer* are further presented.

The user can browse through 13 major training topics on organizational excellence, grouped into two chapters (general information and assessment methods – see Figure 1).

An assessment session consists of self-evaluation against 164 criteria, grouped into 9 key business-related units: leadership, strategies, resource management, human resource management, processes, employee satisfaction, customer satisfaction, community satisfaction, and business performance.

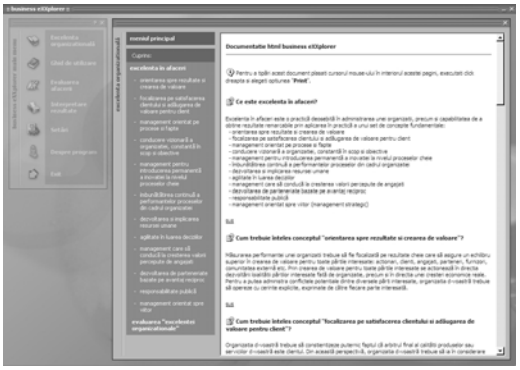

Figure 1: The organizational excellence built-in tutorial.

Each criterion has detailed explanations; the interface is built in such a way that beginners might easily read them, while advanced users might focus just on choosing a conformity level (see Figure 2).

After an assessment session is completed, *business eXXplorer* generates specific reports. The report contains four sections, with both general and in-depth balanced analysis among business possibilities and results.

Figure 3 shows a temple-based visual report. Besides this, some other specific reports are generated by *business eXXplorer* (radar-based, bar-based, balance-based, criteria based, project comparison-based etc.; they are not visualized in this paper).

Moreover, the software-PSS solution elegantly integrates the vectors of innovation proposed at step 5 of the methodology (please review these vectors at page 4 of the paper). Also, during the aggregation process in step 10, the final solution was balanced distributed into product and service, considering information from Table 4 and Table 5 for decision making. This process is a creative one, therefore aggregation does not mean a "mechanical" integration of local solutions; it exploits local solutions to work out a superior global solution which incorporates the strengths of the local solutions and (if possible) brings additional value in the system.

In this context, the targets of the key value characteristics (see Table 2) have been used during the development, testing and validation phases of the final software-PSS solution. Just for informing, the proposed software-PSS solution meets entirely the planned targets.

Figure 2: The assessment session module (with help).

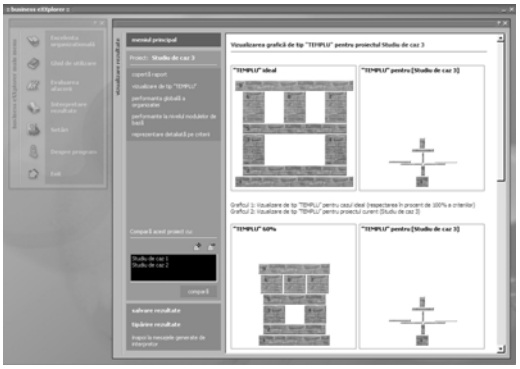

Figure 3: The report browsing module (the "business temple" representation).

5 FURTHER RESEARCHES

Researches to extend the proposed methodology by comprising wider aspects of the PSS's life-cycle, as well as high dynamics in changing stakeholders' values and requirements (both in meaning and intensity) in a continuously adapting global business environment will be further taken into consideration.

6 CONCLUSIONS

A methodology for concurrent planning and design of software product-service systems against a complex set of target-functions is introduced in this paper. The quality of the methodology arises from its potential to propose solutions that incorporate a superior balance between various aspects related to stakeholder's satisfaction, with the consideration of customer's needs as the leading vector. In order to achieve this goal, the methodology exploits the potential of structured tools of performance planning (e.g. AHP, QFD) and innovative problem solving (e.g. TRIZ, CSDT).

The methodology also provides an appropriate framework for adequate distribution of the functions/ features of the solution between product and service in the attempt to maximize value added performances both for customer and producer. The framework breaks down the complexity of the problem by facilitating, at a first stage, a concurrent focalization on separate target-functions to explore local potentials. Later on, the best local results are "combined" in a balanced way to define a mature global solution (or

several mature solutions). From this perspective, the methodology stresses the key role that creativity plays in refining functionalities to a complex problem, based on "local" solutions to specific target-functions. The aggregation algorithm introduced by the methodology is, by itself, an effective mean for supporting the creative process.

Moreover, the vectors of innovation issued by the methodology allow a software designer not only to solve possible conflicts at early stages of the development process, but also help her/him identifying those distinctive features of the solution that are difficult to determine using a classic approach.

The case study introduced in the paper reveals the capability of the proposed methodology to handle both antagonistic aspects during the conceptualization and design phases of a new software-PSS, as well as less tangible issues with value added potential for the decision making process.

7 ACKNOWLEDGMENTS

Financial support from the European Commission within the FP7 research project TECH-IT-EASY/232410 is acknowledged with gratitude.

8 REFERENCES

[1] Futrell, R., Shafer, D. and Shafer, L., 2002, Quality Software Project Management, Prentice Hall.

[2] Barney, S., Aurum, A. and Wohlin, C., 2008, A Product Management Challenge: Creating Software Product Value Through Requirements Selection, Journal of Systems Architecture, 54: 576-593.

[3] Agarwal, N. and Rathod U., 2006, Defining "Success" for Software Projects: An Exploratory Revelation, International Journal of Project Management, 24: 358-370.

[4] Petter, S., 2008, Managing User Expectations on Software Projects: Lessons from the Trenches, International Journal of Project Management, 26: 700-712.

[5] Juristo, N., Moreno, A.M. and Sanchez-Segura, M-I., 2007, Analyzing the Impact of Usability on Software Design, The Journal of Systems and Software, 80: 1506-1516.

[6] Mantere, T. and Alander, J.T., 2005, Evolutionary Software Engineering, a Review, Applied Soft Computing, 5: 315-331.

[7] Pereira, J., Cerpa, N., Verner, J., Rivas, M. and Procaccino, J.D., 2008, What Do Software Practitioners Really Think About Project Success: A Cross-Cultural Comparison, The Journal of Systems and Software, 81: 897-907.

[8] Xie, G., Zhang, J. and Lai, K.K., 2006, Risk Avoidance in Bidding for Software Projects Based on Life Cycle Management Theory, International Journal of Project Management, 24: 516-521.

[9] Jung, H.W., 2007, Validating the External Quality Sub-characteristics of Software Products According to ISO/IEC 9126, Computer Standards & Interfaces, 29: 653–661.

[10] Zhang, X. and Pham, H., 2006, Software Field Failure Rate Prediction before Software Deployment, The Journal of Systems and Software, 79: 291-300.

[11] Bitzer, J., 2004, Commercial versus Open Source Software: The Role of Product Heterogeneity in Competition, Economic Systems, 28: 369-381.

[12] Liu, J.Y.C., Chen, V.J., Chan, C.L. and Lie, T., 2008, The Impact of Software Process Standardization on Software Flexibility and Project Management Performance: Control Theory Perspective, Information and Software Technology, 50: 889-896.

[13] Ajila, S.A. and Wu, D., 2007, Empirical Study of the Effects of Open Source Adoption on Software Development Economics, The Journal of Systems and Software, 80: 1517-1529.

[14] Verner, J.M., Evanco, W.M. and Cerpa, N., 2007, State of the Practice: An Exploratory Analysis of Schedule Estimation and Software Project Success Prediction, Information and Software Technology, 49: 181-193.

[15] Aurich, J.C., Fuchs, C. and Wagenknecht, C., 2006, Life Cycle Oriented Design of Technical Product-Service Systems, Journal of Cleaner Production, 14: 1480-1494.

[16] Aramand, M., 2008, Software Products and Services are High Tech? New Product Development Strategy for Software Products and Services, Technovation, 28: 154-160.

[17] Yang, X., Moore, P., Pu, J-S. and Wong, C-B., 2009, A Practical Methodology for Realizing Product Service Systems for Consumer Products, Computers & Industrial Engineering, 56: 224-235.

[18] Morelli, N., 2006, Developing New Product Service Systems (PSS): Methodologies and Operational Tools, Journal of Cleaner Production, 14: 1495-1501.

[19] Stamelos, I., 2009, Software Project Management Anti-Patterns, The Journal of Systems and Software, article in press.

[20] Han, W-M. and Huang, S-J., 2007, An Empirical Analysis of Risk Components and Performance on Software Projects, The Journal of Systems and Software, 80:42-50.

[21] Lenfle, S. and Midler, C., 2009, The Launch of Innovative Product-Related Services: Lessons from Automotive Telematics, Research Policy, 38: 156-169.

[22] Yang, X., Moore, P. and Chong, S.K., 2009, Intelligent Products: From Lifecycle Data Acquisition to Enabling Product-Related Services, Computers in Industry, 60: 184-194.

[23] Shimoura, Y., Hara, T. and Arai, T., 2009, A Unified Representation Scheme for Effective PSS Development, CIRP Annals: Manufacturing Technology, 58: 379-382.

[24] Büyüközkan, G., Ertay, T., Kahraman, C. and Ruan, D., 2004, Determining the Importance Weights for the Design Requirements in the House of Quality Using the Fuzzy Analytic Network Approach, International Journal of Intelligent Systems, 19: 443-461.

[25] Wang, H., Xie, M. and Goh, T., 1998, A Comparative Study of the Prioritization Matrix Method and the Analytic Hierarchy Process Technique in Quality Function Deployment, Total Quality Management, 9: 421-430.

[26] Akao, Y. and Mazur, G.H., 2003, The Leading Edge in QFD: Past, Present and Future, International Journal of Quality & Reliability Management, 20: 20-35.

[27] Altshuller, G., 2000, The Innovation Algorithm: TRIZ, Technical Innovation Centre.

[28] Brad, S., 2008, Complex System Design Technique, International Journal of Production Research, 46: 5979-6008.