17th International Conference on
AEROSPACE SCIENCES & AVIATION TECHNOLOGY,
ASAT - 17 – April 11 - 13, 2017, E-Mail: asat@mtc.edu.eg
Military Technical College, Kobry Elkobbah, Cairo, Egypt
Tel: +(202) 24025292 – 24036138, Fax: +(202) 22621908

Transdisciplinary Quality System Development Lifecycle Model: Enhancement and Application

Mohamed Asaad Abdelrazek[*], Amir Taher El-Sheikh[†], M.Zayan[‡], A.M. Elhady[§]

Abstract: Due to the increasing complexity in systems development as well the highly competitive environment, it has become apparent that new ways of reducing system development time, minimize cost, enhance organizational efficiency, increase customer satisfaction, and improve the quality of the final systems are required. The Transdisciplinary Quality System development lifecycle (TQSDL) Model is one of the new systematic approaches models used as a design management tool for developing the complex systems based on scientific principles. This model is applied across the whole systems development lifecycle. The model processes are based on the Axiomatic Design (AD) and Quality Function Deployment (QFD) tool. One of the factors that influence the quality and the cost of the final systems is the supply chain. Also, one of the largest difficulties during the development of complex systems is tracking and expecting the changes in development cost due to changes in customer needs or function requirements.

In this paper, the TQSDL model is extended to cover the supply chain process by adding a new domain to TQSDL domains. This new domain is introduced to manage the relationships between system components and all suppliers by using the QFD tool. As well as, Dependency Structure Matrix (DSM) will be integrated into the TQSDL process to improve information management and to address the interdependency between the system components and the interrelation between activities during systems development. Moreover, DSM with a new characteristic vector will aid to capture the changes in the development cost due to the changes in customer needs or function requirements. The enhancement in TQSDL model aims to provide a complete framework and systematic thinking to the designers and technical managers during the whole system development lifecycle. Moreover, it will also support the decision makers on whether or not to implement changes to a design.

Keywords: Transdisciplinary Quality System Development Lifecycle Model, Axiomatic design, Quality function deployment, Dependency Structure Matrix, Systems development lifecycle.

Nomenclature

{SI}: stakeholders' identification vector	[SA]: stakeholders' allocation matrix
{FR}: functional requirements vector	[R]: requirement matrix
{CN}: customer need vector	[C]: constraint matrix

[*] Egyptian Armed Forces, mohamed35.asaad@gmail.com
[†] Administrative Control Authority, amir.arafa@gmail.com
[‡] Head of R&D Sector, Nilesat, m.zayan@nilesat.com.
[§] NExSat-1 Project Manager, NARSS, bakr20081@gmail.com.

{DP}: design parameter vector
{IC}: input constraint vector
{SC}: system component vector
{PV}: process variable vector
{FTC}: functional test case vector
{CTC}: component test case vector
{SUPI}: supplier identification vector
{DPC}: design parameter cost vector

[D]: design matrix
[CA]: constraint allocation matrix
[SS]: system structure matrix
[P]: process matrix
[FT]: functional test matrix
[CT]: component test matrix
[SCA]: system component allocation matrix
{CC}: components cost vector.

1. Introduction

The system development process of complex systems such as satellite, airplanes, submarines, car,…etc take the collaborative effort of a hundred even thousands of people over several years. These systems themselves consists of thousands of components, all of which must be designed, assembled, integrated and tested to get the right system with the right quality at the right time as well as, with the right price to achieve the customer satisfaction. From this picture, there are two large systems are involved in the system development process, teams involved in the development process and system components. In order to design and manage the complex system development process, TQSDL model will be used as a design management tool.

The goal of this paper is to enhance the TQSDL model which is proposed by us [1] to cover all the system development dimensions [2]. Moreover, in order to be able to capture and manage the interactions between system components and the interrelation between the design teams, component-based DSM, and activity-based DSM are integrated into TQSDL process respectively. Also, With the help of the traceability feature of TQSDL model and the integration of the DSM into TQSDL process, the prediction of the change in development cost due to change in customer needs or functional requirements can be estimated. The TQSDL process helps the transdisciplinary system development team throughout the system design and development activities. Moreover, the TQSDL model provides a big-picture and detail view of the system development knowledge.

This paper is organized as follows. Section 2 provides a brief overview of QFD. Section 3 provides a brief overview of DSM. Section 4 presents the TQSDL model. Section 5 describes the enhancement of TQSDL model domains and process. Section 6 describes the estimation of change in development cost by using TQSDL model. Section 7 presents the conclusion of this paper.

2. Quality Function Deployment Overview

Quality function deployment (QFD) had been developed in Japan in the late of 1960 by Professor Yoji Akao. It defines as “QFD provides specific methods for ensuring quality throughout each stage of the product development”[3]. By 1972 the power of QFD had been proven well at Mitsubishi Heavy Industries. The benefits when using QFD properly are 50% reduction in the time needed to develop the system, and 20% -60% reduction in start-up costs, and 30% -50% reduction in design changes and 20% -50% lower warranty claims.

The main objectives of implementing QFD are:

a. Collect all the customer needs.
b. Translate the vague customer needs into technical characteristics.
c. Provide a quality product or service that increase customer satisfaction.

A growth in customer satisfaction by 1% was linked with a 2.37% increase in the return on investment. Moreover, a decrease of customer satisfaction by 1% was linked with a decrease of 5.08% in the return on investment [4].The Quality function deployment format is shown schematically in Fig. 1 [5].

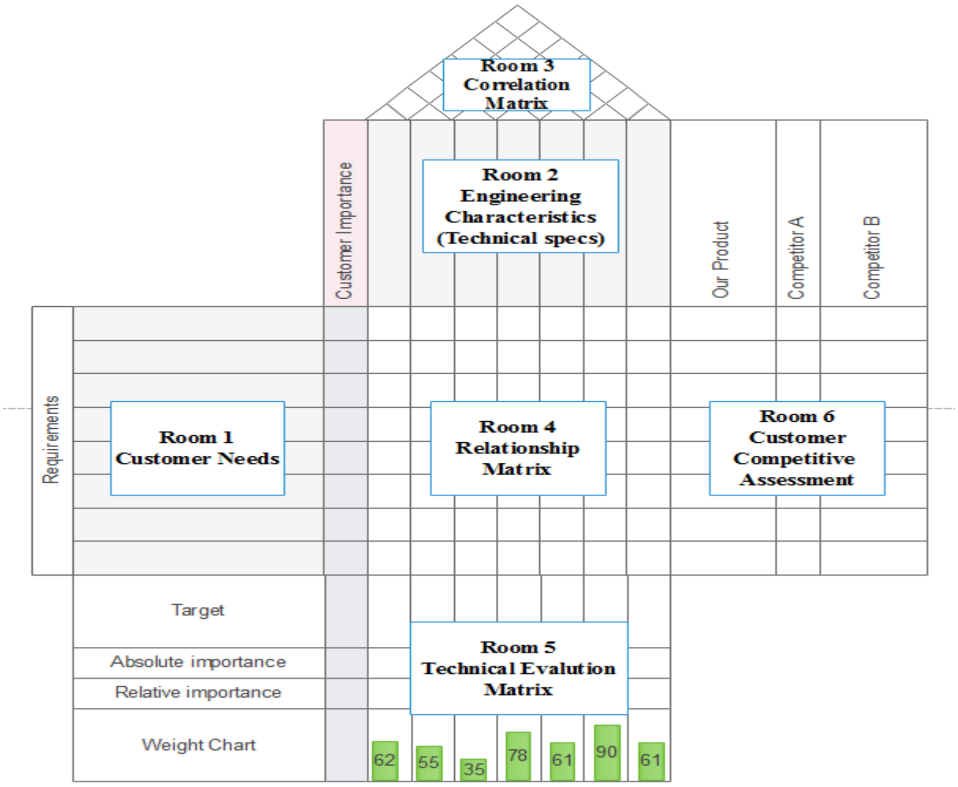


Fig. 1. Quality function Deployment

Room 1is the “customer needs” or “voice of customer”, which describe the needs and hopes of the customers. These requirements tend to be qualitative and in the customer’s expression. The customer needs importance will be represented in the customer importance column on a scale 1 to 5. Scale 5 represents very high importance and scale 1 represents very low importance [6].

Room 2 is the engineering characteristics (Technical specification) which describe technical characteristics of the system in measurable terms. In this room, the engineers list the engineering characteristics that are likely to affect one or more of the customer needs.

Room 3 is the correlation matrix (trade off). Each cell in the correlation matrix is a measure of the possible correlation of two different engineering characteristics. The correlation relationship indicates whether it is a positive supporting relationship or a negative conflicting relationship between each two engineering characteristics.

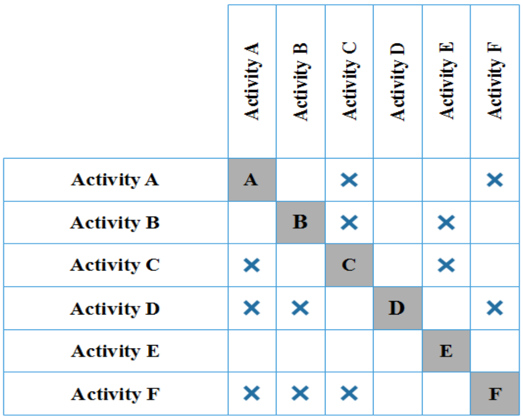
Room 4 is the relationship matrix which highlights the relationship between each customer need and engineering characteristics. Usually, these relationships are measured by strong relationship, medium relationship or weak relationship [6].

Room 5 is the technical evaluation matrix. This room includes the target value for each engineering characteristics which reflect what is needed to assure customer satisfaction. Moreover, this room includes the relative importance and absolute importance for each engineering characteristics. The relative importance and the absolute importance are evaluated by using the relationship matrix and customer needs importance.

Room 6 is the customer competitive comparison which rates how well our system performs on each requirement against the best of our competition (SWOT analysis).

3. Design Structure Matrix Overview

The dependency structure matrix (DSM) is introduced by Don Stewered in 1981[7]. The DSM has become popular as tools of planning system development, systems engineering, project planning and management, and organizational development [8]. The DSM is a square matrix-based tool with identical row and column labels which is used to address the interdependency between the system components, activities, parameters, and teams [9]. In the example of activity-based DSM in Fig. 2, activities are represented along the diagonal. The dependency of activity on another is represented by off-diagonal marks. Reading across a row reveals all of the activities whose output is needed to perform the activity corresponding to that row. Reading down a column reveals which activities receives information from the activity corresponding to that column. For example, activity B receives information from activity C and E, and provide information for activity D and F.



	Activity A	Activity B	Activity C	Activity D	Activity E	Activity F
Activity A	A		×			×
Activity B		B	×		×	
Activity C	×		C		×	
Activity D	×	×		D		×
Activity E					E	
Activity F	×	×	×			F

Fig. 2. Task-Based DSM

There are four types of DSM. These types are divided into two main categories, static and time-based as summarized in Fig. 3 and Table 1 [8],[10].

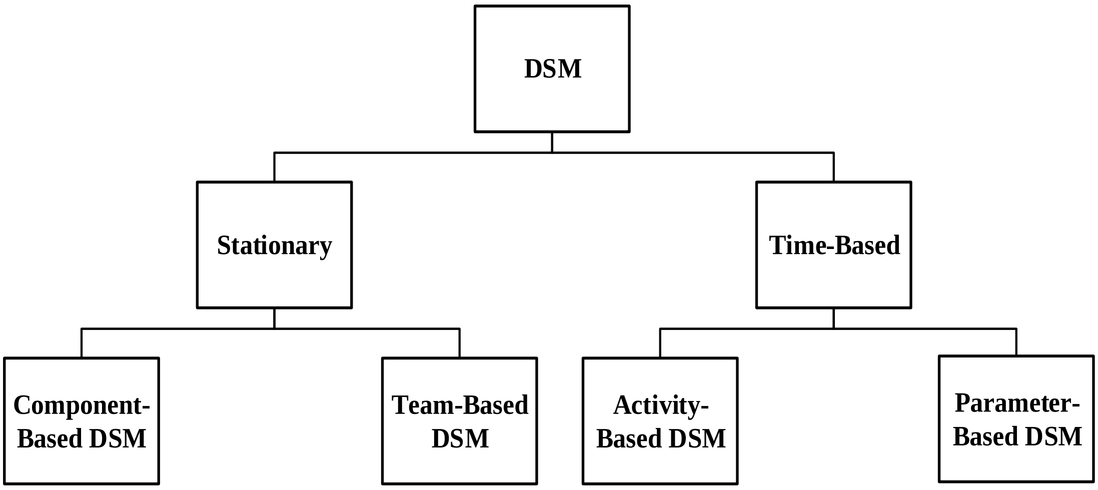


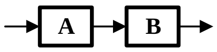
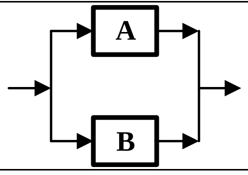
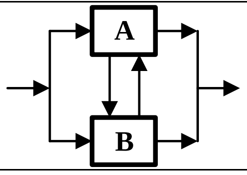
Fig. 3. DSM Types

Table 1 Different types of data that can be represented by DSM

Type	Application	Analysis Method
Component-Based DSM	Used for modeling multi-component relationship	Clustering
Team-Based DSM	Used for modeling multi-team interface	Clustering
Activity-Based DSM	Used for modeling activity input/output based on activities information flow	Partitioning & Tearing
Parameter-Based DSM	Used for modeling low-level relationship between parameters	Partitioning & Tearing

There are three different configurations of the DSM for describing the relationship amongst system component, activity, task, or teams as summarized in Table 2 [11]. In sequential relationship (decoupled configuration), design element B (component, activity, team, or parameter) require information from design element A, this means that design element A must be completed before design element B. In parallel relationship (uncouple configuration) design elements A and B don't require information from each other, this means that both design elements can be performed in parallel. In coupled relationship, design elements A and B require information from each other, this means that both design elements have to be performed in parallel with iterations involved.

Table 2 DSM Configuration

Three configurations that characterize a system			
Relationship	Sequential (Dependent)	Parallel (Independent)	Coupled (Interdependent)
Graph Representation			
DSM Representation	A B A ■ B X ■	A B A ■ B ■	A B A ■ X B X ■

4. Transdisciplinary Quality System Development Lifecycle Model

4.1 Fundamentals of Axiomatic Design Theory

The concept of axiomatic design is proposed by Nam P. Suh in the late of 1970s and has been finalized around 1990. Axiomatic design is defined as "prescriptive engineering design theory that provides a systematic thinking and scientific basis for making design decision" [12]. This theory can be applied to product design, organization, systems design, and software. The axiomatic design consists of four concepts:

a. Domains.
b. Design hierarchy.
c. Zigzagging.
d. Two design axioms.

The design is the interplay between “what we want to achieve” and “how we choose to satisfy the need”, consequently, the scientific theoretical basis that gives structure to the design process is established by the axiomatic design method [12]. To arrange and systematize this interplay process AD creates four domain as shown in Fig. 4:

a. Customer domain.
b. Functional domain.
c. Physical domain.
d. Process domain.

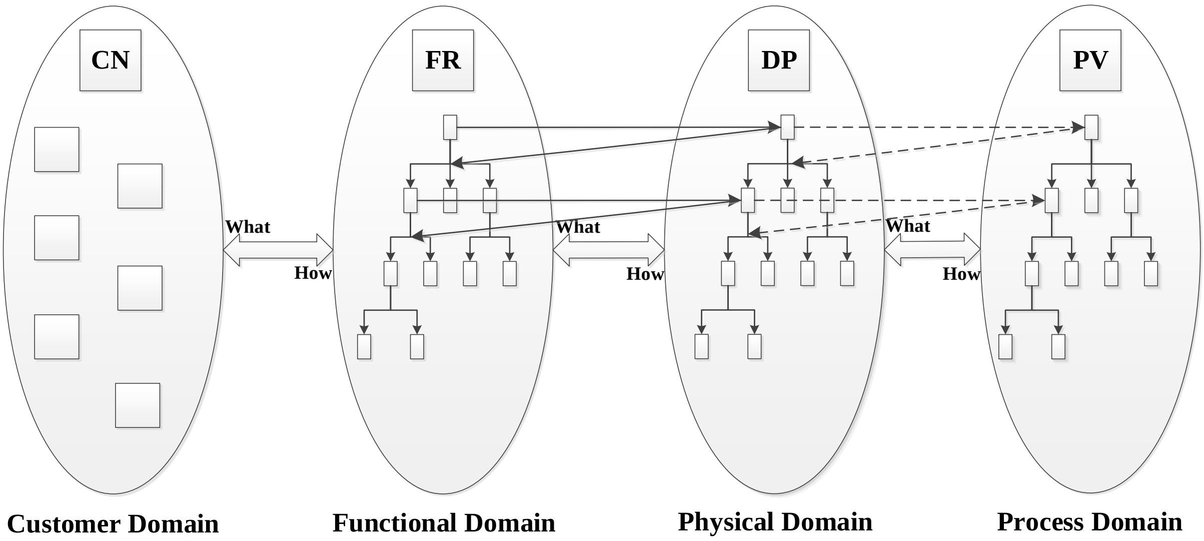


Fig. 4 Axiomatic Design Domains

The customer domain is characterized by the needs that the customer is looking for in a system. In the functional domain, the customer needs are specified in functional requirements (FRs) and constraints (Cs). To achieve the specified FRs, design parameters (DPs) is conceived in the physical domain. Finally, a process that is characterized by process variables (PVs) in the process domain is developed to produce the system specified in DPs. Through the decomposition process of the highest level of FR, DP, and PV, the hierarchies will be established, which are a representation of the design architecture. The decomposition process is performed by zigzagging between two adjacent domains, the left domain is “what” domain and the right domain is “how” domain (Fig. 4).

Based on the experience and observation, Suh proposed two design axioms that govern the design process [13]. The first axiom is called Independence Axiom and the second axiom is called Information Axiom. During the mapping process, the right design decision will be made by using independence axiom. In the case of the presence of more than one solution that satisfies the independence axiom, the information axiom will be used. The axioms are formally stated as[12]:

a. Axiom 1: maintain the independence of the functional requirements (FRs).
b. Axiom 2: minimize the information of the design.

The axiomatic design doesn’t support the whole system development lifecycle. Consequently, Bulent Gumus propose a Transdisciplinary system development lifecycle (TSDL) model to overcome this weak point [14]. In TSDL model, one new domain and four new characteristics vector are added as shown in Fig. 5. The test domain includes the test activities which are represented by two characteristic vectors (components test case (CTCs) and functional test case (FTCs)). Input constraints (IC) and system components (SC) are added to the functional domain and physical domain respectively.

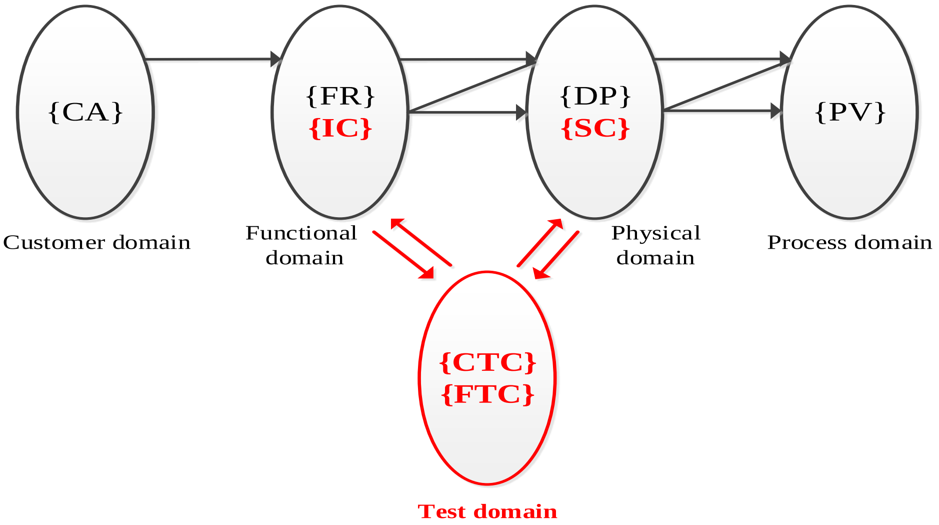


Fig. 5 TSDL Domains

4.2 TQSDL Model Domains

The successful management of new system development is an important source of gaining competitive advantage. Moreover, with the increasing in competition, customer satisfaction is a growing concern of every business. The improving of customer satisfaction will lead to increasing of customer loyalty. An increase in customer loyalty by 5% can increase the profit of a business by 100% [4].TQSDL Model is one of the new systematic approaches models used as a design management tool for developing the complex systems based on scientific principles. This model has been proposed by us to improve the TSDL model in the area of domain entity and the process of system development by adding a new domain and integrating QFD in the development process respectively [1]. The TQSDL approach can be used in design and development of systems, products, services, organizations and software.

To obtain a high level of customer loyalty, we have to understand the customer needs very well and obtain a full picture about our stakeholders. The customer domain in the TSDL model is characterized by the needs that the customer is looking for in a system [12]. Stakeholders' identities are not included in the customer domain. Furthermore, stakeholders' identification is the first and the important step in the system development process. This step helps to ensure that no important needs are neglected. For these reasons, the stakeholder domain has been added to TSDL model domains (Fig. 6) [1]. Moreover, QFD has been integrated into the TQSDL process to control the mapping between Stakeholder domain and customer domain. The stakeholder domain is characterized by the stakeholders' identification vector {SI}. A new theorem has been created to control the process of mapping and zigzagging between the stakeholders' domain and customer needs domain.

- **Theorem (Allocate Stakeholders {SI} to Customer Needs {CNs}):** "The stakeholders that are identified {SI} from the stakeholders' identification and analysis techniques are allocated to {CNs} without restricting to independence axiom. Several stakeholders allocate to one need or multiple needs mapping to one stakeholder" [1].

The mathematical relationship between the characteristics vectors can be written as:

$$\{SI\} = [SA]\{CN_i\} \qquad (1)$$

$$\{FR\} = [A]\{DP\} \qquad (2)$$

$$\{CN\} = [R]\{FR_i\} \qquad (3)$$

$$\{CN\} = [C]\{IC\} \qquad (4)$$

$$\{FR\} = [D]\{DP\} \qquad (5)$$

$$\{IC\} = [CA]\{DP\} \qquad (6)$$

$$\{DP\} = [SS]\{SC\} \qquad (7)$$

$$\{SC\} = [P]\{PV\} \qquad (8)$$

$$\{FR\} = [FT]\{FTC\} \qquad (9)$$

$$\{SC\} = [CT]\{CTC\} \qquad (10)$$

The detail description of the TQSDL model domains and the process will be presented in the next section.

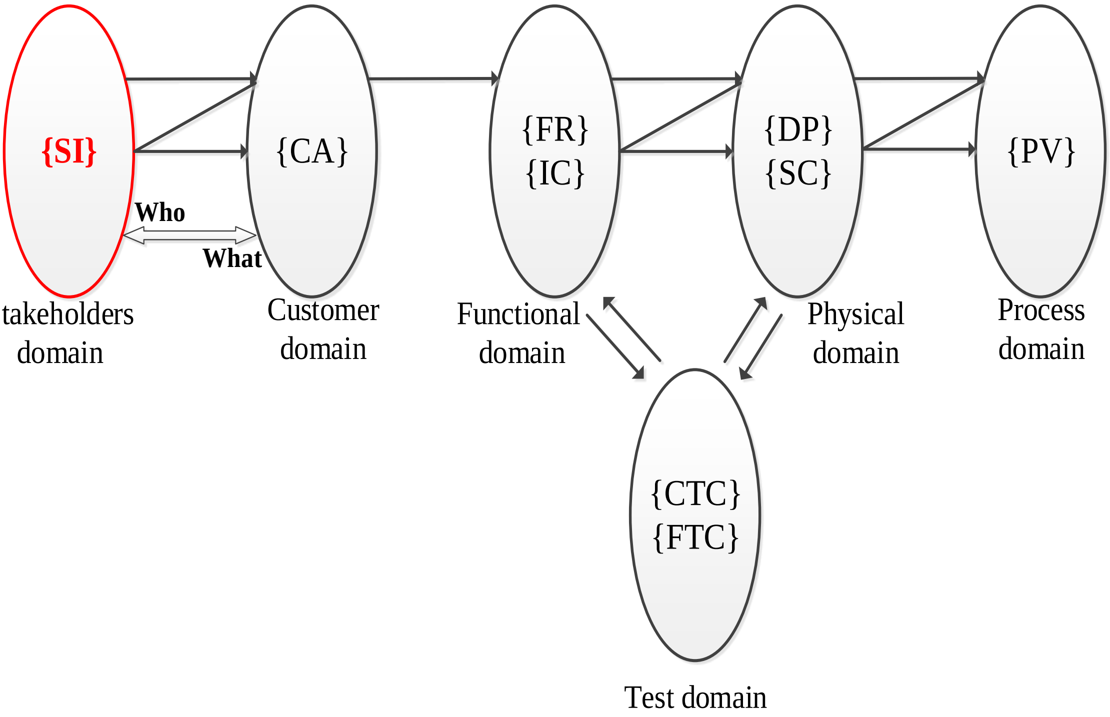


Fig. 6 TQSDL Domains

5. Transdisciplinary Quality System Development Lifecycle Model Domains and Process Enhancement

5.1 Enhancement of TQSDL Model Domains

Based on ever increasing pressures on companies to improve the quality of their final systems and reducing both costs and time it takes to supply systems to the customer, the companies have to be good in all dimensions of the system development. There are three dimensions of the system development [2], technology dimension, demand dimension, and supply chain dimension. In our model both technology dimension and demand dimension are covered. Demand dimension is given in stakeholders' domain and customer domain. Technology dimension is presented in the functional domain and physical domain. For these reasons, a new domain (supply chain domain) will be added to the TQSDL model to cover the supply chain dimension as shown in Fig. 7.

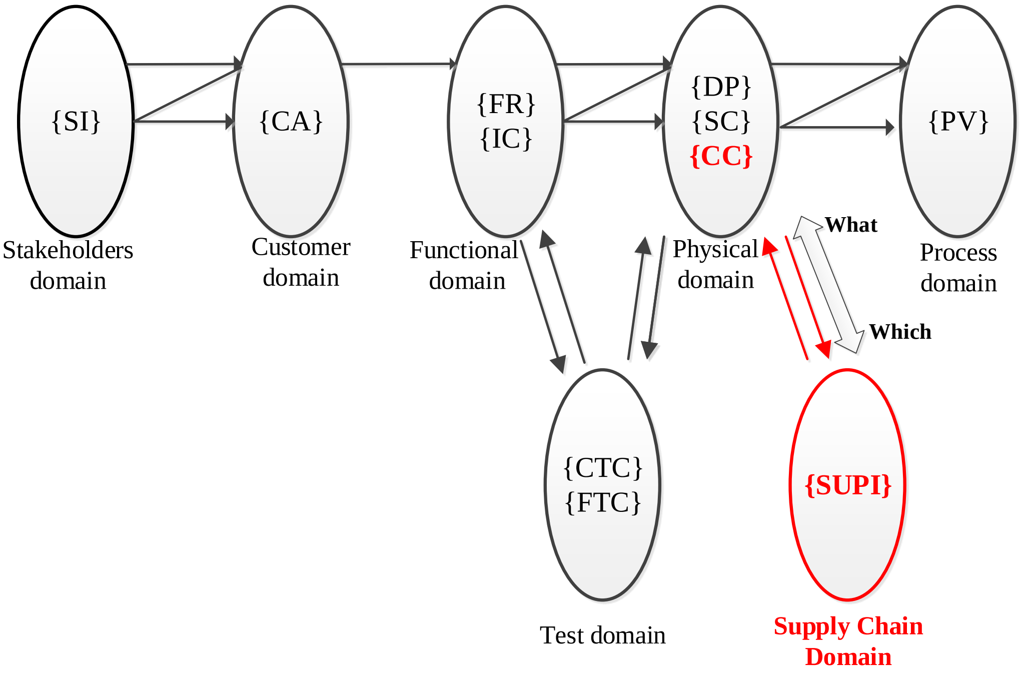


Fig. 7 Modified TQSDL Model Domains

The supply chain is often represented as a network which connects between supplier, manufacturer, distributor, retailer, customer, and service center. This network can be divided into three subnetworks: demand, supply, and service as shown in Fig. 8. The supply chain domain is focusing on the supply subnetwork (as shown in dashed domain in Fig. 8) and its interaction with the physical domain as shown in Fig. 7. This new domain is characterized by the supplier identification vector {SUPI}. For each system component, all the possible supplier will be identified. The relationship between the physical domain and the supply chain domain requires a what-to-which rather than a what-to-how mapping. The mapping process between the physical domain and supply chain domain can be mathematically expressed as:

$$\{SC\} = [SCA]\{SUPI\} \qquad (11)$$

The first axiom isn't applicable for the previous equation because it is possible that multiple system components will be supplied by one supplier as well as several suppliers can supply the same system component. In the case of there are many suppliers can supply one system component, the analytical hierarchy process (AHP) can be used to select the best one based on multiple attributes. A new theorem is created to supplement the axiomatic design and TQSDL model theorems. This theorem is used to control the process of mapping and zigzagging between the physical domain and supply chain domain.

- **New theorem (Allocate system components vectors {SCs} to supplier identification vector {SUPI}):** "The system components vector {SC} is allocated to supplier identification vector {SUPI} without restricting to independence axiom. Multiple system components mapping to one supplier or Several suppliers allocate to one system component."

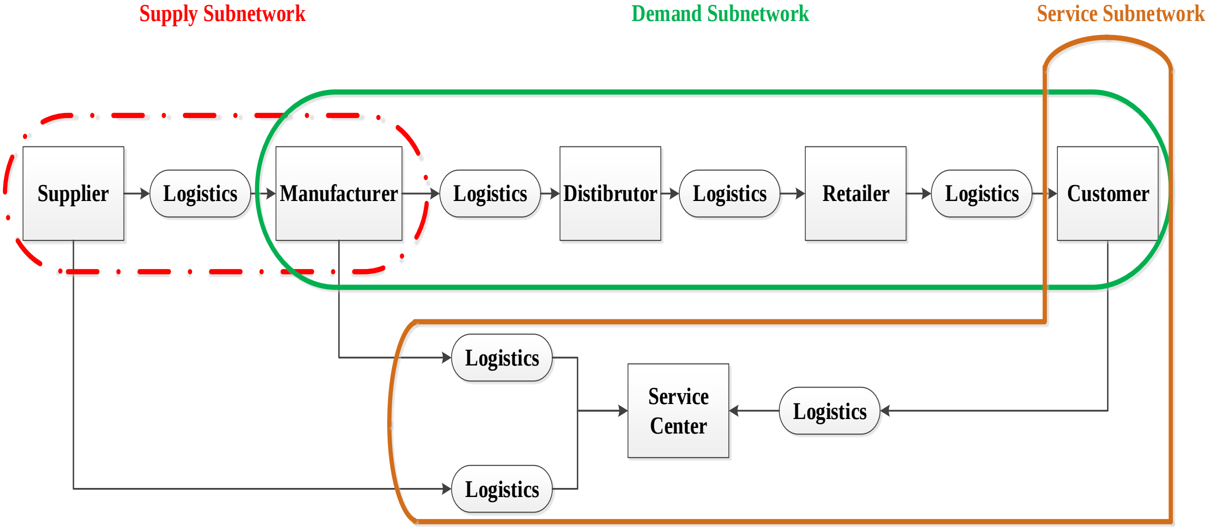


Fig. 8 Supply Chain Networks

5.2 Enhancement of TQSDL Process

A roadmap of the TQSDL model process is presented in a V-shaped process as shown in Fig. 9. The process starts with stakeholders identification and the output will be recorded in the stakeholders' register which contains at least, name, organization, location, role in the project and contact method [15]. Errors in gathering or predicting the customer needs can lead to incorrect system or to complete project failure. Therefore, the mapping between stakeholders and customer needs are performed by QFD to provide a visual relationship between each stakeholder and each customer needs, and to ensure that all stakeholders' needs are gathered.

From analyzing the QFD, it ensures that each stakeholder (each row in QFD chart) has at least one need. If this does not occur, check the relationship matrix again or run a new session with the stakeholder to get the missing needs. On the other hand, in the case of existing customer needs (each column in QFD chart) without a relationship with any stakeholder, it means that this customer need is not requested. The achievement of this extra need will lead to an increase in the total cost of the system. This extra need will be removed, or review the relationship matrix again to get the missing relation.

The weight of each stakeholder in the project will be determined by using the degree of importance. The importance will help to determine the effect on customer satisfaction in case of failure to implement one of the customer needs. The importance of rating the customer needs will be used to determine which CNs have priority and should receive the most attention to achieve customer satisfaction.

Usually, customers use vague expressions in defining their needs. Therefore, the mapping of customer needs to functional requirements will be divided into two stages. First, translating the customer needs into critical to quality (CTQ) characteristics, then translating the CTQs to FRs and ICs. These two stages will be performed by QFD.

CTQs are the translation of customer needs into quantitative terms. The main benefit of translating CNs to CTQs is to break down the CNs into identifiable and measurable terms. Moreover, these terms can be used later as input to the final acceptance test to determine the customer satisfaction degree. The importance rating of CTQs will be used to determine which CTQs have priority and should receive the most resources allocation.

In the second stage, each CTQs will be defined in terms of functionality in order to determine a set of FRs which can help to fulfill CTQs.
After the completion of mapping CNs to FRs and ICs, the top level of DP and SC should be proposed. Once FR/DP/SC top level is developed the design decomposition and zigzagging process will be started to get system architectures. During the decomposition process, the ICs will be allocated to the DPs. In the case of there are several DPs that could satisfy the FR and several SCs could be used to apply DP, a second axiom can be used as a quantitative measure of a given design, and thus, it is useful in selecting the best design from different designs alternatives that may be accepted with regard to the independence axiom [12]. At the end of this step, the system architecture will be finished. The PVs will be defined for each component in the system component hierarchy.

The process of cost estimation in the industry are performed using either parametric or roll-up technique [16]. In TQSDL model, a new characteristic vector ({CC}) will be added to the physical domain to get the cost breakdown structure of the system components. Components cost {CC} will be used to estimate the cost of each DP by mapping the DPC to system components cost breakdown structure. The mathematical relationship between the cost of each DP and the CC can be written as:

$$\{DPC\} = [SS]\{CC\} \tag{12}$$

System structure matrix [SS] is used in cost equation because it represents the mapping between DPs and SCs.

The TQSDL process isn't prepared to address the interrelation between development tasks and the interactions between system components. Therefore, DSM will be integrated into the TQSDL process to overcome these weak points in the model. Moreover, DSM will help to predict the change in development cost due to change in CNs or FRs. To get the interrelation between tasks, we will use a technique to obtain DSM from design matrix (DM) [17]. This technique is proposed to get activity-based DSM from DM at an early stage of the design process when the most important decisions about the system are made. To develop the activity-based DSM from DM, only FRs, and DPs are needed. The DSM which is developed from this technique is the real activities which are performed by the teams to design the systems based on the leaf level of the DPs. The DPs at the leaf level of the tree are the parameters that engineering actually work on during the design process. This technique will be explained in the next section.

System components hierarchy represent functional interaction only, while the system components interact with each other physically as well as functionally. For this reason, the component-based DSM will be developed. The component-based DSM represent the interaction between components in the system architecture. There are four types of interactions as shown in Table 3 [9], and will be described in detail in section 6 Table 6.

Table 3 System components interaction types

Interaction	Description
Spatial	Specifies needs for adjacency or orientation between two elements
Energy	Needs for energy transfer/exchange between two elements
Information	Specifies data or signal exchange between two elements
Material	Specifies materials exchange between two elements.

During the design decomposition and zigzagging process between FR/DP/SC to get system architectures, suppliers will be identified for each system components. The QFD tool may take various forms depending on the type of the problem to be solved [18]. In this model, the QFD tool will be used to manage and visualize the relationship between each system component and suppliers. This step is very important to cover the supply chain dimension in system development process [2]. From analyzing the QFD, it ensures that each system component has at least one supplier. If this doesn't occur, the designer must take into account to change this system component. The relative importance of each supplier will help to determine which supplier can deliver more than one components and which supplier deliver one system component. The correlation matrix will help to determine the correlation between a supplier in order to coordinate and integrate activities in the supply chain. This process will help the designer to coordinate its purchasing and production and create an integrated plan for the supply chain. In the case of there are more than one supplier can deliver the same system component, AHP can be used to choose which supplier will deliver this component based on, for example, time, cost, and quality attributes.

The component testing is performed to verify that the SCs successfully satisfies the FRs and ICs allocated to them by one or more of four alternative verification methods: analysis, inspection, demonstration or test. At least one CTC should be developed for each SCs.

The functional tests are developed to verify that the system satisfies the top level of the FRs. The FTCs should be developed to fully cover all the top level FRs. Moreover, during the execution of the final acceptance test, CTQs will be used to measure the degree of customer satisfaction and ensure that the final system achieves the customer needs.

By using the mapping matrices and the characteristic vectors, relationships between any entities from all domains as well as entities in the same domain can be easily identified. Therefore, it is easy to find out whether all customer needs and functional requirements are satisfied by the design parameters and system components. It is also possible to find out whether each and every functional requirement and system component is tested.

With this systematic approach, the technical manager can make the right decision and get the goals that were identified at the beginning of the development system within budget and time limits.

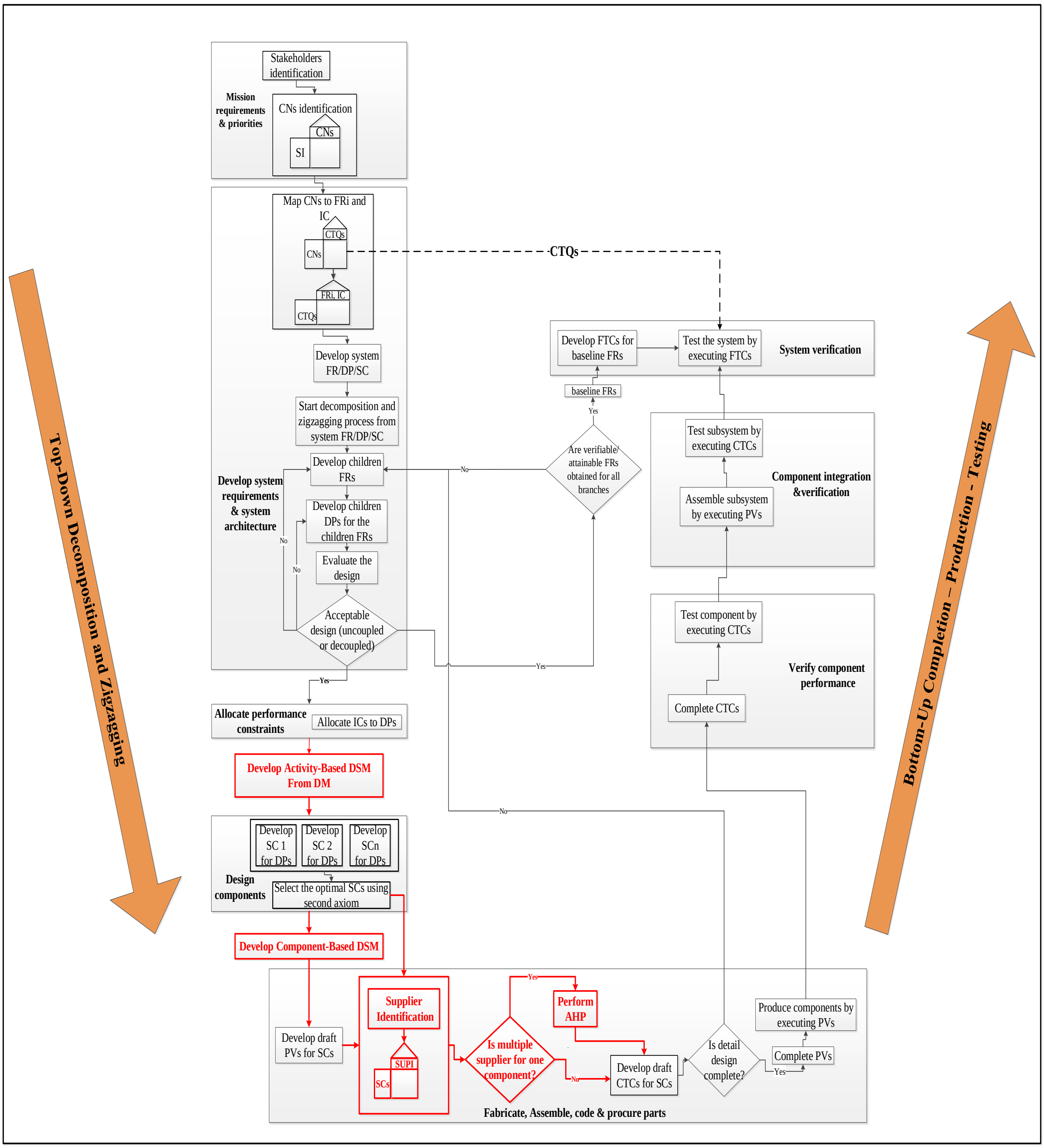

Stakeholders identification
Mission requirements & priorities
CNs identification
CNs
SI
Map CNs to FRi and IC
CTQs
CNs
FRi, IC
CTQs
CTQs
Develop system FR/DP/SC
Start decomposition and zigzagging process from system FR/DP/SC
Develop children FRs
Develop children DPs for the children FRs
Evaluate the design
Acceptable design (uncoupled or decoupled)
Develop system requirements & system architecture
No
No
Yes
Allocate performance constraints
Allocate ICs to DPs
Develop Activity-Based DSM From DM
Develop SC 1 for DPs
Develop SC 2 for DPs
Develop SCn for DPs
Design components
Select the optimal SCs using second axiom
Develop Component-Based DSM
Develop draft PVs for SCs
Supplier Identification
SUPI
SCs
Is multiple supplier for one component?
Yes
No
Perform AHP
Develop draft CTCs for SCs
Is detail design complete?
Yes
Complete PVs
Produce components by executing PVs
Fabricate, Assemble, code & procure parts
Develop FTCs for baseline FRs
Test the system by executing FTCs
System verification
baseline FRs
Yes
Are verifiable/ attainable FRs obtained for all branches
No
Yes
No
Test subsystem by executing CTCs
Component integration &verification
Assemble subsystem by executing PVs
Test component by executing CTCs
Verify component performance
Complete CTCs
Top-Down Decomposition and Zigzagging
Bottom-Up Completion – Production - Testing

Fig. 9 Modified TQSDL Process

6. Estimating Change in Development Cost Based on TQSDL Process

Because customer needs often change during the system development, the FRs and DPs will be also changed. For the system development to be a success, it is vitally important for designers to understand the impact of changing on the lifecycle cost [19]. Life-cycle cost is the total cost of a system in each phase of its existence: development, production, and operation. One of the largest problems in developing complex systems is estimating the change in development cost due to change in CNs or FRs. The development cost is equal to the summation of material cost and labor cost. To address the change in development cost, the component that would be affected and the reworks to develop the new components must be identified. In the late of 2004, Jeziorek proposed a method to estimate the change in development cost based on FRs-DPs mapping [20]. The weak points of his method are as follows. First, his method based on the axiomatic design process, and in AD system components, functional test and component test aren't identified. Second, he proposed that to alleviate the workload to define all activities that are needed to develop each component, the components will be divided into groups within each group common activities are assumed to apply. This is rather a reductionism's view. This proposal isn't real because each component has own design activities and tests.

The TQSDL process provides a good framework to estimate the change in development cost. In our proposed method we will use the traceability feature of TQSDL process, SCs hierarchy, components cost breakdown structure (CC), component-based DSM, and activity-based DSM to estimate the change in development cost. Because this information has been already produced during the execution of TQSDL process, the time needed to estimate the change in development cost will be saved. Moreover, The activity-based DSM which is developed from DM is the real activities which are performed by the teams to design the system components. This matrix is developed at an early stage of the design process when the most important decisions about the system are made. Also, the mapping and zigzagging process between physical domain and test domain present the real tests which are performed on SCs and FRs.Consequently, it will support the decision makers on whether or not to implement changes to a design. In the rest of this section, an example will be given.

A design change is usually implemented by a designer either because the customer needs or functional requirements directly have been changed. DPs are then modified to meet the changed FRs. Firstly, we have to identify the list of the component that will be affected due to change in the CNs or FRs as well as the related component test or functional test. The relationship between FRs and DPs, as seen in Table 4, will be presented by design matrix (equation 2).

Table 4 Design Matrix (FR-DP Relationship)

	DP1	DP2	DP3	DP4	DP5	DP6	DP7	DP8	DP9	DP10
FR1	X									
FR2		X			X				X	
FR3			X							
FR4		X		X				X		
FR5			X		X				X	
FR6		X	X			X				
FR7		X	X	X			X			
FR8	X				X	X		X		
FR9			X						X	
FR10					X	X	X		X	X

Assume that FR1 will have to change in order to satisfy the change in customer need. Consequently, DP1 will have to change also. Because DP1also affects FR8, DP8 will have to be changed in order to compensate for the change in DP1 and still satisfy FR8. The result of the change to FR1 is that DP1 and DP8 will have to change.

Table 5 DPs-SCs Relationship

	SC1	SC2	SC3	SC4	SC5
DP1	X				X
DP2	X	X		X	X
DP3			X		
DP4		X	X		
DP5	X	X		X	X
DP6			X		
DP7			X		X
DP8					X
DP9		X	X		
DP10	X	X			

From the system structure matrix (equation 7), we can find the system components that will be affected by the changes. From [SS] matrix, SC1 and SC5 will be affected by the changes to DP1 and DP8 as shown in Table 5. Thus the output from the SS matrix is a list of components affected by the functional changes. In order to capture the physical interactions between SCs, component-based DSM will be used as described in section 5.2.

Table 6 Component-Based DSM

	SC1	SC2	SC3	SC4	SC5
SC1	X				
SC2		X			
SC3			X		
SC4	X	X		X	
SC5	X				X

By using component-based DSM, the list of affected SCs from a spatial, information, energy, and material interfaces can be captured as shown in Table 6. Suppose that there is information interface between SC1 and SC4, and energy interface between SC1 and SC5. Also, Suppose there is the spatial interface between SC2 and SC4 as shown in Table 6.The input to component-based DSM is the SC1 and SC5, and the output is SC1, SC4, and SC5. From SS matrix, the list of affected DPs due to the changes in SCs can be listed. The input to SS matrix is SC1, SC4, and SC5, and the output is DP1, DP2, DP5, DP7, DP8, and DP10. Moreover, by using component test matrix (equation 10) and functional test matrix (equation 9), the list of affected component test and the functional test can be detected. In the end of this step, the complete list of SCs, CTCs, and FTCs have been identified.

Secondly, the amount of rework required for each system components will be calculated to measure the impact of a change on the development time. Then, the cost of labor due to change can be determined. Activity-based DSM will be used to calculate the amount of rework for each activity.

There are two methods to get the activity-based DSM are explained as follows. Firstly, obtain the activity-based DSM through interviewing the designers and managers that are involved in the system development, as well as reading the design documentation. This method works well when the system development process in detail design phase. But, in our case, we need the activity-based DSM at an early stage of the design process when the most important decisions about the system are made [17].

Secondly, obtain activity-based DSM from DM. This technique is used to create activity-based DSM at an early stage of the design process. The technique consists of three main steps [11], [17], [21], [22], :

a. Construct a design matrix (DM), equation 2.
b. Choose the output variables in each row of DM. it was demonstrated that the only valid output variables choices are the elements on the diagonal of DM [17].
c. Permute the matrix by exchanging row so that all the output variables are on the diagonal. Rename each row according to the names of the columns.

The motivation of transferring a DM into a DSM comes from linear algebra, how to solve a the system of linear equations using substitution. Now, the activity-based DSM can be determined as shown in Table 7.

Table 7 Activity-Based DSM

	DP1	DP2	DP3	DP4	DP5	DP6	DP7	DP8	DP9	DP10
DP1	X									
DP2		X			X				X	
DP3			X							
DP4		X		X				X		
DP5			X		X				X	
DP6		X	X			X				
DP7		X	X	X			X			
DP8	X				X	X		X		
DP9			X						X	
DP10					X	X	X		X	X

Partitioning of the activity-based DSM is needed such that the new activity-based DSM arrangement doesn't contain any feedback marks, thus transforming the activity-based DSM into a lower triangle form [7]. this action is performed by reordering of the matrix rows and columns. In the case of the difficult to eliminate all the feedback marks, the designer's objective changes to moving the feedback marks as close as possible to the diagonal. The value of this process is providing smoother information flow where all requisites information for an activity is available before the activity start as shown in Table 8.

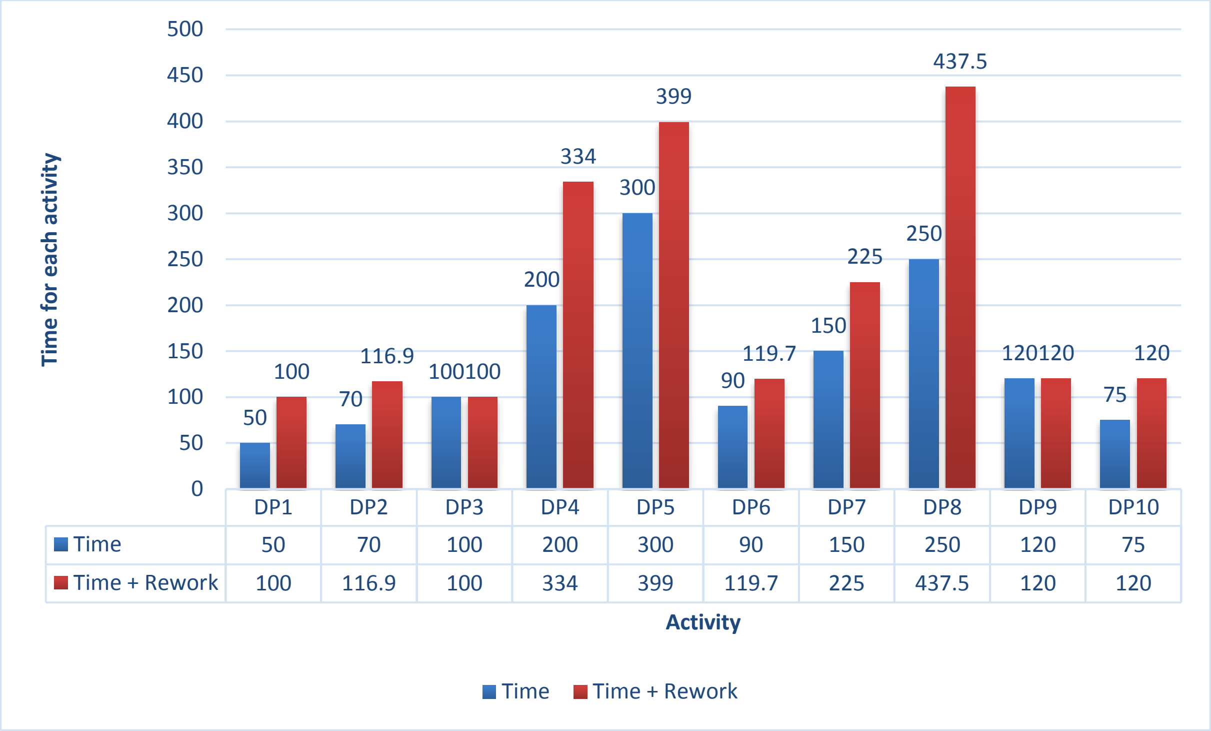
Table 8 Partitioning Activity-Based DSM

	DP1	DP3	DP9	DP5	DP2	DP6	DP8	DP4	DP7	DP10	**Rework**	
DP1	X										**1/1**	**100%**
DP3		X									**0/1**	**0%**
DP9		X	X								**0/2**	**0%**
DP5		X	X	X							**1/3**	**33%**
DP2			X	X	X						**2/3**	**67%**
DP6		X			X	X					**1/3**	**33%**
DP8	X			X		X	X				**3/4**	**75%**
DP4					X		X	X			**2/3**	**67%**
DP7		X			X			X	X		**2/4**	**50%**
DP10			X	X		X			X	X	**3/5**	**60%**

The input to activity-based DSM is the affected DPs due to the change in SCs. All the X's in the column of the affected DPs are marked as necessary to rework as shown in Table 8. The rework percentage will be added to the total development time for each activity to get the total time of each activity. Then, the total time will be multiplied by the cost per hour of each activity to get the total labor cost. The calculation of rework percentage can be written as (assume all interfaces activities in each activity are equal in time):

$$Rework\ for\ each\ activity = \frac{number\ of\ affected\ interfaces\ activities}{Total\ number\ of\ interfaces\ activites} \qquad (13)$$

Finally, the total change in development cost equal to the summation in the component change cost, test change cost and labor change cost. Based on this data, the decision makers can give a decision on whether or not to implement the changes to the design. The development time before functional requirement change and after the change are plotted versus the activities in Fig. 10. This figure illustrates that DP3 and DP9 activities haven't been affected, and DP1 activity has been completely affected by the change in the functional requirement FR1.



	DP1	DP2	DP3	DP4	DP5	DP6	DP7	DP8	DP9	DP10
🟦 Time	50	70	100	200	300	90	150	250	120	75
🟥 Time + Rework	100	116.9	100	334	399	119.7	225	437.5	120	120

Fig. 10 Activity Development time

7. Conclusion

The successful management of new system development projects is an important source of gaining competitive advantage. In the present work, the TQSDL model provides a robust structure and systematic approach based on a scientific basis to cover all the system development dimension (technology dimension, demand dimension, and supply chain dimension). The TQSDL model uses in design and development of the system, product, organization, and software.

TQSDL model provides an easy-to-follow process to perform requirement management, process management, change management, information management and logistic support. Consequently, Successful implementation of the TQSDL model will assist the systems engineering manager for implementing the intended roles in a more effectively manner and develop the complex systems that achieve customer satisfaction.

The main difference between the previous TQSDL model which is proposed by us and the enhanced TQSDL model are as follows:

a. The enhanced TQSDL model has a supply chain domain with SUPI characteristic vector.
b. The enhanced TQSDL model has a new theorem that is used to control the process of mapping and zigzagging between the physical domain and supply chain domain (Allocate system components vectors {SCs} to supplier identification vector {SUPI}).
c. The enhanced TQSDL model has the CC characteristic vector in the physical domain.
d. The enhanced TQSDL model has a supplier allocation matrix for allocating SCs to SUPI.
e. The enhanced TQSDL model has the CC hierarchy (cost breakdown structure for components).
f. The enhanced TQSDL process has the activity-based DSM.
g. The enhanced TQSDL process has the component-based DSM.

h. The enhanced TQSDL process has QFD to mapping system components to the suppliers.

The TQSDL process inherently documents full traceability of customer needs to constraints and functional requirements, to decomposition solution, into solution domain, and then into system components. This enables the managers to ensure that the customer satisfaction is achieved. Moreover, helps the designer to analysis the impact of customer need changes (change management) on the system. The benefits of Integrating DSM into TQSDL process are as follows:

a. Capture the physical interaction of system components.
b. Obtain the activity interrelation in the early stage of the design process when the most important decisions about the system are made.
c. The technique used to obtain DSM from DM enables us to use DSM as a system analysis tool and a process management tool.

A TQSDL model framework provides a structure to cost engineer to provide a quick estimation of the change in development cost. Moreover, it will also support the decision makers on whether or not to implement changes to a design.

8. References

[1] M. A. Abdelrazek, A. T. El-sheikh, M. Zayan, and A. M. Elhady, “Systems Engineering Management using Transdisciplinary Quality System Development Lifecycle Model,” *Int. J. Mech. Aerospace, Ind. Mechatron. Manuf. Eng.*, vol. 11, no. 1, pp. 115–125, 2017.

[2] M. Sahlin, “A Decision Framework for Integrated Synchronized Development of High Tech Product,” *ICAD2002*, pp. 1–7, 2002.

[3] D. Maritan, *Practical Manual of Quality Function Deployment*. 2015.

[4] G. Tontini, “Integrating the Kano Model and QFD for Designing New Products,” *Total Qual. Manag. Bus. Excell.*, vol. 18, no. 6, pp. 599–612, 2007.

[5] F. Pakdil, F. B. Işın, and H. Genç, “A Quality Function Deployment Application using Qualitative and Quantitative Analysis in After Sales Services,” *Total Qual. Manag.*, vol. 23, no. 12, pp. 1397–1411, 2012.

[6] L.-K. Chan and M.-L. Wu, “A systematic approach to quality function deployment with a full illustrative example,” *Omega*, vol. 33, pp. 119–139, 2005.

[7] “descripton-design-structre @ www.dsmweb.org.” .

[8] T. R. Browning, “Applying the design structure matrix to system decomposition and\nintegration problems: a review and new directions,” *IEEE Trans. Eng. Manag.*, vol. 48, no. 3, pp. 292–306, 2001.

[9] A. Yassine, “An Introduction to Modeling and Analyzing Complex Product Development Processes Using the Design Structure Matrix (DSM) Method,” *Urbana*, pp. 1–17, 2004.

[10] E.-P. Hong and G.-J. Park, “Decomposition Process of Engineering Systems Using Axiomatic Design and Design Structure Matrix,” *Proc. ICAD2009*, 2009.

[11] M. D. Guenov and S. G. Barker, “Requirements-Driven Design Decompostion: a Method for Exploring Complex System Architecture,” *ASME Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf.*, pp. 1–7, 2004.

[12] N. P. Suh, *Axiomatic Design: Advances and Applications*. 2001.

[13] S. Lo and M. G. Helander, “Use of axiomatic design principles for analysing the complexity of human–machine systems,” *Theor. Issues Ergon. Sci.*, vol. 8, no. 2, pp. 147–169, 2007.

[14] B. Gumus, A. Ertas, D. Tate, and I. Cicek, “The Transdisciplinary Product Development Lifecycle model,” *J. Eng. Des.*, vol. 19, no. 3, pp. 185–200, 2008.

[15] Project Management Institute, *A guide to the project management body of knowledge (PMBOK ® guide)*. 2013.

[16] T. R. Browning and S. D. Eppinger, "Modeling impacts of process architecture on cost and schedule risk in product development," *IEEE Trans. Eng. Manag.*, vol. 49, no. 4, pp. 428–442, 2002.

[17] Q. Dong and D. E. Whitney, "Designing a Requirement Driven Product Development Process," *Proc. DETC 2001 ASME 2001 Int. Des. Eng. Tech. Conf.*, 2001.

[18] K. Prasad and S. Chakraborty, "A quality function deployment-based model for materials selection," *Mater. Des.*, vol. 49, pp. 525–535, 2013.

[19] J. Hintersteiner, "Addressing changing customer needs by adapting design requirements," *Int. Conf. Axiomat. Des.*, pp. 290–299, 2000.

[20] T. Lee and P. Jeziorek, "an Exploratory Study of Cost Engineering in Axiomatic Design: Creation of the Cost Model Based on an FR-DP Map," *Proceeding ICAD2004 Third Int. Conf. Axiomat. Des.*, pp. 1–7, 2004.

[21] D. Tang, G. Zhang, and S. Dai, "Design as integration of axiomatic design and design structure matrix," *Robot. Comput. Integr. Manuf.*, vol. 25, no. 3, pp. 610–619, Jun. 2009.

[22] X. Cheng and C. Chen, "Applying independence axiom and design structure matrix to product module division," *2010 Int. Conf. Mech. Autom. Control Eng. MACE2010*, pp. 581–584, 2010.