Proceedings of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2010
August 15-18, 2010, Montreal, Quebec, Canada

DETC2010-28821

COMBINING USER WORKFLOW AND SYSTEM FUNCTIONS IN PRODUCT DEVELOPMENT

Thom J. van Beek[1]
Delft University of Technology
Intelligent Mechanical Systems group
Delft, The Netherlands
T.J.vanBeek@TUDelft.nl

Tetsuo Tomiyama
Delft University of Technology
Intelligent Mechanical Systems group
Delft, The Netherlands
T.Tomiyama@TUDelft.nl

Abstract

This research addresses two aspects of multi disciplinary product development; understanding what the customer needs and translating this knowledge effectively into a transparent system architecture. A clear overview and understanding of this architecture can be obtained by using function models. How to make an effective translation from the customer needs to function models is not trivial. The approach taken in this research is to discuss, develop, negotiate, align and model the intended users' workflow of the system. Multiple stakeholders (e.g. buyer, user, marketers, application experts, architects, designers and third party suppliers) should participate in this process. The user workflow is systematically mapped to a system function model. The contributions of this paper can be divided in two. First a systematic and easily applicable design method to capture and validate user requirements in a multi disciplinary design process using explicit workflow models is proposed. Secondly a systematic method of translating user workflow models to system function models will be given.

INTRODUCTION

For a successful product development process it is essential to have a clear understanding of what the customer needs and then translate that into a successful product [1]. A product is successful if it does what the customer wants it to do. What the product does is determined by the system architecture. The system architecture is defined here as a mapping of *what* the system should do (functions) to *how* it should do that (components). A clear mapping of customer requirements to system requirements is therefore an essential input to the system architecting process and eventually, a successful product.

Industrial design engineers and software engineers have been educated with a more central role for the customer and user in their product development process compared to mechanical engineers. In the mechanical engineering domain, general practice is to capture the top-level customer requirements once, at the beginning of product development , and then focus at the technology for the remainder of the process. The user often is not properly given the opportunity to express or develop his or her needs and is not really involved in the design process. Therefore user-centered-design needs to find its way to the mechanical engineering level of product development.

In literature, approaches can be found that focus on the customer requirements side or on the system architecting side. The Function and Key drivers method (FunKey) [2] method deals with relating system functions to key drivers and requirements in a matrix. This method gives a good approach to relating key performance drivers to architecture. This provides the system architect with an overview of how his decisions can be traced back to the architecture. The focus in Bonnema's work lies with the architect. This paper proposes to take the external view to the customer explicitly into the equation of systems architecting.

The CAFCR model presented by Muller [3] does propose a decomposition of the architecture into five main views that capture the need of the customer, the functions, the product performance and the design of the product from the conceptual

[1] Address all correspondence to this author.

and realization points of view. The contribution of Muller's work lies in presenting a method to create different architectural views including the one of the customer. However, Muller does not specify how to implement these methods in a structured way that resembles the hierarchical and iterative nature of the product development process.

Research on scenario based design gives us valuable methods for including the externally focused view at the user from a product development standpoint. It includes the user in the product development process by modeling his or her behavior in the form of stories. In [4] a scenario is defined as: '*Explicit descriptions of the hypothetical use of a product'*. The scenarios are analyzed and used to influence the design activity. Scenarios appear in the form of textual stories or storyboard like representations very understandable to humans. Scenarios have strength in describing the use aspects, but usually do not provide a structured and decomposable way of translating them into system architecture.

A use-case in UML [5] is an example of a scenario based technique that tries to connect to system architecture. The level of detail in use-cases found in literature is however not high. Furthermore, use-case modeling is quite software specific and does not seem to be applicable to product development processes including other disciplines like mechanics, physics and electronics.

There is a need for a design method that incorporates both the view of customer requirements and the view of translating these requirements into a system architecture. The needed design method should; fit the hierarchical and decomposable nature of the product development process, be easy to understand for all stakeholders and be applicable in a real industrial setting.

The main contributions of this paper can be divided in two. First a systematic, practically applicable and integral design method to capture and validate user requirements in a multi disciplinary design process using explicit workflow models is proposed. Secondly a systematic method of translating user workflow models to system function models will be given.

This paper will proceed by giving some related work found in literature that is used as a foundation of the presented research. The section that follows explains both the design method and the proposed models step-by-step taking the product development cycle of the V-model as a reference. Experiences with applying the models in a real case study with our industrial partner are given to illustrate the method. Finally some discussions and conclusions will be presented to the reader.

RELATED WORK

The idea of using workflow models to capture user requirements in this research was inspired by the field of business process modeling. In business process modeling workflow management and workflow management systems are used to make sure that the proper activities are executed by the right resource at the right time [6]. Although a detailed literature review on business process modeling is not in the scope of this paper, a description of the modeling concepts adopted from workflow modeling will be given in this section. This will be used as reference for later sections.

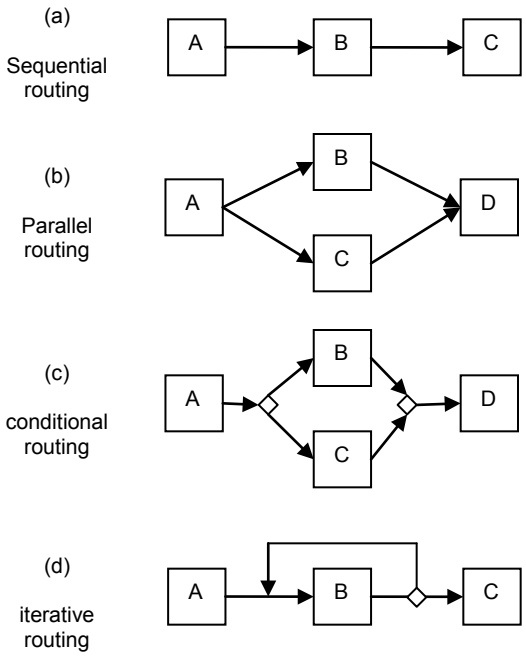


FIGURE 1. FOUR WORKFLOW ROUTING CONSTRUCTS BY VAN DER AALST [6]

Workflow

From the work of Van der Aalst [6] the following valuable definitions and descriptions of a workflow process description are adopted. A workflow is *case* based. A *case* is the actual work that is being handled by executing *tasks* in a specific order. The *workflow process definition* specifies which tasks need to be executed in what order. This is what is commonly known as a flowchart. Every task has *pre-* and *post-conditions*. Conditions correspond to causal dependencies between tasks and can be true or false. The pre-condition is a prerequisite for the task to be executed and the post-condition holds true after the task is executed. Most tasks are executed by a *resource.* A resource is either a machine or a person or both. In this paper the products' user corresponds to a resource for the system workflow for example. Multiple resources per workflow can occur. A task that needs to be executed on a specific case is called a *work item.* A work item being executed by a resource is called an *activity.* An example of an activity can be: execute the task of 'send bill to customer' for the case of 'product sold to customer Jack' by resource 'administrator Claire'.

Four common patterns in workflow modeling are depicted in fig 1. These patterns can be used to construct the workflow

processes. Figure 1 shows sequential (a), parallel (b), conditional (c) and iterative (d) workflow routings that are most commonly used in workflow modeling.

Although out of scope for this paper, workflow modeling is computerizable. By formalizing the workflow it can be simulated and analyzed. (High level) Petri nets are a common way of modeling workflows in a computer understandable language. Tasks map to Petri net *transitions*, conditions to *places* and cases to *tokens*. Many types of analyses are available and can be classified into verification, validation and performance analyses.

The *Workflow Management Coalition* (WMC) defines a workflow management system as [7]: *A system that completely defines, manages and executes workflows through the execution of software whose order of execution is driven by a computer representation of the workflow logic*. The WMC focuses on software execution of workflows. Examples of executable workflows are found in many places nowadays, e.g. handling tax declarations, online order requests, web shop shopping carts, creation of online mail account, paycheck management. All these processes are highly automated on one hand, but heavily depend on human interactions on the other.

In this research multi disciplinary products (e.g. printer copiers, transportation systems, mobile devices and medical systems) are considered incorporating disciplines like mechanics, physics, electronics and not just software. Defining, managing and executing workflows for these products cannot solely be based on the execution of software. What the targeted systems do have in common with business processes is that proper functioning heavily depends on interaction with their users. Workflow modeling is a good way of modeling user interaction with a system. Therefore just the modeling part of workflow management systems will be adopted in this research. The connection to an application executing the workflow will be different for a tangible system.

Systems engineering process

For this research the systems engineering V-model [8] was chosen as a reference diagram on which to map the proposed models and method. The V-Model (fig 2) describes the systems engineering process as a letter V. The diagram should be read left to right starting at the top-left leg of the V with user requirements and moving down, going in depth and creating a system architecture and finally, more detailed design. The bottom part of the V represents the production and assembly phase. Moving up the right leg represents component level testing, systems integration tests and system launch respectively. Validation of the product design is done by comparing the realized system at system launch to the user requirements modeled at the start of the product development process. Modeling user requirements and translating them into system architecture resides in the upper and middle left part of the V model. It is however not a one-way process, iterations

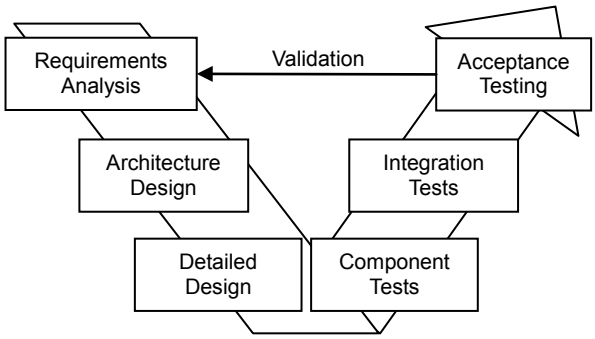


FIGURE 2. V-MODEL OF SYSTEMS ENGINEERING

between all these phases of systems engineering are important as stressed in many design methodologies (e.g. [1, 9]).

Systems architecting can be defined as the mapping of system functions to system components. Functions in system architecting being the 'what the system should do' and components being the 'how the system should do that'. The field of Function Modeling (FM) covers the mapping of functions to components.

Function – Behavior – State modeling

This paper adopts the Function-Behavior-State (FBS) modeling technique presented in [10, 11]. The advantage over other FM techniques is that it associates the functional descriptions with structural elements via the behavioral layer [12]. This paper only uses the concepts of *function, behavior, state* and *entity* and their relations as they are defined in the FBS framework:

- *Function*: A subjective description of behaviors of physical systems. Functions can be considered as a bridge between human intention and physical behavior of artifacts. Functions can be hierarchically decomposed into lower level sub-functions. Functions can be defined as 'to do something'. In FBS models function nodes take the form *function body (objective entities)*. The verb being the function body, the objective the entity affected by the function and through the function-behavior relation, a function carrier entity is connected that realizes the function. E.g. 'to move (puck) with a stick '.
- *Behavior*: A behavior is an objective category defined by sequential changes of states of a physical structure over time. This change of states has an influence on the environment and it is perceived as the impact of the behavior.
- *State*: States are the different modes of a physical system or entity. Changes of these modes, state transitions, are the underlying cause of behaviors.
- *Entity* (Component): An atomic physical object that has different states, hence the capability to generate behavior.

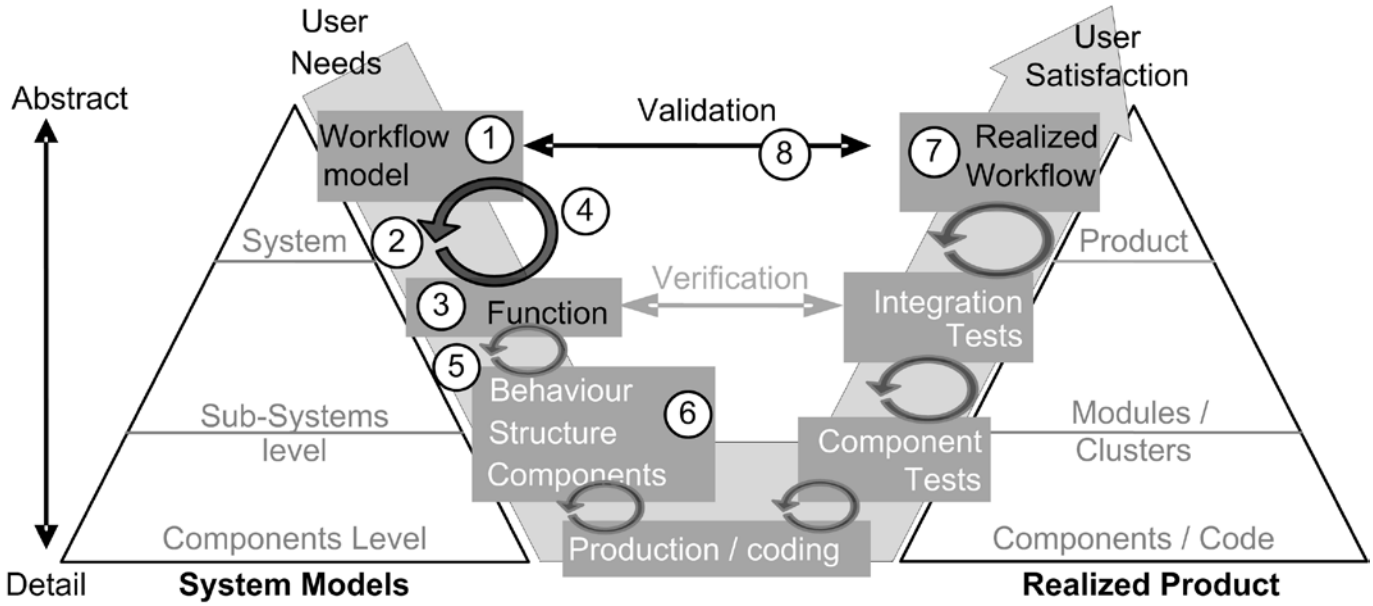


FIGURE 3 WORKFLOW MODELING METHOD MAPPED ON THE PRODUCT DEVELOPMENT PROCESS. SHOWING STEPS OF PRODUCT DEVELOPMENT PROCESS IN PARALLEL WITH SYSTEM COMPLEXITY

More detailed descriptions of relations among the FBS concepts are discussed can be found here [11, 13].

To create a FBS model, FBS literature generally proposes to start with a functional decomposition of the system, connecting the leaves of the function tree to a network of behaviors and states of entities concurrently. Besides paper models, the FBS modeler tool was developed [13, 14]. The models in this research were not put in the FBS modeler, but do use the FBS framework as a basis.

METHOD DESCRIPTION

The method proposed in this paper considers models, and the process of applying these models. First the steps in applying the models will be discussed. The proposed method is mapped onto the systems engineering V-model in fig 3.

Process

Depending on the situation in a product development environment, different stakeholders can be identified. For the purpose of this paper the following stakeholders are identified as important during the implementation of the product development process:

- *Customer*
  - *User*. The person or people that use the system or interact with the system.
  - *Buyer*. The person or organization that buys the system. Can be the same as the user.
- *3rd Parties*. Business and development partners involved in the product development process. Typically responsible for parts of the system.
- *Marketing / Application.* product development organizations' internal experts on user requirements.
- *System architects*. Developers responsible for creating system architecture during the product development process.
- *Design team* members. A team of people from multiple disciplines, both technical and operational, from within the product development organization.

Figure 3 shows a graphical representation of the method mapped onto a V-model illustrating the product development process combined with the pyramid by Muller [3] illustrating system complexity. The pyramid depicts the complexity of increasing amount of details that is created as the system is being developed. The grayed out items are part of the global method, but are only briefly discussed here since they are outside the focus of the contribution of this paper.

Step 1: Creating Workflow Models

Together with the System Architect, the marketing and application experts manually create a high level of abstraction workflow model based on recurring interviews, presentations for and meetings with the users.

Depending on the specific design task, more than one user workflow model might be needed (e.g. system set up, system operation and cleaning of the system). If related, the multiple

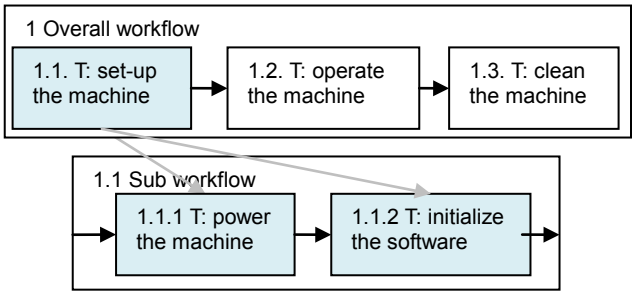


FIGURE 4 WORKFLOW DECOMPOSITION

workflows in their turn can be combined into a super workflow model, see fig 4. Going over the tasks in the workflow one-by-one, systematically checking for decomposition opportunities, guides a structured way of hierarchical workflow modeling.

The high level of abstraction models have only very few *task* nodes and describe the global workflow of the system being designed in a graphical manner. They focus on 'what' the system should do during normal use.

As the product development process proceeds, more detailed information on the system use becomes available from interaction (questions and answers) with the user. This information can be put in the models iteratively by decomposing the tasks further into sub-workflow models. This is illustrated in fig 4. The task 'Set-up the machine' is decomposed into a sub-workflow model containing two tasks, namely; 'power the machine' and 'initialize the software'. Manual task coloring is used to indicate decomposition relations among tasks. Tasks in different workflow model levels with similar colors are related by decomposition.

The hierarchical model created during this process facilitates discussions and communications between stakeholders on all levels of abstraction. High level of abstraction models can for example be used to inform new stakeholders, who intermediately join the project, on the status of design. Pencil annotations in detailed paper workflow descriptions facilitate design team efforts on covering all the small, but necessary user workflow steps.

Design team reviews of the workflow trigger updates and iterations of the models throughout the initial phase of the product development process. When the model becomes mature enough, a project milestone should freeze and consolidate the agreed workflow with all stakeholders. Having the user involved in this process prevents surprises in the final phases of the product development process.

At least two valuable types of analysis can be applied to the workflow model in our application. The first analysis is time duration of the total workflow. This is valuable design information. The time needed to complete the workflow is usually one of the parameters that determine if the user accepts the proposed workflow. Once estimated per task, the time durations have to be verified. Manual analysis is feasible for

TABLE 1. WORKFLOW TASK ATTRIBUTES

Attribute:	**Description:**
Label	Textual description of the task in human understandable language
Min. duration	Estimation of the minimum duration time of the task. This facilitates time performance analysis of the workflow.
Max. duration	Estimation of the maximum time duration of the task. The difference with the Min. duration gives the range.
Development Owner	Specifies which development stakeholder party is responsible for realization of the specific task in the total workflow.
Task ID	Unique identifier. See fig 4 for an example of hierarchical ID's for tasks.
Super workflow	ID of the super-workflow. If no super workflow exists, this task is the main task.
Predecessors	List of task ID's directly preceding this task and are connected by an arc. If no predecessors exist the specific task is a source.
Successors	List of task ID's directly succeeding this task and are connected by an arc. If no successors exist the specific task is a sink.
Function ID	ID of the connected function..
Resource	Textual description of the possible resource needed to trigger this function.

sequential workflows, but timed Petri net analyzers have standard functions to perform this operation more effectively for complex workflow constructs [6, 15].

The second analysis is applying the workflow model to the system as a recipe. The system architect should make sure that the system as he designs it facilitates executing the workflow. In step 2, system functions will be connected to workflow tasks to establish a relation between the recipe (tasks) and the execution entities (F-B-S). The recipe consists of a sequence of tasks that should be fired. Depending on the nature of the workflow, different sequences of firing can be possible. Take the conditional routing in fig 1 (c) for example; the sequence could be either (A,B,D) or (A,C,D). Both recipes should be supported by the FBS model of the system. Finding the sequences could be automated using Petri net analysis techniques like reachability and coverability analysis [6], but that is considered outside the scope of this paper.

Two types of graphical workflow node representations are used to accommodate the four workflow patterns in fig 1 in practice. Rectangles are used for normal tasks and diamonds for

decision points in the workflow. The modeled choice can be the users' to make or it may represent a 'control choice' of the system. In the second case the choice should be part of the system functions. The conditional routing construct as in fig 1(c) represents a choice. The other workflow patterns in fig 1 are supported by just using rectangles for workflow tasks.

All tasks have attributes. A list of attributes used in this research is given in table 1. All these attributes need to be determined manually at this moment. By filling in the attributes systematically for all tasks, valuable design data is captured and documented. The process of entering this data can however be quite labor intensive. If in future research the process described in this paper will be computerized, the list in table 1 will give a first concept of the attributes for the task object.

The output created in this step is:

- Common stakeholders understanding of what workflow will look like.
- Consolidated and user agreed hierarchical workflow model document

Step 2: Workflow – Function Development

In parallel to step 1, the workflow model is developed into a FBS model by the system architect. Step 2 covers the actual mapping of intended use of the system to system properties. Following the hierarchical workflow decomposition, the function tree is constructed. This is not a one-to-one mapping. Multiple tasks can be connected to the same function but with different parameter values, e.g. 'move table up' and 'move table down' could point at the same function 'to move table' but with different goal states, 'up' and 'down' respectively.

Systematically going through the workflow tasks, by following the sequence, all applicable tasks can be connected to a function. This connection will be called T-F. Each task should be connected to a leaf of the function tree. The leaves of the function tree are not decomposed further at that moment and are connected to realization behavior and states. To find out which function should be connected to the task the label attribute is analyzed. The following items can occur:

- Function body (required at least once)
- Objective
- Function carrier
- Post condition state
- Pre condition state

The more items out of this list can be found, the more precise the task-function relation can be described. An example task label is: '*put* the *moving puck in the goal* with the *stick*'. The attributes of the mapping would then be:

- Function body = *put*
- Objective = *puck*
- Function carrier = *stick*
- Post-condition = *in the goal*
- Pre-condition = *moving*

TABLE 2. T-F ATTRIBUTES

Attribute:	**Description:**
T-F ID	Unique identifier
Task ID	ID of the connected task
Function ID	ID of the connected function.
Function Body	Textual description of the function identified in the connected task. This should be a verb
Objective	Textual description of the entity that is affected by the function.
Function carrier	Textual description of the entity
Pre-condition	Textual description of the Pre-conditional state of this function.
Post-condition	Textual description of the goal state of objective after realizing this function.

Identifiers for both types of conditions are words like: *to, in, on,* etc. Another word found regularly in task labels is '*and'*. Especially in the high level workflows people tend to combine multiple task descriptions in one task item, e.g. 'swing stick AND put the moving puck in the goal with the stick'. When this is encountered, either the task can be divided into separate tasks, or the workflow task can be mapped onto more than one function.

In the process of connecting the workflow to the functions information the system architect will find out that information and knowledge about the system is not yet sufficient. Therefore he or she needs to synthesize. The result of the synthesis can then be added to the FBS description and thereby the T-F support the synthesis process.

Practically the workflow–function connection can be displayed as an arrow between the task and a function. The attributes of the arrow correspond to table 2. The complete set of all these mappings can be combined in a Domain Mapping Matrix (DMM) [16] mapping from the workflow to the functional domain. Through this connection the workflow time duration estimations can be propagated to the function model for future verification purposes. The same holds for the *Resource* attribute of the task defined earlier.

Step 3: Function modeling

In parallel to the workflow modeling of step 1, the system architect develops a FBS model of the system. In step 3 the functional part ('F' in 'FBS') of the FBS model is constructed. Input needed to create the model is:

- Workflow model and connections to the T-F functions
- system architect's and other stakeholders' expertise and competences in all disciplines
- Knowledge on and documentation about existing, related system architectures
- The framework of FBS and a basic understanding of FBS

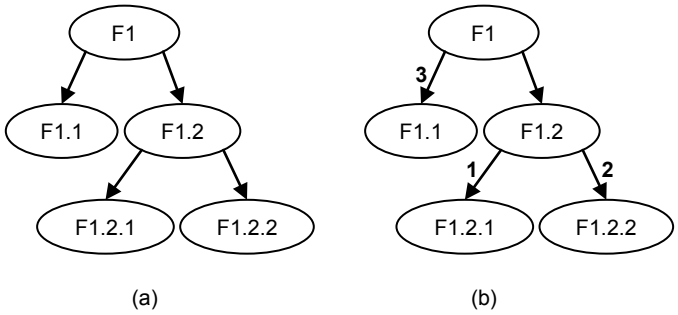


FIGURE 6. SCHEMATIC FUNCTION MODEL. A.) CONVENTIONAL FBS STYLE. B.) AUGMENTED WITH A ORDER FOR A GIVEN WORKFLOW RECIPE.

For a detailed description of developing the Function-to-Function (F-F) decomposition part of FBS modeling, the reader is referred to [13, 17]. This section will discuss how the T-F relations influence the F-F modeling process.

F-F modeling in FBS is the process of how to decompose functions into sub-functions and how to relate them. For this two categories of F-F relations are given in literature [13, 17], namely:

- Causal decomposition
- Task decomposition

Let f be decomposed into sub-functions *f1, f2, f3… fn* and *b1, b2, b3…bn* the behaviors that embody these functions. If *f1* is realized by *b1* and *f2* by *b2*. The definition is that causal decomposition of *f1* and *f2* occurs if the behaviors *b1* and *b2* are causally related. As can be seen from this, causal decomposition of functions originates from the behavioral layer of the FBS model (bottom-up) and can therefore be considered as feedback on the designers initial function model. Since the T-F relation is not concerned with behaviors, causal decomposition of functions cannot be supported by the T-F connection. Knowledge generated in the behavioral level about causal decomposition could however be valuable feedback from the function model to the workflow. See step 4.

Task decomposition is the process performed by the architect while making the function model (mostly top-down). In order to fulfill *f*, the sub-functions *f1* and *f2* are needed. The system architect uses his/her knowledge and competences to create this decomposition. This process can be supported by capturing the knowledge and agreements in the workflow model. Taking the workflow decomposition as the guide for the function decomposition is like checking a list.

The recipe created by the workflow contains sequence information. This information can be used to denote the sequence in which functions should occur. Visually displaying the sequence information in the function model increases the readability of the function model. In FBS such an ordered

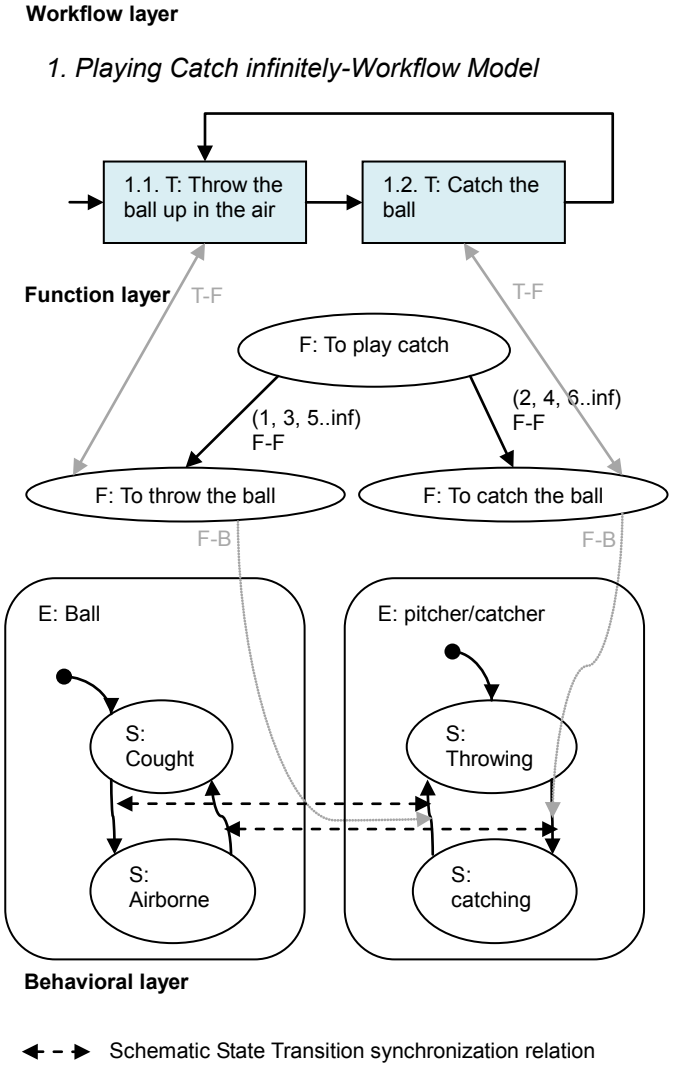


FIGURE 5 EXAMPLE OF WORKFLOW-FUNCTION MODELS INTEGRATED

sequence, and therefore time, notation is not available. The order of functions occurring can be displayed by annotating the function decomposition relations/arrows with numbers. The order of the sequence can be derived from the workflow sequence of tasks. Like already discussed, the sequence doesn´t give a causal ordering. It just feeds back the order of functions. See fig 5 for an example. In fig 5 the sequence of functions occurring is *F1.2.1, F1.2.2, F1.1* and *F1.2.2* again.

Step 4: Function modeling - Workflow feedback

Because of the other inputs for the function model mentioned in step 3 besides the workflow, it might happen that functions are introduced in the function model that require a separate task in the workflow. The generation of additional

functions in the function model could for instance be done by reasoning about behaviors as discussed in step 3. The system architect feeds this information back to the workflow models and updates them. This step closes the iteration cycle of the workflow – function connection.

Step 5: Function – Behavior/State relations

Realizing the function requires developing relations to behaviors and states of entities. In the function verb (objective) description the verb is the function body and the objective an object. The F-S relation was mentioned by Shimomura in [17] to denote this relation. Besides the objective there is also a function carrier that usually is an object. So the function; 'to fix the work piece by the clamp' has two F-S relations, namely; the relation between the function and the work piece and the function and the clamp. The F-B relation is well documented in literature [10] and out of the scope of this paper.

Step 6: Behavior/State model

FBS behavior and state part of the model is not the focus of this research. Mostly the FBS modeling was adopted, but it was adapted to explicitly show possible states of the entities involved. This was done by using a 'light' finite state representation. The models are not simulated at this moment. Just the graphical representation was mimicked.

The bottom part of fig 6 shows this representation using an example of a two function connected to entities in the behavioral layer to an entity with states.

Step 7: Creating realized workflow report

When the system development process has reached its final phase the system needs to be validated. A comparison between the workflow model developed in the product development process and the realized workflow will provide the necessary feedback. A report of the realized workflow is needed as a comparison counterpart for the workflow model. Many different forms and shapes of this report may be created, e.g. pictures, movies, verbal explanations, step-wise textual descriptions of the use.

An important requirement for this report is that the form chosen supports the step-wise or decomposed nature of the workflow models. This makes it easier to systematically, step-by-step, check if all items of the modeled workflow correspond to the realized workflow. Another requirement is that the report must have a low threshold to create. Since ideally the user is requested to create this report, it should be quick and low-cost. A simple bulleted list, textual document has this step-by-step nature and was used for the validation of the case study in this approach.

Step 8: Validation of system

Validation is very important to measure the performance of both the product and the product development. This step ultimately gives the feedback on how well the workflow model created in the early phases of the process matches the real workflow of the user. With the detailed workflow already available, product validation in this method is quite simple. The workflow models are manually compared with the report created in step 7. A quantitative and/or qualitative measurement on the resemblance between the two documents gives a good indication on product performance. This should be accompanied by asking the user two basic questions:

- Does the realized workflow match your expected workflow? (validation)
- How well do the realized workflow and product satisfy your needs? (quality)

The result is valuable product development feedback and should be added to the project archive.

CASE: INTRA-OPERATIVE MRI

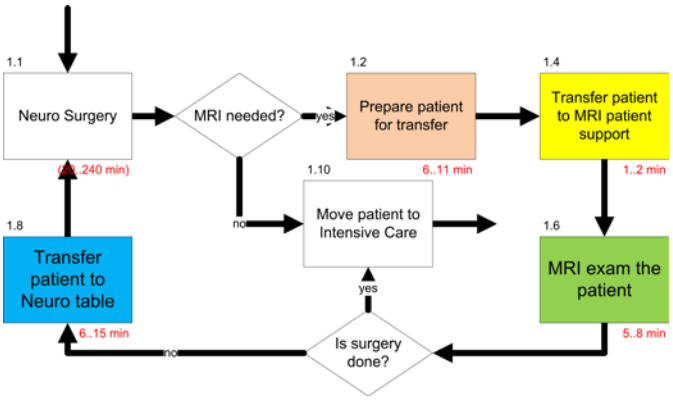
During neurosurgery with an intra operative MRI suite, the neurosurgeon needs to navigate through the brain in search for the specific area of interest (e.g. tumor). A Magnetic Resonance Image (MRI) provides images of the brain tissue of the patient. The location and geometry of the surgical instruments are mapped real-time on the MR images giving the surgeon feedback on what he is doing. After a while the surgery itself will have changed the situation. The intra operative MRI makes this possible during surgery instead of post surgery. This reduces the need for re-surgeries, reduces the risks for the patient by quick recovery, results in fewer costs and allows the surgeon to inform the parents/relatives about the success of the surgery immediately after the procedure when the patient is in recovery.

Project characteristics of the specific product development process are:

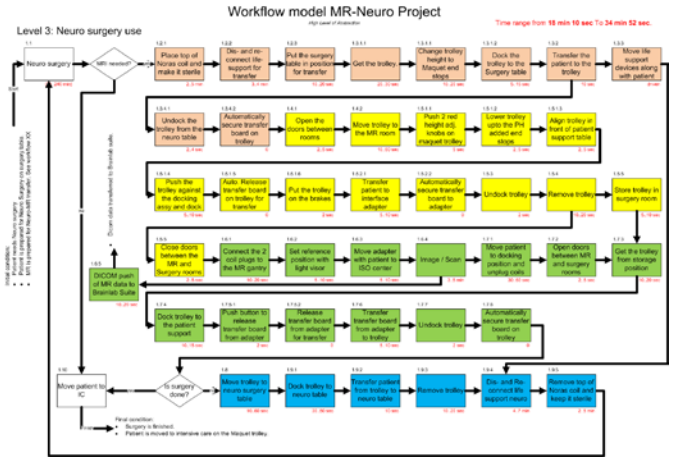
- A patient centric solution
- A multi disciplinary, multisite project team
- The product would have to be the integrating factor between multiple existing medical modalities.
- The project was based on partnerships
- Based on a minimal amount of system changes to the existing MRI system.
- A short project timeline (less than 1 year between start of project and first surgery)

Re-use of existing components was a pre-condition. Four main sub-systems needed to be integrated:

- Surgery table and trolley (3rd party)
- Neuro-navigation software and hardware (3rd party)
- Head fixture (3rd party)
- MRI system (main developer)



(A)



(B)

FIGURE 7 CASE WORKFLOW. LEVEL ONE (A) AND LEVEL FOUR (B)

The first three sub-systems in the list are from third parties. Among the stakeholders involved were; users, buyers, third party co-developers, third party suppliers, marketing, designers and system architects.

Three 'main' workflows were determined; setting up, surgery and breaking down. Presenting the detailed workflows is not possible due to space limitations in this paper. But fig 7 (b) gives a global overview of the detailed fourth level workflow. The first level surgery workflow is displayed in fig 7 (a).

As you can see in fig 7 (a), the first level workflow consists of 8 nodes. This workflow was decomposed into three more levels. The fourth level, fig 7 (b), workflow description had 48 nodes. The last two levels just fit on an A3 piece of paper and were communicated to the stakeholders on paper and digitally. Time estimations were added to the individual nodes to calculate global time duration. The function models are not displayed here, but could be presented on A3 paper as well.

The detailed workflow description was presented to the customer for review. The users agreed with the proposal. Afterwards the customer requested the model for internal communication to other users.

The validation process focused on product quality, customer satisfaction, and workflow comparison. Product quality and customer and user satisfaction were measured qualitatively through interviews and a questionnaire. The realized workflow was compared with the modeled workflow in two ways; the users had prepared their own realized workflow description and a real surgery was observed by representatives of the development team.

DISCUSSION

The successful product delivery was a result of many factors. The work described in this paper was just one of them. By having a common understanding and view on the intended workflow, a shared mental model of the design task was created. The workflow model consists of things necessary to satisfy the user and nothing additional. This minimal but sufficient approach kept the focus on the user needs. This also had a positive influence on keeping the schedule, because no deviations occurred. A product was delivered that does what it was supposed to do with a transparent, lean and modular architecture.

The workflow models ended up in several project documents. It ended up in the customer requirement specifications project presentations. Furthermore they were used as a communications medium during the design workshops.

From a developers point of view this case is considered a success. The system does what the customer expected and therefore gets a high 'score'. Especially the workflow models were of great value in, making the user needs explicit, and allowing for a simple but thorough validation process. Although the FBS model indirectly provided valuable input to the system architects in getting their interfaces clear and understandable, a small barrier was observed in really adopting the FBS model or creating it themselves. Research on how to apply FBS models better in industrial settings remains on the agenda.

As indicated in step 3, knowledge about causal relations between behaviors and functions created in the FBS model from reasoning about behaviors can be used and fed back into the workflow models. This is an interesting direction for further research.

For this specific case, the product development process was quite complex. The quality of the product was high because it totally lived up to the user's expectations by reusing existing functionality/products where possible, but having a minimal impact on the existing architecture. Arriving at this lean, modular, transparent and simple architecture was the objective in the first place. Iteratively updating and checking

the workflow model with information from both the user's and the system's perspective ensured that the lean architecture proposed would support the envisioned use of the system. By good system architecting, the complexity of the system itself could be kept under control. Applying the method to an even more complex case can be researched further.

CONCLUSIONS

This paper proposed a systematic design method to capture and validate user requirements using human understandable and decomposable workflow models. Connecting these workflow models to the system architecture through functions ensures that the system being developed will satisfy the user.

The use of workflow model helps engineers to better be able to determine functions of a system. Taking the workflow as a starting point for decomposition is easier than starting directly starting from the function domain of FBS.

Workflow models consist of clearly visibly distinguishable nodes. Decomposing these nodes one-by-one into new nodes guides the system architects in systematically creating a system decomposition. The human understandable language used in the models helps keeping that process understandable.

Workflow models help closing the product development cycle. At the start of development workflow models are created by all stakeholders including the user, and the same models can be used for closing the loop through system validation.

Research directions for future work should address and test scalability of the proposed design method. Another interesting research direction can be found in the behavioral reasoning domain of FBS. Creating better systems architecting based on reasoned out behavioral knowledge is an interesting research direction.

ACKNOWLEDGEMENTS

This work has been carried out as a part of the DARWIN project at Philips Healthcare under the responsibilities of the Embedded Systems Institute. This project is partially supported by the Dutch Ministry of Economic Affairs under the BSIK program.

REFERENCES

1. Pahl, G. and W. Beitz, *Engineering Design: A Systematic Approach*. 1996, Berlin: Springer-Verlag.
2. Bonnema, G.M., *FunKey Architecting, An Integrated Approach to System Architecting Using Functions, Key Drivers and System Budgets*. 2008, University of Twente: Enschede, The Netherlands.
3. Muller, G., *A multidisciplinary research approach, illustrated by the Boderc project*, in http://www.gaudisite.nl. 2007.
4. Anggreeni, I. and M.v.d. Voort, *Classifying Scenarios in a Product Design Process: a study towards semi-automated scenario generation*, in *CIRP Design Conference 2008*. 2008, Springer: Enschede, The Netherlands.
5. Booch, G., J. Rumbaugh, and I. Jacobson, *Unified Modeling Language User Guide, The (Addison-Wesley Object Technology Series)*. 2005: Addison-Wesley Professional.
6. Van der Aalst, W., *The application of Petri nets to workflow management.* Journal of Circuits Systems and Computers, 1998. **8**: p. 21-66.
7. WFMC, *Workflow Management Coalition Terminology and Glossary (WFMC-TC-1011)*. 1996, Workflow management Coalition: Brussels.
8. Forsberg, K., et al., *The relationship of system engineering to the project cycle.* Center for Systems Management, 1994. **9**: p. 11.
9. Beitz, W., *VDI 2221: Systematic approach to the design of technical systems and products*. 1986, VDI society for product development, design and marketing.
10. Umeda, Y., et al., *Function, Behaviour and Structure, Application of Artificial Intelligence in Engineering V, Vol 1: Design, JS Gero*. 1990, Computational Mechanics Publications, Boston.
11. Umeda, Y. and T. Tomiyama, *FBS Modeling: Modeling scheme of function for conceptual design*, in *Proc. of the 9th Int. Workshop on Qualitative Reasoning*. 1995. p. 11-19.
12. Erden, M.S., et al., *A Review of Function Modeling: Approaches and Applications.* AIEDAM: Multi Modal Design, special issue. in press.
13. Umeda, Y., et al., *Supporting conceptual design based on the function-behavior-state modeler.* Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 1996. **10**(4): p. 275-288.
14. Yoshioka, M., et al., *Physical concept ontology for the knowledge intensive engineering framework.* Advanced Engineering Informatics, 2004. **18**(2): p. 95-113.
15. Murata, T., *Petri nets: Properties, analysis and applications.* Proceedings of the IEEE, 1989. **77**(4): p. 541-580.
16. Danilovic, M. and T. Browning, *Managing complex product development projects with design structure matrices and domain mapping matrices.* International Journal of Project Management, 2007. **25**(3): p. 300-314.
17. Shimomura, Y., et al., *Representation of design object based on the functional evolution process model.* Transactions-American Society of Mechanical Engineers Journal of Mechanical Design, 1998. **120**: p. 221-229.