INT J COMPUT COMMUN, ISSN 1841-9836
8(5):662-672, October, 2013.

Remarks on Interface Oriented Software Systems Modelling

D. Bocu, R. Bocu

Dorin Bocu, Răzvan Bocu*
Department of Mathematics and Computer Science
Transilvania University of Brasov
Romania, 500036 Brasov
d.bocu@unitbv.ro
*Corresponding author: razvan.bocu@unitbv.ro

Abstract: This paper explores the foundations regarding the systematic usage of the concept of interface in order to sketch a methodological approach, in which the fundamental perspectives that guide the abstracting of a software system solution (referred to as UP, SP and BP in the paper) are unified, with the goal to optimally derive the behaviour of the system from its structure. Moreover, this is very useful for opening new avenues in order to address the shortcomings that are provoked by changes, considering a software system that is conceived at an industrial scale.
Keywords: user perspective, structural perspective, behavioural perspective, interface.

1 Introduction

Have all the issues that impede the realization of a quality software system been solved? We can't even speak about something like this. This is especially true nowadays, when the topological diversification of software systems has gone so far [2]. There are two reasons that will prevent this process from ceasing and will determine further amplification of it:

- An increasing number of the human activities are optimized or taken over by IT systems.
- The technologies that underline the creation of the IT systems bear an overwhelming strategic repositioning on the map that pictures the latest developments in the field.

In these conditions, the endeavour to inventorize the already existent paradigms and technologies becomes increasingly difficult. This process should have the essential goal of simplifying the choice of the paradigms and technologies that are necessary in order to roll out an IT project [3]. In this paper, we study several aspects that are related to the possibility of modeling the solution of a software system, while easing the natural demarche through which the switch from structure to behaviour is accomplished [7].

Although experts are largely unanimous regarding the qualitative determination of a system's behaviour by its structure [4], the experts themselves do not give up the search towards the identification of some new thought schemes concerning the relation between structure and behaviour in the process of a software system solution's modelling [5].

2 A Short Account on the State of The Art

Naturally, we can position ourselves far away from the beginnings regarding the technologies that are used in order to model the solution of a software system. Using a simplified and non-exhaustive approach, we can observe the following:

- In order to roll out the analysis activity in relation to an information system at a reasonable quality level, there are standard investigative methods and languages, with which the analysis results can be represented. The endeavour to optimize the streams of an information system involves the necessity to first realize a precise map of these streams. This is the main goal of the analysis activity. Naturally, considering a complementary layer, the analysis process is also intended to discover the requirements of the people that sustain the activity of the software system, regardless of the executive or managerial status of those who are involved [2]. We can offer examples of tools, with which it is possible to rationalize the activity of analysis: flow diagrams, Gantt diagrams, PERT diagrams, flow and control diagrams, UML and BPMN, etc. [1]. The technologies that feature footnotes represent two state-of-the-art approaches that aspire to the status of universal tools for the modelling activity [3].

- The progresses that have been achieved towards the rationalization of the modelling process are remarkable. It can be stated that in the war of methodologies an armistice has been reached, around the idea according to which the ideal solution for the specification of a modelling system is represented by the adoption of some rigour exposure formulae that is associated to a modelling process with the help of a notation that cautiously combines the formal correctness with the visual ingredients. It is a beneficial armistice both for developers and for the CASE tools developers.

- Considering the previous remarks, can it be stated that the modelling activity is free from syncopes? It seems to be accurate to answer no. The modelling has continuously progressed, but it still faces open problems. As we have already mentioned, one of these problems is approached in this paper: the problem concerning the relation between the structure and the behaviour. More precisely, the idea that is highlighted is, briefly: how should the modelling activity be approached in order to ensure a reliable and flexible passage from the structure of a software system to its behaviour.

The structure of a software system, considering different levels of abstractization, represents a potential that has to be efficiently valued considering every phase of a software system solution's maturation [8].

If we do not work efficiently, we'll have to schedule additional iterations, which correct the modelling errors or eliminate the unintentional shortcomings of the modelling demarche [6]. What can we undertake in order to be more efficient in the modelling activity? An attempt to answer this question can be found in the following sections of this paper.

3 The Historic Premises of a Quality Modelling Endeavour

It has already been mentioned in Section 2 that specialists have constantly done their best in order to streamline the modelling of a software system [9]. With the risk to alienate part of the specialists that have already studied the same topic, it can be stated that the summary of these efforts can be expressed as follows:

From the structuring activity, using a methodic abstractization, towards object orientation. The structuring activity emphasized and sometimes turned into a fetish the necessity to structure data and the operations, following specific rules, with the goal to realize reliable and easy-to-maintain software systems.

During the era when the structuring activities used to prevail, mapping the solution domain on the problem domain hadn't been considered a clear priority. The art of modelling used to

be an example of an approach that had been exclusively preoccupied by its own topology and coherence.

The structuring activity has obviously brought an increase of clarity regarding the modelling and its subsequent demarche, that is the implementation.

The structuring paradigm showed its weaknesses in relation to high-complexity modelling problems. Thus, we have to put together some other ideas apart from the structuring paradigm in order to be successful in such cases. We have to grant special attention to different abstractization modalities, as abstractization imposes itself as a fundamental tool for taking over the complexity. As a consequence, specialists have theorized and elaborated progressive abstractization procedures, both in relation to the universe of the data and to that of the operations. Abstract data types can be considered a kind of a synthesis of the approaches that held at their core the methodic abstractization. This is a synthesis that has anticipated the revolutionary paradigm that is represented by object orientation. Is object orientation a paradigm without problems? Aspect oriented programming suggests that the answer is no. The same message comes, although not as a reproach, from the component oriented programming, as well. Where do the most powerful criticisms of the object oriented programming come from? Surprisingly, they come from those that intensely and methodically plead for the importance of the object oriented modelling activity. Many of the object orientation concepts and principles are accompanied by indications and contraindications. Even the programmers that discerned the importance of a quality design for the success of an IT project feel that something is not in order with object orientation. It is true, no human creation can avoid the criticism of the human being himself. Even if nothing can be objected in relation to the technical aspects of a technology, which is unlikely, the problem of taste remains open. This is a problem regarding the correspondence between the technology's offerings and the user's expectancies, considering the three tiers: syntactic, semantic and pragmatic. In OOP (Objected Oriented Programming), we can state that the world can be assimilated to a structured collection of objects. Thus, we speak about a rigorously-defined set of objects that interact. Any interaction involves the existence of some actors and a communication protocol. Here lies the problem: the communication protocol is always different for every new software system. It is clear that a generally valid communication protocol cannot be specified for all the software systems. Nevertheless, we are convinced that we can "draw" a communication framework that is bound to help at specifying the communication protocols amongst the actors of a particular software system.

4 Short Inventory of Essential Problems, which are Frequent in the Modelling Activity

The systems that belong to the surrounding world are structured entities collections, due to the fact that they have the potential to operate and co-operate.

We have been aware that things are like this for a long time now. Even if we cannot precisely anticipate the future states of the systems that determine the operation of other systems, we are at least able to know, with approximation, the cause of the changes that we witness. Every systems developer, but also every researcher that studies a system, looks for the most reliable way to increase the quality of their own demarche. Consequently, what are the main issues that impact on the general endeavour for progress in relation to the systems development and modelling? Following, they are presented according to a sequence that does not yet suggest anything:

- The limited human capacity to analyse the structure and the behaviour of the systems. The limitation that we refer to in this case is quantitatively and qualitatively measurable.

We cannot think considering rhythms that resemble to the power of the continuum. We think according to rhythms that are consistent with the discrete structures, on which is founded the difficult-to-define human intelligence. The discrete systems are not necessary examples of approximate knowledge, which are prone to fast moral wear. Humans have created systems, be them theoretical or practical that have successfully passed the test of time, which is the surest proof of their quality.

- The intrinsic complexity of any creative demarche. Modelling the creativity is a priori a chimera. Nevertheless, it is within human reach to stimulate the creativity through the automation of all the activities, which together with methodical repetition and optimization become algorithmizable. Freed up from the concern to consume time for performing some routine activities, the human being has all the time in the world to be creative, that is to push forward his fight with the unknown from within himself and from the universe.

- Being characterized by the limited capacity to analyse the structure and the behaviour of the systems, and considering that any creative step features a non-conventional complexity, the human being pushes his boundaries through the use of the paradigms. Any paradigm, regardless of the field of knowledge, is an instrument for treasuring some qualitative accumulations that relate to the human creativity. Any paradigm has the role of keeping active the flame of knowledge discovery, provided it becomes a tool that is generally accessible. Thus, it can be stated that paradigms resemble to their creators: they are being born, they mature, they flourish for a while, and then they are replaced by some other paradigms.

- The most precise history of a certain field of knowledge is the story of the never-ending chain of paradigms, which have fuelled the illusion of forward movement in that respective field of knowledge. The history of the happenings inside each paradigm is the only one that has the power to legitimize the joy of living timelessness, but also the need to periodically return to its ever-renewed sources. We dare to assert that the perpetual precariousness of the paradigms, to which the human being relates as a knowledge agent, is a trick that is used with the certainty that the human reasoning is able to push the boundaries of knowledge deeper into the unknown. The goal of each paradigm, not always clearly stated, is to bring an increase of knowledge in a certain field. The initial universality of each paradigm is abandoned as soon as the achievements that have been registered in relation to it proved its conjunctural capabilities.

- Although modelling is an activity that is made possible only from inside a paradigm, which impose a certain rigour to the approach, obeying the exigencies of the rigour is, generally speaking, problematic. Humans prefer to model as if they were speaking in the mother tongue,by making slalom through ambiguities, correctness, metaphors, formal abstractions with limited reach, and some other methods that are used to explore the incomprehensible. This natural predisposition of the humans conflicts with the need to discipline the modelling activity, which is claimed by the tools producers and also by the management of the IT projects that is intensely focused towards the success of these projects.

We have presented through the problems that have been introduced, part of the reasons that provoke the seemingly chaos that thrones in the world of the modelling paradigms. This chaos is, considering a more attentive analysis, the expression of the continuous search effort that aims to enrich a given paradigm or propose a new paradigm. In this paper, we present the main ideas of a modelling paradigm that establishes the importance of the concept of interface as a methodical bridge, which links the structure and the behaviour of a software system.

5 Concepts and Fundamental Principles that Are Used in the Interface Oriented Modelling of the Software Systems

5.1 Preliminaries

In accord with the objectives of this paper, in this section we attempt to sketch the essential conceptual infrastructure, which is necessary in order to practice the interface oriented modelling in the IT industry. This infrastructure obviously refers to the concepts with the help of which the interface oriented modelling may become operational, with promising results. Figure 1 presents the role that is played by the concept of interface in the modelling activity, considering very high levels of abstraction.

In *Figure 1*, the two actors that are designated by Developer and Client abstract the modelling expert that is perceived in one of the two fundamental hypostases: models producer or models user in his own modelling activity. In any of the two hypostases, the modelling expert can address the complexity-related issues by granting the proper attention to the concept of interface. Although what Figure 1 suggests is as correct and obvious as it can be, this does not mean that any modelling expert automatically becomes a winner, if he understands the message transmitted by Figure 1. What is, in fact, the message?

The mind of the one that models has to get used to perceive systems, that correctly and coherently relate to their environment through the concept of interface. Thus, we solve two problems that have a major impact on the quality of the models that we elaborate: the streamlining of the service traffic between software systems and the assurance of the modular continuity in relation to the software systems.

We could end the presentation of this paper here but, it wouldn't be a wise decision. Considering the *Figure 1*, we cannot deduce with sufficient clarity how the interface can help to optimize the relation between the structure of a system and the behaviour that is built on this structure. In order to provide additional clarity to our demarche, we make some preliminary remarks:

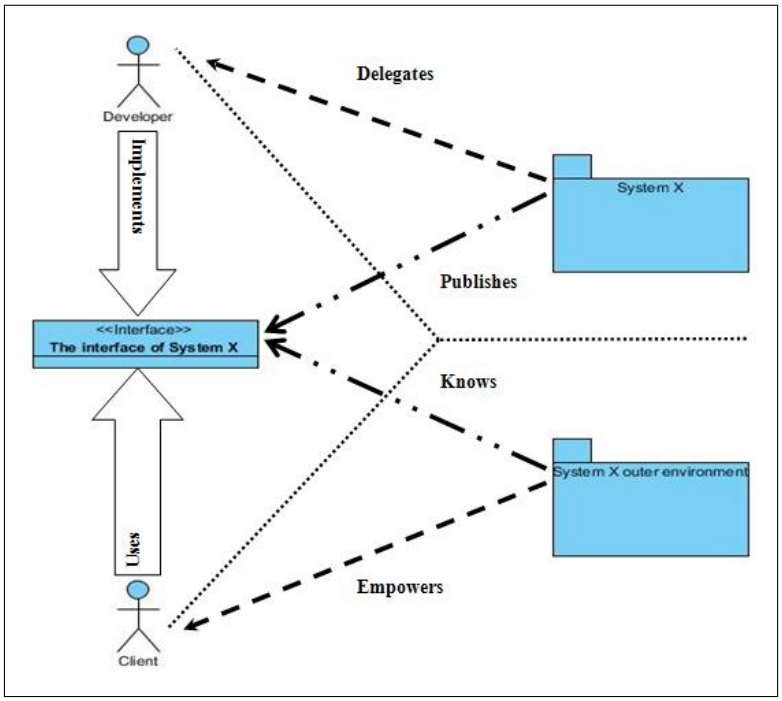

Figure 1: The interface: a fundamental tool for a modeling activity of the real world in IT

1. The modelling activity is an endeavour in which the abstractization is the essential tool that allows for the complexity to be mastered and modelled.

2. The modelling activity is spatially organized: on the horizontal, the semantics that correspond to different perspectives are structured; on the vertical, the abstraction levels of these perspectives are structured.

Consequently, it is necessary to specify the concepts, with which we operate inside the perspectives at various abstraction levels. We plan to deal with this problem, at least considering several iterations. At each iteration, we present a diagram, which is connected to an abstraction level that summarizes the concepts that exist at that level.

The beginning of the scientific knowledge passes through the concept of system. Considering that the modelling is a remarkable kind of knowledge, we can infer that the legitimate origins of the modelling belong to the area that is determined by the concept of system. The concept of system represents equally a modelling tool and, also, a tool for representing a certain type of modelling approach. As a modelling tool, the system can be explained through making use of three perspectives: the user perspective (UP), the structural perspective (SP), and the behavioural perspective (BP).

The necessity to utilize the three perspectives is relative. Representing different levels of maturation of a modelling demarche, and approached in an iterative manner, the three perspectives feature as a terminal point the decryption of the systems' behaviour, as it can be noticed in *Figure 2*. It is also natural to reflect on the links that are established between the three perspectives.

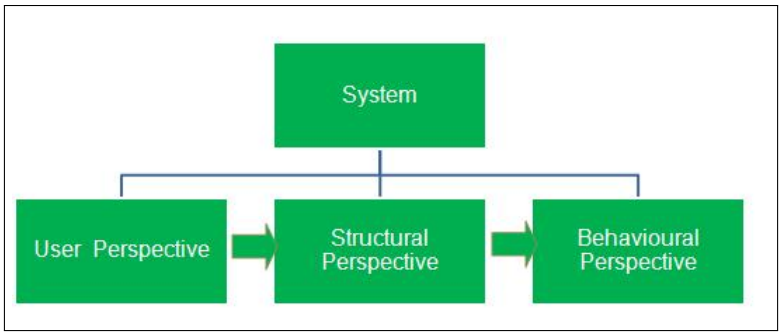

Figure 2: The main perspectives that are used for modelling a system

It can be assumed that based on the knowledge acquisition that is organized according to UP, the elaboration of SP becomes more natural, while considering an approach that favours the correct specification of BP, considering any level of detail.

Therefore, we speak about naturalness in the specification and preparation process of an optimal framework that is suitable to specify the behaviour. The link that is established between the three perspectives goes even further. This will be better understood as soon as the concepts that are part of the interface oriented abstractization base infrastructure are introduced.

5.2 The Concept of Interface in the Software Systems Modelling

Before proceeding to the actual brief description of each perspective and to their methodology driven assembly, we define, among other things, the concept of interface, while considering the highest level of abstractization.

In the practice of software systems modelling, it is called interface a protocol that allows for the communication between two objects to occur. The communication between two objects

takes place through a messages exchange. In order for the communication to be feasible, it is necessary that the sending object (the object that requires a service) knows the interface of the object that receives the message (the object that will provide the service).

This is the way the concept of interface can be understood, provided it is documented from an operational perspective. From a conceptual perspective, the notion of interface bears a series of meanings, which are probably more important for the quality of the software systems modelling activity.

An interface is a model that abstracts the messages exchange between two objects, with two essential goals:

1. The safeguarding of the fact that the objects that send messages to the object that owns the interface use a calling scheme, which remains unchanged in a reasonable time span.

2. The exemption of the object that receives messages from the task to adapt to possible changes from the environment it is related to.

In other words and considering the perspective that is induced by these de?nitions, an interface is correctly specified if the object it serves (service provider for the environment it is a part of) is, at the same time:

- temporarily deggrevated of the responsibility to adapt to the environmental changes;

- free to modify the implementation of the services that are published through the interface.

We are now entitled to ascertain that the interface is a double-sided abstraction tool:

- Provides stability for the clients that call it, which entitle us to state that the interface features genuine structural ambitions, because it regulates, in the long run, the manner according to which the clients of the owner-object may call the object's services.

- Relieves the object that receives a messages of the task to adapt to the environmental changes; thus, the interface offers the ideal framework for the respective object's behaviour optimization.

Without trying to state that this is the optimal variant, we promote the idea that the correct transition from a system's structure to its behaviour involves the methodical utilization of the concept of interface.

With the wish to clarify the framework related to which the contents and the connections of the three perspectives in *Figure 2* are presented, we introduce in the following paragraphs, two principles that feature a great power of generalization and up-to-datedness for the general knowledge process and, consequently, for the modelling activity.

5.3 The Principle of Consistency and Deductive Conditioning of the Perspectives

The logic that underlies the modelling activity is the one that governs the general knowledge. It is known that in order to model or explain the reality, the human being needs to analyse this reality. In the case of high-complexity systems, the analysis activity is organized in such a way that the human capacity to reason sequentially may found an approach, whose outcomes are comparable to a hypothetical knowledge/modelling exercise that is based on parallel processing capabilities. It could be stated that one of the secret aspirations of the human mind is the wish

to relate to this reality not as to a sum of parts that are put together by a structure with a certain granularity, but as to a whole whose parts are articulated according to rules that are known in detail. The latter set of rules features an essential importance, which is nevertheless secondary as compared to the importance of the whole.

It is essential to provide two assertions:

The regular human perceives sets of components that are structured according to the adequate observation and research possibilities;

The demiurge perceives systems inside of which the components move according to the implacable logic of the principles that form the foundation of the universe.

These are the two extreme approaches, which can be tried in the activity of knowledge acquiring/modelling. Given the precariousness of the paradigms that are elaborated in order to support demiurge-like approaches, we can only optimize the regular human-like approach. In such an approach, the complexity is taken over in a progressive manner, while the knowledge effort is mapped on several perspectives, in order to enhance the accuracy of the analysis. Each perspective has the goal to relate with specific means to the same semantics.

The consistency of each perspective and the deductive conditioning of the perspectives are two requirements that have the value of a principle in the activity of modularization/knowledge acquiring, in general.

The hypothetical continuity of a modelling approach is assured through an accurate relation to the principle of consistency and deductive conditioning of perspectives. The problem regarding the hypothetical continuity of a modelling approach is linked to another invariant of any modelling approach: the change as an immanent attribute to any real system.

Additionally, whether it is about the law of the entropy, or about the on-the-fly re-definition of the structural equilibria, the change is an important variable in relation to any system's modelling.

Consequently, the problem can be formulated as follows: is the model that has been elaborated capable of reacting to changes according to the level at which changes operate? Or, the changes provoke shock waves that affect the system in an uncontrolled manner? The continuity of a modelling demarche would ensure that the developer enjoys some added comfort, in the case he confronts with various types of change exhibitions.

5.4 The Principle of the Perspectives Overlap in the Modelling Activity

Although it can be considered a truism, we indicate another fact that is used by abstractization techniques in the software industry, without trying to problematize: the overlap of the perspectives. The idea of the perspectives' overlap can be presented as per the following paragraphs.

Considering that an abstractization method involves a multi-perspective approach, the connection between the perspectives is not only a deductive constraint (with the meaning stated in Figure 2, which indicates the qualitative progress of the abstractization), but also the overlap inside each perspective, which confirms the quantitative interdependence of the perspectives.

The modelling represents, at each moment, a mixture of evolution in time and specific state. As an example, the elaboration of the user perspective is premise for the structural perspective, but the ingredients it is composed of are of the type UP, SP and BP. Naturally, this composite is always characterized by the name of the ingredient that is documented at the highest degree, mainly UP in this context. Thus, the iterative and incremental nature of the human approach to modelling becomes manifest. This is valid both from a natural perspective, and considering the human struggle towards attaining efficiency in relation to the modelling activities.

Following, we describe the manner according to which the concept of interface, when used

at different abstraction levels of a software system solution, can contribute to disciplining and making more efficient the specification of the perspectives, which are mentioned in Figure 2.

5.5 The Methodological Unification of the Perspectives in the Modelling Activity

Although it seems exaggerate to consider such aspects, it is necessary to note that the identification of a software system's structure is just a premise for understanding its behaviour. Possibly, an explanation for this reality resides in the different ways that are used to perceive the structure and the behaviour.

While the structure needs to be mainly invariant and flexible as an additional feature, the behaviour requires mainly flexibility and has to be invariant as an additional feature.

The above assertion should become even clearer if we added the following explanatory note: while the invariance of a system is perceived based on the rigour of the relations that are established between its component parts, the system's behaviour is perceived based on the results that it produces while it operates.

The apparent pressure that exists between the two types of perspectives (SP and BP) can be "attenuated" by carefully handling the concept of interface. The careful handling involves the correct understanding of the roles that are played by the interface in relation to the partnerships it is involved in. Thus, in the context that is determined by SP, the shape of the contract between interfaces and the implementation-related resources can modify, while the content of the contract should be invariant. This becomes clear if we watch the semantics that is presented in *Figure 3*. In the context that is determined by BP, the shape of the interactions is temporarily frozen, while the content of the interactions can suffer modifications.

Let us study the semantics that is shown in *Figure 4* in order to understand the meaning of the above statements.

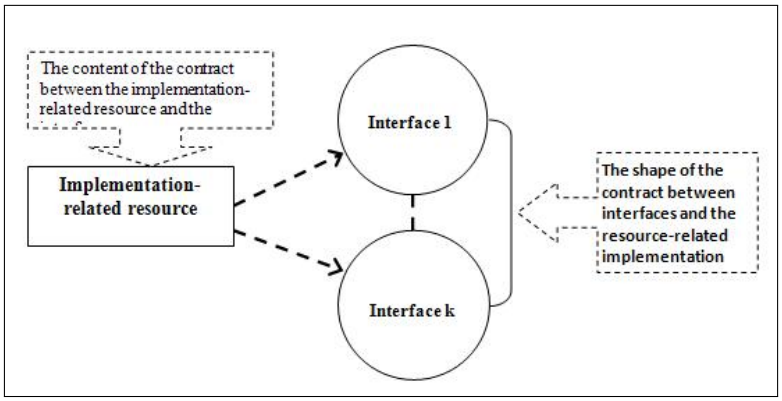

Figure 3: The role of the interfaces in the context of SP

Considering the previous considerations, we can also study the representation in *Figure 5*. This discusses the natural link between two abstractization circuits of a software system's solution:

- The **small circuit** (interior), which enhances the principle of the deductive conditioning of the perspectives, but also the principle of the perspectives overlap considering each step of the modelling effort.
- The **big circuit** (exterior), which presents the defining elements that support the consistency of the three perspectives (UP, SP, BP).

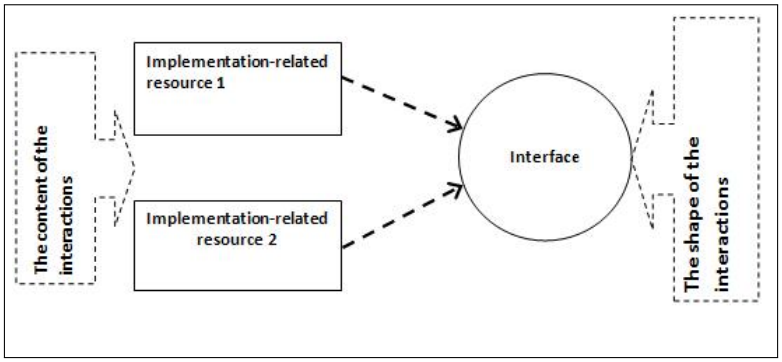

Figure 4: The role of the interfaces in the context that is determined by BP

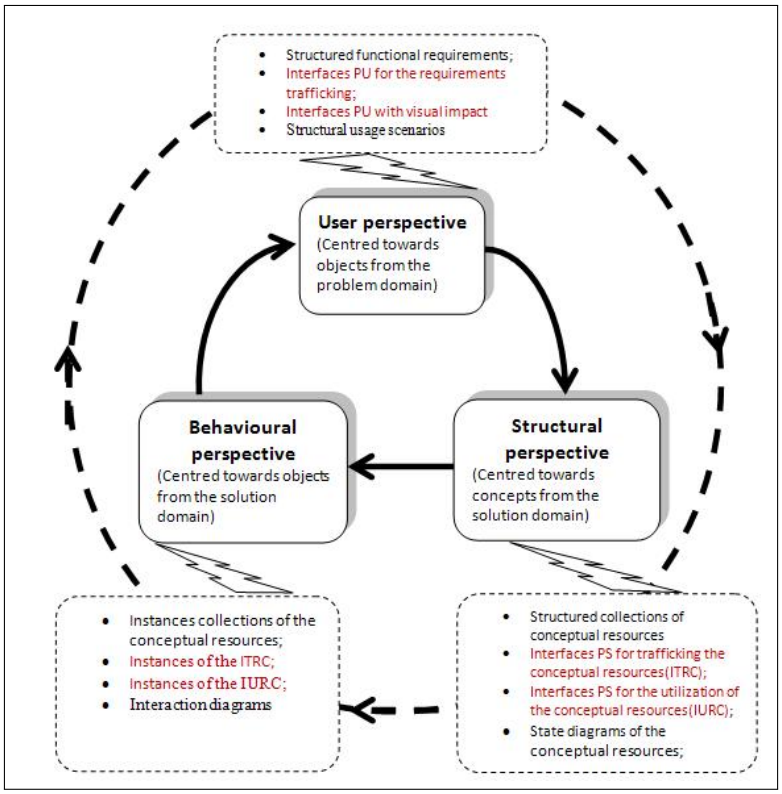

Figure 5: Unifying the perspectives in the IT systems modelling activity

6 Conclusions and Future Developments

We are now able to state that the methodological unification of the three perspectives in the modelling activity can be realized in an artisanal manner, or through methodically using the concept of interface. This involves the usage of the concept of interface, considering the particular usage perspective, as a stabilizing tool or as a flexibility-related tool in connection to different aspects of the solution, as it has been suggested by the proper remarks that have been made.

In a future paper, we plan to approach the problem of drawing a detailed framework that is bound to foster the derivation of a software system's behaviour from its structure.

Bibliography

[1] Bloch, J.; *Effective Java - Second Edition*, Addison-Wesley, 2008.

[2] Craig, I.; *Object-Oriented Programming Languages: Interpretation*, Springer, 2007.

[3] Garzas, J.; Piattini, M.; *Object Oriented Design Knowledge - Principles, Heuristics and Best Practices*, Idea Group Publishing, 2007.

[4] Martin, R.C.; *UML for Java Programmers*, Prentice Hall, 2003.

[5] Gomez, J.; Cachero, C.; OO-H Method - Extending UML to Model Web Interfaces, *Information Modeling for Internet Applications*, 144-173, 2003.

[6] Rector, A.; Axioms and templates: distinctions and transformations amongst ontologies, frames, and information models, *Proc. of the K-CAP Conference*, 73-80, 2013.

[7] Espinoza, L.; Espinoza, H.; Feng, W.; Modeling a Facilities Management and Information System by UML, *Proc. of the ITNG Conference*, 65-70, 2013.

[8] Filip, F.G.; A Decision-Making Perspective for Designing and Building Information Systems, *INT J COMPUT COMMUN*, ISSN 1841-9836, 7(2):264-272, 2012.

[9] Marian, Z.; Czibula, G.; Czibula, I.G.; Using Software Metrics for Automatic Software Design Improvement, *Studies in Informatics and Control*, ISSN 1220-1766, 21 (3):249-258, 2012.