International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 2, Issue 5, May 2014)

Generalized Model for Interactive Multimedia Tools Development, Used for Software Interface Implementation

Maya Stoeva[1]

[1]PhD candidate and Assistant at Faculty of Mathematics, Informatics and Information technology, Plovdiv University "Paisii Hilendarski", 4003 Plovdiv, 236 "Bulgaria" blvd., Bulgaria

***Abstract* — The power of applying models for application development and their interfaces, regardless of the technology used, is proven over time. With them are achieved: better structured system; saves time and money from the wrong directions for development of software; A better coordination between the different units in the case of larger projects. Multimedia interactive tools are part of software development, which not only became fashionable, but its importance continues to grow. Despite their greater use in practice, they are not available enough models yet to be used in their implementation. Famous ones have specific pros and cons of different types of projects that apply. This fact has provoked the author to do research in this area and to create a non-linear generalized model, combining the advantages and disadvantages of the known models. Thus, need to be adapted for various types of interactive applications. The purpose of this article is to describe the author created "generalized model". Other aim is to show where it is used to now, and where can be find applicability. In the considered model are defined and described specific processes that need to be taken under attention when developing a multimedia application.**

***Keywords* — Generalized model, interactive tool, multimedia application, software interface, prototype, prototyping, wireframing, specific multimedia controls, usability**

I. Introduction

The advent of new technologies and interactivity complicate the interface design of applications. The change in the ways of human interaction with the computer and the wide entrance of mobile smart devices create a new set of specific interactive tools. Their aim is to facilitate the process of building a user interface.

The implementation of interactive multimedia applications through models helps us clear the ideological, conceptual level what will be a future product. They enable us to generate a number of specific scenarios for the behaviour of the system, which can be easily changed in the development process.

Part of this process is making prototypes (prototyping), which are a valuable tool for developers in the later stage – implementation.

The author's aim is to give a brief overview and analysis of specific existing models, to describe the author's generalized model for the development of this type of interactive support tools and its applicability in practice. This all to become the basis of a detailed study and research over the years of own observations and practices in this area. Under attention is taken two most famous researches in this area - an interactive model of Rosemary Caffarella [1] and IDE interfaces Mobi-D of Angel Puerta from Stanford University [2].

II. Preliminary Notes

Interactivity as a concept has different meanings according to the context in which it is used. In our case we will assume its notation as a physical interaction between the user of the system and the system itself. (But it can be between users and application, between two or more software.) Example is mouse clicking, writing text from the keyboard, speech command; touch the screen of a monitor or mobile device and results from these actions. As interactive response actions, suggest the opposite side. That is the mechanism of interaction should motivate itself a feedback. It should be well thought out in order to produce the desired effect resulting from its use.

There are several groups of interactivity which must be taken into consideration in the description of the model. Interactivity can be understood as a multimedia possibility of a user to exert influence on the content or form of communication in real time. Thus, the user is involved in the end result of the interaction. Considering this type of interactivity in 2003 year Rafaeli [3] describes three levels of interactivity:

1) Two-stage - non-interactive communication.
2) Reactive communication
3) Fully interactive communication.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 2, Issue 5, May 2014)

Distinguishes three categories of interactivity:

1) *Navigation interactivity* – allowing the user to move through the software, using the relevant hyperlinks or buttons.
2) *Functional interactivity* – allowing users to interact with other users, and receive data from other systems such as RSS feeds, for example.
3) *Adapted interactivity* – allows customization of the system to the needs and views of consumers.

According to what is done in the space, the interactivity can be according: the time – **synchronous** and **asynchronous**; the participants – **one-to-one**, **one-to-many**, **many-to-one**, **many-to-many**; relative location – close remote. Each of these types has its advantages and limitations affect the realization of a particular interactive application. For example, [4]:

Many-to-Many – suggesting more than one view of the application has more than one nomination tool can be used to account for trends in consumer behavior; **Man-to-Man** – the answer can be provided completely (people are different), the reactions are based on human experience, participants have the ability to train during the interaction with the system, in contrast to other types of players (computers, mobile devices, software who need additional programming semantic logic); **Machine-tools** – can be conducted at nearby players, remote or mixed, allowed to multitasking (i.e. parallel can be made and other ways of interaction), physical contact is not possible (this can be an advantage or a disadvantage according to goals) can conceal identities; **Synchronous** – does not allow multi-tasking (as is performed in real time is possible only occurrence of an event in a given time), requires coordination , and more.

Because of these features of interactivity present in the model part, devoted to the logical process of choosing the right kind of interactivity.

III. Patterns And Styles For MODELING

When developing models for interactive tools we have the ability to create multiple user interfaces to meet the requirements of different platforms and subsequent implementation.

This is not reduced efforts for the proper design, but it helps. Real responsibility to choose the right kind of controls and interactivity remains, but because the very function and structure of the objects are similar in different technologies, we can describe them only once to a meta-level.

This defines two parts of the model - an area with objects (which we'll call it "domain objects") and areas with different scenarios of behavior and performance of these sites (" domain features"). According the way describes a model and its objects are identified three basic styles [5]:

A. *Flowing models*

Most of the modeling process using this method (over 69% according to recent studies [5] and [9]). The special feature of this style of modeling is that all activities - major and minor are placed in the system objects themselves, followed by the sub-objects. Later it can be integrated into the other larger objects or roles. But this nesting in one or other of them is difficult to use and generally deprive the system from flexibility. Therefore this method is not suitable for the "general model", maybe just for local example.

B. *Models, human-oriented, natural language*

This style covered models based on natural – spoken languages (subject-predicate-object) and is used by 10% of designers [5] and [9]. Like previous models style here again reflects activities in the direction roughly with distinct roles and objects. Again the tasks themselves are embedded in the individual units. Compared to the previous model only one connector (link) shows what are the links between activities. In the Fig. 1 can be seen role card of subjects, predicates and actions portraying areas representing the modeled objects. In this way this model with the available tools and paradigms of natural language away from the needs of modeling interactive tools.

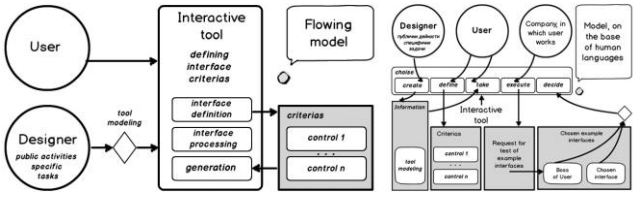

Fig. 1 - Flowing model and model, made on the base of natural languages

C. *Mixed models*

The processes here contain features of the both model, described above. According to statistics, 21% used this mixed technique [5] and [9]: In the Fig. 2 can be seen that the control flow between activities here is not consentient. The roles rather are used to trigger actions and objects are used to model the flow of data between activities.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 2, Issue 5, May 2014)

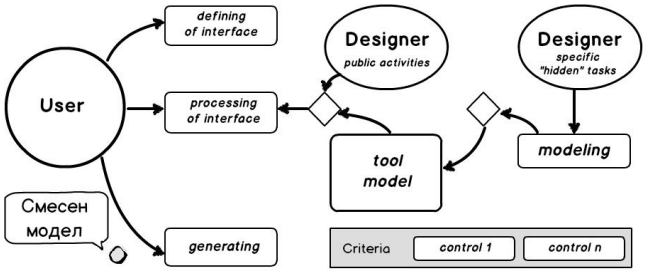

Fig. 2 – Mixed model

D. Caffarella model [1]

Here comes the time to complete the classification with the interactive model of Caffarella [1], which will later be extended to generalized model for the development of interfaces. Since it is difficult to be assigned to some other three, put it in a separate subsection. First, as it focuses on the processes, and the architecture of the application is defined as a part of them is not always constant greats (at a time may be subject to modifications). Second – has flexibility that others do not to this extent. On the ideological level, this model has neither beginning nor end. At each stage we can strike the necessary adjustments and return to a more advanced version or skip a particular step and come back later to it. The model itself is not linear, which in our case is one of the main advantages. It presents modeling of software as a dynamic and ever-changing process, organizing targets for in-depth development and analysis. Caffarella's model is a set of 12 main process in which the designer is actively involved. Decisions get left in the hands of the designer responsible for modeling software product. The main processes are: 1. Recognition context or what the needs and the scope of the future product; 2. Establishment of a rich base for support and collaboration; 3. Ideas for program development; 4. Sorting and prioritization formed ideas; 5. Development of program objectives; 6. Design plan with instructions/steps; 7. Develop a plan for transferring knowledge; 8. Formulation of a strategy for evaluation; 9. Development of recommendations and discussion of results; 10. Selecting formats, schedules, additional participants in the process; 11. Preparing budgets and marketing plans; 12. Coordinating resources and associated development meetings and events.

Modeling of the program structure is still good practice dictates that take into account and incorporate. This is the use of design patterns. They are architectural structural solutions that help solve the most often situations that arise in a subsequent program implementation.

They give us the non-use of certain algorithms or implementations as design patterns independent of programming language. They are grouped into four main groups: creating patterns, structural patterns, behavioral patterns and architectural patterns [6]. Regardless of which of them will choose according to their needs, using it will reach at least improve and speed up the design process of the interface. If we spend time to create a library of reusable widgets, we will help us to achieve easier consistency. And we meet in them hidden design principles. Reasons for not paying attention in the preparation of the latest models and design patterns are few. First, the principles are more theoretical and patterns describe specific instances of these good principles and strategies. Second, the principles guide us where to go, but don't show us how exactly. Templates are a bridge between them and the specific design implementation. Third, if the principles tell us what to do, the patterns explain us why and when to use them.

IV. GENERALIZED MODEL DESCRIPTION

The aim of the generalized model is to show the main aspects through which it passes in the time of development interactive tools for implementation of interfaces. It was developed as described above Caffarella [1] model and extended and adapted for the particular type of software that describes.

In general, one interactive tool includes the following parts connected to each other:

1) Domain or area with the objectives and requirements that must satisfy the tool.
2) Domain or area of tasks to be undertaken to achieve the objectives.
3) Design template that matches the architecture of the tool, helps to create a sample frame for this structure.
4) Domain with different types of interactivity that this tool will use.
5) Domain with user interfaces, including controls needed for the implementation of interactivity.
6) Domain with types of devices that will use interactive model.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 2, Issue 5, May 2014)

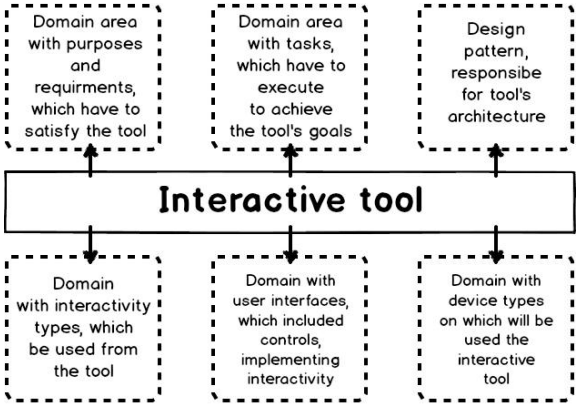

Fig. 3 – Model of Interactive tool

The user unit, which will interact with the tool after implementation of the model and the designer himself, remain as external participants in the scheme, but with the ability to influence the system.

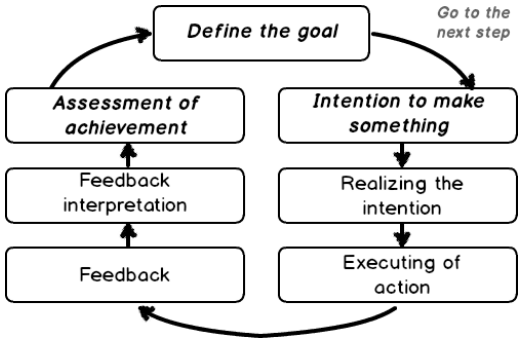

Fig. 4 – Activities' Cycle of Norman

That is why the next generalized model aims to cover all of these areas and help their faster and better definition. Since interactive actions in nature are cyclical (Activities' Cycle of Norman [7], see Fig. 4), and interactive systems are quite complex, the author believes that only a nonlinear model, which has no fixed beginning or end would be the more appropriate. Like that in every development step, at any time, we can make the necessary adjustments without affecting significantly the overall system. Generalized model is composed of the following processes:

A. Defining type of the interactive tool

Clarify that it will be only used in the organization or will be accessible to other users external to it, such as future or current clients. If the latter, should determine whether access will be on the remote server or in the cloud. The last has a number of advantages, even at this stage as a model.

In the process of prototyping, which we will discuss later, the cloud technology by sharing resources help to: facilitate cooperation, allows to edit the pattern everywhere, in its very essence is embedded interactivity, things are posted in real time, the system (in this case the model) alone build itself, financially is cheaper (this is immediately visible when budgeting). At this stage, we are taking into account the dynamics of the future system: low/high. From where and who will supply us with the necessary information for future product. Determine the level of access to our context. Here is the place to do comparative analysis with similar existing interactive systems, if any. Also, after clarifying the type of interactivity here is better able to identify technologies and means for its realization. It is important to clarify the advantages, disadvantages and limitations with which we must comply with selections affecting even on themselves prototypes that will be made in the next step.

B. Building connections between designer, project managers, developers and clients/users of the future software

The project manager is constantly in touch with all others. This is necessary for the fastest system building. Also helps for later future support and cooperation. This is a research phase for "Who are we dealing with?“. First clients' impressions and requirements are very important. Here we try to ask appropriate questions to clarify what is required on first glance from the future application. In other words, determine the needs of customers/clients.

C. Collecting ideas for program development

This process can be assumed as iterative. This is because often it may be necessary to return to it. Here is the time for inspiration, ideas, prototyping and testing them before the implementation. Also for defining interactive controls and tools that will be used and restrictions will have. The ideas can come from us, from the customer or special development groups for massive generation of ideas such as "brainstorm" workgroup. May be provoked by the work of other designers, planners are more abstract developed based on concepts, proposals and more. And to make the design of software better, it needs a good clear description of various software workflows. For that purpose we need to build prototypes (prototyping - wonderful technique for tracking multiple ideas quickly, without large investments and additional costs). Moreover, thus reducing erroneous assessments and easier emulate the real world. Prototypes usually pass through a series of changes in the overall duration of the project.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 2, Issue 5, May 2014)

So we should be always ready for additions of new features and changes. Therefore, it is better to build prototypes where it is possible as modules. Here comes the role of using design patterns. After presenting his prototype to consumers, you need to get feedback. And after stage II we target group and the formulating of expectations for the program is easier. As we must not forget that in this model is not a problem for us to go back in prototyping to activity II. In some cases II and III can merge to optimize the model, when it comes to smaller projects.

D. Sorting ideas and order them by priority. Define goals and tasks, which we put with our tool.

First we need to define clearly what are the criteria for sorting, and then proceed to the order of priorities. We should make a backup of the rejected ideas, in case they need to turn back to them. Then determine what will make this program and what do we expect to return us. Objectives themselves have to be well defined and understood by all participants.

E. Planning and creating a plan for information transfer.

Here we have to prepare the timetable and schedule of tasks and subtasks. Last two must be clearly defined and described in order to be understandable to any party, why is needed concrete time for a task (it is good when define deadlines always to leave extra time for extreme circumstances). We need to allow time for training new people via writing documentation or broadcasting mentors, involved in the process of developing the interactive tool.

F. Creating scenario for evaluation of the developed application.

Define different criteria for evaluation of the instrument, according to its context. Here we need to consider: first - what and why we want to evaluate - performance, speed, design flexibility; second - to describe consumer audience or system to be evaluated, the information that is needed and required; Third - how to make this assessment and in what format (test, game, interview); fourth - including semantics and meaning of evaluation (marking links in the process of evaluation), what will help us results and whether it will bring us a new information or confirm previous one.

G. Processing the results of the evaluations and recommendations extraction

Without this step, the prior evaluation loses its meaning. This is the time for a detailed analysis of where our tool had success and where it fails, regardless of the interface or functional perspective.

It is better to describe the results in carefully prepared reports, which the format is defined in the previous stage. Once the situation is clearer, we can write recommendations and observations, if have any, we can proceed to some of the other processes' adjustments, where is necessary.

H. Preparing budgets and marketing plans like take in mind example resources (not only like money, but also like time), accompanying development for meetings and events

At this stage of modeling have to be determined the approximate cost of the project, including the indicative price for the development, delivery and evaluation of interactive tools. As funds and technology that implements interactivity are constantly changing and build, is good to include extra oriented costs for future detailed development or changes. This is especially true, if it is a tool designed for mobile devices. As a sub-activity here are differentiated aspects such as finding funding and evaluation of return, budget management with regular reports and analysis. For marketing plan itself is well to take the benefits of the application compare to other similar and to prepare promotional materials, if it isn't for internal use, in order to reach a larger audience.

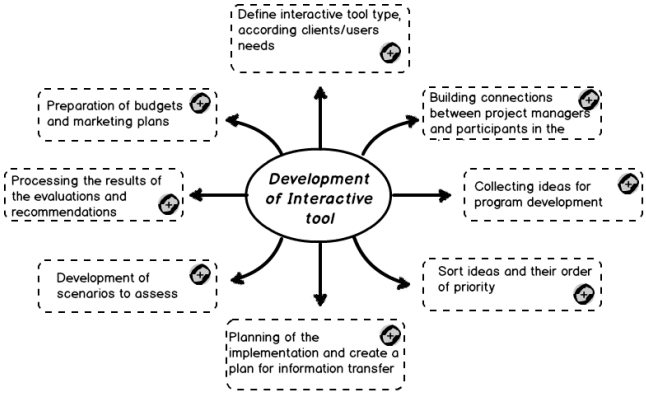

Fig. 5 – Generalized model

V. Applications Of Generalized Model

Until this moment the generalized model is used in a couple of real working author's projects. First of it is reported and printed of the International Conference "From DeLC to VelSpace" [10]. It refers to the modeling and implementation of interactive web tool for training and testing of pupils/students from different schools. Second usage is for modeling and developing the graphic LC constructor – a designer tool for logic circuits [11].

Other two are focused in other areas – one is modeling and design of HTML5/PHP interactive sketch-flow designer (for "sketching" and prototyping of web applications), and the second is related to the creation of model and prototype of visual and administrative part of the commercial interactive multimedia mobile shopping application. As the power of the latest is in the developed administration, which controls the whole design (described by meta language YML) of the mobile system, which is running under iOS and Android operating systems.

General model can be utilized for other future developments of various types of interactive tools to help designers, developers like shorten design and implementation time, and improve the interaction between them. This is achieved through the non-linear structure, which gives us flexibility. And to the participants in the project – an opportunity for the easy return at any time to a previous stage, rearrange or skip certain stages of it. A listing of future users in the process of idea generation and prototyping of applications improves efficiency, because the easier it becomes clear what is required of the future system. And rejection of irrelevant options for software versions, on the prototype level, saves time and money.

VI. Conclusion

Over the time, technology and means of interactivity are constantly changing. This versatility leads to changes in the methods of software and their interfaces development. Before were available only mouse, keyboard and several input-output devices. Now we have the opportunity to communicate also with our voice, sounds, and touches. Therefore, the use of described generalized model in implementation of interactive tools will be even longer updated alternative to the more specific design methods. Methods depend on the specific technology and resources for implementation of the interactivity.

REFERENCES

[1] Caffarella, R. S. (2002). Planning programs for adult learners: A practical guide for educators, trainers, and staff developers (2nd Ed.). San Francisco: Jossey-Bass.]. Exhibit 2.1, pp. 23-24.

[2] A. Puerta, "Model-Based Automated Generation of User Interfaces," Proc. Nat'l Conf. Artificial Intelligence, MIT Press, Cambridge, Mass., 1994, pp. 471-477.

[3] Sheizaf Rafaeli.,,Interactive Media face Artificial Consumers and marketing theory must re-think", http://www.computer.org/portal/web/guest/home

[4] Michigan state university, http://learndat.tech.msu.edu/teach

[5] England, D., Palanque, P., Vanderdonckt, J., Wild, P. (Eds.), Task Models and Diagrams for User Interface Design, 8th International Workshop, TAMODIA 2009, Brussels, Belgium, September 23-25, 2009, ISBN 978-3-642-11796-1

[6] Gamma E., Helm R., Johnson R., Design Patterns, 2005, Softpress, ISBN 9546853526

[7] Norman D., The Design of Everyday Things, 1990 r, Kindle Edition

[8] Heather H. Boyd, Evaluation Specialist, Citation: Program Development and Evaluation, Ways to Improve the Quality of Your Program Evaluations, Quick Tips #9, University of Wisconsin-Extension, Madison, WI. © 2002

[9] Herrmann, T.: SeeMe in a nutshell – the semi-structured, socio-technical modeling method, http://www.imtm-iaw.rub.de/imperia/md/content/seeme/seeme_in_a_nutshell.pdf

[10] Stoeva M., Krushkova M., "Modeling and implementation of interactive web-based system for education in game method", International conference "From DeLC to VelSpace" at Faculty of Mathematics and Informatics, Plovdiv University "Paisii Hilendarski", dedicated to 10-years from the beginning of "Distributed eLearning Center (DeLC)" project, 26-28 March'2014, Plovdiv town, Bulgaria

[11] Kiskinov H., Radev V., Stoeva M., "A Graphic Constructor For Logic Circuits Design", International Journal of Recent Development in Engineering and Technology, ISSN 2347 - 6435 (Online) Volume No. 2, Issue No. 4, 24-29 page, April 2014