A METHOD OF CONTEXT-SENSITIVE HELP GENERATION USING A TASK PROJECT[1]

Valeriya Gribova

Abstract. The article presents a new method to automatic generation of help in software. Help generation is realized in the framework of the tool for development and automatic generation of user interfaces based on ontologies. The principal features of the approach are: support for context-sensitive help, automatic generation of help using a task project and an expandable system of help generation.

Keywords: Ontology, task model, context-sensitive help, user interface development

ACM Classification Keywords: I.2.2 Artificial intelligence: automatic programming

Introduction

One of the basic quality criteria of any software is learnability (reducing learning time). A principal characteristic of learnability is intuitive understandability of a user interface; however, a help system in software is also required. Complexity and functionality of software are increasing every year. As a result, the number of users who know all features of an application program (according to statistics, users are familiar with about 10% of application program functions) is decreasing. Therefore availability of a help system for users is important feature software.

Development of a help system is a costly and time-consuming task. Most of software has some context-free help realized as static guidance or a tutor. Static guidance can be realized in the form of an on-line help system. Nevertheless, searching information in large help systems is difficult for users, and cost of maintenance is very high (when an application program is modified, the help system must be modified as well). Tutors also have shortcomings because they teach users only some aspects of using an application program and have high cost of maintenance in the life cycle of an application program. Context-sensitive help realized in some model-based interface development environments [1,2] has a number of advantages over context-free help. The most important of them are automatic generation of a help system and using a current status execution of an application program when an answer in the help system is being generated. Nevertheless, the grave disadvantage of this system is that it cannot be expanded.

In this article an approach to automatic generation of expandable context-sensitive help is proposed. The help generator is a component of the tool for development and automated generation of user interfaces based on ontologies [3]. To provide help the help generator uses a task project. It is a component of the interface project to be used to generate the executive code of the interface. The task project is a tree. To expand the number of context-sensitive help types a script language has been developed. Recently the context-sensitive help generator has been implemented and introduced into the tool for development and automated generation of user interfaces based on ontologies.

Tools for help generation

There are next types of help in software, namely, context-free and context-sensitive. Traditional help systems are context-free and realized either in the form of a static guidance system or of a tutor.

A static guidance system provides help that is defined as a canned text at the development stage. To learn some aspects of software features users have to read a part or parts of the guidance. There are several kinds of this help: books, instruction manuals and on-line static help. On-line help is developed by authoring tools in various formats (HTML help, HTML-based Help, JavaHelp, Oracle Help, Adobe Portable Document Format – PDF, Macromedia Flash, WinHelp, AP Help, and others).

[1] The presented work has discussed on the KDS-2007. It has corrected in compliance with remarks and requests of participants.

Compared with instruction manuals, on-line static help is more convenient. However, it is difficult to find required information for large on-line static help so developers make a particular section called "help for help". In some cases it is useful to know a special query language. The help system and the application program are not interrelated.

Tutors simulate some application program behavior; so make the process of learning easy for the user. However, they teach users only the main functions of an application program. To find out other functions of the application program, users have to use instruction manuals and on-line static help. Development and modification of tutors is expensive and time-consuming, because all the alterations in the application program must be represented in the tutor.

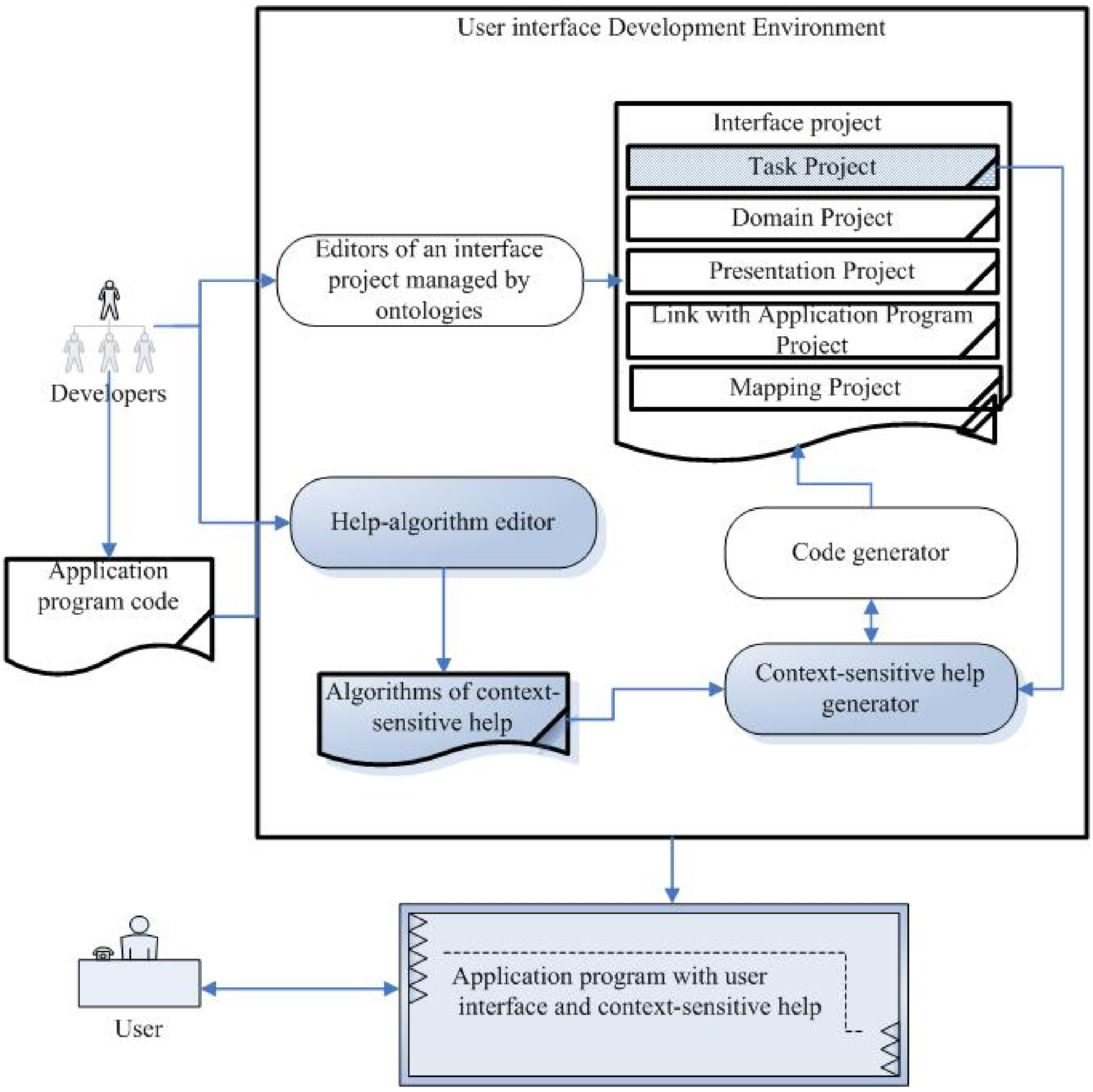
User interface Development Environment
Interface project
Task Project
Domain Project
Presentation Project
Link with Application Program Project
Mapping Project
Editors of an interface project managed by ontologies
Developers
Help-algorithm editor
Code generator
Application program code
Algorithms of context-sensitive help
Context-sensitive help generator
Application program with user interface and context-sensitive help
User

Fig. 1 The basic architecture of the expanding system of context-sensitive help-generation

Context-sensitive help is a kind of on-line help. There are two types of the context-sensitive help: conceptual and procedural. The conceptual help describes a framework of an application program, knowledge required to interact with it and the meaning of interface elements. The procedural help describes functions and operations of an application program. Unlike the conceptual help, it concentrates on user's tasks.

Implementation of the context-sensitive help is an important but very expensive task because it addresses to internal data structures. The most difficult objective is to provide conformity between the help system and the application program during the life cycle.

Some of the model-based user interface development environments (MB-IDE) have facilities for generation of context-sensitive help using an interface model. The principal method of context-sensitive help generation is based on transition networks. The transition network consists of a set of vertexes and arcs. Vertexes describe a set of valid states. Arcs show allowable user's actions. A help generator scans the transition network to form the context-sensitive help. A number of MB-IDE use a task model for help generation. The task model represents the tasks that users need to perform with the application program. It describes tasks by hierarchically decomposing each task into sub-tasks (steps) until leaf tasks become operations supported by the application program. Various specification languages are used for performing task models, for example, LOTOS, CCS [4,5]. Usually help systems can give answers to a set of questions which are task-related (*):

- Why this task is not available?
- How to realize this task?
- How to activate this task?
- What tasks can I perform now?

The basic advantages of context-sensitive help are coupling with the interface model (the task model is a component of an interface model) and automatic generation of help which makes the cost of development and maintenance of context-sensitive lower compared with other types of help.

Nevertheless, help systems support only a fixed number of questions. To add a new question to a system it is necessary to develop a new version of the help system. Tasks do not have an executing status so the quality of the help is lowered.

An approach to help generation

The state of the art in help generation enables us to set following some principle:

- Help should be context-sensitive;
- Help should be automatically generated at the time of user interface generation;
- Help should be invoked in any period of user's interaction with a user interface;
- Help should give answers to the set of questions marked * (see above);
- A help system should be expandable.

Context-sensitive help is realized in the framework of the tool for development and automatic generation of user interfaces based on ontologies (Fig. 1).The main idea of an ontology-based approach [3] to user interface development and generation is to form an interface project using ontology models which describe features of every component of the project and then, based on a component of this project (task project), generate a code of the user interface. The components of the interface project are:

- A domain project,
- A task project,
- A presentation project,
- A project of link a user interface with the application program (An application program project),
- A dialog scenario project,
- A mapping project.

Every component of the interface project is developed by a structural or graphical editor managed by an ontology model.

The domain project determines domain terms, their properties and relations between them. These terms describe output and input data of the application program and information on the intellectual support of the user.

The task project determines the tasks users can implement using the application program.

The presentation project determines a visual component of the interface and provides support for various types of the dialog.

The an application program project determines variables, types of their values shared by the interface and the application program, protocols for communication between the application program and the interface, addresses of servers and methods of messages transfer.

The dialog scenario project determines abstract terms used to describe the response to events (sets of actions executed when an event occurs, sources of events, modes of transfer between windows, methods of the window sample selection, and so on).

The mapping project determines relations between components of the interface project.

A context –sensitive help generator uses the task project of the interface project and a current task (the name of an executed task). The task project is a tree. The root of the tree is marked by the name of an abstract task which represents a set of application program tasks. The task project can be reused to design other interfaces if their application programs have similar features. Nonterminal vertexes of the tree are marked by names of abstract tasks. Terminal vertexes are marked by names of elemental tasks. The elemental tasks are tasks that cannot be divided into sub-tasks. Arcs of the tree do not have any marks; they link vertexes indicating the tasks hierarchy.

Every set Y can be divided into four types of sub-sets. The set Y={Y1, Y2,…,YN} is formed from marks of vertexes which are direct descendants of a vertex marked by X. Every sub-set of the Y set establishes relations among tasks. These relations are: choice, interleaving, synchronization and deactivation. These relations are taken from notations developed for specifying concurrent systems (LOTOS) [4].

- The sub-set called "choice" means that every task from the sub-set can be implemented; nevertheless, users can not start implementing any task until the previous task, i.e. the task which has begun implementation has been completed.
- The sub-set called "interleaving" means that every task from the sub-set can be implemented; users can start implementing any task of the sub-set while the previous task from this sub-set is been executed.
- The sub-set called "synchronization" means that only successful implementation of the sub-set task allows the user to implement other tasks of the sub-set.
- The sub-set called "deactivation" means that as soon as a task from the sub-set has been completed, implementation of the other tasks of the sub-set is broken.

A process of the interface project design according to the ontology-based approach, in particular, requires that the developer of a user interface in the mapping project determine links between every task from the task project and interface elements from the presentation project. By handling an interface element, the user initiates implementation of a task linked to this interface element. As soon as an event of the interface element occurs, implementation the task begins. It means execution of an action chain defined in the scenario dialog project. For example, a task called "activation of the expert

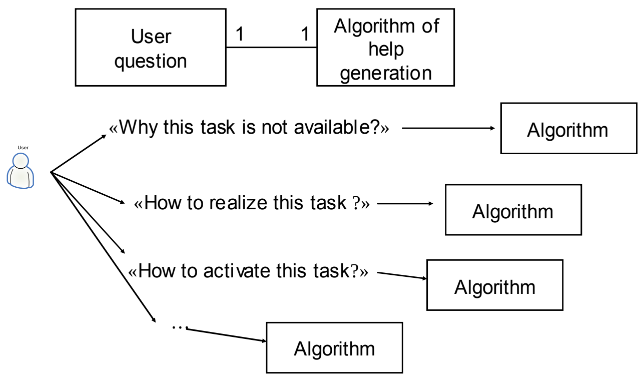
User question
1
1
Algorithm of help generation
User
«Why this task is not available?»
Algorithm
«How to realize this task ?»
Algorithm
«How to activate this task?»
Algorithm
…
Algorithm

Fig. 2. A principle of forming answers on user's questions

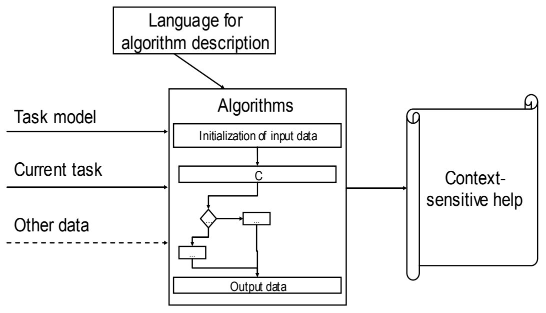

Fig. 3. A sceme of context-sensitive help generation

system" is defined in the task project. This task links to a push-button (an interface element) called, "activation of the expert system", in the mapping project. When an event called "keystroke" appears in the interface, i.e. it means that the user presses the push-button called by the name of this task, "activation of the expert system", the following chain of actions defined in the scenario dialog project begins to be executed: "to save data", "to pass data to the application program", "to represent new dialog window", etc.

To realize context-sensitive help, every chain of actions in the scenario dialog project must be extended by a system function. It passes the name of the executing task to the help generator. Inclusion of this function in the scenario dialog project automatically produces a sub-system of the help generator. Then, while interacting with an application program, the user invokes context-sensitive help. The help generator based on the user's query chooses an appropriate algorithm for help generation (Fig 2). Input data for this algorithm are: the name of the executing task and the task project (Fig. 3). To generate context-sensitive help the developer adds a set of interface elements used to call context-sensitive help to the interface project.

The principal requirement to the system of help generation is its expandability. To realize this requirement a script language for describing algorithms of help generation is proposed. This language consists of imperative constructions and a set of system functions which allow a new algorithm to be described. Using a structural editor the developer can add a new algorithm (a new kind of help generation). The added algorithm is transmitted to the XML-file and included in the algorithm base. Recently the main kinds of help generation mentioned in the requirements (see above) have been developed.

Conclusion

In this article an approach to automatic generation of context-sensitive help is proposed. The basic idea of the approach is to add an expanding system of help-generation to the tool for user interface development based on ontologies. The main task of the system is to form answers to user's queries using a name of the executed task and the task project. To date a prototype of the system has been developed at the Intellectual Systems Department of the Institute for Automation and Control Processes, the Far Eastern Branch, the Russian Academy of Sciences.

Acknowledgements

The research was supported by the Far Eastern Branch of Russian Academy of Science, the grants «An expandable system for quality monitoring».

Bibliography

a. Moriyón, R., Szekely, P., Neches, R.: Automatic Generation of Help from Interface Design Models. In C. Plaisant (ed.): Proceedings of CHI'94. New York: ACM Press 1994 (pp. 225-231).

b. Palanque, P., Bastide, R.: Contextual Help for Free with Formal Dialogue Design. In Alty J.L., Diaper D., Guest S. (eds.): Proceedings of HCI'93. Cambridge: Cambridge University Press 1993.

c. Gribova V., Kleshchev A. From an ontology-oriented approach conception to user interface development International //Journal Information theories & applications. 2003. vol. 10, num.1, p. 87-94.

d. Paternó, F., Faconti, G.: On the Use of LOTOS to Describe Graphical Interaction. In Monk A., Diaper D., Harrison M.D. (eds.): Proceedings of HCI'92. Cambridge: Cambridge University Press 1992 (pp. 155-174).

e. Sukaviriya, P., Foley, J.D.: Coupling a UI Framework with Automatic Generation of Context-Sensitive Animated Help. In Proceedings of UIST'90. New York: ACM Press 1990 (pp. 152-166).

Author's Information

Gribova Valeriya – Ph.D. Senior Researcher of the Intellectual System Department, Institute for Automation & Control Processes, Far Eastern Branch of the Russian Academy of the Sciences: Vladivostok, +7 (4323) 314001 e-mail: gribova@iacp.dvo.ru, http://www.iacp.dvo.ru/is.