การประชุมวิชาการเครือข่ายวิศวกรรมเครื่องกลแห่งประเทศไทยครั้งที่ 17
15-17 ตุลาคม 2546 จังหวัดปราจีนบุรี

The Efficient Search Technique for Mechanical Component Selection

Tanunchai Jumnongpukdee[1] Apichart Suppapitnarm[2]
National Metal and Materials Technology Center, National Science and Technology Development Agency
114 Thailand Science Park, Paholyothin Road, Klong Luang, Pathumthani 12120, Thailand
Tel: 0-25646500 Ext. 4357 Fax: 0-25646370 E-mail: tanuncj@mtec.or.th[1], apichas@mtec.or.th[2]

Abstract

In this paper we propose our implementation of the efficient search technique for mechanical component selection and demonstrate its application in the selection of bearings. Comparing our technique with conventional data retrieval methods, three advantages were identified. First, our proposed method could reduce time to search significantly. The technique allows users to search and filter for parts in many different consecutive paths according to the user's interests, so that the required parts are retrieved more directly and as intended. Second, as the technique has been implemented with great customization during the search process, the users can then view more and more details of the parts as they like, in any order field, and can stop the search anywhere once they have found the required parts. Finally, because our method has been developed with a powerful object-oriented concept i.e. Microsoft® Visual C# on the Microsoft® .NET framework, the implementation can also be easily applied for other engineering related projects. Our proposed technique has been presented as an integral part of a mechanical component selection framework where each component can be searched and retrieved efficiently from their relevant database. It is anticipated that the technique will also be useful for the development of searching for other products that required similar levels of searching complexity.

Keywords: Engineering search technique, dictionary of product characteristics, mechanical component database

1. Introduction

Currently, most of the dictionaries of product characteristics available in the form of electronic media, especially for engineering components such as bearings, are published in online catalogs or scanned paper catalogs on the internet and intranet. Although these give designers some advantages in the selection process, there still remained a few problems particularly when designers have to do several iterative search for the mechanical parts. One of the reasons is that there is a lack of well structured information retrieval as most currently available are just scanned paper catalogs with an alphabetical list of part characteristics. With this structure, the search mechanism can be performed only once, each with a specified set of keywords, and the most closely matched solutions or mechanical parts, in this case, are then offered to designers. Although the keyword search structure is considered efficient and most widely used for general purposes, such mechanism is not suitable for use in mechanical component selection such as bearings, gears, shafts etc. There are several considerations for each mechanical part and changes in the requirements or designer's intentions occur frequently during the design activities. For these reasons, the efficient search structure should offer designers some flexibility in the selection process, including the number of specified fields/considerations and the sequence of the fields, and allow the designer to filter information as needed. In this paper, we attempt to develop such a search structure, fitted with convenient graphical user interface, for mechanical component selection and demonstrate its application in the selection of bearings. The search program allows the user to filter and view more and more details of mechanical parts in many steps and in any order field, and the user can stop the search anywhere. For example, the user can first select the most important field, views the details of available parts at this point and then selects another interesting field and views the (closer) details of the parts and so on, until the parts meeting all requirements and intentions are identified.

Section 2 provides some background and reviews related to this work. Our system architecture and its implementation are then described in Section 3. A few conclusions are given in

Section 4 and possible future work related to the program is outlined in Section 5.

2. Background and Related Work

Most of online dictionaries of product characteristics in the field of mechanical component selection, such as those in [1] and [2], consist of scanned paper catalogs which can not easily be processed, retrieved and searched. In particular, when the designer needs to select several mechanical components, each with many different considerations, scanned paper catalogs with one-time keyword search offer very low level of matching accuracy during the search process and consequently, could not offer the suitable parts required by the designer. In the case of deep groove ball bearings for example, the designer can require considerations in static load, dynamic load and mass of bearing in one application and only static load and mass in another application. Different levels of requirements as well as different considerations in various mechanical components cause the keyword search structure for general purposes unsuitable for the engineering selection. By our experience, the dictionaries of product characteristics that have a more advanced search structure than a one-time keyword search do exist [3]. However, there are still a few limitations associated with this structure. Although such structure allows the user to specify the search in many requirement fields, the user has to identify the fields at the beginning and then input each requirement in those fields before the search could be performed. The user cannot view a set of solutions or parts during each step of the search and hence, the search still performs like a one-time keyword search algorithm, except that each keyword is identified in a more explicit format. Although the required parts could be identified precisely from catalogs with the above search structure, its presentation narrows the exploration of potential alternatives, and hence, the design overall, as the user cannot see such alternatives during each step of the search.

Considering the complexity of the search algorithm with different numbers and sequences of fields, if, for example, a programmer needs to write if-case conditions for fields A, B and C, there are a total of 15 possible paths that the user can select before conducting the search. These are A, B, C, AB, AC, BC, BA, CB, CA, ABC, ACB, CBA, CAB, BAC and BCA. The next question is that if we have several fields, more than 10 or 20 fields, say, then the number of paths to be constructed in the program will be very large. Table 1 gives the relationship between the number of fields and the number of possible paths.

Table 1: Relationship between the number of fields and the number of possible paths

Number of fields	Possible paths that user can select
2	2+2 = 4
3	3+6+6 = 15
4	4+12+24+24 = 64
5	5+20+60+120+120 = 325

In the book of Modern Information Retrieval [4], there is a computer program, called the filter flow model, which was developed by Young and Sheneiderman [5]. It allows users to filter more and more in a few attributed types or fields. However, the program has further limitation in the number of fields, only about 3-4 attributed types are allowed. The reason for such limitation could be speculated by the relationship given in Table 1. Our technique offers an alternative way that can manage a large number of fields efficiently and hence the technique can be applied to the selection of various mechanical components.

3. System Architechture

The system architecture of our search technique was established based on the Microsoft® .NET framework [6]. .NET framework is a set of Microsoft® core technologies for connecting information, people, systems and devices. Examples of .NET modules include windows forms and web forms, as we know well. Microsoft® .NET framework enables a high level of software integration through the use of XML (extensible markup language) web services. Our proposed program was written mainly with the Microsoft® Visual C# .NET, which is a hybrid between C++ and Microsoft® Visual Basic. Microsoft® Visual C# .NET enhances capability of C++ such as ease of reusable codes, and that of Microsoft® Visual Basic such as convenience of building a graphical user interface on windows form.

Figure 1 shows the system architecture of our proposed framework. The system architecture has three tiers as three-tier architecture can be disconnected in order to reduce overload of database server. One is the data tier which consists of multiple databases, in this illustration, the bearing database, and OLEDB. The middle tier consists of OleDbConnection, OleDbCommand and OleDbDataAdapter. Data tier allows us to retrieve a database with OleDbConnection class, by specifying a database name, host or server name and a security type. The read and write properties of OleDbCommand class are first set to their initial

values. The SQL statement is then filled into OleDbCommand class. There are a number of classes associated with the architecture. These are .NET framework class libraries. OLEDB is a set of interfaces for data access. OLEDB provides a flexible and efficient database architecture for accessing and manipulating all types of data. These interfaces will be used by data-consuming application. Our structure uses SQL server as a storage for the data tier.

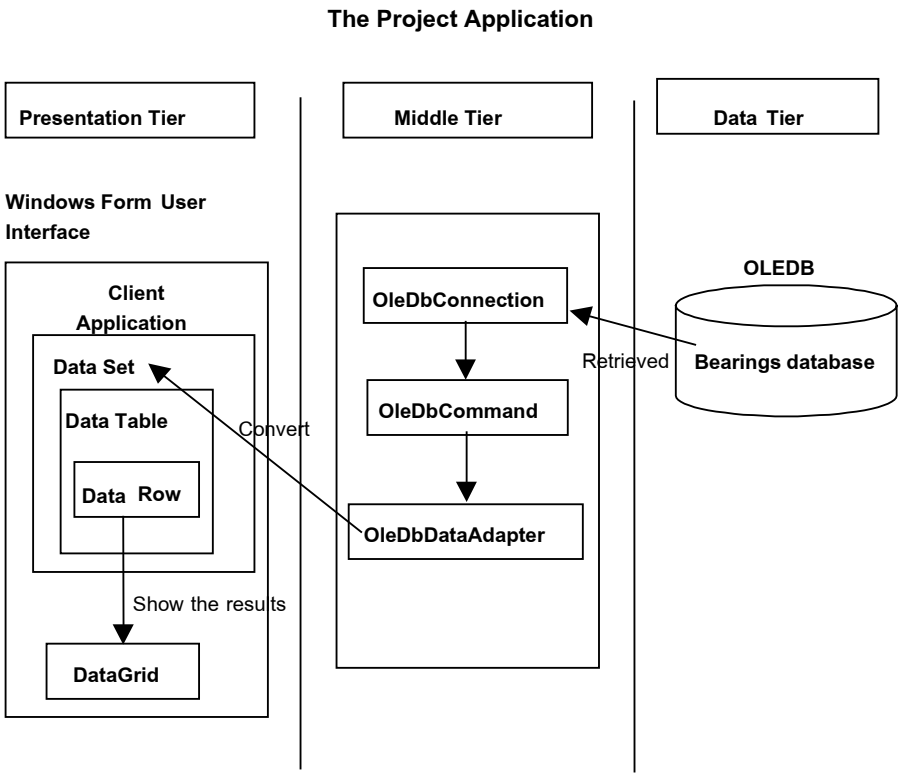


Figure 1: System architecture of our proposed framework

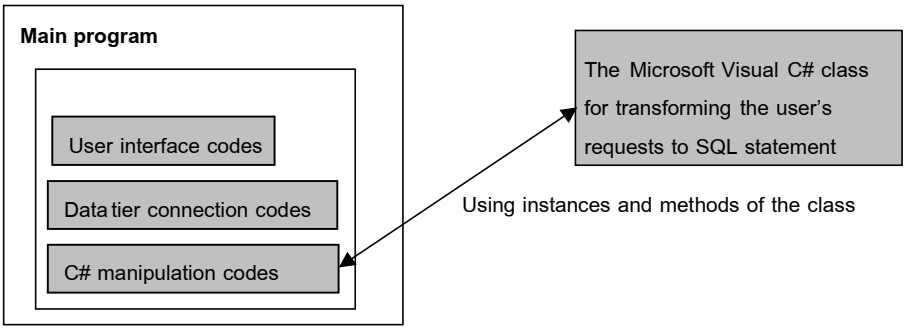


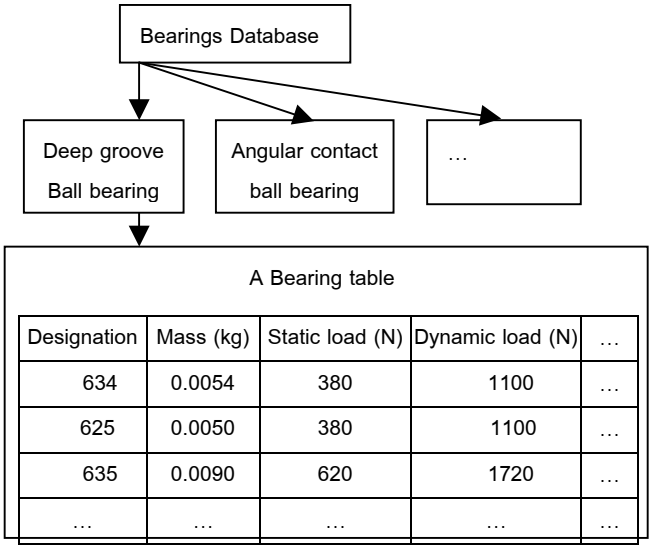
Figure 2: Main components of the program

The last is the presentation tier consisting of DataSet, DataTable, DataRow and DataGrid, one could be a subset of another, as shown in Figure 1. The key task of the presentation tier is to support the windows user interface. While DataSet is in the presentation tier, the datasource where it retrieves data from is in the data tier. OleDbDataAdapter class in the middle tier serves as a bridge between DataSet and the datasource by sending the command to retrieve the required set of data. DataTable is a subset of DataSet. DataRow is a further subset of DataTable. A set of queried results is then shown through the view command in DataGrid. DataSet, DataTable, DataRow and DataGrid are also .NET framework class libraries. Figure 2 shows the major components of the program Implementation. The main program of the development consists of a set of user interface

codes, data tier connection codes, and manipulation codes with the Microsoft® Visual C# class. The user interface codes retrieve what the user requires and then display the solution or the part found after processing at any steps of the search upon the user's requests. The data tier connection codes send the commands, connect to the server and pull the required information from the data tier. The C# manipulation codes create particular instances in order to use the class methods so that they could transform the user's requests to SQL statements as well as SQL syntax. That is the C# manipulation codes are associated with OleDbCommand in the middle tier.

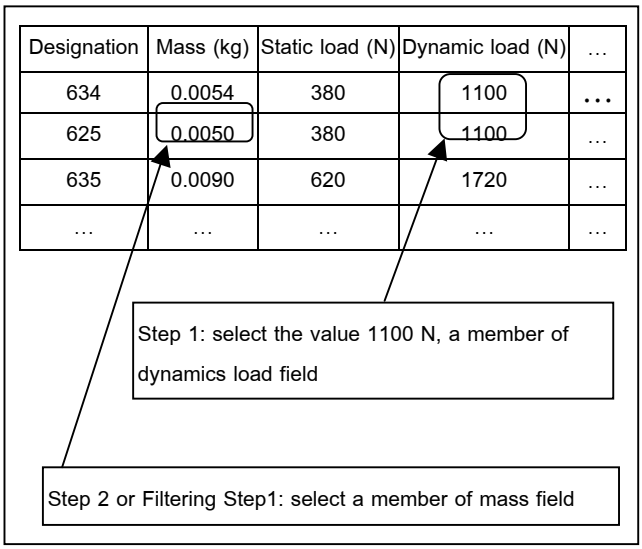
The developed system architecture is applied to bearing selection as a case study in this paper. The structure of bearing database normally consists of several types of bearings, such as deep groove ball bearings, angular contact ball bearings, spherical roller bearings, single thrust ball bearings, etc. Each type of bearings also requires numerous and different design considerations which can result in a large number of fields in the database. The structure of bearing database is shown in Figure 3. For ease of illustration, only 3 fields are presented for deep groove ball bearings. In the actual database, each type of bearings could have up to about 10 – 20 fields. Users can select any number of the required fields, with each of the fields in any order, to search for the parts.

For example, if a user selects a required dynamic load of 1100 N (step 1) and then chooses to view the results, the parts with designations 634 and 625 will both be available, see Figure 4. However, if the user would like to filter more with the required mass of 0.0050 kg (step 2) before viewing the results, only the part with designation 625 will be available. Users can view the results at any step and filter the results as much as they like to narrow the scope of feasible parts that fit their design tasks. Figure 5 summarizes the search and filter mechanism as a flowchart of instructions for the users to use the program.



Designation	Mass (kg)	Static load (N)	Dynamic load (N)	…
634	0.0054	380	1100	…
625	0.0050	380	1100	…
635	0.0090	620	1720	…
…	…	…	…	…

Figure 3: Structure of bearing database



Designation	Mass (kg)	Static load (N)	Dynamic load (N)	…
634	0.0054	380	1100	…
625	0.0050	380	1100	…
635	0.0090	620	1720	…
…	…	…	…	…

Figure 4: Identification of required parts from the search and filter mechanism

A few program interfaces during the search and retrieval operations are shown in Figure 6. Figure 6.1 consists of two listed boxes and a DataGrid. First, users have to select a field, or an attribute type, from the "Fields" listed box (on the right) representing all bearing considerations that are available in the bearing database. Once they press the "Select" button, the "Details" listed box (on the left) will show all requirements corresponding to the chosen field. After selecting a required value and the users wish to view the results, they can do so by pressing the "Result" button. The DataGrid will then display the results at this stage with all attribute types. The users can choose to view the configuration of each part by highlighting such a part in the DataGrid and then click "See Picture". The configuration of the part will then appear in a new window as shown in Figure 6.2. The users can also check the units for the requirement fields by clicking "See Unit", so that the new window detailing such units will appear as shown in Figure 6.3.

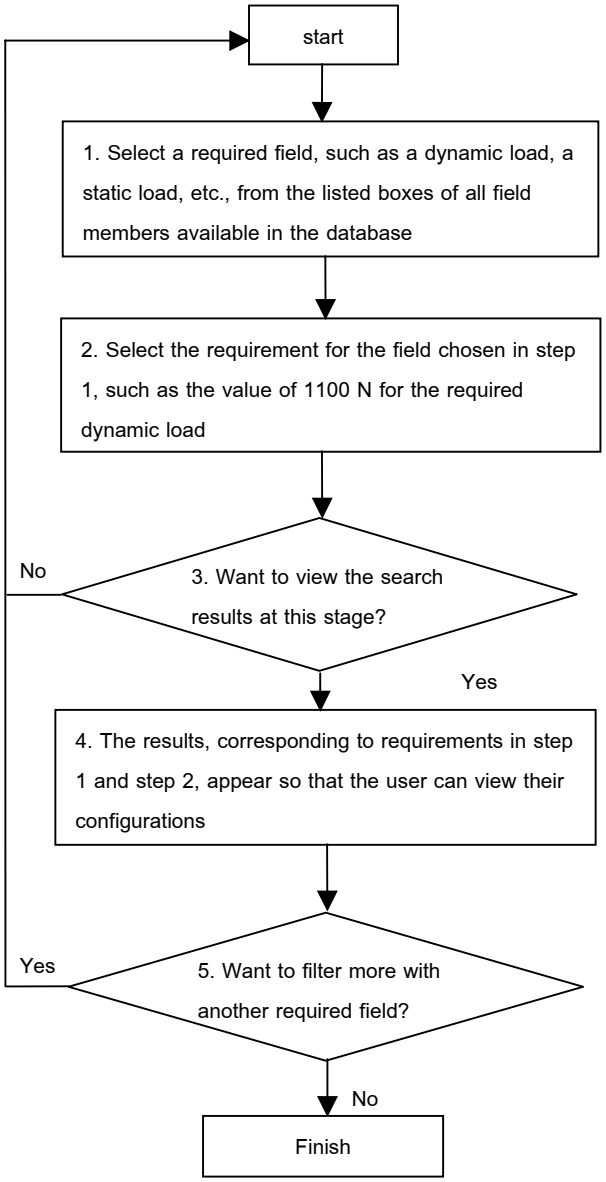


Figure 5: Flowchart of operating instructions to use the search and filter program

4. Conclusions

4.1 The proposed search technique, developed on the Microsoft® .NET framework, is particularly suitable and efficient for mechanical component selection, such as bearings, as it can properly manage a large number of fields that are normally faced in the engineering design environment.

4.2 Within the certain structure of engineering components, the proposed program can reduce time to search, comparing with the scanned paper catalogs, as the matching accuracy with one-time keyword search is low.

4.3 The program offers flexibility during the search and retrieval operations. The user can select, in any order, any number of required fields or considerations. The user can also choose to view the results at any stage of the search. This is particularly useful for designers during the trial-and-error stage of conceptual design, as the designers can see many potential alternatives and this encourages design exploration.

4.4 The program has been written with Microsoft® Visual C#, a powerful object-oriented programming language with ease of customization, so that it can be readily combined with and reused by other engineering database projects.

5. Future Work

5.1 Ongoing work to complete the selection for other mechanical components such as gears, shafts etc. are being considered so that the total framework for mechanical component search can be achieved.

5.2 The proposed technique could be directly linked to other mechanical application programs, such as UniGraphics®, AutoCAD® or others related, with the support of .NET framework. This means that the technique could play a significant role in streamlining the design activities for mechanical engineers and designers.

5.3 With the support of Common Language Runtime (CLR) from the Microsoft®.NET core technology, the implemented codes will be compatible with other programming languages from Microsoft®, such as Visual C++ and Visual Basic, and from third parties, such as Lahey's Fortran. Hence, the codes can be integrated to enhance the capabilities of many complex software written in those programming languages.

References

[1] Scanned paper and online catalog on the internet of bearings, http://www.bcibearings.com.

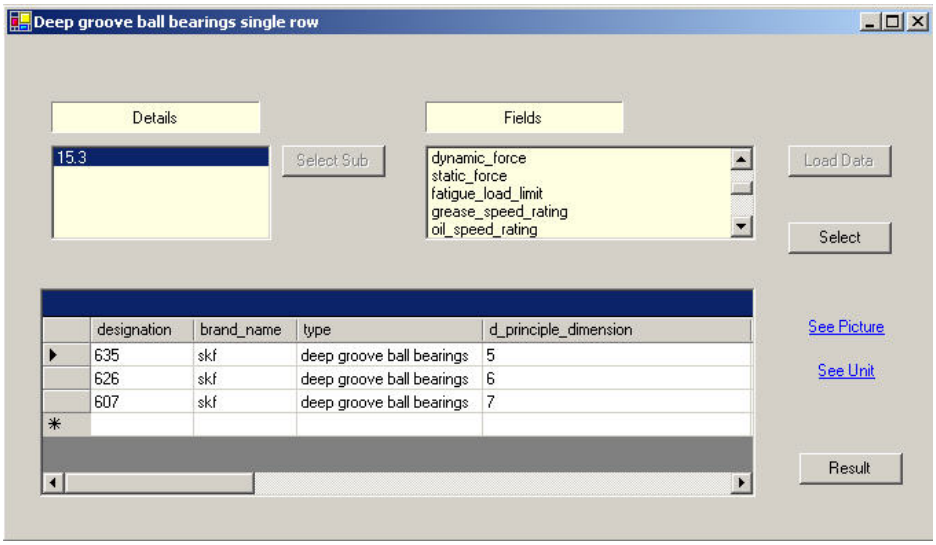
[2] Scanned paper and online catalog on the internet of bearings, http://www.xzsybearing.com/chan-pin-sj-english.htm.

[3] S. Sander, and Mogge, C., "Lösungssuche in Engineering-Netzen–Stand der Technik und Zukunftsperspektiven", In: Meerkamm, H. (Ed.), "Fertigungsgerechtes Konstruieren-Beiträge Zum 9. Symposium 1998", Universität Erlangen-Nürnberg, 1998.

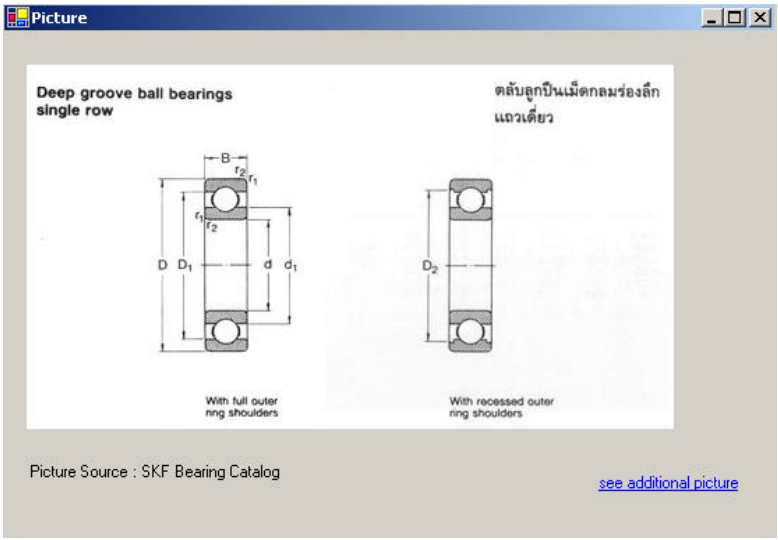
[4] R. Baeza-Yates, and Ribeiro-Neto, B., "User Interface and Visualization", Modern Information Retrieval, ACM Press, New York, Addison-Wesley, 1999, pp. 284.

[5] D. Young, and Shneiderman, B., “A graphical filter/flow model for boolean queries: An implementation and experiment”, Journal of American Society for Information Science, July 1993, Vol. 44, No. 6, pp. 327-339.

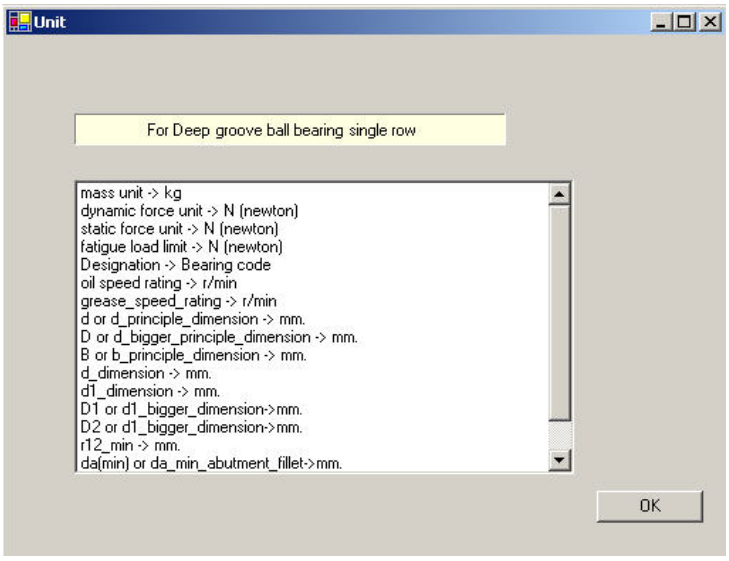
[6] Microsoft, “Visual Studio .NET Walkthrough”, Microsoft® Corporation, Chapter 2: Distributed application, 2001, pp. 36-37.



(6.1)



(6.2)



(6.3)

Figure 6: Examples of program interface