Reusable General Component in Unknown Domain

Dr. Buthainah F. AL-Dulaimi  Amer Abdulmejeed Abdurehman
Computer Department/College of Education for Women/University of Baghdad

Abstract:
Component reuse has become more important because the developer begins the process development by investigating the available library for the pr-existing components that achieve the desired functions. The paper presents the reusability via constructing general component and reuses it in unknown domain. The practical work describes the construction of the general component filter with its applicability to library, employee, and seminar application systems is introduced.
Keywords: component, reuse, model, application, system, domain

1. Introduction

A software process model is an abstract representation of a software process. Each process model represents a process from a particular perspective, and thus provides only partial information about that process. The generic models are not definitive descriptions of software processes. Rather, they are abstractions of the process that can be used to explain different approaches to software development. You can think of them as process frameworks that may be extended and adapted to create more specific software engineering processes.

a. The waterfall model This takes the fundamental process activities of specification, development, validation and evolution and represents them as separate process phases such as requirements specification, software design, implementation, testing and so on.

b. Evolutionary development This approach interleaves the activities of specification, development and validation. An initial system is rapidly developed from abstract specifications. This is then refined with customer input to produce a system that satisfies the customer's needs.
c. Component-based software engineering This approach is based on the existence of a significant number of reusable components. The system development process focuses on integrating these components into a system rather than developing them from scratch.

1.2. Component-based software engineering

In the majority of software projects, there is some software reuse. This usually happens informally when people working on the project know of designs or code which is similar to that required. They look for these, modify them as needed and incorporate them into their system. In the evolutionary approach, reuse is often essential for rapid system development. This informal reuse takes place irrespective of the development process that is used. However, in the last few years, an approach to software development called component-based software engineering (CBSE), which relies on reuse, has emerged and is becoming increasingly used.

This reuse-oriented approach relies on a large base of reusable software components and some integrating framework for these components. Sometimes, these components are systems in their own right (COTS or commercial off-the-shelf systems) that may provide specific functionality such as text formatting or numeric calculation. The generic process model for CBSE is shown in above Figure. While the initial requirements specification stage and the validation stage are comparable with other processes, the intermediate stages in a reuse-oriented process are different. These stages are:

a. Component analysis Given the requirements specification, a
search is made for components to implement that specification. Usually, there is no exact match, and the components that may be used only provide some of the functionality required.

b. Requirements modification During this stage, the requirements are analysed using information about the components that have been discovered. They are then modified to reflect the available components. Where modifications are impossible, the component analysis activity may be re-entered to search for alternative solutions.

c. System design with reuse During this phase, the framework of the system is designed or an existing framework is reused. The designers take into account the components that are reused and organise the framework to cater to this. Some new software may have to be designed if reusable components are not available.

d. Development and integration Software that cannot be externally procured is developed, and the components and COTS systems are integrated to create the new system. System integration, in this model, may be part of the development process rather than a separate activity.

Component-based software engineering has the obvious advantage of reducing the amount of software to be developed and so reducing cost and risks. It usually also leads to faster delivery of the software. One of the main targets in problem solving is to obtain a general view of the problem area. Some functions are domain-specific while some are more general but have domain-specific implementation. An example of the latter would be the achieving of the search operation. [1, 2, 3, 4, 5, 6]

2. The Proposed General Component

A one common operation is widely used by different applications is search and filtering operation. The achieving of the search operation could be expressed in generic terms that did not dictate the structure of the opening file or database in the domain. A general component filter is constructed to achieve search operation into the existing data in the environment. The application system...
library is constructed to apply the component filter without any modifications. Therefore a component filter was designed to be independent of the domain to which it is applied. The component filter performs searching and overcomes the constraints that concern structure and scope. The component filter is flexible in the sense that it can be changed to achieve additional special features. The component filter can be included in different management applications: the employee, and seminar application systems.

2.1. The Structure of Component Filter

The structure of the component filter is based on three parts defined in general descriptions to be able to read the opening file or database in the data environment and to achieve the search and filtering operation according to given criteria. The first part is an array of elements to hold names of the fields that belong to the opening file or database in the environment. The second part contains all the operators that are needed for the comparison in the condition expression upon records. The third part contains the expressions that are given by the user in order to filter data records in the environment and to be used in the condition expression. The three parts represent the primary key in the search and filtering operation, in case of the need for secondary key the logical operators that are represented by and, or operators are added to the component filter. The secondary key has the same arguments of the primary key. The case sensitive option is included in the component filter when the foreign language is used. The structure of component filter is illustrated in figure (1) and the fields of the component filter are illustrated in the form that is shown in figure (2).

![Figure 1. The Component Filter Structure](image.png)
3. The Component Filter Processes

The component filter has two selections one without condition expression, which means that no processing steps are required to formulate filter condition expression. Therefore, all data are located. The second selection contains condition expression given by user to be formulated for use in search and filtering data. The fields in the component filter form that is shown in figure (2) are checked. The procedure proceeds to the next step in the process if the fields contain valid data. The values are entered into the fields by selection from the opening list that contains the available values. The values can also be entered directly by the user without using the list selection according to user’s desire. After checking the fields, the valid fields are formulated to be the condition expression in order to be used for search and filtering data. The Skeleton of Component Filter Process is illustrated below.
#DEFINE NORECSFOUND_LOC "No records were found which meet your query. Please try another query."

DO CASE
CASE THIS.VALUE = 1 &&SET FILTER condition
    SET FILTER TO
    fexp1=ALLTRIM(thisform.cboFields1.DisplayValue)+ALLTRIM(thisform.cbooperators1.DisplayValue)+ALLTRIM(thisform.txtExpr1.Value)
    fexp2=ALLTRIM(thisform.cboFields2.DisplayValue)+ALLTRIM(thisform.cbooperators2.DisplayValue)+ALLTRIM(thisform.txtExpr2.Value)
    IF !EMPTY(ALLTRIM(thisform.cboFields2.DisplayValue)) AND ; !EMPTY(ALLTRIM(thisform.cbooperators2.DisplayValue)) AND ; !EMPTY(ALLTRIM(thisform.txtExpr2.Value))
    IF thisform.optgrpAndOr.option1.Value=1 oexp=" AND "
    ENDIF
    IF thisform.optgrpAndOr.option2.Value=1 oexp=" OR "
    ENDIF
    fexp=fexp1+oexp+fexp2
ELSE
    fexp=fexp1
ENDIF
    SET FILTER TO &fexp
LOCATE
* Check if no records found
    IF EOF() =MESSAGEBOX(NORECSFOUND_LOC," ")
    RETURN
ENDIF
CASE THIS.VALUE = 2 &&SET FILTER TO all
    SET FILTER TO
    LOCATE
ENDCASE
RELEASE THISFORM
4. The Component Filter within Applications

The application of a reusable component was tested to more than one application system such as library, seminar, and employee systems. The component filter is the representation of some aspect of a system that maybe used to develop systems increasingly based on the integration with the component filter to form complete applications. The component filter applicability will be shown on library, employee, and seminar application systems.

5. The Library System

The purpose of building the library system is to demonstrate the advantage of using the component filter in system design. The system archives the registers of the library and automates the daily work of the library. The daily work of the library includes borrowing, and returning various types of item such as books, magazines, research papers, etc. Most of the work in the library depends on search and filtering operation. The operation concerns the work of looking up for books, authors, delayed borrowers, etc. The search and filtering procedure in the system is replaced by the component filter without any modifications. The component filter is integrated within the rest components of the system, which causes building the system rapidly. The Overall Structure Chart of Library System is shown in figure (3).
5.1. The Library System Implementation

The main menu of the system is shown in figure (4).

A-The search selection invokes the component filter to be executed. The form in figure (2) is displayed for entering the condition expression to get the required record for update or displaying or delete operations. The update operation can be achieved by changing field values in the form, illustrated in figure (5).
display operation contains options next and previous for displaying records according to the current position of the file pointer as shown in figure (5). The delete operation removes the current record from the book file. The selection of delete option causes a message to be displayed to confirm the deletion.

B- Second function contains Borrow, Return, List; The selection of borrow option causes the form in figure (6) to appear and be ready to enter new borrow records. The form contains the fields related to borrow operation only.

![Figure 6. The borrow form of new records](image)

The calling occurs when entering the book number from the selection menu in book number field. The part number field is filled with the required value. After that the book name will be displayed in its position in the form since the entered book number exists in the book file. The name of the borrower, his identity number and the borrower date has to be entered. After all the information is entered the record can be saved in case of choosing borrow option. To delete the borrow record the cancel option can be chosen to remove the record and the second option in the submenu is the returning operation, which deals with the borrower who returns books or he delays to return the book.
Figure 7. The return form of new record

The first field in the form is the borrower name that is entered from the selection menu, then the rest fields contents will be displayed in the form. To confirm the right record is displayed we can choose the search option. The fields need values to be entered are the identity number and the return date. The record can be saved in case of choosing the button return else will be canceled. The cancel return button is chosen to remove the called record from the file.

C-The third choice in the submenu is for printing a list of who failed to return what he borrowed from the library. In case of selecting this option the form in figure (8) is displayed.

Figure 8. The borrow list

Third function is the Print operation. This function prints the existing data. The choice of print function causes the search form in figure (6) to appear on the screen for entering condition expression if data are to be filtered.
5. The Employee Application

The employee application is a simple application to demonstrate the ability of applying the component filter to another application without any modifications. The application archives the information of employees according to their degrees and employing date. The application contains employee file and a number of operations are accomplished on this file. The application chart is shown in figure (9).

![Figure 9. The Employee Structure Chart](image)

The figure shows the main operations involved in the application to manipulate employee file. The new activity is required to enter new employees, the display operation shows the existing employees. The selected employee can be deleted or printed. The find activity is the search operation, which is replaced by the component filter without any changes as shown in figure (10).

![Figure 10. The Application Employee and component Filter](image)

5.1. The Implementation of Employee Application
The main menu of the system is shown in figure (11).

![Figure 11. The Employee Main Form](image)

The form contains the main menu of the employee application activities and the fields of the employee file. The new button is used to enter new employees on the blank record. The next selection is the display button that is used to display records of the employee file, next the current record or previous one of the current file pointer according to the chosen button as shown in figure (13).

![Figure 13. The Employee Display Form](image)
The delete option is used to remove the selected record. The selection is achieved by choosing find button that invokes the component filter to be executed and achieve search operation among records as shown in figure (2). The print option is used to display a list of employees in the application and the format of the list is shown in figure (14).

![Print Preview](image)

<table>
<thead>
<tr>
<th>Employee ID</th>
<th>Section</th>
<th>Degree</th>
<th>Birth Date</th>
<th>Employment Date</th>
<th>School Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Training</td>
<td>PhD</td>
<td>05/09/69</td>
<td>12/12/81</td>
<td>Computer</td>
</tr>
<tr>
<td>2.0</td>
<td>Maintenance</td>
<td>MSc</td>
<td>09/10/60</td>
<td>11/11/81</td>
<td>Computer</td>
</tr>
<tr>
<td>3.0</td>
<td>Internet</td>
<td>BSc</td>
<td>01/10/72</td>
<td>02/01/86</td>
<td>Management</td>
</tr>
</tbody>
</table>

Figure 14. The Employee List

6. The Seminar Application

We want to build a model of a series of seminars. A seminar, being a process, is by its nature intangible, and often we identify or remember a process by the time it existed and the elements that were part of the process. A seminar is characterised by: 1) an opening time and a closing time; 2) a specific person being the speaker; and 3) the seminar being with a set of other seminars that are prerequisite to understanding this seminar.

A model object for a specific seminar could contain information about the opening and closing time, the speaker, and
In this system, the time and the type of elements involved in the process span a universe (time and speaker) in which processes can be said to be located. In other words, the model object can refer to the phenomena that are related to the existence and identity of the seminar, the intangible phenomenon.

The system contains two files one represents the parent file, which is the seminar file, and the child file is the reference file, their relationship is shown in figure (15), and the system structure is shown in figure (16).

Figure 15. The files Seminar, reference and their relationship

Figure 16. Top-Level Structure of Seminar System
A- Factoring Seminar File Activities: The activity level includes operations needed to work with the seminar file as illustrated in figure (17). The operations include entering new data to the seminar file. Activities print, display, and delete, are requested after the required seminar is selected. The selection occurs within the activity find.

The find activity is used for searching the required seminar from a series of seminars or seminar references. The activity can be built rapidly by using the component filter. Therefore the structure of the system is integrated with the component filer as illustrated in figure (18).
B- Factoring Reference File Activities: The activity level includes operations needed to work with the reference file as illustrated in figure (19). The operations include entering new data to the reference file after entering the data in the parent file, which is the seminar file. Activity delete is requested to remove the selected child.

![Activities](image)

**Figure 19. The Factoring Reference File Activities**

6.2. The Seminar System Implementation

The main form is shown in figure (20). The form in the figure contains the main activities in the system that manipulates the seminar file.

![Seminar Main Form](image)

**Figure 20. The Seminar Main Form**
The procedures in the system include entering new seminars in the seminar file with their associated information in the reference file after choosing their own new button. The form in figure (21) displays seminars and their references depending on the direction option selections, which are next, and previous according to the current file pointer.

Figure 21. The Seminar Display Form

The search operation is implemented after choosing the find option to locate the required seminar. The component filter is invoked. After finding the required seminar, the display and delete operations can be used. The delete removes the displayed data. The delete option exists in two positions, one for parent and another for child. In case of print option the form in figure (22) is shown. It contains seminars and the prerequisite seminars.
7. Conclusions

Component reuse does not happen by accident, system designer must plan to reuse existing components that are lacked in complex domain. Hence, reuse is to reduce costs over the entire life cycle of a system and shortens development time. Reuse component filter to build different management applications help accomplishing the systems quickly, since the support can be in both the aspect of system functionality and aspect of architecture.

References
