A CGI Based Approach for Remotely Executing a Large Program for Integration of Design and Manufacture over the Internet

Daizhong Su and Nariman Amin
Department of Mechanical and Manufacturing Engineering,
The Nottingham Trent University, Burton Street, Nottingham NG1 4BU, U.K
E-mail: daizhong.su@ntu.ac.uk nariman.amin@ntu.ac.uk,

Abstract

This paper proposes an Internet based system for geographically dispersed teams to collaborate over the Internet for the purpose of integration in design and manufacture. As one of key issues of the system, a CGI (Common Gateway Interface) based multi-user method has been developed to remotely execute a large size software package via the Internet. A gear optimisation software package has been chosen as the application area. The package implements genetic algorithms to search the best solution for gear design. The objective is to implement this in Internet and to collaborate between different locations. The package is located and run on the host server; after the completion of the program, the results are sent back to the client. To accomplish this, a combination of CGI, HTML, JavaScript, Java, multi-user environment and Internet security techniques has been utilised.

1 Introduction

The Internet technology has evolved rapidly over the last decade. It has been employed widely in business, industry, government and academia. In the engineering sector, the application of the Internet and its associated World Wide Web (WWW) is found in numerous areas including marketing, management, design, manufacturing, production, planning, customer-service, etc.

This fast growing area has attracted researchers' great attention, for example, Cheng et al [2] and Pan et al [11] proposed a Java and AI based system for the implementation of design agility and manufacturing responsiveness. Huang et al made contributions to design for X over the Internet [12]. With the European Union's support, an international team worked on a collaborative research project of a global engineering network [13]. Shackleford and Proctor developed the Java-based tools for development and diagnosis of real-time control system [14].

The advent of the Internet and WWW, has sparked new activities among researchers into the development of the systems that allow collaboration from remote sites by sharing ideas and information. This is particularly crucial for geographically dispersed engineering teams which demand for improved communication and cooperation among the teams in different locations.

The research reported in this paper proposes an Internet-based system for geographically dispersed teams to collaborate over the Internet for the purpose of integration in design and manufacture. As one of the key issues of the system, a method to remotely execute a large size software package over the Internet has been developed. It is a CGI based multi-user method, which combines the techniques of CGI, HTML, JavaScript, Java, multi-user environment and Internet security. A gear design optimisation package was chosen as vehicle for the development of the method,

In this paper, the overview of the Internet-based collaboration system and the gear optimisation software is given first, followed by detail description of the CGI based multi-user method.

2. Overview of the Internet-Based Collaboration System

2.1 The Internet-based Collaboration System

As an international collaborative project, an Internet based collaboration system has been proposed, which consists of several intelligent hybrid systems (IHS), each of which is located in an individual site, with functions of integrating design and manufacture for tasks assigned to the site; the individual systems are then integrated via the Internet's World-Wide Web, forming a global system. In this way, the capabilities of each partner of the international collaborative team are not only enhanced by the local IHS, but also enhanced by utilising other IHSs within the network. This also enables a task to be conducted agilely and effectively.

The Internet-based collaboration system is sketched in Figure 1. This system is for the collaboration in the domain of mechanical transmission and six partners from five countries are involved. Within an individual site, a site-controller controls all activities within the site and communicates with other sites via the web server through the network interfaces. Individual intelligent hybrid systems (IHS) are located inside the distributed agents, each of which is responsible for integrated design and manufacturing tasks in a specified area. The agent controller also is in charge of the use of resources (databases, equipment and software tools). The resources in an agent are accessible for other agents. The IHS is illustrated in Figure 2 and further information can be found in [10].

2.2 The CGI based Multi-user Method

One of the techniques required for developing the Internet-based collaboration system is how to remotely execute a large size software package over the Internet. A partner site within the collaborative team may have a considerable large size software package owned by the site only, and other partners may need to access it. Considering the large size of the program, as well as the copyright and security reasons, the package should not be downloaded from the host web site to the client's site. Usually this type of package may not be written in Java, and hence is not platform-independent. It also has to be considered to allow several users to run the package at the same time.

There are three techniques for developing a method to run a software program over the Internet:

- To download the program from the host web site to the client's web site. Then the program is executed in the client's site entirely. This is a relatively easy method, but there are several issues to concern: the host would lose control over the program, the client must obtain permission to have the program, the program must be platform-independent, and, for a large size program, there may be problems of download time and the client's computer space.
- To write the program in JavaScript or Java. Because JavaScript and Java are platform-independent, the program can be run over the Internet. However, JavaScript or Java may not be suitable for all applications and to convert an existing large program written in other language into JavaScript/Java may be not feasible in terms of cost and time.
- To let the program reside within the host site and the clients from their sites access the program via a CGI or a similar program which is located in the host server. This method can overcome the problems of the above two methods, but more development work is required.

To meet the requirement of the Internet based collaboration system, the last method is adopted for the research to remotely execut a program over the Internet. A gear optimisation software package [9] was chosen as a vehicle for the development. The package was originally developed using Visual Basic and C++, and with the CGI based approach, it can be run by several clients from different web sites while the package residents in the host site.

To accomplish this, a combination of HTML, CGI, JavaScript and Java programming is used. The HTML file gives all the necessary interface between the client and the server. It allows a user to insert the desired parameters into a series of text boxes. The parameters are then sent to the server which would be accessible by CGI files. When the user clicks on an appropriate button, a CGI program, which is located on the server, is initiated by the HTML code. The CGI program parses the data and invokes the gear optimisation program, which is a C++ executable program located on the server. When the execution is completed, the results are sent back to the client. This is considered to be a secure approach since the program is not downloaded to the user’s machine and the client has no access to the source code.

The development of multi-user environment is also an important issue of this research. Since the gear optimisation program uses the input/output files as temporary space to store data, conflict arises if more than one user is running the software. A method is proposed to overcome the above problem: each user has specific username and password which when permission is given, a space is allocated on the server to the user, so that all the interactions occur within that space. This method prevents the conflict between the existing users. This will be explained in more detail later in the paper.

In the following sections, a brief description of the gear optimisation program is given first, followed by the description of the method used to implement this software in Internet; then an overview of the problems is given that could arise in a multi-user environment, and a method is provided to overcome these problems. Finally, the development work regarding the Graphical User Interface (GUI) and the security are presented.

3 Gear Optimisation Software

3.1 Functions of the Software

Optimisation of gear design is an important issue for improving the performance of the machines or the systems. However, gear design to professional standards, such as ISO (International Standard Organisation), AGMA (American Gear Manufacturer Association) and British Standards, is a complicated task which makes gear design optimisation very difficult.

A gear optimisation software package OptMMGear [9] has been successfully developed by the Mechanical Design and Concurrent Engineering Research Group at The Nottingham Trent University. The package consists of three parts:

- a friendly graphical user interface (GUI)
- a genetic algorithm (GA) program and
- a numerical analysis program for gear strength calculation to BS 436.

The GUI was developed using Visual BASIC, and the other two were written in C++. The GUI is used for design data input, setting-up optimisation specifications (goals, weight factors, population size and number of tests), and display of results. The optimisation is conducted by the GA program. The numerical analysis program is invoked by the GA program in the optimisation process to calculate the tooth strength. All the three parts are fully integrated into a single software environment. The package is used to optimise the design of external spur and helical gears with involute tooth profile.

Up to nine gear design parameters can be optimised, including tooth facewidth, module, pressure angle, helix angle, rack tip radius, addendum coefficients, addendum modification (tooth profile shift) coefficients, and number of pinion teeth.

There are five optimisation objectives including minimising facewidth, minimising centre distance (variable centre distance only), reducing the difference in tooth root bending stresses between the pinion and wheel, increasing contact ratio, and reducing the difference in tooth tip sliding speed between the pinion and wheel.

Three design constraints are considered during the optimisation, including that tooth root bending stress cannot exceed the allowable stress, tooth contact stress cannot exceed the allowable stress, and sliding/rolling speed ratio cannot exceed 3.

3.2 The Optimisation Program.

The optimisation is conducted by the GA program. The key features of the GA programming, *cascade procedure* and *genome cording*, are presented in this section. For further detail, see [1].

(a) Cascade Procedure. The GA optimisation is conducted in a cascade procedure of three tiers as in figure 3. The first tier of the optimisation uses the values from the initial design to start its optimisation process, and the parameters are adjusted in the search of a global optimum. The process continues until 70% of the genome population are identical. At this point, the information encoded within the converged genome is decoded forming the solution to this tier and the intermediate 'warm' starting values for the next. The GA optimiser is initiated again with the solution from the first tiers, applying a narrower, more accurate band to the search. The increase in resolution localises the search in the region of the global maximum, enabling the accuracy of optimisation to be increased. Again, the search is repeated until the limiting percentage of the population are identical, at which point the converged genome is decoded to form the final solution.

(b) Genome Construction. The genome consists of 9 genes and the bits for each gene as shown in figure 4. The genome has been constructed in such a way that both fixed and variable centre distance gear designs can use the same genome. Parameters that are used within both designs form the first section of the genome, while the two extra genes, (x_2 and z_1), required for the variable centre distance are located at the end. This structure allows the additional parameters to be ignored when not required, i.e., the case of no-fixed centre distance. Their permanent inclusion within the genome will not affect the GA process, as they are ignored and during crossover the positions of the transferred bits are maintained, therefore never letting the contents of these genes affect the search.

3.3 Optimisation Process and Program Structure

The optimisation process performs genetic algorithm. Genetic algorithm is known to be computationally expensive. One of the main factors which affects the execution time is the population. According to the tests done by Wakelam [1], a combination of 2000 for the population size and 10 for the number of tests would give the optimum result. The user is recommended to use this set-up, however, if a quicker result is required, the user has the option of reducing these values. This however might degrade the performance. Therefore, there is a trade off between the performance and the processing time.

Figure 5 shows the structure of the optimisation software in a simplified form.

As shown in figure 5, the user is required to input via the GUI the necessary information including design specification (power to be transmitted, speed ratio, etc.) and GA parameters (population size, number of tests, etc.). These values are then written into the input files. The Visual Basic program then calls the optimisation program to perform genetic algorithm. It will first read the data from the input files and starts the genetic algorithm. After the execution is completed, the results are written into output files which are then displayed to the user.

Upon completion, the package provides the information of the optimised design as follows:

- The performance of the optimisation process can be displayed in graphical form. The search paths for the goals are provided to allow the user to ensure that the solution has repeatability.
- The final design is displayed giving its major parameters and its performance, including the improvements on the initial design and the stresses acting upon the design.

The optimisation process may be time consuming depending on the size of the population defined, the number of tests and the number of parameters selected.

4 Internet Solution

4.1 Key Issues

Platform-Independence and Security

One of the main concerns of implementing the design software in Internet is the ability of the client to run the software from any machine, whether it is UNIX, Windows, etc, from any part of the world. As mentioned in the previous section, the GA optimisation program is written in C++ and compiled in windows environment. If the user is working on the UNIX machine, downloading this software would not be compatible with its environment and it cannot be executed. Therefore the first criterion is to run the software on the server. This would also improve the

security, since the user would not have access to the program. Using this method, the speed of the program would depend on how powerful the server is, rather than the user's machine.

User Interface

As mentioned earlier, the user interface of the original software has been created using Visual Basic. Visual Basic is an object-oriented language which is used to create graphical windows based application. Since Visual Basic is platform dependent, it cannot be simply used for the Internet application. Therefore, the user interface needs to be created using an alternative language – HTML, which is platform independent and the codes are interpreted locally, i.e. on the client's machine.

HTML has a feature called *form* which makes it possible to obtain information from the user. This could be in the form of text boxes, check boxes, radio buttons, etc. After the user inserted the data and pressed the *submit* button, all the data within the form will be transferred to the CGI program located on the server. The CGI program can then perform the necessary calculation on the data and send the results back to the user.

HTML documents are static, plain-text files and their function is totally informative. They do not have the ability to perform any calculations or validations on the data submitted to the server. To make the web page more dynamic and user interactive, JavaScript could be used. JavaScript is also written as text which is added to the HTML code in a text format and are interpreted and run by the web browser whenever the user retrieves the web page and it does not need to be compiled into a program.

The main reason that JavaScript has been used in this project is to check the data before passing them to the server. JavaScript processes the form locally without contacting the server, which would reduce the time dramatically.

Common Gateway Interface (CGI)

CGI is a program located on the server and can communicate with the HTML forms. It can be written in any programming language such as Java, C/C++ and perl which can read from standard input, write to standard output, and read environment variables. The data that the CGI program receives from the form is in the form of one long string that needs to be parsed first. Subsequently the CGI program processes the data and sends the results back to the user, in the form of an HTML page.

The CGI programs that have been developed for this research have been written in C++ language.

There are two different methods of sending the data to the server: GET and POST. The GET method sends its data as command line input, i.e. as one long string assigned to an environment variable, whereas the POST method sends

all data values as one line of text to standard input device which must be read within the program [3]. The amount of data that can be retrieved from a GET is considerably smaller than that from a POST. Since the number of data needs to be obtained from the user is considerably large, it was decided to use POST to transmit the data.

4.2 The Structure

The method is based on the client/server approach. Processes that need access to data located on the server which performs some calculations, or tasks that require high performance from the server machine, belong to the server [4]. Figure 6 shows the structure of the method which would enable the user to run the GA software and obtain the results from a remote location. As shown in figure 6, the HTML and JavaScript are located and interpreted locally, i.e. on user's machine, whereas the CGI and the optimisation program are executed remotely, i.e. on the host server.

The user would insert all the required information, and presses the *Submit* button. Before passing the data to server, JavaScript would check if any of the data is wrong, i.e. out of the permitted range. If there is an error, then JavaScript would prevent transferring the data to the server and will display a message box informing the user what the error is. If the data is correct, then the values are passed onto the CGI program located on the server. An example of JavaScript that checks the value of the population size is shown below:

```
<script language = Javascript>
function validate()
{
   if (document.forms[0].population.value<0 || document.forms[0].population.value>10000 ||
   (isaNumber(document.forms[0].population.value)==false))
   {
      document.forms[0].population.focus();
      document.forms[0].population.select();
      alert("Wrong data for 'Population Size' !\n Please insert a value between 0 and 10000");
      return false;
   }
}
</script>
```

Figure 7 shows the screen dump of the program when JavaScript has detected an error in the input.

The CGI program first parses the data and after applying some calculations on the data, it will write the data into the input files. It will then call the optimisation program which is a C++ executable file. This program will then start

running on the server. This works the same way as it was explained in section 2.2. The optimisation program will read the data from the input files and starts the genetic algorithm. It will take some time, depending on the values given, before the results are out. In this situation, if for any reason the connection is lost between the client and the server, it will not create any problem, since the program is running on the server independent of the client. When the execution of the optimisation program is completed, it will write the results into the output files.

A flag has been used to check if the optimisation is completed. When the user clicks the *Result* button, the CGI program checks the value of the flag. If the execution is not completed, it will inform the user, otherwise it will display the results on user's screen.

5 Multi-User Environment

The method described in section 4 works perfectly as long as we guarantee that only one client is using the software at a particular time. We cannot and should not guarantee such a situation. One of the merits of implementing this software in Internet is the ability of multi users to utilise this software simultaneously. What problems could arise with the current method?

5.1 An example of problem using the current method

A situation that would create a problem has been shown graphically in figure 8. We assume that User 1 makes a connection with the server. The data is sent to the CGI program and the optimisation program starts running. At this moment, the connection with the server might be lost, or since the user knows that it will take some time before the execution is completed, he might decide to come back later to check the results, as indicated in figure 8a. So, the optimisation program is running on the server and it will be completed after a period of time and the results will be written into output files. Meanwhile User 2 makes a connection with the server and starts running the software. The program is completed and the results are written into output files, in another words, the results of the User 1 will be overwritten by the results of User 2, as shown in figure 8b. When User 1 returns to check the results, wrong information will be displayed on his screen.

This is a crucial point that has to be dealt with. There must be a technique to overcome this conflict.

5.2 The solution for a multi-user environment

It was decided to give each user an identity and a space on the server, so that all the interactions could occur within that allocated area. In this way, there would be no conflicts between different users. This is shown in figure 9.

The first time the users log in (http://domme.ntu.ac.uk/people/namin/personal/password.html), they are required to input their username and password. If permission is given, then the CGI program will create a directory on the server under the cgi-bin directory. The name of this directory would be the same as the username. This is where all the interactions occur for this user.

The next step is to copy all the CGI and HTML files into this directory. The structure was designed in such a way that only one copy of optimisation would be needed. This would avoid copying the optimisation program, which is considerably large, into the directory every time a user logs on. However, the path of the directory needs to be passed on to it and also to all other CGI files. This is a vital step, otherwise the optimisation program will not have any information about where it should be reading the data from and where to write the output files.

HTML *form* has the ability to pass arguments to the CGI programs. This is called the *hidden element*. This characteristic is used to pass the directory name, i.e. username, to the CGI programs. Consequently, we need to add this to the HTML form dynamically. This is done by the CGI program. It will update the HTML files and adds the hidden value to the HTML *form*. The final step would be to open the file index.html within the current directory. From this point onwards, all the interactions and reading/writing from/to files occur within this location.

When the user submits the data, the CGI program will write the data into input files within this directory and then calls the optimisation program and passes the directory name to it. The optimisation program will read the data from this directory and after the execution is completed, the results are written into output files in that directory. So, when the user presses the *Result* button, the correct data is sent to the user.

When the results are displayed to the user and the user logs off, the CGI program will delete all the files copied into this directory and finally removes the directory itself. This is to reduce the space occupied on the server to minimum.

The program has been tested using various input data and the results have been compared with the original software running on a single workstation. The results proved to be successful. An example of the optimisation results executed remotely is given in figure 10. The program has also been tested using remote stations and from different platforms, all of which have produced successful results.

6 Graphical User Interface

The process of designing a web site, even though at the first sight seems to be quite straightforward, requires a lot of considerations. It is important to know who the client is. The users of this software are design engineers. This group of clients does not look to web sites for entertainment. They want an effective information delivery and they are not interested in flashy graphics [5].

The user interface, i.e. the Web pages, was carefully designed to make the pages more user friendly and also to increase the speed of downloading by limiting the pages to only the necessary images. In addition to this, all the possible errors that could occur, were taken into consideration. The objective is that the user should never see a server error [6]. If a programming error occurs, the CGI program should trap it and send a detailed error page to the browser. This would reassure the users and refer them to the administrator.

After the program has displayed the result, the user has the option of viewing the result graphically. Java applet is used which reads the results from the files located on the server and plots the graph on the screen. The process is shown in figure 11. The flowchart illustrates how the chart class is passed three vectors. The first is needed to determine the number of generations used in the search, which is necessary to scale the horizontal axis. Two other vectors are passed, each containing data produced for each generation of gear. So, for each generation a point is drawn on the graph.

Before the charts can be drawn, all data must be scaled to the size of the graphs within the frame. This is achieved by searching through the vectors for maximum and minimum values. Once found, a suitable scale is calculated by:

Scale = (Max value – Min value) / Min value

For every generation a value is scaled against the vertical axis. However a suitable offset is needed so the convergence can clearly be seen as the focus point of the graphs. This offset is calculated as:

Constant = Scale * 4

The new data values can now be generated:

New data = (old data * scale) + constant

This data is used to produce the graphs using the graphics class of the Abstract Windowing Toolkit (AWT).

The optimisation process searching for a optimum value of facewidth and centre distance has been displayed in the graphs shown in figure 12. Peaks represent the beginning of a test and convergence can be seen when the graph levels to a single value.

7 Security Concerns

Recently there have been concerns about the security of using CGI programs[7]. CGI program allows the client to call external programs on the server, as in this project where it invokes the C++ executable program which performs genetic algorithm. If the hackers manage to access the system, they could execute a CGI program that would delete or modify the files on the server.

One method of minimising the security risk is to limit access to the site from Internet by designing a firewall. An Internet firewall is a security mechanism that separates the internal network from the global network and filters the information that are passing through, refer to figure 13 [7].

It has also been recommended to use a compiled language such as C++ rather than the interpreted language like Perl for writing CGI programs [8]. With the compiled language, the intruder could access the binary form but not the source code, whereas with the interpreted language, the source code is readily available for the intruder.

8 Discussion and conclusion

The research reported in this paper proposed an Internet-based system for geographically dispersed teams to collaborate over the Internet for the purpose of integration in design and manufacture. As one of key issues of the system, a CGI based multi-user method has been developed to remotely execute a large size software package over the Internet.

A structure based on the combination of CGI, HTML, JavaScript and Java programming has been developed. The problem with multi-users utilising the software simultaneously has also been discussed and a solution was provided.

It was mentioned that the optimisation process is very much time consuming. The program is located on the NT server which has a relatively good processing power. Nevertheless, in times of high demand, vital processing may be taken away from the server and it might slow down the server dramatically. This problem is going to be addressed in the next stage of the research.

The method that has been described in this paper accomplishes the objective of this research. However, this is not the only method currently available. One available option is using Servlet instead of CGI. Servlets are Java classes loaded into and invoked by a Web server [15]. Servlet is found to be faster than CGI, the main reason is the fact that Servlet is loaded only once and the connection remains alive until the administrator stops it. Whereas every time a connection is requested to a CGI, the server must load the CGI. A research is currently being carried out to implement the gear optimisation using Java Servlet instead of CGI, and comparison of the two methods will be conducted.

The paper also discussed the issue of security problems and reducing the risk to minimum. It has to be taken into consideration that once we give permission to external users to access our web site, there will always be a security risk. There will always be hackers who would strive to break into the system. Therefore, the decision of using the Internet to collaborate with remote clients, depends mainly on the balance between the advantages of implementing the Internet and the security risk. Nowadays most companies are tending to choose the former option.

Acknowledgment

The authors thank Dr Wakelam for his contribution of the GA package development and Mr T Cooke for his contribution of Java programming, which were related to this research.

References

1. M. Wakelam, “Intelligent Hybrid Approach for Integrated Design.”, Ph.D. thesis, The Nottingham Trent University, UK, October 1998.
2. K. Cheng, D.K. Harrison and P.Y. Pan, “Implementation of Agile Manufacturing – An AI and Internet Based Approach”, proceedings, 13th International Conference on Computer-Aided Production Engineering, Warsaw, June 1997, pp. 273-278.
3. Stevens, “Kicking and Scripting: JavaScript and CGI”, Dr. Dobb’s Journal, April 1997, pp. 92-114.
4. Pierce, “Combining Java and CGI Scripts, Communicating between Client and Server”, Web Techniques, Vol. 2, Issue 9, September 1997, pp. 49-52.
5. P. Shikli, “Designing Winning Web sites for engineers”, Machine Design, November 6, 1997, pp. 30-40.
6. M. Kuchling, “A CGI Framework in Python”, Web Techniques, Vol. 3, Issue 2, February 1998, pp. 43-6.
7. C. Kou and F. Springstell, “The Security Mechanism in The World Wide Web (WWW) and the Common Gateway Interface (CGI) – Example of Central Police University Entrance”, Proceedings, IEEE 31st Annual 1997 International Carnham Conference on Security Technology, 1997, pp. 114-119.
8. “The World Wide Web Security FAQ”, http://www.w3.org/Security/Faq/wwwsf4.html.
9. D Su, M Wakelame, K S Henthorn and K Jambunathan, “A software package for evolutionary optimisation of gear design”, in: Evolutionary Design by Computers -- CD-ROM, P Bentley (ed), Academic Press, 1999.
10. D Su and M Wakelam, “Intelligent hybrid system for integration in design and manufacture”, Journal of Material Processing Technology, Vol 76 (1998), Elsevier Science S A, pp 23-28.
11. P Y Pan, K Cheng and D Harrison, "Java-based Systems: an Engineering Approach to the Implementation of Design Agility and Manufacturing Responsiveness", proceedings, 14th International Conference on Computer-Aided Production Engineering, Durham, April 1999, pp. 86-93.
12. G Huang, J Shi and K L Mak, "Internet-Based Design for X", proceedings, 14th International Conference on Computer-Aided Production Engineering, Durham, April 1997.
13. J. Gausemeier, et al., "Global Engineering Network - weltweiter Informationsverbund zur Staerkung der Innovationskraft in Produktentwicklungsprozessen", VDI-Berichte 1302, 1996, pp. 39-52.

14. W. Shackleford and F. Proctor, "Java-based Tools for Development and Diagnosis of Real-time Control System", Proceedings, 1998 ASME Design Engineering Technical Conferences, Paper No. DETC/CIE-5530.
15. P. Clip, “Servlets: CGI the Java Way”, Byte, May 1998, pp. 55-56.

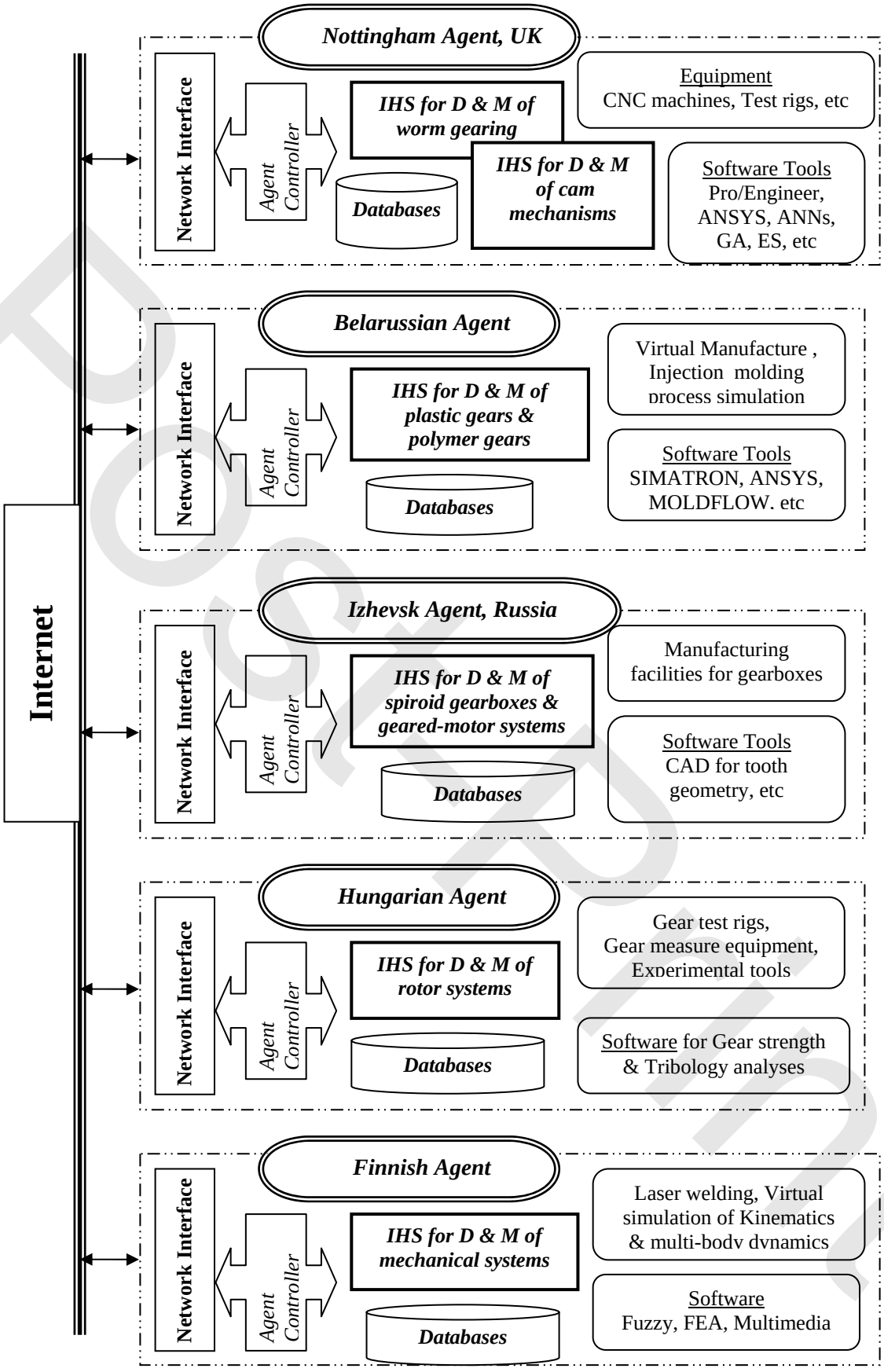

Figure 1. Internet-based collaboration system

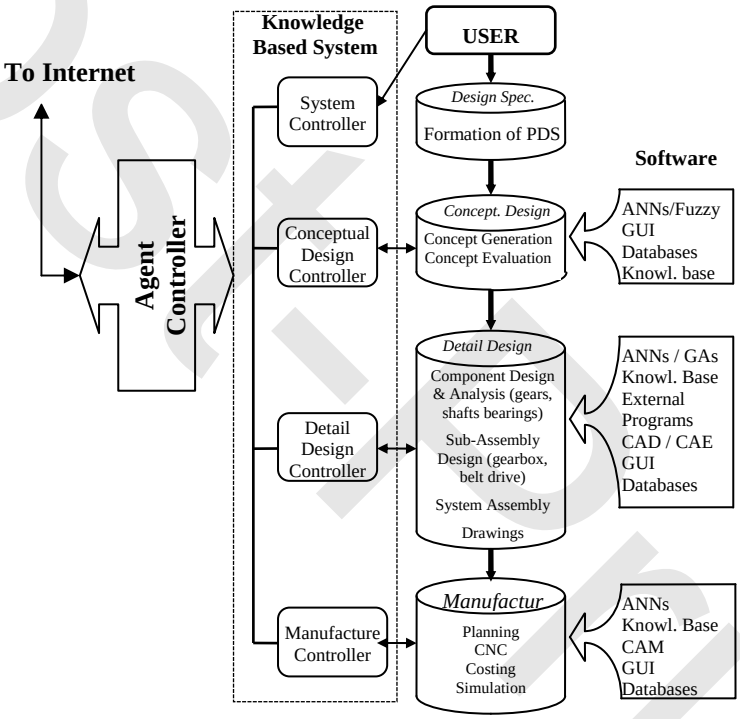

Figure 2. Intelligent hybrid system

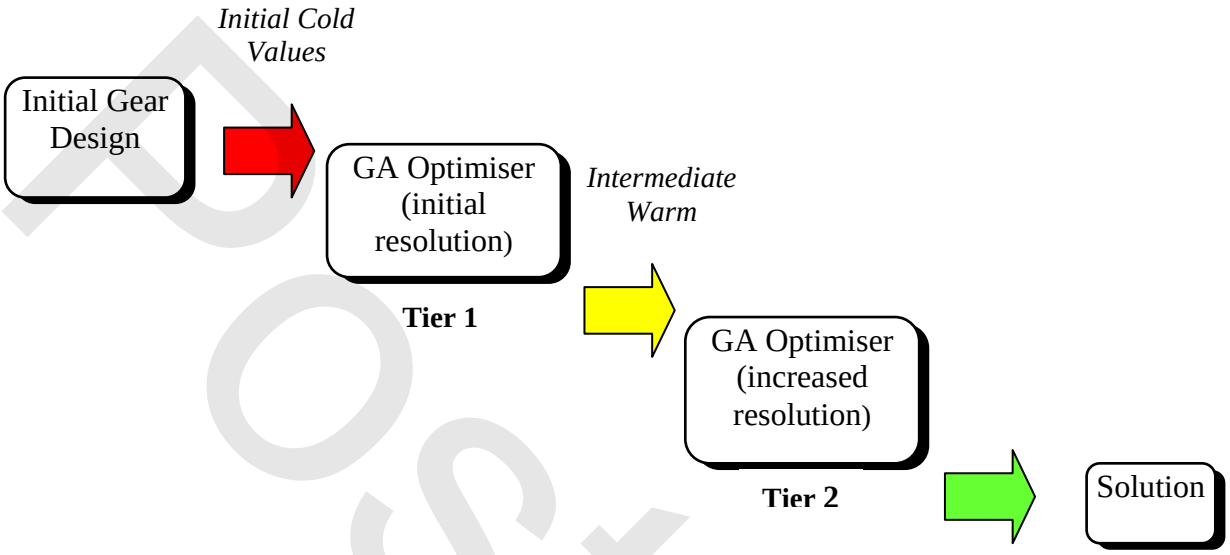

Figure 3. The cascade procedure of GA optimisation

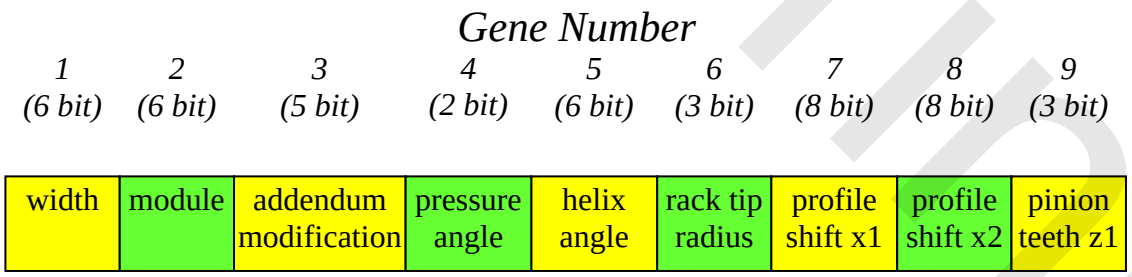

Figure 4. Genome combination

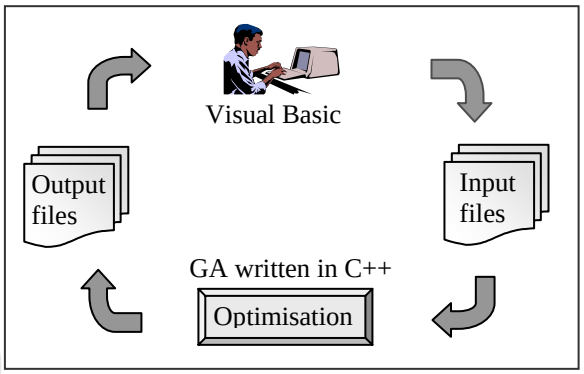

Figure 5. Structure of the original gear optimisation software

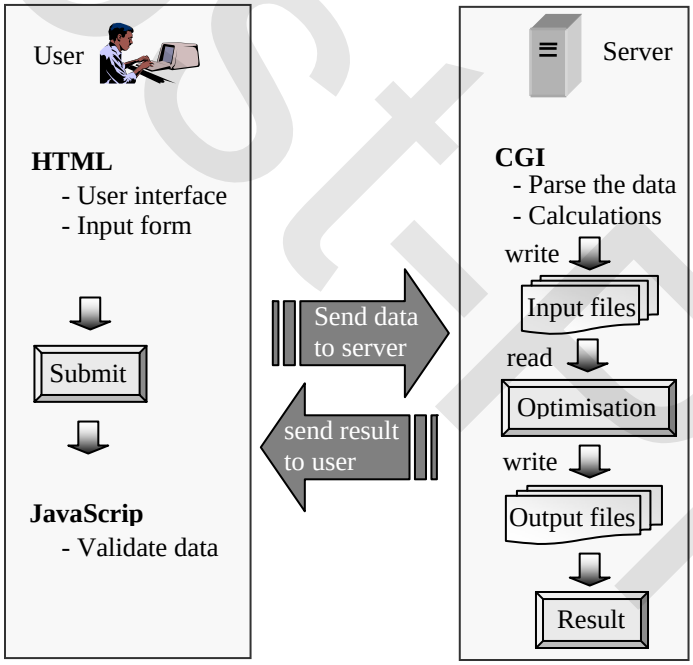

Figure 6. Structure of the Proposed Method

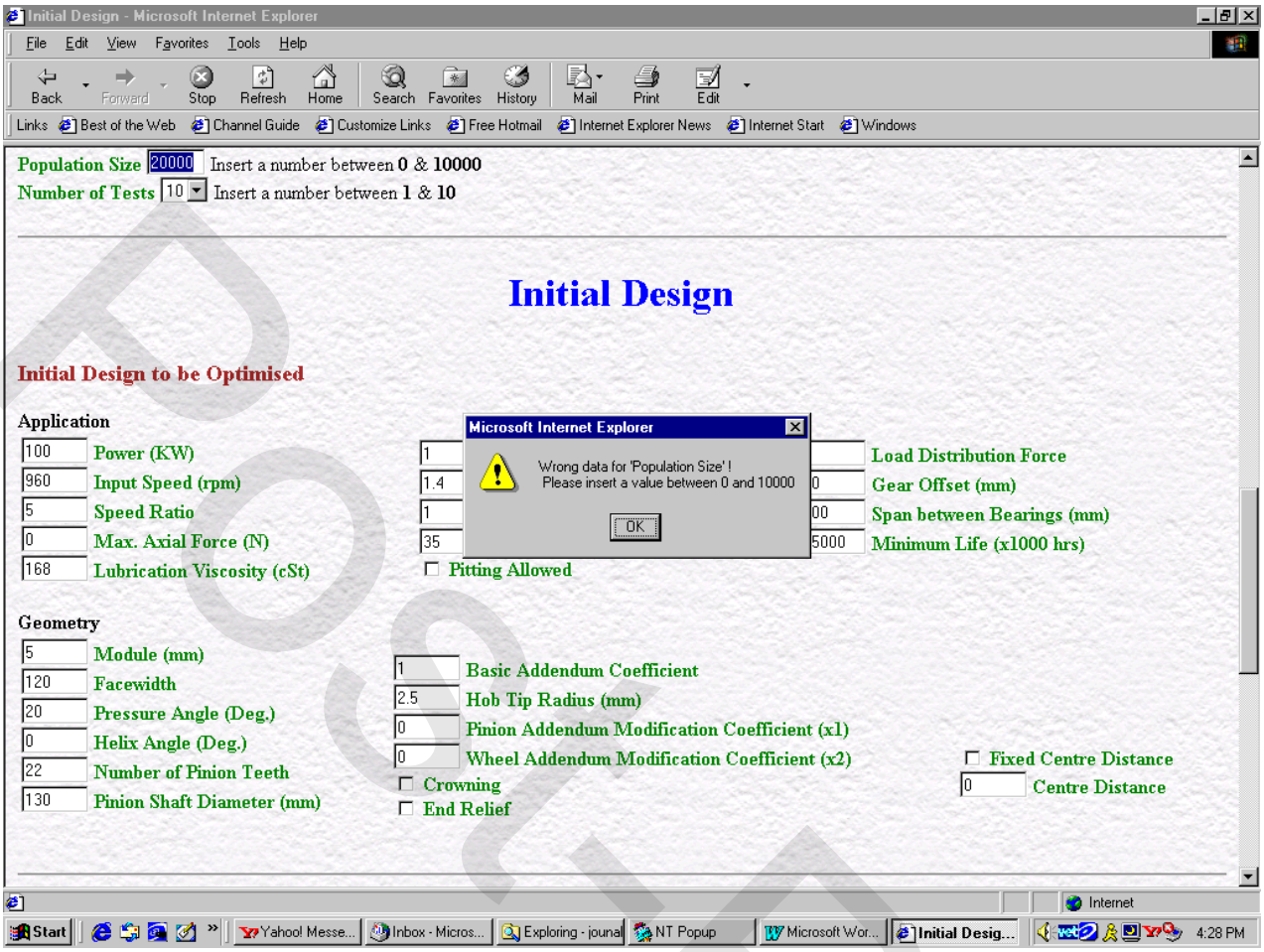

Figure 7. JavaScript message box informing an error in the input

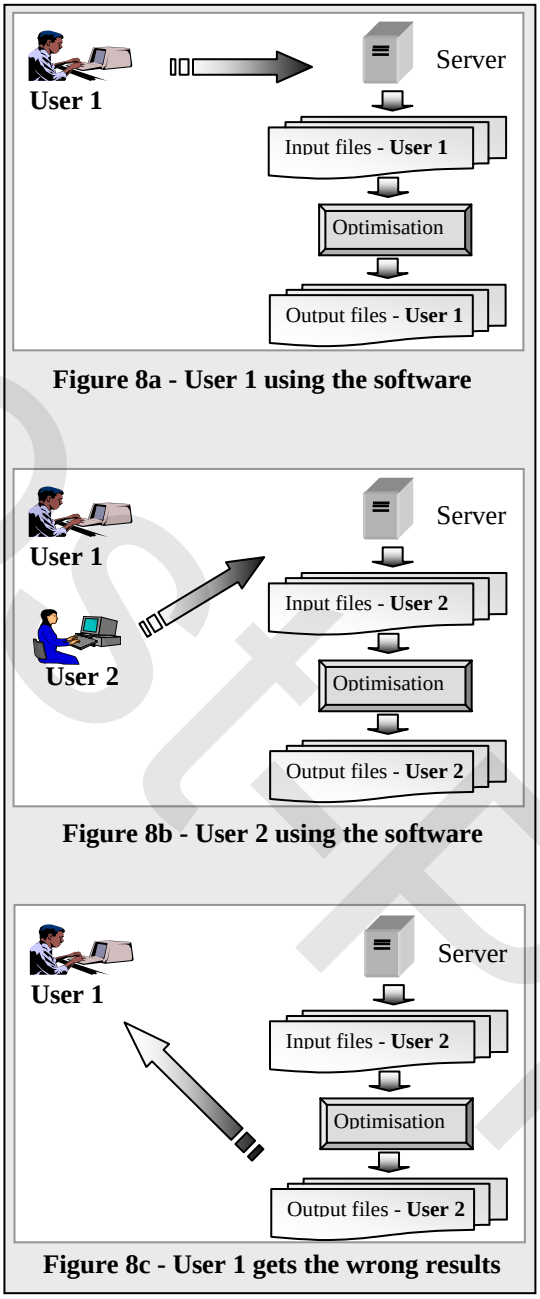

Figure 8. An example of a problem in a multi-user situation

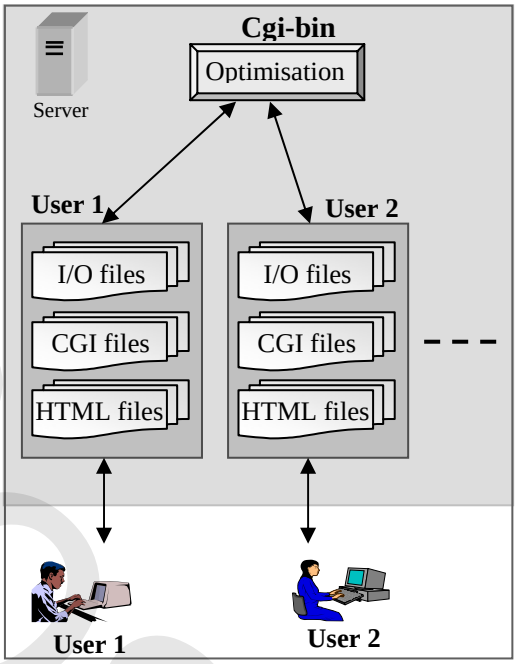

Figure 9. The proposed structure for a multi-user environment

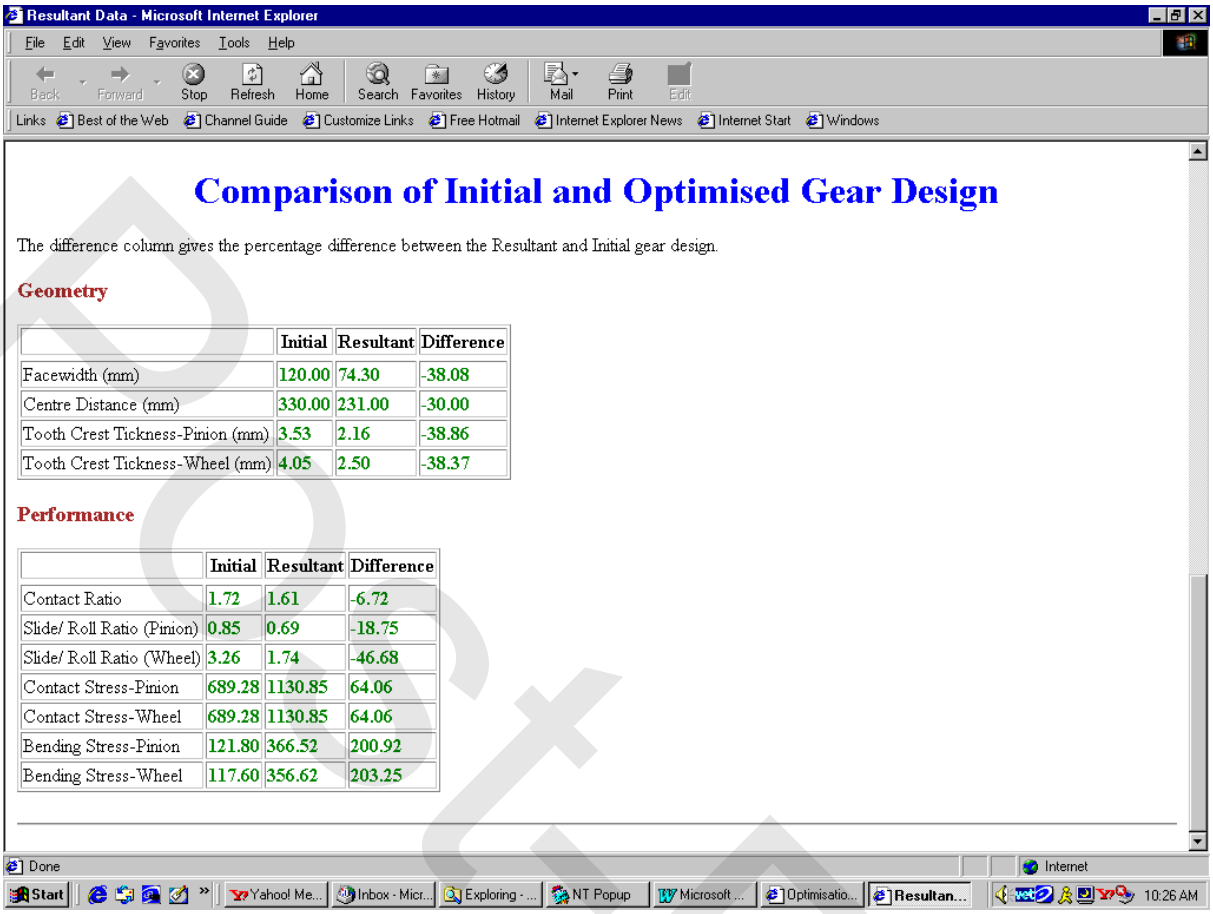

Figure 10. Results of the optimisation program displayed on the client's machine

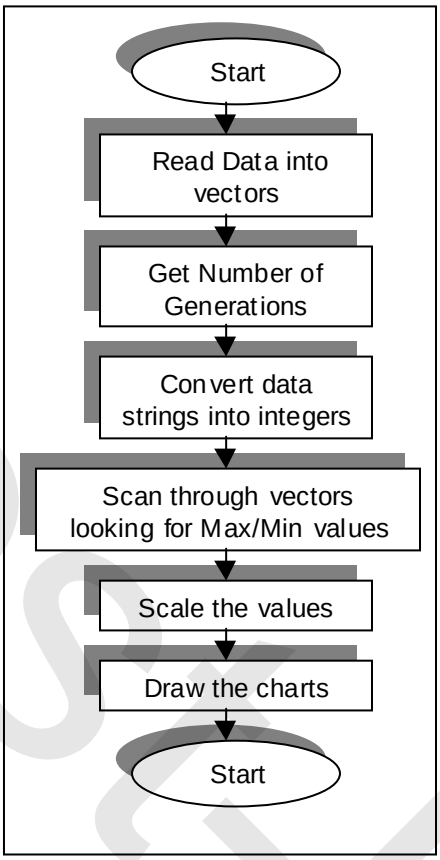

Figure 11. Optimisation charts flowchart

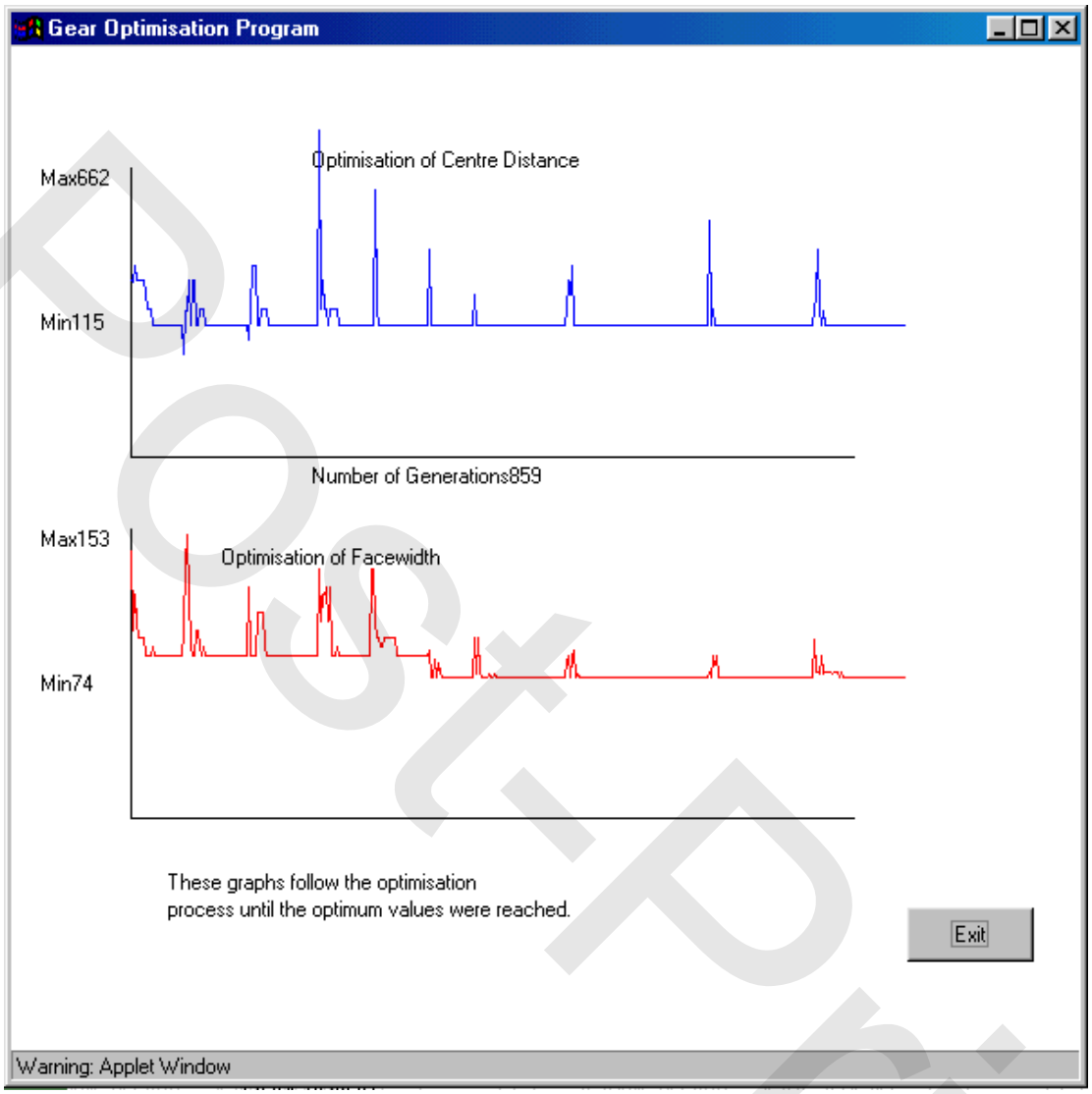

Figure 12. Java applet displaying the result graphically

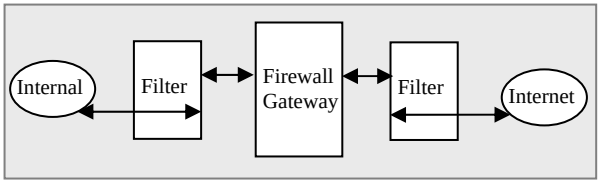

Figure 13. Firewall for Internet