A multi-objective genetic algorithm for software personnel staffing for HCIM solutions

Enrique Jiménez-Domingo,
Computer Science Department, Universidad Carlos III de Madrid,
Av. Universidad 30, Leganés, 28911, Madrid, Spain
enrique.jimenez@uc3m.es

Ricardo Colomo-Palacios,
Faculty of Computer Sciences, Østfold University College,
B R A Veien 4, 1783 Halden, Norway
ricardo.colomo-palacios@hiof.no

Juan Miguel Gómez-Berbís
Computer Science Department, Universidad Carlos III de Madrid,
Av. Universidad 30, Leganés, 28911, Madrid, Spain
juanmiguel.gomez@uc3m.es

Abstract

The pervasive potential of artificial intelligence techniques in business scenarios has gained momentum recently through the combination of traditional software engineering disciplines and cutting-edge computer science research areas such as neural networks or genetic algorithms. Following this approach, MORGANA is a platform to perform competence oriented personnel staffing in software projects by means of a multi-objective genetic algorithm. The system is designed to be part of global human and intellectual capital management solutions. The main goal of MORGANA is to assist software project managers, by providing a comprehensible artificial intelligence-based formal framework to optimize efficiency and improve person-role fit.

Keywords: Genetic Algorithm; software personnel, person-role fit, artificial intelligence.

1. Introduction

One of the crucial tasks in project management resides in efficient and expedient formation of project teams. A failure in the formation of project teams can manifest in a decrease of service quality, unattainable project deadlines (with expected financial penalties), and loss of credibility (Strnad & Guid, 2010). In the software engineering research field, personnel and staffing are crucial aspects for the success of a project (Casado-Lumbreras et al., 2009; Colomo-Palacios et al., 2014; Colomo-Palacios et al., 2013; Colomo-Palacios et al., 2010; González-Carrasco et al., 2012). Task scheduling and resource allocation in software development projects constitute extremely hard problems and they represent two of the principal challenges of software project management due to its sheer complexity (Chang, Christensen, & Zhang, 2001). Properly assigning people to development roles is crucial for creating productive software development teams (Acuña & Juristo, 2004). On the other hand, wrong assignments may result in significant loss of value due to understaffing, under-qualification or over-qualification of assigned personnel, and high turnover of poorly matched workers (Naveh, Richter, Altshuler, Gresh, & Connors, 2007). Software managers typically assign people according to their experience, heuristic knowledge, subjective perception, and instinct (Acuña, Juristo, & Moreno, 2006). Doing this without the help of proper computer-enabled tools is tedious, repetitive and time-consuming (Naveh et al., 2007). Furthermore, common project management tools such are not effective at managing optimization problems.

However, Genetic Algorithms (GAs) are flexible and accurate addressing software project management issues, and represent an important tool for project management automation (Alba & Francisco Chicano, 2007). To the best of author's knowledge there are some efforts devoted to combine the power of genetic algorithms with project staffing, but these efforts have not taken into account aspects crucial to the formation of teams, such as work dynamics and interpersonal issues.

In this scenario, many solutions from psychology to artificial intelligence have arisen. MORGANA (Genetic Algorithm Staffing for Software Engineering Projects) is a hybrid proposal that combines psychology and artificial intelligence. The proposal pools, on the one hand, the competence paradigm and, on the other hand, the use of GAs to build a proper solution based on state of the art technologies and paradigms. The main idea is to build a system that will be able to cover both personnel characters and abilities and professional skills to create adapted work teams that ensure an adequate performance and achievement of the goals. In addition, the system has two different versions: one that allows to obtain the best general team and other in which is possible to establish the requirements for a predefined project in order to accommodate the best team for developing concrete tasks inside the project. The two versions and other characteristics of the system will be detailed throughout the document.

Finally, the system is designed to be part of global solutions for the management and development of human and intellectual capital (HICM) in the cloud as presented in (Colomo-Palacios, Fernandes, Sabbagh, & de Amescua Seco, 2012; Colomo-Palacios, Fernandes, Soto-Acosta, & Sabbagh, 2011). This fact leads to special features and non-functional requirements that the system must address in order to be suitable for this kind of environments.

2. Background

Developing effective selection approaches represents a pivotal task for managers and companies alike (Chien & Chen, 2008). As a result of this pressure, the application of decision support systems on personnel selection and recruitment has been increasing in the last years. Thus, expert systems are shown to have matured to the point of offering real benefits in many of their applications including, among other aspects, personnel issues (Garcia-Crespo et al., 2009). In this scenario, GAs have been pointed out as accurate solutions for project management issues (Alba & Francisco Chicano, 2007).

A Genetic Algorithm (GA, for short) is an artificial intelligence procedure introduced in the 1970s by Holland based on the theory of natural selection and evolution. GAs are adaptive metaheuristic search methods premised on the evolutionary ideas of natural selection by employing a population of individuals undergoing a selection in the presence of genetic operators. GAs do not guarantee the discovery of the global optimum, but in many cases a near-optimal solution is often acceptable. A GA is essentially an optimization process that allows obtaining accurate solutions in problems of different nature. The algorithm makes an exploration of the greatest solutions search space generating a random initial population of individuals which later get through the processes of reproduction, crossover and mutation (Mitchell, 1998). Each individual represents a possible solution for the problem and it is composed for chromosomes or genes. Due to non-acceptable solutions generation, some genes are divided and mutated to modify them in order to obtain more propitious individuals or solutions. Each individual is evaluated using the fitness function that provides a value of how good the individual is and, consequently, the solution. The individuals are ordered and depending on this order they will go through different phases prior to obtaining a better solution on each execution, losing the worst individuals and improving the population level. GAs have been widely used for staffing: general personnel assignment (Toroslu & Arslanoglu, 2007), general project scheduling (Kılıç, Ulusoy, & Şerifoğlu, 2008), multi-project scheduling (Gonçalves, Mendes, & Resende, 2008), training assignment (Juang, Lin, & Kao, 2007), selection (Chien & Chen, 2008) or team formation and staffing (Celik, Er, & Topcu, 2009; Strnad & Guid, 2010; Wi, Oh, Mun, & Jung, 2009) citing some of the most recent and relevant issues. GAs are also present in software engineering projects issues in aspects like project scheduling (Alba & Francisco Chicano, 2007; Chang et al., 2001; Chang, Jiang, Di, Zhu, & Ge, 2008), effort estimation (Singh, Kaur, Bhatia, & Sangwan, 2010) and temporal prediction (Amoui, Salehie, & Tahvildari, 2009). Even project staffing and team formation has been tackled with GAs (Silva, Motta, Santoro, & Oliveira, 2009). But in spite of all of these, there are not systems covering both sides of personnel management in the team formation context, taking into account human factors such as character and personal behaviour and also professional skills. Another difference with the existent systems is the fact that MORGANA allows the user to choose what option is more adequate for each

concrete case and also to decide the requirements that a team should fulfil to find the best possible solution. These are enough reasons to consider the system presented in this work.

3. System Core

In this section different aspects of MORGANA are depicted, raising the problems found, the design decisions and other significant information to contribute a better understanding of the system properties.

3.1. *Human resources in software development teams*

As stated before, the problem of human resources management is not trivial. It requires a deep and detailed analysis of each of the variables to be taken into account. In order to do so, it was necessary to carry out a state of the art study focused more in business aspects than in technology. This study provided a more appropriate knowledge for the environment in which we are working.

Once this process was finished and we acquired a more detailed knowledge about the problem and matters involved, several development options were studied in order to achieve the objectives raised since the beginning to build a system able to create work teams adapted to the requirements established by the user. After that processes of acquisition and analysis, the personnel capabilities were divided in two groups (Colomo-Palacios et al., 2013):

- General abilities: Those inherent to technological or specific knowledge.
- Professional skills: Those linked to previously acquired knowledge through the study or experience.

At this point, it was important to establish the different abilities contained in the system that will define each person for his/her subsequent evaluation inside a team. Another aspect that raised attention is the fact that each ability has a relationship with the rest of the abilities, and this fact had to be controlled and managed by the system itself. Thus, it was possible to establish what abilities should appear with a higher value in the members inside a team and what of them should be lower or avoided in order to achieve a better result for the group.

Following (Acuña et al., 2006; Acuña & Juristo, 2004), more significant general competences were selected and introduced in the system in order to achieve an important level of description that allows the system to obtain remarkable results, conclusions of the studies performed and adequate recommended solutions for the users of the system. The general competences presented in the system are:

- Analysis
- Decision making
- Independence
- Innovation/creativity
- Judgment
- Tenacity
- Stress tolerance
- Self-organization
- Risk management
- Environmental knowledge
- Discipline
- Environmental orientation
- Customer service
- Negotiating skills
- Empathy
- Sociability
- Teamwork/cooperation
- Co-worker evaluation
- Group leadership
- Planning and organization

These competences capture the most important aspects in software environments and allow a complete representation of the requirements that can be entailed for developing task in this environment. Dealing with professional competences and using the information provided in (Shackelford et al., 2006), the subset of competences selected to be incorporated to the system are described below:

- Programming Fundamentals
- Computer Architecture and Organization
- Operating Systems
- Net Centric
- Platform Technologies
- Human-Computer Interaction
- Intelligent Systems (AI)
- Information Management (DB)
- Information Systems Development
- Analysis of Business Requirements
- Analysis of Technical Requirements
- Software Modelling and Analysis
- Software Design
- Software Verification and Validation
- Software Evolution (Maintenance)
- Software Process
- Software Quality
- Distributed Systems
- Security: Implementation and Management
- Systems Integration

Yet another key conceptual element of the system are roles. Since inadequate assignment of personnel and problems among project team members are identified as two of the main human factor related difficulties affecting software project success (Nelson, 2007), a person's adjustment to a role is also crucial, because usually, roles inherit competences in many organizations.

As previous specialized studies have underlined (Acuña, Juristo, & Moreno, 2006; Colomo-Palacios, Tovar-Caro, García-Crespo, & Gómez-Berbís, 2010), different roles can be considered when working in software engineering projects. The ones used in the system are Team leader, Quality manager, Requirements engineer, Designer, Programmer, Maintenance and support specialist, Tester and Configuration manager.

The choice of these capabilities comes from the aim to capture the widest range possible, allowing us to represent in a faithful manner all the aspects to be taken into account when treating with people and, specially, when forming teams. This is a very complex task, thus it is important to have an additional aid to perform it in an efficient way. The results obtained after the experimentation process provided us significant information about how to manage work teams and relationships established between persons and their abilities.

3.2. *Human resources in software development teams*

In this section, we will present the design solution adopted for the problem. This solution offers viability and positive performance to the system. To achieve that, the appropriate techniques have been used, having in mind general requirements such as adaptability, adequacy and reusability. The aim of the system is to form work teams maximizing the performance of the group through their members' abilities interactions, taking into account personal and professional factors. In this section different design decisions taken will be detailed. First of all, it was necessary to establish a division between the different components of the system. This division was made thinking in the best use, reuse and next modifications.

3.2.1. Individual

An individual encapsulates all operations related to the creation and manipulation of the possible solutions provided by the system. Each individual is composed of members' numbers chosen by the user to compose the team. Each person on the system has an identifier that allows locating him univocally to know and manage his abilities. Hence, the identifier is only required in the individual definition, what simplifies the rest of operations that must be performed during the execution of the GA.

3.2.2. Fitness functions

The fitness function is the most important part of a GA. The result of this function execution provides a value of how good is a solution and determines the evolution of the population. After the study of different possibilities in the fitness function development, we finally decided implementing one fitness function for each member's ability. This is due to a better management of the results provided, making possible the analysis of each ability in the team separately and treating interactions and relationships of each ability in the team in a different and more appropriate manner. Each fitness function follows different objectives and provides a value adapted to the desired proximity value, allowing the GA to evolve favorably.

3.2.3. Genetic Algorithm

The cycle followed by the GA is the one showed in Figure 1:

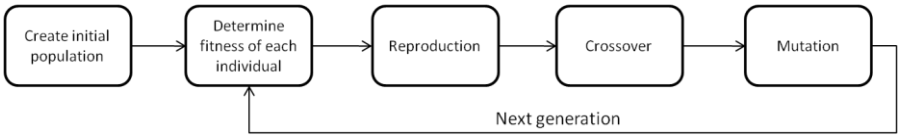

Fig. 1. GA life cycle

As shown previously, the structure of our GA follows the classical model using genetic operators of reproduction, crossover and mutation. The different steps of the GA will be detailed in the following paragraphs.

3.2.3.1. Create initial population

The first step in a GA is to create the initial population. This is a random process that provides a population of individuals that generally does not allow obtaining precise results. In this problem, the initial population is created by assigning people using their identifiers randomly to complete the necessary members on a team (number previously indicated by the user of the system). During this process the repetition of members in the same team is controlled, thus, it is not possible that the same person appears twice in the initial population.

Once the length of the population is adequate and we have the necessary individuals, all of them forming a team, is time to go through the next phase of the GA.

3.2.3.2. Determine the fitness of each individual

Now it is time to execute fitness functions. This is the most important step of a GA because an appropriate fitness function development will determine the accuracy of the solution.

As it has been explained previously, a different fitness function will be applied to each member's abilities. This decision increases development complexity, but it improves control over the relationships between abilities and it ultimately establishes the most suitable values that a work team should obtain for each of them. Despite this differentiation between abilities, some of them had the same behaviour and it was not necessary to develop a different fitness function in these concrete cases.

In order to calculate the fitness function it needs to be established the most suitable value for each ability to be included in a perfect team. Then, a classification of values suitability is established and depending on the value obtained for a

concrete team, a weighted grade is provided for the team regarding this ability (between 0 and 10). Some abilities require higher values for a team, while others present preferable lower values or precise moderate results.
The fitness value for the individual comes from the addition of the weighted values obtained from each fitness function. This provides a clear idea of how good is the team formed and its suitability to the established requirements.

3.2.3.3. Reproduction

The method implemented for the reproduction is the tournament method. The tournament selection method is adopted here to decide what chromosomes should survive from one generation to the next. This method works selecting randomly a fixed number on individuals and comparing its fitness values to choose the one with higher fitness. The chromosomes that survive will compose an intermediate population that will be used to perform crossover and mutation operations.

3.2.3.4. Crossover

The aim of crossover is to mix the genetic code in each chromosome to obtain better individuals than in the previous generations. For MORGANA, the crossover works taking two consecutives chromosomes and performing the type of crossover desired. This is to ensure that all individuals are crossed. Two types of crossovers are included in the system:

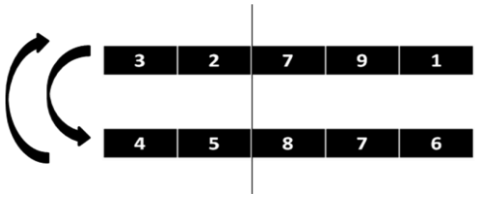

Fig. 2. Simple crossover

Simple crossover. It exchanges the middle of genetic code of both individuals. Establish a cut-off point in the middle of the chromosome and exchange the same part of it between both individuals as it can be seen in the figure 2.
Multipoint crossover. It exchanges the genetic code existing between two random points in the chromosome. Two points are selected randomly and the region inside these two points is exchanged between both individuals. Figure 3 shows the process:

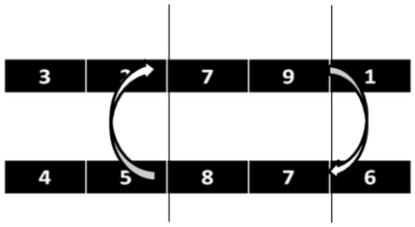

Fig. 3. Multipoint crossover

The newly crossed chromosomes are then combined with the rest of the chromosomes to generate a new population whose individuals will undergo mutation operations.
A situation that emerges with crossover operations is the high possibility of introducing the same person more than once in the same team. This was solved checking it prior to calculating the fitness value of the team and adding only once the values of this repeated member. This solution favours the discard of individuals with repeated members. A lower number of members in a team usually leads to lower productivity, but sometimes, if the members of the team work good enough, it is possible to achieve good results with less resources.

3.2.3.5. Mutation

The mutation operation is used to ensure the genetic variability by randomly changing some genes of the chromosome, allowing the appearance of individuals with new features and avoiding the population stagnation.

It is performed, first of all, by calculating the number of positions in the whole population. After that, and using the previously indicated percentage of mutation, a random number is obtained for each position in the population. If this randomly obtained number is lower than the percentage value, the member identifier is exchanged for other member randomly obtained. If the number is higher, the member stays in the team.
This classical mutation performance brings again the state of possible repeated members in the teams. As it can be seen in the previous case, if mutation causes this situation, the system will proceed taking only once the repeated member when calculating the fitness value.

3.3. *The MORGANA approach*

In this section all GA aspects and approaches besides the different fitness functions included and other elements that allow the correct performance of the system will be detailed. The aim of a GA is to provide a group of individuals which, from an initial population, improves previous generations until those who represent work teams adapted to user desires are obtained. The developed algorithm generates results that can be monitored in every moment to compare the results obtained from different tests and the system evolves following the next steps:

- The fitness value of each individual is initialized to 0 in every execution.
- The fitness value is calculated by the addition of the suitable functions.
- The individuals of the population are ordered to be able to know which of them are better.
- The tournament method is applied to the population individuals until the population's size is complete. Note that individuals for the tournament method are selected randomly, thus, some of them may be selected more than once whereas others will never be. To avoid the loss of the best individual, the elitism technique has been implemented. This way, the best individual of each generation is passed to the next one without the modifications caused by the genetic operators phase.
- The desired type of crossover operation (simple or multipoint) is executed in two consecutive individuals of the population. It ensures that all individuals will be crossed.
- The mutation operation is applied to ensure the genetic variability and avoid the stagnation. The population percentage that is mutated will be introduced by the user using the interface.
- The last step is to calculate the fitness value and reorder the population.

This process will be repeated as many times as generations are executed obtaining as a result more accurate and appropriate individuals to the user preferences each time. The basic performance of the system and the GA in which it is based has been detailed. Now it is time to introduce the different approaches implemented and the fitness functions associated with them.
The system presents two versions clearly differentiated. The first one is a side which aims to form a well balanced team taking into account all the abilities that are present in the system. The second version creates the teams depending on the user preferences or concrete project requirements, trying to adapt teams the most to these demands.
Both versions and the different fitness functions used to achieve the objectives of the systems are detailed below.

3.3.1. Well balanced team creation system

This version of the system is focused on forming the best global team, in other words, to design a team to maximize all the abilities, not focusing in some specific ones. Moreover, the system will prioritize those individuals that have considerable knowledge in most of the subjects instead of experts in a concrete area or task.
It is well known that the fitness function is the fundamental part of every GA, since its evolution will go in a different sense depending on them. The development of this function is the most complicated part in this kind of systems due to the number of variables that have to be taken into account to achieve the creation of suitable teams. For this case, we decided that the best option was to develop as many fitness functions as abilities a person could have. Therefore, it is straightforward to obtain a value weighted between 0 and 10 and allow knowing easily the score of a member and, eventually, of a team.

Furthermore, for this version of the system, an adjusted addition of every fitness function will be performed. The result of this operation will be expressed like a percentage and this value will represent the team adaptability level to the requirements previously established.

3.3.2. Adapted team creation system

The adapted team version of the system is in charge of providing work teams that fulfil certain requirements established by the user. These requirements are introduced in the system using the interface and represent concrete project requirements.

The difference between the previous version and this one resides in the GA focused on finding members that fulfil the aforementioned requirements, leaving aside other abilities. For this reason, the system tends to find experts in specific subjects instead of people who have moderate knowledge in different areas.

The fitness function performance is very similar to the previous one, but now the adjusted addition will be performed giving higher weight to abilities required in the preceding step, providing only little importance to the rest of the abilities. This method ensures obtaining more adequate members to address the task they will be assigned to.

As happened in the other version of the system, the score of each ability goes from 0 to 10, to obtain, after that, its value as a percentage.

3.3.3. Fitness functions

After the analysis of different alternatives, we decided to develop one fitness function for each ability present in the system in order to establish a better management of the preferences, relationships and scores between a person's capabilities. Each capability has a different behaviour and this should be captured by the system. In spite of this, some abilities have a similar behaviour and can be grouped, avoiding redundancies. It is important to remember that all values are weighted in order to identify precisely the level of each one. All the fitness values are calculated using the following formula:

$$F = \sum_{n=1}^{c} \left(\frac{\sum_{n=1}^{i} (C_v * f(C_v))}{i} \right) \quad (1)$$

In Eq (1), "i" represents the number of persons in each team, "c" is the number of capabilities taken into account in the system (listed in section 3.1), Cv is the value of one concrete capability in the member and f(Cv) is the transformation function applied. This transformation function will vary depending on each capability due to a different behaviour of them, i.e., for one concrete capability higher values will be better for a team configuration (Teamwork/cooperation) and for others lower or moderate values will be better (Leadership). These variations will be taken into account when calculating the fitness value.

It is easy to see the way Eq. (1) works and how the performance of the group is directly influenced by the value of each person's capabilities and by the transformation function applied. This function is different depending on the type of fitness associated to it. The groups in which all the fitness functions have been divided and the way they work are widely explained below. It is important to clarify that the values taken as frontiers in the functions have been chosen studying the more accurate values for each competence and with experts' recommendation. These experts will participate in the evaluation process that is explained in section 4. In addition, these values that these experts have considered the most suitable ones can be easily modified if the user considers it necessary.

3.3.3.1. Positive Fitness

The abilities that are included in this kind of fitness functions are those in which the ability level of each member is higher, so the value of the group increases. Examples of these abilities are "group work", "analysis", "design", etc. With a high value of "group work" in the members of a team it is more probable that they may work well together.

Moreover, the function in this type of fitness will provide a higher general value when adding considerable high score capabilities results by modifying the transformation function in the way that is represented below:

(2)

$$f(C_v) = \begin{cases} 1, & C_v \geq 7 \\ 0.7, & 7 > C_v \geq 4 \\ 0.4\,, & 4 > C_v \geq 0 \end{cases}$$

Also, depending on each capability, the score provided or the range of values can change between them, thus, different capabilities with the same kind of fitness can obtain different values at the end.

3.3.3.2. Moderate Fitness

This type of fitness functions includes those abilities beneficial for the group where members have a moderate level of them. E.g. "leadership" when more than one individual with elevated higher level of leadership exists, conflicts are more likely to occur, but a lower value of leadership in all the members of a team may unleash in a situation of stagnation or difficult decision making. The perfect situation will be the involvement of some people in the team with higher values that could manage the group or, in other case; a well-balanced group where all the members have an acceptable level of leadership and the addition of its levels create a good environment inside the team. For these reasons moderate group values are more valuable, establishing penalizations when the value reaches extremes. The function applied here works in the same way as the previous one. However, it prioritizes moderate group values so intermediate values will generate the highest scores.

(3)

$$f(C_v) = \begin{cases} 0.3, & C_v \geq 8 \\ 0.7, & 8 > C_v \geq 6 \\ 1, & 6 > C_v \geq 4 \\ 0.7, & 4 > C_v \geq 2 \\ 0.3, & 2 > C_v \geq 0 \end{cases}$$

3.3.3.3. Negative Fitness

This represents the opposite case of the positive fitness function. It is applied to abilities in which a lower value generates benefits for the group. In this group we find the "cost" of adding each member.

The function in this case works providing a better score to lower values of the group regarding a concrete capability.

(4)

$$f(C_v) = \begin{cases} 0.4, & C_v \geq 7 \\ 0.7, & 7 > C_v \geq 4 \\ 1, & 4 > C_v \geq 0 \end{cases}$$

3.3.3.4. Mood Fitness

This fitness is different from the ones explained before. As it has been previously commented, the mood is a concept that represents the capacity of each worker to work with the rest of the workers inside the company, establishing a one to one relationship between them determined by a numeric value. In the system we assume that all the people in the company have worked with the rest, presenting a more complex and complete situation. MORGANA is able to compare the values of the mood between all the members allocated in a team, providing higher values when the mood level is elevated.

(5)

$$f(C_v) = \begin{cases} 1, & C_v \geq 8 \\ 0.8, & 8 > C_v \geq 6 \\ 0.6, & 6 > C_v \geq 4 \\ 0.4, & 4 > C_v \geq 2 \\ 0.2, & 2 > C_v \geq 0 \end{cases}$$

The function applied in this fitness works in a similar way than the previous cases. It compares the mood values between the members of a team and, as a result, a number is obtained. The higher the number, the higher the score.

3.3.3.5. Role Fitness

Roles are a concept always present in enterprise environments. Each worker has a role assigned that determines competences level and the costs associated to them. Thanks to MORGANA, it is possible to manage the cost of assigning people to projects. Now is when the multi-objective approach merges. We are assuming that all workers sharing the same role bring the same cost associated. So now the objective of the system is to minimize the cost of the team without losing competences values. This situation may derive in members' promotion to higher roles or even vice versa if anyone is able to perform the tasks required by the company, creating a wide space of solutions in which a high amount of situations may be addressed.

The function, in this case, will assign more value to persons with lower costs associated. However, a margin different from the one explained for the negative fitness will be used. The values assigned to these costs will change depending on the role. Nevertheless, in some cases, a person capability will be more important than his/her role. This situation can save money to the company and provide an opportunity for a person to promote to a better position.

4. Evaluation

4.1. *Research design*

With the aim of obtaining results and feedback about the work performed, an evaluation has been carried out. This evaluation has checked all the different aspects concerning all features of the system. In order to do so, real competence information about people was necessary. These data was gathered together from students of two different subjects in their last year of the Computer Science degree program in the Carlos III University of Madrid. The students adopted one of the roles defined for software projects as defined in section 3.1. Once both subjects were finished, the students carried out a 360º evaluation and the results provided information about competences and mood. The tests performed were divided into two phases:

(1) Test 1. The aim of these set of tests was to obtain only one team. This team could be achieved using two versions:

- (a) Search the best general team
- (b) Search the team that fits the most with the requirements desired

(2) Test 2. The aim of these set of tests was to assign all the persons in the system (56 in this case) to a concrete team. This assignation could be performed following the next two objectives:

- (a) The teams generated were balanced, this means, the resultant teams had approximately the same configuration in terms of competences.
- (b) The teams generated were ranked. The first team would have the best configuration possible, the second would be the second better and the last would have the team with lower competences value or the less adapted to the requirements.

In addition, for each of the tests performed, four configurations were adopted to study the different results obtained:

(I) Using mood values for the persons in the system

(II) Using role fit. That meant that each member could only be assigned to the role previously established for him.

(III) Not using the prearranged role and the mood values. This allowed assigning people to roles based only on their competences and not considering their mood values.

The teams formed, using all the possible configurations, were composed of 8 members. For evaluating the performance of the working team selection process, precision and Mean Reciprocal Rank (MRR) measures were used. Precision can be seen as the fraction of retrieved documents that are relevant to the search. Mean Reciprocal Rank measures show the different aspects of how the workers are selected for the teams. MRR was later introduced to be able to measure the differences between several results in a system or queries. In order to compare results with a given standard, a set of experts would analyze groups in order to categorize them according the requirements described in Tests 1 & 2.

4.2. *Sample*

The sample of subjects was composed of students in their last year of the Computer Science degree program at Carlos III University. These students evaluated their competences and their partners using the 360º evaluation method after the development of the courses "Software Engineering III" and "Corporative Information Systems Design". This data will be used as input for MORGANA, allowing the configuration of working teams. The sample was composed of 56 students, 7 women (12.5%) and 49 men (87.5%), with an average age of 26.2. Although this population might not completely reflect future users, most studies in the literature have used academics to provide queries and judge the relevance (Jason Morrison, 2008).
The sample of experts was composed of 4 experts in this scope, 3 men and 1 woman, all of them researchers from Carlos III University, Spain. The average age was 34.2. These experts evaluated the results obtained from 360º evaluation method performed for the students to form working teams based on the competences and mood values showed there.

4.3. *Results*

Once the information about competences and mood were captured using the methods explained in previous sections, a set of experts studied the values obtained and made their own team configuration based on the requirements previously established using the DELPHI method. After that, the results were compared with the output of the system using different methods and standards.
The Delphi Method consists in a separate analysis of the problem made by all the experts involved in the evaluation. Each of them provides a solution and after that, all the experts have to reach a consensus and provide a final solution for the problem presented.
For this study, a set of experts were selected to perform the task. Each expert was asked to provide a team configuration depending on the kind of test performed as explained before and having all the information about people's competences. After that, these experts will discuss its opinions and configurations to provide a final and joint decision of the best team configuration in each case. These final decisions will be compared with the ones provided by MORGANA. Table 1 shows the results of all possible configurations in the system and their precision and MRR marks compared with Delphi results.
The results shown in Table 1 are the average of the results provided by 20 experiments by each test configuration. This setup was designed in order to be as precise as possible and study a more realistic set of solutions within GA.
As Table 1 figures yield, results presented by MORGANA are promising, given that there are no significance differences with the results provided by experts. However, in some cases, MORGANA provides more accurate results with respect to experts output. These differences occur due to two main causes. Firstly, the emotional and personal factors present in experts judgment. Secondly, MORGANA fitness functions behave in particular direction. Next section provides a discussion about results.

4.4. *Discussion*

Human resource allocation, or staffing, is an important issue to be analysed when software development is undertaken as a value-driven business; therefore, a proper usage of every available resource in a software project is very important (Barreto, Barros, & Werner, 2008). Thanks to MORGANA all staffing aspects are covered and it is possible to provide accurate solutions to general or concrete problems and tasks in team building as can be seen in the results showed above.
Results obtained are very satisfactory. Some differences have emerged comparing results. This situation is normal due to the different preferences established by the experts and the influence that exert their personal opinion.
Dealing with this situation it is important to mention that MORGANA is able to isolate from this kind of preferences, centering its efforts in obtaining the most effective team configuration in each case. However, it is also true that results show that the fitness function does not produce perfect results and, thus, requires improvement.
As aforementioned, precision and MRR have been the methods for evaluating the results provided by the system. Precision and MRR measures reflect different aspects of team formation performance. These measures were used before to measure genetic algorithms based systems (Torres et al., 2009), but there are not significant works in staffing or team

formation to establish a boundary in order to compare the results. It is also difficult to compare MORGANA with other software with similar features due to the lack of works in this field of study. There are recent and relevant works related to the formation of software development projects teams e.g. (Ampuero, Baldoquín de la Peña, & Castillo, 2010), but these works are mainly based on the application of patterns, and the comparison with MORGANA is not possible.

Table 1. Results of evaluation tests performed

Test	Type	Subtype	Precision	MRR
1	A	I	7/8 = 0.875	NA
1	A	II	6/8 = 0.750	NA
1	A	III	7/8= 0.875	NA
1	B	I	7/8= 0.875	NA
1	B	II	7/8= 0.875	NA
1	B	III	7/8= 0.875	NA
2	A	I	44/56 = 0.785	0.869
2	A	II	50/56 = 0.892	0.927
2	A	III	35/56 = 0.625	0.780
2	B	I	49/56 = 0.875	0.934
2	B	II	44/56 = 0.785	0.892
2	B	III	40/56 = 0.714	0.846

However, one aspect of the system must be enhanced. Thus, the performance can be improved by refining the fitness functions or creating some more in order to cover more aspects of each capability.

5. Conclusions and future work

Leveraging the potential of Artificial Intelligence technologies in Business Information Systems application domains has been deemed as a critical challenge for a particular number of disciplines that interrelate intensively in optimizing and harnessing synergies in both domains. Nevertheless, a number of research lines in projects encompassed by EU common initiatives among public and private partnerships have found significant interdependencies and positive outcomes in both approaches.

In this paper, authors presented MORGANA, a platform that combines AI-oriented multi-objective genetic algorithms for software development projects, focusing on competence oriented personnel management optimization. MORGANA aims at unifying part of the enabling knowledge-based software project management forthcomings by, as it is shown in the evaluation, finding an optimal team creation strategy.

Future works can be undertaken twofold. On the one hand, potential additional research issues could be the assessment of those teams under particular “over the edge” conditions. On the other hand, based on the particular results of the aforementioned research extension, improving the algorithm with a more accurate fitness function and evaluate the prospective use of neural networks or other evolutionary computing techniques.

References

Acuña, S. T., & Juristo, N. (2004). Assigning people to roles in software projects. *Software: Practice and Experience*, *34*(7), 675–696. doi:10.1002/spe.586

Acuña, S. T., Juristo, N., & Moreno, A. M. (2006). Emphasizing human capabilities in software development. *IEEE Software*, *23*(2), 94–101. doi:10.1109/MS.2006.47

Alba, E., & Francisco Chicano, J. (2007). Software project management with GAs. *Information Sciences*, *177*(11), 2380–2401. doi:10.1016/j.ins.2006.12.020

Amoui, M., Salehie, M., & Tahvildari, L. (2009). Temporal software change prediction using neural networks. *International Journal of Software Engineering and Knowledge Engineering*, *19*(07), 995–1014.

Ampuero, M. A., Baldoquín de la Peña, M. G., & Castillo, S. T. A. (2010). Identification of Patterns for the Formation of Software Development Projects Teams. *International Journal of Human Capital and Information Technology Professionals*, *1*(3), 69–80. doi:10.4018/jhcitp.2010070105

Barreto, A., Barros, M. de O., & Werner, C. M. L. (2008). Staffing a software project: A constraint satisfaction and optimization-based approach. *Computers & Operations Research*, *35*(10), 3073–3089. doi:10.1016/j.cor.2007.01.010

Casado-Lumbreras, C., Colomo-Palacios, R., Gómez-Berbís, J.M., & García-Crespo, A. (2009). Mentoring programmes: a study of the Spanish software industry. *International Journal of Learning and Intellectual Capital*, *6*(3), 293-302.

Celik, M., Er, I. D., & Topcu, Y. I. (2009). Computer-based systematic execution model on human resources management in maritime transportation industry: The case of master selection for embarking on board merchant ships. *Expert Systems with Applications*, *36*(2, Part 1), 1048–1060. doi:10.1016/j.eswa.2007.11.004

Chang, C. K., Christensen, M. J., & Zhang, T. (2001). Genetic Algorithms for Project Management. *Annals of Software Engineering*, *11*(1), 107–139. doi:10.1023/A:1012543203763

Chang, C. K., Jiang, H., Di, Y., Zhu, D., & Ge, Y. (2008). Time-line based model for software project scheduling with genetic algorithms. *Information and Software Technology*, *50*(11), 1142–1154. doi:10.1016/j.infsof.2008.03.002

Chien, C.-F., & Chen, L.-F. (2008). Data mining to improve personnel selection and enhance human capital: A case study in high-technology industry. *Expert Systems with Applications*, *34*(1), 280–290. doi:10.1016/j.eswa.2006.09.003

Colomo-Palacios, R., Casado-Lumbreras, C., Soto-Acosta, P., García-Peñalvo, F.J., & Tovar-Caro, E. (2014). Project managers in global software development teams: a study of the effects on productivity and performance. *Software Quality Journal*, *22*(1), 3-19.

Colomo-Palacios, R., Casado-Lumbreras, C., Soto-Acosta, P., García-Peñalvo, F. J., & Tovar-Caro, E. (2013). Competence gaps in software personnel: A multi-organizational study. *Computers in Human Behavior*, *29*(2), 456–461. doi:10.1016/j.chb.2012.04.021

Colomo-Palacios, R., Fernandes, E., Sabbagh, M., & de Amescua Seco, A. (2012). Human and Intellectual Capital Management in the Cloud: Software Vendor Perspective. *Journal of Universal Computer Science*, *18*(11), 1544–1557.

Colomo-Palacios, R., Fernandes, E., Soto-Acosta, P., & Sabbagh, M. (2011). Software product evolution for Intellectual Capital Management: The case of Meta4 PeopleNet. *International Journal of Information Management*, *31*(4), 395 – 399. doi:10.1016/j.ijinfomgt.2011.04.001

Colomo-Palacios, R., Tovar-Caro, E., García-Crespo, Á., & Gómez-Berbís, J. M. (2010). Identifying technical competences of IT Professionals: the case of software engineers. *International Journal of Human Capital and Information Technology Professionals*, *1*(1), 31–43.

Garcia-Crespo, A., Colomo-Palacios, R., Gomez-Berbis, J. M., & Mencke, M. (2009). BMR: benchmarking metrics recommender for personnel issues in software development projects. *International Journal of Computational Intelligence Systems*, *2*(3), 257–267.

Gonçalves, J. F., Mendes, J. J. M., & Resende, M. G. C. (2008). A genetic algorithm for the resource constrained multi-project scheduling problem. *European Journal of Operational Research*, *189*(3), 1171–1190. doi:10.1016/j.ejor.2006.06.074

González-Carrasco, I., Colomo-Palacios, R., López-Cuadrado, J.L., & García-Peñalvo, F.J. (2012). SEffEst: Effort estimation in software projects using fuzzy logic and neural networks. *International Journal of Computational Intelligence Systems*, *5*(4), 679-699.

Jason Morrison, P. (2008). Tagging and searching: Search retrieval effectiveness of folksonomies on the World Wide Web. *Information Processing & Management*, *44*(4), 1562–1579. doi:10.1016/j.ipm.2007.12.010

Juang, Y.-S., Lin, S.-S., & Kao, H.-P. (2007). An adaptive scheduling system with genetic algorithms for arranging employee training programs. *Expert Systems with Applications*, *33*(3), 642–651. doi:10.1016/j.eswa.2006.06.010

Kılıç, M., Ulusoy, G., & Şerifoğlu, F. S. (2008). A bi-objective genetic algorithm approach to risk mitigation in project scheduling. *International Journal of Production Economics*, *112*(1), 202–216. doi:10.1016/j.ijpe.2006.08.027

Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

Naveh, Y., Richter, Y., Altshuler, Y., Gresh, D. L., & Connors, D. P. (2007). Workforce optimization: Identification and assignment of professional workers using constraint programming. *IBM Journal of Research and Development*, *51*(3.4), 263–279. doi:10.1147/rd.513.0263

Nelson, R. R. (2007). IT Project Management: Infamous Failures, Classic Mistakes, and Best Practices. *MIS Quarterly Executive*, *6*(2).

Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., Lunt, B. (2006). Computing Curricula 2005: The Overview Report. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education* (pp. 456–457). New York, NY, USA: ACM. doi:10.1145/1121341.1121482

Silva, F. E. de O. da, Motta, C. L. R., Santoro, F. M., & Oliveira, C. E. T. de. (2009). A Social Matching Approach to Support Team Configuration. In L. Carriço, N. Baloian, & B. Fonseca (Eds.), *Groupware: Design, Implementation, and Use* (pp. 49–64). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-04216-4_5

Singh, Y., Kaur, A., Bhatia, P. K., & Sangwan, O. (2010). Predicting software development effort using artificial neural network. *International Journal of Software Engineering and Knowledge Engineering*, *20*(03), 367–375. doi:10.1142/S0218194010004761

Strnad, D., & Guid, N. (2010). A fuzzy-genetic decision support system for project team formation. *Applied Soft Computing*, *10*(4), 1178–1187. doi:10.1016/j.asoc.2009.08.032

Toroslu, I. H., & Arslanoglu, Y. (2007). Genetic algorithm for the personnel assignment problem with multiple objectives. *Information Sciences*, *177*(3), 787–803. doi:10.1016/j.ins.2006.07.032

Torres, R. da S., Falcão, A. X., Gonçalves, M. A., Papa, J. P., Zhang, B., Fan, W., & Fox, E. A. (2009). A genetic programming framework for content-based image retrieval. *Pattern Recognition*, *42*(2), 283–292. doi:10.1016/j.patcog.2008.04.010

Wi, H., Oh, S., Mun, J., & Jung, M. (2009). A team formation model based on knowledge and collaboration. *Expert Systems with Applications*, *36*(5), 9121–9134. doi:10.1016/j.eswa.2008.12.031

BIOGRAPHIES

Enrique Jiménez-Domingo is a Researcher and Innovation Consultant in a private I+D Researching Company. He holds a BSc, an MSc and a PhD in Computer Science from Universidad Carlos III de Madrid, where he has been working as Teaching Assistant. He is involved in several Spanish and European research projects and he has been a committee program member of several international conferences in Computer Science. His main research lines involve Artificial Intelligence, Cloud Computing, Semantic Web and Business Processes among others and he has numerous important publications that support his work in this field.

Ricardo Colomo-Palacios is full professor at the Computer Science Department of the Østfold University College, Norway. Formerly he worked at Universidad Carlos III de Madrid, Spain. His research interests include software engineering, IT project management and applied research in information systems including aspects like service governance and big data environments. He received his PhD in Computer Science from the Universidad Politécnica of Madrid (2005). He also holds a MBA from the Instituto de Empresa (2002). He has been working as Software Engineer, Project Manager and Software Engineering Consultant in several companies including Spanish IT leader INDRA. He is also an Editorial Board Member and Associate Editor for several international journals and conferences and Editor in Chief of International Journal of Human Capital and Information Technology Professionals.

Juan Miguel Gómez-Berbís is an Associate Professor at the Department of Computer Science in the Universidad Carlos III de Madrid. He obtained his PhD from the Digital Enterprise Research Institute (DERI) at National University of Ireland, Galway. He received his Master Thesis in Software Engineering from the Swiss Federal Institute of Technology (EPFL) in Lausanne (Switzerland) and a Msc. in Telecommunications Engineering from the Universidad Politécnica de Madrid (UPM). Juan Miguel Gómez-Berbís has published around hundred scientific international publications via books, journals, conferences and workshop contributions. His current research interests include the Semantic Web, Semantic Web Services, Business Process Modelling, formal methods for eCommerce and eBusiness and, recently, Big Data environments.