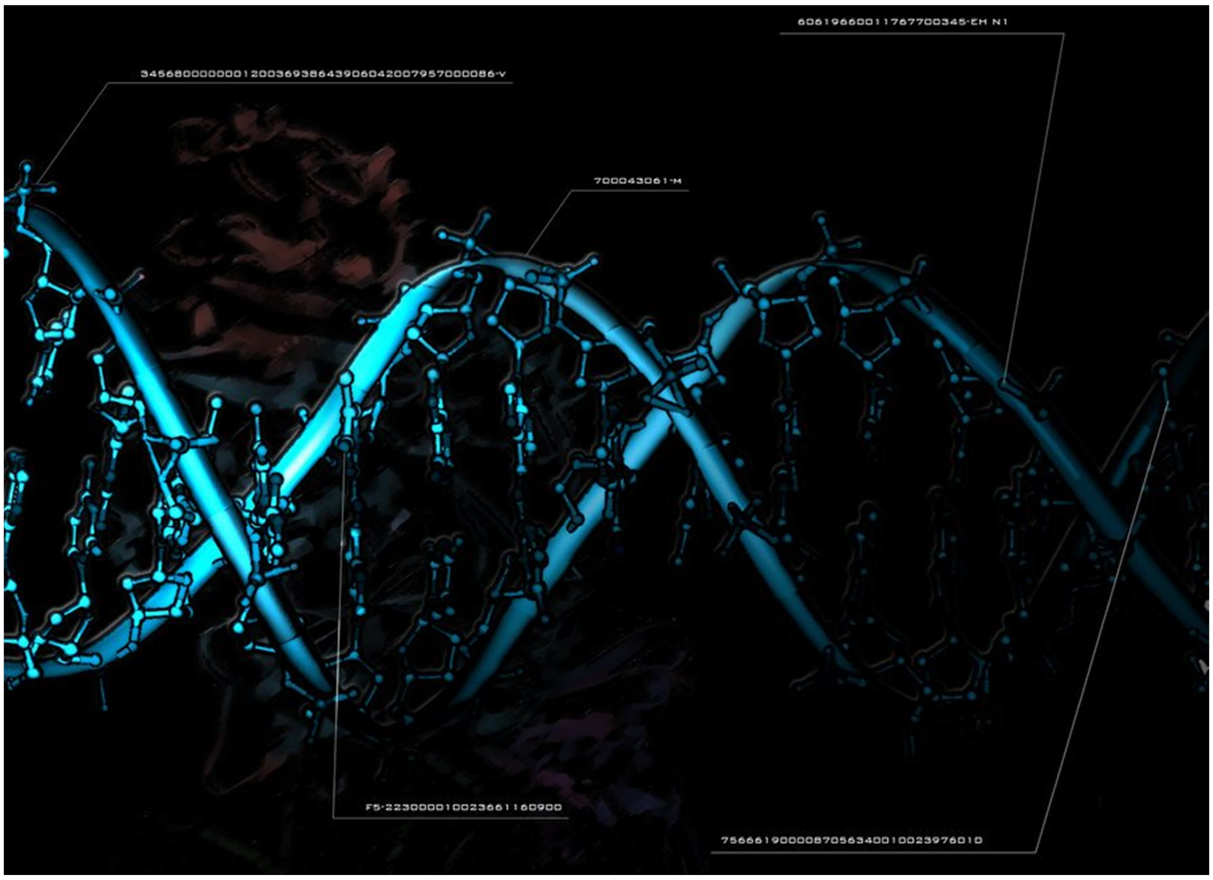
The Simulation-based Multi-objective Evolutionary OptimizatioN (SIMEON) Framework



Ronald Apriliyanto Halim (1531786)

Master Thesis

Delft University of Technology: Faculty of Technology, Policy and Management

Systems Engineering, Policy Analysis, and Management

Modeling, Simulation and Gaming Profile

TU Delft
Delft University of Technology

accenture
High performance. Delivered.

GRADUATION COMMITTEE

Title : The Simulation-based Multi-objective Evolutionary OptimizatioN (SIMEON) Framework

Author : Ronald Apriliyanto Halim
Student no. : 1531786
Place, date : Delft, December 2010

Graduation Committee
Committee chair Prof.dr.ir. A. Verbraeck – TU Delft, Systems Engineering
First supervisor Dr. M. Seck – TU Delft, Systems Engineering
Second supervisor Dr. S. Cunningham – TU Delft, Policy Analysis
External supervisor Dr. S.P. Van Houten – Accenture, Supply Chain Management

MSc. program Systems Engineering, Policy Analysis, and Management
Faculty of Technology, Policy and Management
Systems Engineering Section

Accenture B.V.
Supply Chain Management Department
Management Consulting

PREFACE

Honor and glory to my Lord, Savior, Helper and Friend Jesus Christ as without His grace and faithfulness, I would not be able to finish this master thesis. This master thesis is the embodiment of my deep ambition and desire to use all of my given knowledge and talent in the field of systems engineering and operations research. The project results in a system that I hope can contribute significantly to many scientific and real-world optimization problems.

It is part of my believe that every single entity in this universe is designed with a purpose. Looking from this perspective, it is logical that we as humans, both as individuals and organization, always try to reach our goals in our limited period of life. The efforts to reach these goals are reflected through the decisions that we make in our daily life. However, to make the right decisions is not easy. In fact, people often question whether they have made the right decisions in different segments of their life. Therefore, to help people making the right decisions, SIMEON is built. Although not all decisions we make can be rationalized, I hope SIMEON could contribute to those which can be. This way, certain rational decisions that we have to make in this world can be assisted by the state-of-the art technology that is nature-inspired or to be more precise, this-universe-creator-inspired.

I would like to thank all persons who have helped me to finish this master thesis through four different seasons and with many different challenges. I thank my first supervisor, Mamadou for his outstanding supervision, trust and support. It is definitely a pleasure to work throughout different difficulties as well as amusing times as his partner. I am looking forward to see excellent graduates who come out of his supervision. Further, it is also going to be memorable moments to sit and work together with Prof. Verbraeck to solve the memory leak problem in SIMEON. Those moments definitely had given invaluable contributions both to my technical knowledge and mental state to not give up and eventually solve the problem. Next, I also thank my second supervisor, Dr. Scott Cunningham who has given me many clear and useful feedbacks regarding the content of the thesis. His great keenness in design and optimization theories has proven that he is the right supervisor to work with. Furthermore, I also thank Dr. Stijn-Pieter van Houten as my external supervisor in Accenture for the coaching, challenges and both technical and practical advises that have been given to me. The time I spent in Accenture together with him is a truly valuable experience which has contributed to my personal development. Equally important, the performance and usability of SIMEON definitely benefit a lot from his suggestions and guidance.

Furthermore, I would like to also thank my parents and brother whose supports are always evident through their prayers and encouragements. They always make my determination worthwhile. Next, I also thank my second family, Jonathan Hartsuiker and the covenant generation. Your older brother is still fighting here and will never give up till His glory is revealed. Last but not least, to my IFES colleagues who have been working with me and revealing the hope that I have in Him. May God bless you all.

Delft, December 2010

Ronald Apriliyanto Halim

GLOSSARY

1. **Stochasticity:** problem that arises from the fact that there are variances in the simulation results due to the incorporation of probability distributions in the model.
2. **Experiment:** a set of replications towards the same model, simulator and treatment but with different individual pseudo random-number stream for each replication.
3. **Non-dominated solutions:** solutions that don't have worse values than other solutions with regard to all objective functions.
4. **Pareto Frontier:** A Frontier in the solution space in which all non-dominated solutions are located with respect to different objective functions.
5. **DSOL:** Distributed simulation object library that is developed by Delft University of Technology based on simulation and modeling framework proposed by Zeigler (2003).The library supports multi-formalism modeling and developed based on DEVS formalism.
6. **Object component/module:** type of decision variable that is not quantitative in nature and comes from the object of the simulation model (i.e. qualitative variables, system policies, etc).
7. **Metaheuristic:** "A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions" (Osman & Laporte, 1996).
8. **NSGAII:** Non-dominated Sorting Genetic Algorithm II, a state-of-the-art multi-objective evolutionary algorithm developed by Kalyanmoy Deb in 2002.
9. **Objective function:** a representation of the desired goal in a mathematical form expressing the relationship between the decision variables and the goal.
10. **Actual value:** a value that is obtained by evaluating the objective function by using solutions suggested by the optimization algorithm.
11. **Run control condition:** the specification of experimental conditions by which treatment is exercised to the simulation model. This includes length of the warm-up period, run length, and number of replications.
12. **Decision space:** a notion that describes the area in which all decision alternatives are contained.
13. **Criterion space:** a notion that describes the area in which all values of the objective functions are contained.
14. **Linear programming:** an optimization method to find the best solution of a given linear mathematical model.
15. **Linear mathematical model:** a mathematical model which elements have linear relationship one another.
16. **Extreme points:** points that mark the intersections between constraint functions in a decision space.
17. **Convergence:** a condition where an optimization algorithm has reached stability in its search process. Normally, this is indicated by the absence of quality improvement in the solution found.
18. **Structural variable:** a type of decision variable that represents design alternatives in the structure of a simulation model (e.g. the route from one point to other point, the way an entity is processed in a workstation, etc).
19. **Synergistic:** a condition where there is a high probability that the sampling fitness of the intersecting building blocks will be higher than the sampling fitness values of all intersecting blocks.

Abstract

A powerful combination of simulation and optimization has been successfully applied to solve real-world decision making problems (Fu et al., 2000; Fu, Glover, & April, 2005). Unfortunately, there are scientific and application problems with this method. Firstly, there is no transparent and formal structure to define the integration between simulation and optimization. Secondly, there are challenges to ensure a proper balance between the various desired features of the simulation-based optimization method (i.e. generality, efficiency, high-dimensionality and transparency)(Fu, 2002). This research provides two contributions to the problems above by providing: 1) the design of the framework that addresses the knowledge gap above; 2) the implementation of the framework that fulfills the aforementioned features in Java. The proposed framework is developed based on Zeigler's modeling and simulation framework and the phases of an optimization study in operations research. The test and evaluation show that the desired features are successfully satisfied.

Keywords: framework; simulation; multi-objective; evolutionary optimization

MANAGEMENT SUMMARY

In the real world, decision making often involves making trade-off between the satisfactions of multiple conflicting objectives. This fact can be seen in different problem domains, covering both social and technological systems, where a single or multiple parties are involved. Many studies in the field of decision sciences have been performed to support decision makers in formulating acceptable solutions to this problem. Among those studies, simulation-based optimization (SO) method turned out to be a promising approach to use.

Accenture as a management consultancy company which expertise is in the area of decision making, views the merit of adopting the SO method to improve its technological capability. However, to apply such a method effectively, several research challenges have to be addressed. Firstly, there is a knowledge gap in the formal structure that defines the way simulation and optimization techniques can be integrated. Secondly, and as a result of the first problem, there is an implementation gap in developing a SO method that has a proper balance between the desired features. In order to support Accenture in adopting the SO technology, the objective of this research is:

To design a framework for simulation based-multi objective optimization that satisfies generality, efficiency, multi-dimensionality and usability features to the level where the requirements from Accenture are met by appropriately making reasonable trade-offs between those features.

To accomplish this research objective, Systems Engineering life-cyle phases that are structured in waterfall model are used (Sage & Armstrong, 2000). The design process consists of three phases: system definition; system design and development; and system operational test and evaluation. In the definition phase, detailed analysis of the requirements for such framework is performed. The potential users and literature in the SO field are used as the primary inputs. Based on the formulated requirements, a conceptual design is made. In the design and development phase, the conceptual design is further detailed and implemented in Java to deliver the final product. Finally, in the operational test and evaluation phase the framework is tested with test cases and evaluated by the users.

Next to the design method used, the development of the proposed SO framework is also supported by reviews on the literature and commercial SO packages. It is revealed that many of those packages still lack of the desired features. Extensive delineations on theoretical foundations from both simulation and optimization fields are used to explain the root problems and the possible solutions in detail.

Based on the literature review, the knowledge gap in the SO field is addressed in the conceptualization phase. Zeigler's modeling and simulation framework is used as the theoretical foundation from the simulation field. This is because it has features that are responsive to the elicited requirements. On the other hand, the steps of operations research are used as the theoretical foundation from the optimization field. This results in several definitions and a structure which are valuable to bridge the knowledge gap. Furthermore, a multi-objective evolutionary algorithm, namely the Non-dominated Sorting Genetic Algorithm (NSGA-II) is used as the optimization technique integrated into the framework. This is because it satisfies the desired features. As such, the framework is named the SIMEON (Simulation-based Multi-objective Evolutionary OptimizatioN) framework.

In the design and development phase, system architecture of SIMEON is made based on the conceptual design. This architecture gives an accurate overview of the physical structure of the SIMEON framework and how the sub-systems within interact with one another. Next, the architecture is translated into the detailed design based on which the concrete implementation is performed. An evaluation as to the fulfillment of the desired features is also performed by verifying whether all the technical requirements are successfully met by the implementation. The result shows that the framework has design features that meet the requirements. They cover the generic structure of the framework, the efficient design of the optimization algorithm and the usable user interface.

Next, the operational test phase is performed to assess whether SIMEON has satisfied all the technical requirements on the operational level. To serve this purpose, three multi-objective optimization test cases with different scales and problem domains are used. The first test case uses mathematical model-based optimization problem for which scientific benchmark results are available. The other two test cases use simulation-based optimization problems that are technically relevant to the possible applications of SIMEON in the real world. The test results show that SIMEON is able to find the best solutions for the three test cases.

The evaluation phase is intended to assess the potential contributions of SIMEON to solve the real-world decision making problems and to assess how SIMEON can increase the technological capability of Accenture to solve real-world problems. This is done by taking into account the results of the operational test phase and conducting interviews with the simulation experts in Accenture. The result shows that SIMEON can contribute to decision making process by presenting the optimal trade-offs of the modeled problem in a relatively quick way. This way the decision maker(s) could use this information to make further decision. More importantly, SIMEON has a significant added value when it is used to solve complex simulation-based problems where manual enumeration of all the alternatives is tremendously time consuming or impossible.

Furthermore, SIMEON is able to improve the expertise and technological capability of Accenture to solve real-world decision making problem. The main reason for this being SIMEON's capability to facilitate learning and to create knowledge for the user. Hence, SIMEON can be used to help the clients of Accenture to learn to make the best decisions for their perceived problems. Another reason would be that SIMEON is considered to be useful in improving the confidence of Accenture to propose solutions that can be considered further by the clients.

Finally, looking at how all the technical requirements have been satisfied and translated into the physical design, SIMEON is considered to have accomplished the objective of this research. The key design feature that we use to satisfy the generality requirement is the separation of concerns between the optimization algorithm and the problem. This allows SIMEON to be used to solve problems from different domains. To satisfy the efficiency and multi-dimensionality features, the NSGA-II is implemented as a high performance multi-objective optimizer. Last but not least, usability is satisfied by the design of the user interface.

As the recommendation it is important to note that what SIMEON presents is not an end answer for the decision making problem encountered. The whole decision making steps have to involve all relevant stakeholders and have to be performed with the principles of a fair decision making process. They include openness, protection of core values, incentive for progress and focus to the content (H. Bruijn, de, Heuvelhof, & Veld, 2002). Furthermore, it is also valuable to combine Nash bargaining theory with the SIMEON framework to analyze the possible unique solution that will come out from the decision making process in a multi-actor setting.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. Introduction to Simulation-based Optimization

Multiple objective decision making notion has been prevalent in the real world due to its ability to grasp the core aspect of decision making which is making trade-offs. This fact can be easily seen in many different problem domains, covering both social and technological systems where single or multiple parties are involved (Amodeo, Chen, & El Hadji, 2007; Chin, Houten, & Verbraeck, 2005; Cunningham & van der Lei, 2009; Ding, Benyoucef, & Xie, 2009; Mueller, 2008; Narzisi, Mysore, & Mishra, 2006; Persson et al., 2006). Many studies and efforts in the field of decision science/ operations research have been performed to support the decision makers in formulating acceptable solutions to this problem (Hillier & Lieberman, 2009). Among those studies, simulation and optimization have turned to be prominent approaches that are widely used to assist decision making process (Chin, et al., 2005; Daalen, Verbraeck, Thissen, & Bots, 2009; Hillier & Lieberman, 2009).

Although the developments of simulation and optimization have given substantial contributions in the field of decision making, both methods still suffer some weaknesses that still could be improved (Fu, 2002). Optimization methods, especially those that are applied in real-world problems normally formulate the decision problems into mathematical models which tend to oversimplify the dynamics and the stochasticity of the real system (Vidal & Goetschalckx, 1997). In addition to that, another difficulty comes from intractability to represent complex, non-linear systems into mathematical models (Ding, Benyoucef, & Xie, 2004; Vidal & Goetschalckx, 1997).

On the other hand, simulation models normally only function as system analysis tools which still require further effort from the decision maker to come to the best decisions. This fact can be seen in two paradigms depending on the objective of the simulation study. In the field of risk-management, simulation model is used to study the behavior of the modeled system such that the dynamics that might be produced by the system can be profoundly analyzed and taken into account for further decisions(Magno, 2002; Zeigler, Praehofer, & Kim, 2000). In this context, the type of analysis involved is normally what-if analysis in which new facts, assumptions, knowledge or parametric modification can be introduced to the model to evaluate the resulting behavior. On the other hand, in the context where decisions have to be made to achieve certain goals, simulation study normally facilitates the execution of computational experimentations in which evaluations on pre-defined decision alternatives are performed in regards to those goals (Daalen, et al., 2009; Zeigler, 2003). In this type of objective, what-if analyses are used to explore the feasible alternatives such that the best decision in respect to the previously formulated goals can be found (Sage & Armstrong, 2000).Furthermore, these analyses are typically conducted in an iterative/cyclical manner in which the results of the previous experiments are taken into account to perform the subsequent explorations. This way, the decisions that are represented in the experiments are made adaptively with the help of the evaluations against future uncertainties that are performed by means of computational experimentations (Thierry, Thomas, & Bel, 2008). It is also common to use the help of experts to formulate these decisions/experiments. To provide better understanding of the concept elaborated above, Figure 1illustrates the position of the model builder/experimenter and its relationship with the entities of the widely used simulation and modeling framework that is proposed by Zeigler (2000).In this framework the concept of Experimental Frame (EF) is advocated to capture the context under which the observation to the real system or experimentation to the model can be performed (Traoré & Muzy, 2006). In addition to that, the simulator is the executor of the structure described in the model to generate outcome measures.

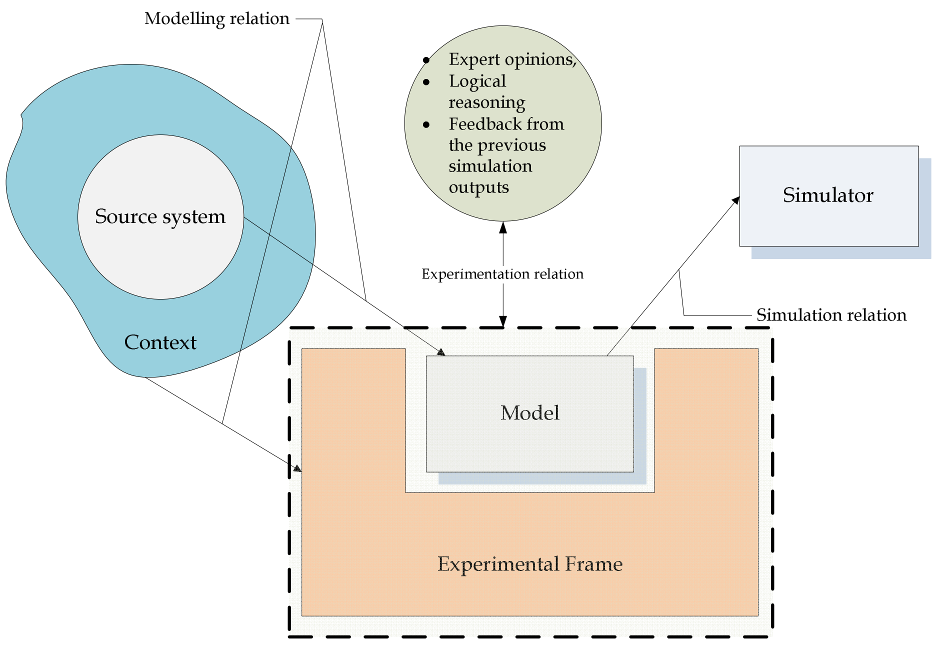


FIGURE 1 MODELING AND SIMULATION FRAMEWORK (ADAPTED FROM TRAORÉ & MUZY (2006))

Unfortunately, the phase where the experiments are carried out can be very inefficient and unfruitful. This occurs if the knowledge of the modelers with regard to the decision alternatives is limited while the amount of the potential alternatives is enormous. Furthermore, although improvements of system performance(s) might be achieved through arbitrary specification of the solutions, questions as to the quality of the solutions often emerge as there is no structure in the approach that is used to solve the decision problems. On top of that, in the case where the modeler manages to optimize the decisions iteratively (based on the previous experimentations or the knowledge of the experts), the re-designing and re-structuring of the simulation model have to be done manually, which is also a tedious work (Wiedemann & Krug, 2003). Therefore, it is clear that there is still an enormous challenge in the process of finding the best decisions if simulation technique is used alone. This is true, especially if the behavior of the modeled system is hard to/impossible to be studied by altering/adapting the decisions iteratively.

Looking at the problem in simulation study, optimization technique can clearly contribute a valuable help, by providing the structure needed to find the best decisions. An optimization technique that is implemented in a computer program even gives the benefit of automating the search process of finding the best decisions (or in optimization theory often called as solutions). As a matter of fact, successful efforts have been reported in combining simulation and optimization methods which nurture the growth of researches in this so-called simulation-based optimization field (April, Glover, Kelly, & Laguna, 2003; Ding, Benyoucef, & Xie, 2006; Fu, et al., 2005). In the simulation-based optimization method, optimization typically functions as the search method that explores the alternative space of a simulation model in such a way that solutions leading to the preferred system performance(s) can be found. Figure 2 depicts the concept of this method.

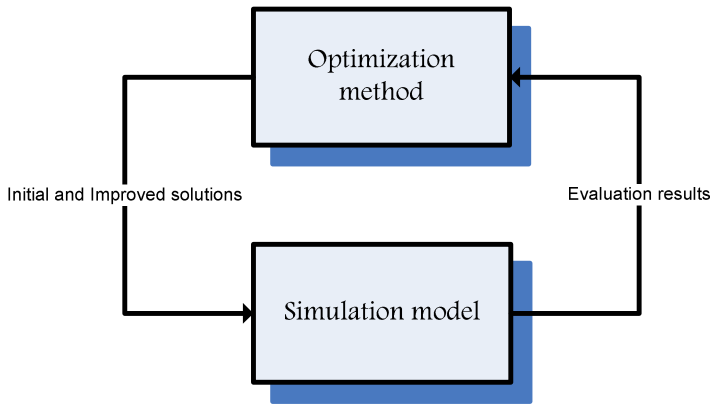


FIGURE 2 SIMULATION-BASED OPTIMIZATION METHOD (ADAPTED FROM APRIL, GLOVER, KELLY, & LAGUNA (2003))

This way, the dynamics and stochasticity of the modeled system are still taken into account while best solutions can be derived without tedious effort (e.g. manually traversing the whole possible alternatives for the input of the simulation model).

However, there is still a fundamental problem in this widely applied simulation-based optimization method: to the best of the author's knowledge, there has not been any formal and detailed explanation on the way an optimization method is integrated into the widely used modeling and simulation framework (Figure 1). Most of the researches that harness simulation-based optimization method don't formally explain with a solid theoretical foundation how the interaction between the optimization technique, the model and the experimental frame is designed (April, et al., 2003; Boesel, Nelson, & Ishii, 2003; Hojun, Zeigler, & Doohwan, 2008; Wang, Rivard, & Zmeureanu, 2005). A simple conceptualization like depicted by Figure 2 is often used instead. This knowledge gap thus, deters the appropriate use of optimization technique by simulation practitioners and vice versa.

Next, despite its successfulness in providing rich insight and strategies to decision makers, simulation-based optimization still faces challenges in providing desirable features for the decision makers, especially when it is used in a problem where multiple and possibly conflicting objectives exist. These conflicting objectives may come from single or multiple decision maker(s). In this type of problem, different objectives have to be taken into account in the decision-making process. This brings about an important requirement for any decision support tools that deals with this problem: they have to be able to facilitate the finding of the non-dominated solutions. In a condition where different objectives conflict one another, it is important to present the decision maker(s) a clear information as to the different non-dominated solutions in regards to the conflicting objectives. This way, the quality of any solutions that may come from the decision making process can be preserved by not presenting the decision maker(s) any dominated solutions. However, this also has a direct impact to the efficiency of a decision support method applied to this problem as the computational time needed to solve multi-objective optimization problem is considerably high (Narzisi, et al., 2006).

Challenges related to simulation-based multi-objective optimization method can also be found by looking at the broader requirements formulated by Fu (2002) which include generality, transparency to user, high dimensionality and efficiency. Among them, generality very often conflicts with efficiency (Fu, 2002; Fu, et al., 2000). This can also be seen from the optimization routines that are being developed in this field. Most of them employ evolutionary-based algorithms which are designed in tight coupling with the modeled-problem to allow efficient exploitation of the problem structure while sacrificing the generality of the method to solve wide range of problems across different domains (Ding, et al., 2006; Ding, et al., 2009; Hani, Amodeo, Yalaoui, & Chen, 2008; Persson, Grimm, Ng, et al., 2006). While usability can often be facilitated with a user friendly user interface, there are many shortcomings pertaining to efficiency and high dimensionality in the simulation-based optimization methods in commercial packages. A review of two popular optimization routines in commercial simulation packages (AutoStat and OptQuest) reveals that both packages lack an efficient multi-objective optimization routine (Fu, 2002). AutoStat approaches multi-objective optimization problem using classical inefficient method in which multiple objectives are aggregated to form single objective using a weight-vector. The drawback of this method is that for different preferences of the decision maker(s), different weight-vectors have to be used and the same problem has to be solved repetitively (Srinivas & Deb, 1995). The similar approach was also employed by OptQuest which started to provide an efficient multi-objective optimization routine allowing the analysis of the Pareto frontier , after the release of engine v 6.5 (http://www.opttek.com/Products / Documentation.html).

Looking at the challenges and problems in the application of simulation-based multi-objective optimization to real-world problems, it is evident that a framework that structures and facilitates effective implementation of simulation based multi-objective optimization is required. There are three contributions to real-world problems that this framework has to make. The first is to increase the effectiveness of simulation as a decision support tool to solve real-world decision making problems by using an optimization method. The second is to facilitate the application of simulation-based optimization to solve multi-objective optimization problem(s). Lastly, is to facilitate the implementation of simulation-based multi-objective optimization which is able to fulfill the desired features mentioned by Fu (2002). This research proposal is, therefore, put forward to design a simulation-based multi-objective optimization framework that is able to address problems above.

The rest of the section is organized as follows. Section 1.1 delineates the research problem of such a simulation based optimization framework. This is then followed by the description of the research questions in section 1.2. Section 1.3 describes the design approaches and steps that will be taken to design the proposed framework.

1.1 Research Problem

This section begins with the problems that are encountered in the application of modeling and simulation in Accenture. Based on the insight regarding the problems, the need of a framework is then substantiated. Next, research problems pertaining to the desired features of the framework are presented to give better understanding on the problems. Finally, a clear problem statement and design objective are presented.

1.1.1 Accenture and Real-world problems of Modeling and Simulation

Accenture is a global management consulting, technology services and outsourcing company with offices and operations in 52 countries, serving clients in more than 120 countries. The expertise and technological capability of Accenture are mainly dedicated to help its clients in entering new markets, increasing revenues in existing markets, improving operational performance and delivering their products and services more effectively and efficiently (Accenture, 2010). One of the strengths that distinguish Accenture from other similar companies lies in its history of technology innovation and implementation, including its research and development capabilities on which they approximately spend 300 million dollar annually (Accenture, 2010).

Modeling and simulation is one of the methods that Accenture uses. Its expertise and technological capability help clients from different industries to become high-performance companies. To date[1], there have been 8 initiatives in Accenture Amsterdam-Zuid to develop simulation models aiming to provide consultancy services in the form of advices for the clients to improve their business performance(s) in the area of supply chain management, logistic and business planning. Five out of those eight initiatives have been turned into modeling and simulation projects in which DSOL[2] -based models are implemented by the simulation experts at Accenture. Those models are:

1. Accenture Workforce planning model
2. PMI Workforce planning model
3. Mushroom market model
4. Fiber glass roll-out model
5. Internet telephony and television service model

The models were developed by applying the separation of concerns principle to the model, the simulator and the context used to study the model similar to the framework proposed by Zeigler (2003), (Figure 1 Modeling and simulation framework). They also encompass different business domains that fall under the expertise of Accenture. Depending on the scope and objectives of the modeling study, the size of the model could vary greatly. This far, the common use of those simulation models is to perform what-if analyses using different business scenarios. However, as simulation applications in Accenture have proven to be capable to perform alternatives evaluations, it is evident that there is an interest to use those simulation models to perform optimization studies or to apply the simulation-based optimization method. With this method Accenture will have an improvement in its technological capability to help its clients in solving real-world decision problems and to accomplish their business goals.

However, there is an inherent challenge in applying such a method. Normally, the number of decision variables that can be analyzed increases proportionately with the size and complexity of the models. This consequently requires knowledge and experiences in specifying the decision variable values of the simulation model. This is a rather challenging situation as there are many decision making problems in the area of supply chain management, logistic and business planning which have enormous complexities in terms of the relationship between the possible alternatives and its resulting system performance(s) (Hillier & Lieberman, 2009; Vidal & Goetschalckx, 1997). In the optimization field this property of complexity is known as non-linearity. Non-linearity entails that a systematic approach, which is normally implemented in the form of an optimization algorithm, is required to efficiently find the alternatives that lead to the desired system performance(s).

Therefore, it is clear that the integration of the optimization method to a modeling and simulation framework currently used by Accenture has a valuable contribution in assisting the search of the best decision(s) and eventually in the effort of delivering trustworthy alternatives/solutions to its clients. From the scientific world, the development of this type of framework falls into the rapidly growing research field called simulation-based optimization. The next section will delineate the high level requirements of such framework in regards to the real-world problems that are faced by Accenture.

[1] 15 April 2010

[2]**DSOL:** Distributed simulation object library that is developed by Systems Engineering group of Technology, Policy and Management Faculty in Delft University of Technology. The library supports multi-formalism modeling and developed based on DEVS formalism.

1.1.2 Research Problems for the Design of Simulation-based Optimization Framework

In the effort of eliciting the requirements of such a framework, the desired features mentioned by Fu (2002) can be used as the basis on which the detailed requirements can be formulated. However, among those features, trade-offs have to be inevitably made. In addition to that, research problems that arise from the requirements of the framework have to be addressed as well.

Generality

Among all the desired features mentioned by Fu (2002), generality is normally of the highest value for both scientific and business worlds as it enables practical implementation of the optimization methods to different simulation problems (Fu, 2002; Fu, et al., 2000). Consequently, this makes a framework that employs problem-dependent optimization techniques becomes less desirable. In the field of supply chain management, logistic and business planning, optimization of the input parameters/quantitative variables of the simulation model (often called parametric optimization) always dominate the requirement. However since the framework is going to be applied to different problem domains, the optimizer should also not be problem-structure dependent.

A widely-used algorithm in the simulation-based optimization which possesses the characteristic of not being problem-dependent is metaheuristics. In the implementation of this type of algorithms, the simulation model is treated as a black-box where the optimizer/algorithm serves as an intelligent module that iteratively suggests solutions to the simulation model based on the evaluations of the previous solutions (April, et al., 2003). There are many algorithms that fall under this category which work based on different mechanisms. Some of them use a single point approach, where the algorithm starts with a single solution and uses the information of the neighborhood structure to guide the solution-finding process (Blum & Roli, 2003). Examples of these metaheuristics are Simulated Annealing (SA), Tabu Search (TS), Iterated Local Search (ILS), Variable Neighborhood Search (VNS), Greedy Randomized Adaptive Search Procedure (GRASP), and Guided Local Search (GLS). Some use population-based approach, where the algorithm starts with multiple solutions and uses a nature-inspired intelligence (such as evolution process) to learn implicitly the structure of the problem through stochastic samplings and performs the exploration of search space accordingly (Blum & Roli, 2003). They are Evolutionary Algorithms (EA) which includes Genetic Algorithm (GA) and Differential Evolution (DE), and Ant Colony Optimization (ACS). Therefore, to satisfy generality feature, metaheuristic algorithm is going to be used in the proposed framework.

Related to this requirement, two generic types of decision variables to be optimized have been defined as part of the requirements: the input parameters/quantitative variables and policy alternatives/qualitative variables of the simulation model. From the scientific world, efforts have been made to make a generic optimizer for structural and qualitative variables of a simulation model (Azadivar & Tompkins, 1999; Pierreval & Paris, 2003). These authors also use genetic algorithms to optimize combination of structure and qualitative variables of a simulation model. However, their approach unfortunately does not provide the functionality to optimize quantitative variable which is important for parametric optimization.

Efficiency

From efficiency requirement, it is important to guarantee the convergence of the optimizer in a reasonable amount of computational time. Among the vast majority of metaheuristic algorithms that have been developed in simulation packages, EA, especially GA is well known for its efficiency (April, et al., 2003; Fu, 2002; Fu, Chen, & Shi, 2008; Konak, Coit, & Smith, 2006). This is because it has advantages over those which use single point approach. Firstly, due to its population based approach, GA is capable of exploring larger area of the search space with smaller number objective function evaluations and therefore giving less computational time than non-population based algorithms (April, et al., 2003; Konak, et al., 2006). This is a very important feature as in a simulation-based optimization method one objective function evaluation requires the execution of the simulation model. Secondly, it also does not require the initial solutions to be close to the optimum solutions as what the case is for the single point approach (April, et al., 2003).Therefore, from the efficiency feature, genetic algorithm is also found to be a strong candidate.

However, it is important to maintain the balance of generality and efficiency on this type of algorithm. An example of possible conflict between generality and efficiency can also be found in the research of Azadivar & Tompkins, (1999) and Pierreval & Paris (2003). Although their researches have opened new opportunity to develop a generic optimizer for structural and quantitative variables, the algorithms that they developed have not been proven yet to be convergent for large-scale and multi-objective optimization problems.

Multi-dimensionality

From the multi dimensionality criterion, it is desirable that the design of the framework should enable the optimization of different types of decision variable against multiple-objective functions. This requirement comes from the fact that there exist alternative solutions that may involve changes in the components of the modeled-system and also from the fact that there exist multiple and possibly conflicting objectives to optimize. Consequently, the framework design should be able to facilitate the generation of Pareto-optimal solutions and the necessary change in the simulation model based on which types of decision variable are to optimize. For this purpose, an automatic simulation model generator might be needed (Azadivar & Tompkins, 1999).

Usability

The last problem that has to be addressed in designing the framework is to ensure the usability of the framework such that both the expert and non-expert users may make use the framework comprehensively according to their knowledge level. An important desired feature for the expert users is the extendibility of the framework that allows them to perform any necessary extensions to increase the efficiency and efficacy of the framework. On the other hand, it is desirable to have an interface that can facilitate the non-expert users to model optimization models and to perform simulation-based optimization studies easily. Hence, to reach the targeted usability, the design of the interface should be based on the characteristics of the users. To serve this purpose, a list of technical requirements will be made using inputs from the users.

1.1.3 Problem Statement

The problem exploration has shown that there are research problems that have to be addressed in constructing a framework for the simulation-based multi-objective optimization method. Two major problems and three sub-problems can be formulated as follows:

1. There is a design and implementation gap concerning the framework that integrates optimization techniques to Zeigler's modeling and simulation framework.

Sub problem statements:

1.1 There is a design challenge to design a framework that is able to solve multi-objective optimization problems using combined approach of simulation and optimization.

1.2 There is an implementation gap concerning the adaptation of the genetic algorithm to the framework such that it provides the desired degree of generality and efficiency.
1.3 There is a design and implementation gap concerning the interfaces of the framework that are able to facilitate the desired multi-dimensionality and usability of the framework.

2. There is a knowledge gap as to how this new framework will contribute to improve the user's expertise and technology capability in solving real-world problems.

1.1.4 Research Objective

This research is conducted as the graduation project for the master programme in Systems Engineering, Policy Analysis and Management of Delft University of Technology. The project is commissioned by Accenture. The main research objective of this project is to design a framework for simulation based-multi objective optimization which satisfies the following desired features:

1. generality
2. efficiency
3. multi dimensionality
4. usability

to the level where the requirements from Accenture are met by appropriately making reasonable trade-offs between those features.

From scientific point of view, this design project contributes to the development of the generic framework to solve simulation-based multi-objective optimization problems. On the other side, this project also contributes to the application of such method to solve the real-world problems faced by multiple industry sectors.

1.2 Research Questions

Following the research problem that has been formulated in previous section, following main and sub research questions are formulated.

Main research question

What does the design of a simulation-based multi-objective optimization framework that increases the effectiveness of simulation-based decision support tools look like?

Sub research questions

1. How can metaheuristic techniques be integrated into the framework of modeling and simulation that is currently used to solve real-world problems?

2. How can we adapt genetic algorithm such that it can provide the desired generality and efficiency for the framework?

3. What does the design of the interface components that are able to facilitate the necessary changes in the simulation model look like?

4. To what extent this framework can contribute to the decision-making process where there exist multiple and possibly conflicting objectives?

5. How can this framework improve the expertise and technological capability of the user in solving real world decision-making problems?

1.3 Design Approach

The philosophy of the design approach that is used here follows the Kantian inquiry system. This inquiry system views truth content of a system to be located in both its theoretical and empirical components (Mitroff & Turoff, 1973). Furthermore it also puts a great emphasis on the definitions of the possible alternatives/methods to reach the pre-defined objectives and goals (Mitroff & Turoff, 1973).

Based on the philosophical foundation above, we postulate that this research project makes use knowledge and concepts of systems engineering that requires both theoretical and empirical supports in its approach to deliver valid answers for the research problems stated above. As such, solid theoretical foundations and rigorous experimentations will be used to design the components of the system. Furthermore, design alternatives will have to be prepared and implemented whenever limitations are encountered in the previous design efforts to realize the system.

To be able to accomplish the main objective of the research and answer all the research questions above, there are steps that have to be systematically taken. Following are the description of the steps relevant for each design question.

1. How can metaheuristic techniques be integrated into the framework of modeling and simulation that is currently used to solve real-world problems?

    - Study the literature on the development of simulation-based optimization methods, particularly those which employ genetic algorithms. Conduct analyses on how the available simulation models that are built using the widely-used modeling and simulation framework achieve its objectives.

    - Study the literature about the design of the interface that allows loose coupling between simulation model and the optimization module.

    - Experiment with currently available generic optimizers ISSOP.

    - Implement the interface design and test it.

1. How can we adapt genetic algorithm such that it can provide the desired generality and efficiency for the framework?

    - Study the literature on the implementation of genetic algorithms that have been proven to be generic yet convergent for small and large scale problems.

    - Implement the algorithm using available libraries e.g. jMetal library (Durillo, Nebro, Luna, Dorronsoro, & Alba, 2006)

    - Do experiments/case studies to test the convergence of the algorithms (both for small and large scale problems

2. What does the design of the interface components that are able to facilitate the necessary changes in the simulation model look like?

    - Derive from the requirements, components of the framework that are required to facilitate the necessary changes in the simulation model related to the types of the decision variables to optimize.

- Conceptualize the design of those components.
- Implement the components in a programming language.

3. To what extent this framework can contribute to the decision-making process where there exist multiple and possibly conflicting objectives?
    - Study the literature on the applications of multi-objective optimization techniques for real-world problems.
    - Study the literature on the possibility to use the framework in a multi-actor system.
    - Conduct experiments to the framework using test cases.
    - Translate the results of the simulation and optimization study to concrete recommendations and assess its potential contributions to the performance(s) of the real-system.
    - Perform interviews to the experts regarding the added value of the framework in solving multiple objectives decision making problems.
4. How can this framework improve the expertise and technological capability of the user in solving real world decision-making problems?
    - Study the literature on the applications of simulation-based multi-objective optimization techniques to solve real world problems.
    - Interview the experts regarding the requirements of the framework.
    - Translate the requirements into the framework using systems engineering steps.
    - Perform interviews with the experts regarding the contributions of the framework to solve real-world problems.

There is an expected drawback with regard to the design approach for the framework. It is foreseen that the framework will still require considerable amount of computation resource. One obvious source of this problem comes from the required simulation replications to smooth out the stochasticity of the model. This could be worsened by the size of the simulation model and number of decision variables to optimize. Limitation to the number of the decision variables to optimize might come as the result of the study.

Furthermore, it is also important to note that in this research we do not deal with multiple tactics and hidden agendas that exist in the multi-actor environment. A clear formulation of multi-objectives optimization problem is the starting point of the research. The definition of this kind of optimization problem will be elaborated further in the second chapter.

Figure 3 below depicts how all abovementioned research questions are transformed to thesis chapters through the design approaches. Systems engineering life-cycle phases (Sage & Armstrong, 2000) are proposed to facilitate the design of the framework and to ensure the fulfillment of the user requirements.

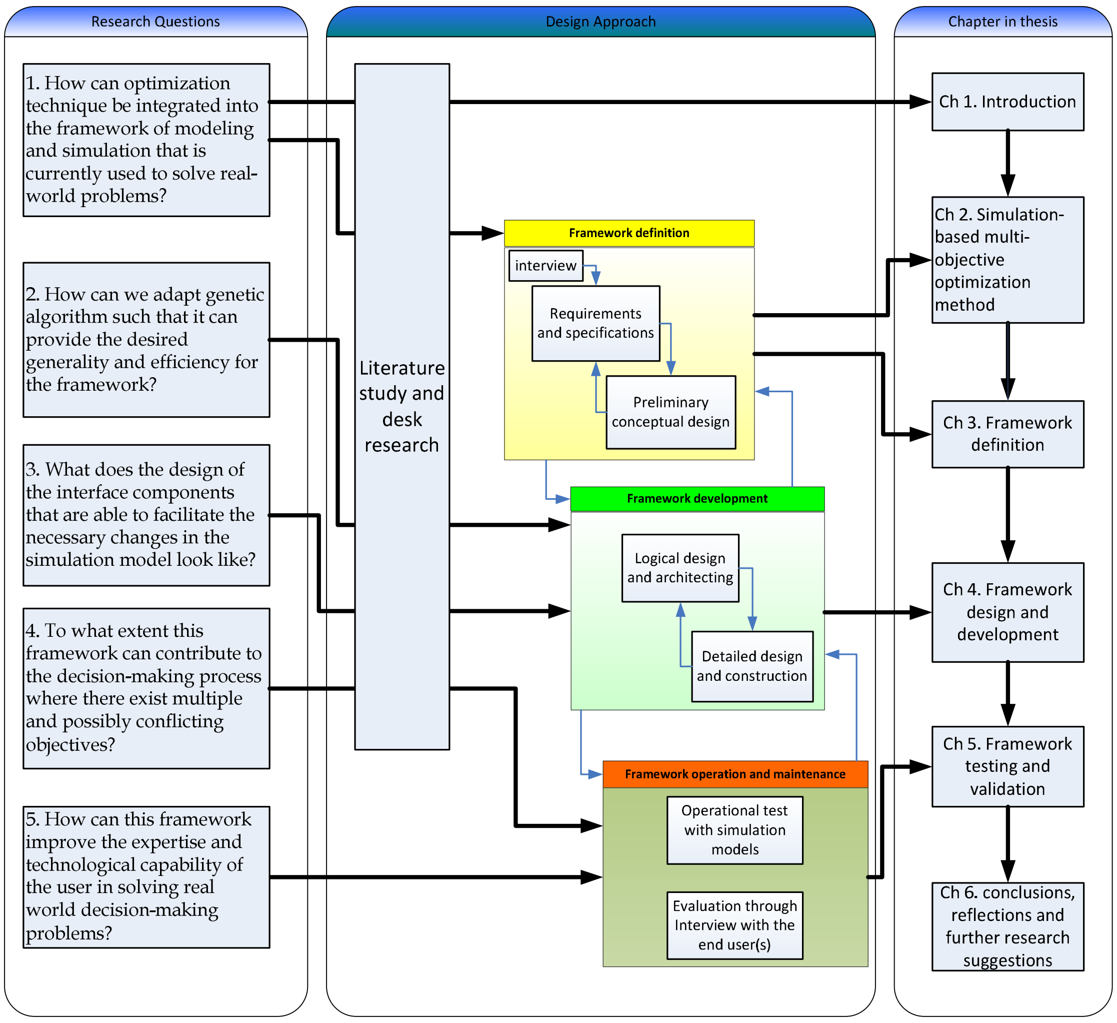

Research Questions
Design Approach
Chapter in thesis
1. How can optimization technique be integrated into the framework of modeling and simulation that is currently used to solve real-world problems?
2. How can we adapt genetic algorithm such that it can provide the desired generality and efficiency for the framework?
3. What does the design of the interface components that are able to facilitate the necessary changes in the simulation model look like?
4. To what extent this framework can contribute to the decision-making process where there exist multiple and possibly conflicting objectives?
5. How can this framework improve the expertise and technological capability of the user in solving real world decision-making problems?
Literature study and desk research
Framework definition
interview
Requirements and specifications
Preliminary conceptual design
Framework development
Logical design and architecting
Detailed design and construction
Framework operation and maintenance
Operational test with simulation models
Evaluation through Interview with the end user(s)
Ch 1. Introduction
Ch 2. Simulation-based multi-objective optimization method
Ch 3. Framework definition
Ch 4. Framework design and development
Ch 5. Framework testing and validation
Ch 6. conclusions, reflections and further research suggestions

FIGURE 3 TRANSFORMATION OF THESIS RESEARCH QUESTIONS TO THESIS CHAPTERS THROUGH DESIGN APPROACH

2. Simulation-based Multi-objective Optimization Method

In this chapter the theoretical foundations that lead to the development of generic simulation-based multi objective optimization framework are presented. To begin the chapter, an introduction to the concept and development of simulation-based optimization method in the real world are presented. This is to provide an overview on how this method has been developed and applied in both academic and real worlds. Next, the elaboration on the Zeigler's modeling and simulation framework and how simulation has been applied to solve real world problems are presented. This would then bring the discussion into the optimization problem that occurs in the real-world: multi-objective optimization problem. To provide a clear understanding of the problem, the theoretical foundation of this problem is going to be presented. More importantly, the multi-objective optimization algorithms that have been used to solve such a problem are also going to be detailed. Finally, the last section proposes the development of a simulation based optimization framework by addressing the problems that are found in the current simulation-based optimization methods. The redefinition of the second research question is also performed to narrow down the focus of the research and to expose the novelty of this research.

2.1 Simulation-based Multi-Objective Optimization Method

Simulation-based optimization is a method that stems from the rapid and successful development of simulation and optimization techniques. The integration of those two methods has often been considered as one of the most successful developments in the field of computer science and operations research (Fu, 2002). One of the major drivers of the development of this method comes from the growing need of using simulation as system modeling and analysis tool as well as optimization as a system design tool (Daalen, et al., 2009; Pierreval & Paris, 2003). However, it is also interesting to note that the successfulness of this method is not independent from the rapid developments of many other intersecting disciplines such as artificial intelligence and systems science. Looking at the contributions of those disciplines this far, it is expected that in the future more developments in simulation-based optimization methods will also be assisted by the developments in those disciplines.

In the field of artificial intelligence, simulation-based optimization has a similar position that of reinforcement learning technique -a subset of machine learning techniques. Similar to this technique, simulation-based optimization shares a similar goal to develop an intelligent system that learns over time using the information obtained through the previous learning cycles/optimization iterations (Gosavi, 2003). This way, the resulting system configuration is a product of intelligent improvements over time, that take place to better accomplish the system's pre-defined goal(s). It is also interesting to note that there have been many algorithms that are developed in the reinforcement learning field that have given valuable contributions in the development of simulation-based optimization especially in the domain application of agent based modeling (Kazemi, Zarandi, & Husseini, 2009; Klein, Bourjot, & Chevrier, 2009).

In addition to that, the notion of simulation-based optimization has also long been framed in the general systems theory that forms the theoretical foundation for all decision support systems that are developed with system perspective. In the specification of different types of system that is provided by Klir (2001), simulation-based optimization can be considered as a goal-seeking system in which optimization problem is used to define its goals, elements and the relationships between them (Klir, 1991). It is therefore clear that simulation and optimization techniques are integral part of this field of study which developments provide direct contribution to simulation-based optimization method.

In a simulation-based optimization method, the simulation model normally functions as the evaluator of the objective functions that are to be optimized by the optimization algorithm, making the simulation model becomes embedded within the routines of the optimizer. The user is normally provided with the flexibility to set the parameters of the algorithm, the initial condition and also the run control of the simulation model. Once the entire necessary configuration has been set, the optimizer will start with initial solutions for which evaluations using the simulation are performed. The results of the evaluations are used by the optimization algorithm to generate new solutions that are expected to be better than the previous solutions. This loop continues until the stopping criteria of the optimization algorithm are reached. Figure 4 depicts the concept of this process.

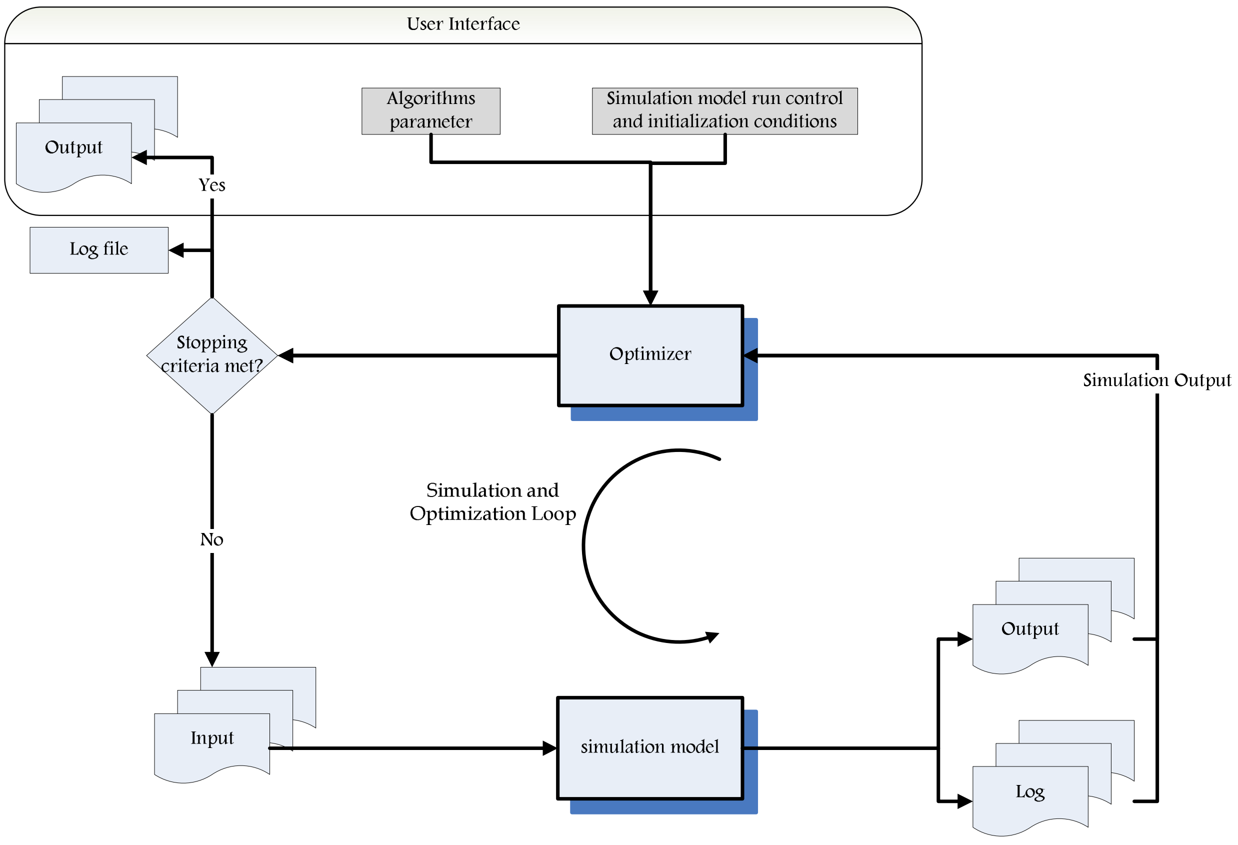


FIGURE 4 SIMULATION-BASED OPTIMIZATION PROCESS

Driven by the successful applications of this approach, the growth of simulation-based optimization method can now be clearly seen in the field of modeling and simulation. Nowadays, almost all commercial off-the-shelves (COTS) discrete-event simulation packages have an integrated optimizer within. Table 1 provides an overview of various optimizers that are integrated in different COTS simulation packages. It is quite remarkable that most of those packages employ metaheuristic algorithm with genetic algorithms as the most prevalently used optimization method. This development is expected as the result of the advance in the research of metaheuristic and the rapid growth in computing technologies (April, et al., 2003). There are several advantages of metaheuristic in comparison to the other optimization algorithms which also make metaheuristic popular in this application:

1. It does not require gradient information
2. It can be adapted such that it is not problem structure dependent

However, there are also weaknesses of the metaheuristic algorithms such as they cannot provide a guarantee that the solution produced is the optimal solution and they might not be as efficient as an algorithm that is specifically designed to solve optimization problems with a special structure or decision space.

TABLE 1 VARIOUS OPTIMIZERS IN COTS SIMULATION PACKAGES

Optimizer	Vendor	Simulation Platform	Multi-objective optimization method
AutoStat	AutoSimulations, Inc.	AutoMod	Single objective genetic algorithm with weighted sum method on the objective functions
OptQuest	OptTek Systems, Inc.	AnyLogic, Arena, Cyrstal Ball, CSIM 19, Enterprise Dynamics, Micro Saint, ProModel, Quest, SimFlex, SIMPROCESS, SIMUL8, TERAS	Scatter search, Tabu search and Neural network, supports the allocation of non-dominated solutions and weighted sum method
Optimizer	Lanner Group, Inc.	WITNESS	Adaptive thermo statistical simulated annealing, with weighted sum method on the objective functions
ISSOP (Intelligent System For Simulation and Optimization)	DUALIS GmbH IT Solution	eM-Plant, Arena, Enterprise Dynamics, Speedsim, DOSIMIS3, MATLAB	Quasi-gradient-strategy, CENUM - component wise enumeration, EVOL - Evolutionary strategy, SIMCARLO -Monte-Carlo-strategy and SIMGEN Genetic algorithm, with weighted sum method on the objective functions

Despite of the successfulness of those packages in combining optimization and simulation, there are generally two important aspects that simulation and optimization researchers would be concerned about:

1. What is theoretical foundation that forms the framework used to perform the integration of simulation and optimization?
2. Why do most of those packages use only weighted-sum method in formulating the optimization problem?

These questions normally rise because of:

1. The importance of using a clear framework which can facilitate a clear understanding on the technical design of the simulation and optimization techniques.
2. The importance of using the appropriate method to solve real-world optimization problems which are mostly multi-objectives in nature.

The answers for these concerns are essentially the important ingredients to develop the trust of the practitioners and researchers on the flourishing simulation and optimization technology. Relevant to the concerns above, we can see that solid theoretical foundations that lead to the answers for those questions are important. Therefore, the following sections will discuss the framework that is used in this research and followed by the discussion on the optimization techniques that are appropriate for real-world problems.

2.2 Zeigler's Modeling and Simulation Framework

A well-known framework in the field of modeling and simulation is the one that is proposed by Zeigler (2000). This is because this framework is firmly rooted in systems theory and developed with a clear specification in the implementation level. In this framework, the System-Model-Simulator (SMS) view is used to define the entities and relationships that exist in modeling and simulation process. There two relationships that are depicted in this framework: the modeling and the simulation relations (Figure 5). The modeling relations explain the extent to which the real system is modeled as a simulation model while the simulation relation explains the separation of concerns between the simulation model and the execution of that model using the simulator (Traoré & Muzy, 2006).

Furthermore, the experimental frame concept explains the separation of concerns between the model and any data gathering (e.g. statistical measurements), and any control efforts (e.g. starting and stopping of the simulation) that are not performed in the real system. To serve these functions, the concept of experimental frame is formalized with three components which govern the experimentation on a model: the generator, transducer and acceptor (Traoré & Muzy, 2006; Zeigler, et al., 2000). The generator component produces the input segments to the simulation model. These input segments are normally specified in time-indexed values representing the events that occur in the model.

Those input segments are then processed by the model using certain state-transition functions to calculate all the resulting state changes. Next, the transducer maps the output variables into outcome measures of interest. Finally the acceptor component monitors the validity of the experiments by checking whether the values of the outcome measures of interest violate the pre-defined constraints of the experimentation.

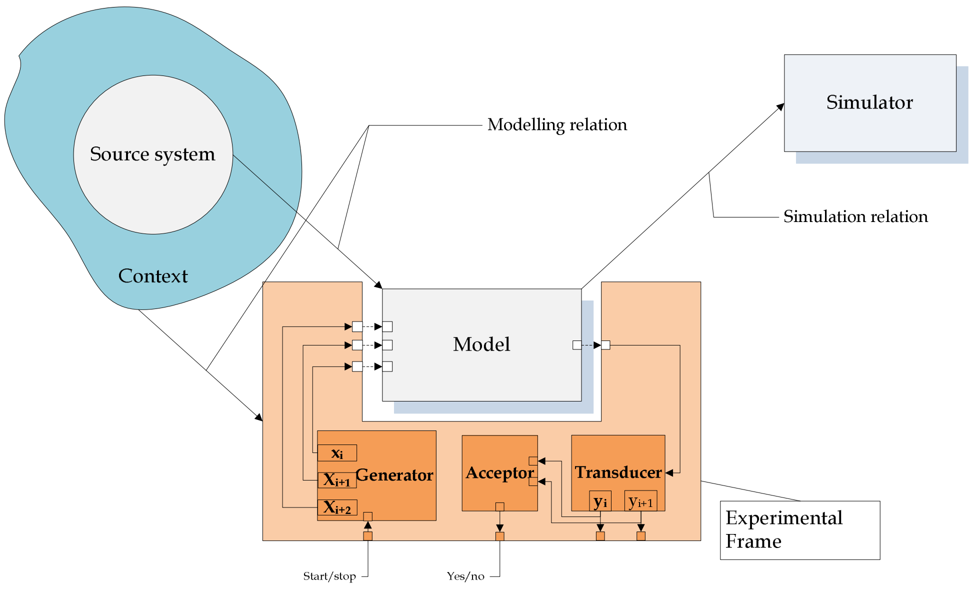


FIGURE 5 ZEIGLER'S MODELING AND SIMULATION FRAMEWORK (ADAPTED FROM TRAORÉ & MUZY (2006))

Given the framework above, we can shift our focus to the experimentation phase of the simulation model. This is the phase where what-if analyses are performed to achieve the objective of the modeling study. If the objective is to achieve goals in regards to certain system performance, the simulation model is used to evaluate the effect of certain alternatives defined by the user of the framework in respect to those goals. In this case, the simulation model is conditioned to a certain set of circumstances that is defined in a so-called treatment (Daalen, et al., 2009).This treatment is thus, the embodiment of the decision alternatives and it consists of the following elements:

- Specification of input data
- Collection of input data
- Initialization conditions
- Run control conditions,
- Specification of output data

Hence, in an experiment, a specific treatment is given to the simulation model, and the model is run to produce the outcome measures of interest (Figure 6). In the effort of optimizing the system performances, normally a number of treatments that involve different collections of input data are specified and executed in the experiments. However, this can be very challenging and sometimes impossible to do manually if the model is too complex or the relationship between the input data (or often called decision variables in optimization study) and the outcomes measure of interest is not understandable. Not to mention if there are wide ranges of values in which the input data can be specified.

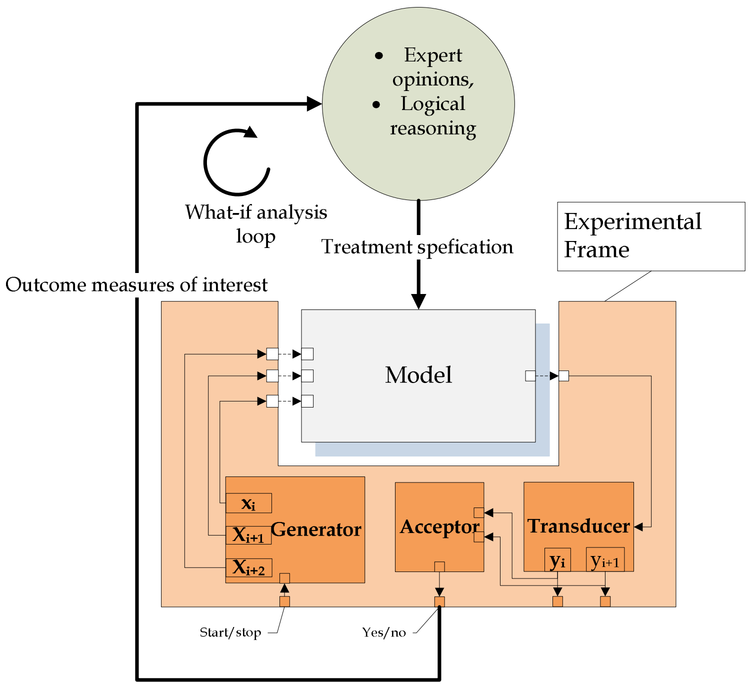


FIGURE 6 EXPERIMENTATION PROCESS (ADAPTED FROM TRAORÉ & MUZY (2006))

This is what also substantiates the challenge of performing optimization with simulation in the real-world. There are many complex real-world simulation models that have numerous decision variables which values can be specified to form a treatment. A couple of examples would be the workforce planning model and supply chain models of Accenture. Both models could have numerous decision variables which make them very tedious if not impossible to be optimized manually. Furthermore, in these models, multiple goals can be defined depending on the interests of the organization.

To deal with such a problem, we therefore need to use a structured approach that allows us to find the values of those decision variables in such a way the desired system performances can be optimized. In this case, Operations Research (OR) techniques can be very well used to perform such structured approach. According to Hillier & Lieberman (2001), there are several steps that can be used to systematically find the optimal solutions:

1. Define the problem of interest and gather relevant data.
2. Formulate a mathematical model to represent the problem.
3. Develop a computer-based procedure for deriving solutions to the problem from the model.
4. Test the model and refine it as needed.
5. Prepare for the ongoing application of the model as prescribed by management.
6. Implement.

While the discussion on the modeling and simulation framework has substantiated the problem, following sections will give an elaboration on the construction of optimization problem that is relevant in our research and how to solve them.

2.3 Multi-Objective Optimization Problem (MOOP)

In this section the fundamental theory of the optimization problem that is going to be dealt by the framework is delineated. The section begins with the formal definitions of single and multi-objective optimization problems to delineate the formal distinction between them and this is then followed by the notion of non-dominated solutions which gives the important distinction in the way to deal with the MOOP in comparison to single objective optimization problem. To provide a clear understanding upon the theory, an example of MOOP is presented and elaborated at the end of the section.

A single objective optimization problem is normally presented in the following form:

Minimize $z = f(\boldsymbol{x})$ (2.1)

Subject to $g_i(\boldsymbol{x}) \leq 0$, $i = 1,2, \ldots, m$ (2.2)

$\boldsymbol{x} \geq 0$ (2.3)

$f(\boldsymbol{x})$ is the objective function, where $\boldsymbol{x} \in \mathbb{R}^n$ is a vector of n decision variables, and $g_1(\boldsymbol{x})$ are inequality constraint that consists of m functions that shape the feasible area. This area in decision space is denoted as S with following definition:

$S = \{\boldsymbol{x} \in \mathbb{R}^n \,|\, gi(\boldsymbol{x}) \leq 0, i = 1,2, \ldots, m, \boldsymbol{x} \geq 0\}$ (2.4)

However, in modeling a decision problem as an optimization problem, we can model different criteria or system outputs as single or multiple objective(s) that have to be optimized. This type of optimization problem belongs to the class of multi-objective optimization problem (MOOP). In such a problem, often the optimization of one objective results in a decrease of the solution quality for the other objective(s) (Deb, 2005).

Following is the formulation of a multi-objective optimization problem:

Minimize $[z_1 = f_1(\boldsymbol{x}), z_2 = f_2(\boldsymbol{x}), \ldots, z_q = f_q(\boldsymbol{x})]$ (2.5)

Subject to $g_i(\boldsymbol{x}) \leq 0$, $i = 1,2, \ldots, m$ (2.6)

$\boldsymbol{x} \geq 0$ (2.7)

To provide graphical visualization regarding the problem, often the multi-objective optimization problem is drawn in both decision space and criterion space. While S is used to denote the feasible region in the decision space, we can define Z as the feasible region in the criterion space with following expression:

$Z = \{\boldsymbol{z} \in \mathbb{R}^q \,|\, z_1 = f_1(\boldsymbol{x}), z_2 = f_2(\boldsymbol{x}), \ldots, z_q = f_q(\boldsymbol{x}),\ \boldsymbol{x} \in S\}$ (2.8)

where $\boldsymbol{z} \in \mathbb{R}^k$ is a vector which contains the values of q objective functions for particular x values. This means, Z is a set of values that are mapped by the $f_q(x)$ for all points in S.

- **Non-dominated solutions**

Given the definitions of decision space and criterion space above, the following section describes the important notion of non-dominated solutions which is nowadays used widely to approach multi-objective optimization problems.

Fundamentally, multi-objective optimization problem is different from single objective optimization problem. In single objective optimization, the search process is focused on finding one best solution that is superior to all other solutions. In the case where there are multiple objectives to optimize, it is not always the case that there exists a solution that is optimal in terms of all the objectives due to *incommensurability* and *conflict* among the objectives (Gen, Cheng, & Lin, 2008). In case there is conflict among the objectives, a solution that is optimum in one objective may be suboptimal along the other objectives. In this condition, there is normally a set of solutions that cannot be compared with each other without adding additional information (such as preference structure upon the objectives). These kinds of solutions are normally regarded as *non-dominated* solutions or *Pareto optimal solutions* (Marler & Arora, 2004). Non-dominated solutions have the characteristic that their optimality cannot be improved further without sacrificing at least one of the other objective functions (Marler & Arora, 2004). A frontier is normally formed by the non-dominated solutions in the criterion space Z and it is called the Pareto front. For each non-dominated solution in the Pareto Front, there is a corresponding point in the decision space *S* that is called *efficient* or *non-inferior* solution (Marler & Arora, 2004). On the other hand, a point in *S* is considered efficient if and only if its corresponding point in *Z* is a non-dominated point. The concept of Pareto optimal solutions and dominated solutions is shown in Figure 7.

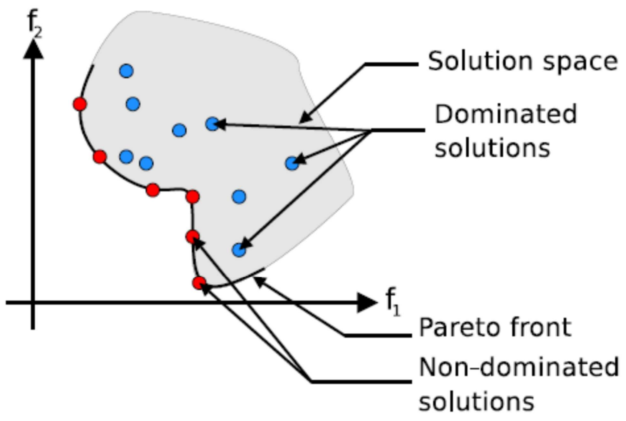


FIGURE 7 VISUALIZATION OF PARETO FRONT

Definition 1.1 A solution $z^1 \in Z$ is considered to be non-dominated by other solutions if and only if (for a maximization case) there does not exist other solutions $z \in Z$ such that:

$z_k > z_k^1$, for $k \in \{1,2,\ldots,q\}$ and (2.9)

$z_l \geq z_l^1$, for all $l \neq k$ (2.10)

Definition 1.2 A solution x^1 is defined as efficient if and only if (for a maximization case) there does not exist other point $x \in S$ such that,

$f_k(x) > f_k(x^1)$, for $k \in \{1,2,\ldots,q\}$ and (2.11)

$f_l(x) \geq f_l(x^1)$, for all $l \neq k$ (2.12)

A point in the decision space is *efficient* if and only if its image is a non-dominated point in the criterion space Z.

To provide a better understanding to the above definitions, the following linear programming problem is exemplified:

Max $f_1(x_1,x_2) = -x_1+3x_2$ (2.13)

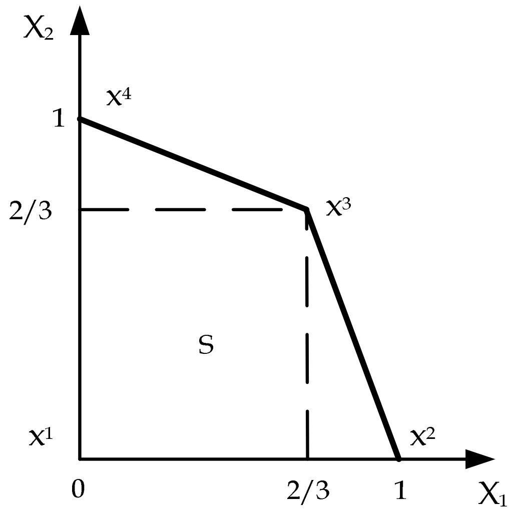
Max $f_2(x_1,x_2) = 3x_1+x_2$ (2.14)

s. t. $g_1(x_1,x_2) = x_1+2x_2-2 \leq 0$ (2.15)

$g_2(x_1,x_2) = 2x_1+x_2-2 \leq 0$ (2.16)

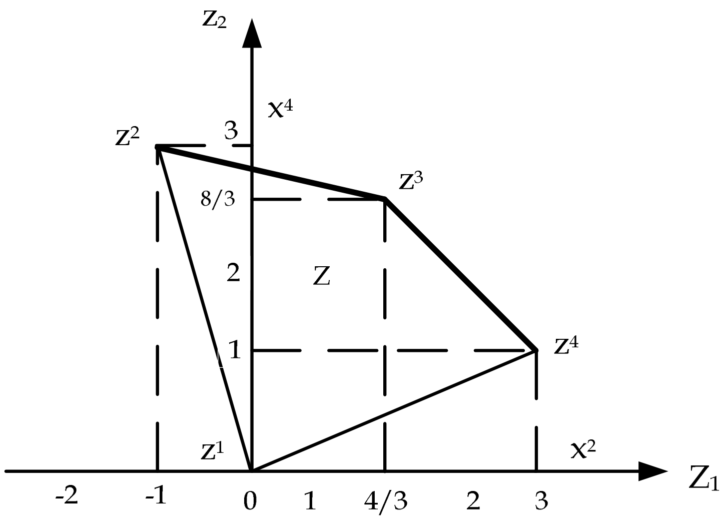
$x_1, x_2 \geq 0$ (2.17)

The feasible region *S* in the decision space is shown in Figure 8. The extreme points in the feasible region are $x_1(0,0)$, $x_2(1,0)$, $x_3(2/3,2/3)$, and $x_4(0,1)$. The feasible region *Z* in the criterion space is obtained by mapping set *S* by using two objectives (Eqs. 1.13 and 1.14), as shown in Figure 9. The corresponding extreme points are $z_1(0,0)$, $z_2(-1,3)$, $z_3(4/3,8/3)$ and, $z_4(3,1)$. From the figures we observe that both region are convex and the extreme points of *Z* are the images of extreme points of *S*. Looking at the points between $z_4$and z_3, it is noted that as $f_2(x_1,x_2)$ increases from 1 to 8/3, $f_1(x_1,x_2)$ decreases from 3 to 4/3, and accordingly, all points between $z_4$and $z_3$are non-dominated points. In the same way, all points between $z_3$and z_2 are also non-dominated points. The corresponding efficient points in the decision space are in the segments between point $x_2$and x_3, and between point $x_3$and x_4, respectively.



$x^1 = [0,0]$
$x^2 = [1,0]$
$x^3 = [2/3, 2/3]$
$x^4 = [0,1]$

FIGURE 8 DECISION SPACE S



$z^1 = [f_1(x_1), f_2(x_1)] = [0,0]$
$z^2 = [f_1(x_2), f_2(x_2)] = [-1,3]$
$z^3 = [f_1(x_3), f_2(x_3)] = [4/3,8/3]$
$z^4 = [f_1(x_4), f_2(x_4)] = [3,1]$

FIGURE 9 CRITERION SPACE Z

- **Convexity of the solutions space**

Another important notion related to the shape of the solution space is convexity. A solution space is convex if every pair of points within the solution space can be connected by a line segment which points also lie within the solution space. Based on that definition, it is noteworthy that the shape of a Pareto front can be non-convex if the shape of the solution space is non-convex. Figure 7 and 10 exemplify non-convex shapes of the Pareto front. Although non-convex shapes of Pareto frontier are relatively uncommon, examples are noted in the test cases presented by Deb (2002).

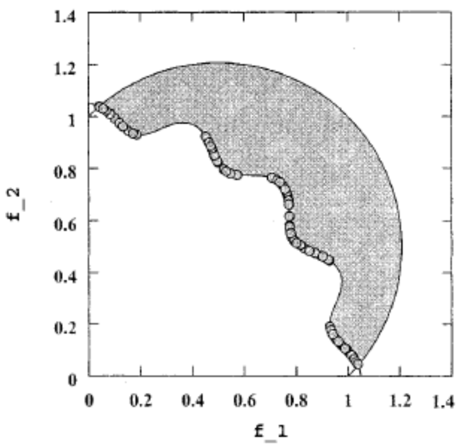


FIGURE 10 NON CONVEX SHAPE OF PARETO FRONTIER (RETRIEVED FROM DEB (2002))

Non-convex Pareto front adds a difficulty to the optimization process as some classical multi-objective optimization algorithms are not able to find non-dominated solutions that are located in the non-convex region of the front. Next section will elaborate this problem further.

2.4 Multi-Objective Optimization Algorithms

Given the formal definitions of multi-objective optimization problem above, different methods to solve this problem are now presented as part of the OR steps explained in section 2.2. Nowadays, there is a wide selection of multi-objective optimization algorithms that one can use to solve MOOP that ranges from classical to the metaheuristic methods. Surprisingly, most of the optimizers in the various simulation packages presented in Table 1 still employ these classical methods. Therefore, to provide an objective assessment in selecting the suitable optimization method for the framework, both classical and advanced methods will be explained. The section begins with two generally used classical multi-objective optimization methods and subsequently presents the state-of-the art multi-objective evolutionary algorithms from which the algorithm for the framework is chosen.

2.4.1 Classical Multi-objective Optimization Methods

The delineation of the classical methods that are presented in this sub-chapter is intended to highlight the problem of inefficiency and inefficacy in the finding of the non-dominated solutions if those methods were used as part of the proposed framework. The main characteristic of the classical methods presented here is that they alter the formulation of the multi-objective optimization problem such that the available optimization algorithms can be applied by taking into account the complexity of the optimization problem as well.

- **Weighted-sum method**

This approach aggregates the different objectives into a single objective by multiplying each objective with a relative weight that is provided by the decision maker(s). This is the most simple and widely used classical approach. Furthermore, to avoid the scaling effect which may be caused by different scaling factors used in the objective functions, normalization is normally applied. A single objective function can then be formed by summing the weighted-normalized objectives. Following is the mathematical representation of such method.

$$\text{Minimize } F(\boldsymbol{x}) = \sum_{n=1}^{N} w_n f_n(x) \quad (2.18)$$
$$\text{s. t. } g_j(\boldsymbol{x}) \geq 0 \quad (2.19)$$
$$h_k(\boldsymbol{x}) = 0 \quad (2.20)$$
$$x_i^{(L)} \leq x_i \leq x_i^{(U)} \quad (2.21)$$

$w_n \in [0,1]$ is the weight of the n-th objective function. It is a normal practice to weight the n objective functions such that its sums would equal to 1 or $\sum_{n=1}^{N} w_n = 1$. Once this problem is solved by any optimizer, the result will be a single pre-destined optimal solution on the Pareto front. In principal, different non-dominated solutions can be found by changing the weight vector. However there are some drawbacks of this method:

1. Different weight vectors have to be evaluated perpetually to find a solution that satisfies the decision maker(s) (Ragsdale, 2008). For decision maker(s) who have a difficulty to specify appropriately the weight vector, the process of fine-tuning the weight vector can be tedious and inefficient.
2. Applying the same weight vectors to different multi-objective problems doesn't necessarily result in the discovery of Pareto-optimal solutions (Srinivas & Deb, 1995).
3. The method is not able to find the non-dominated solutions which are located on the non-convex portion of the Pareto front (Deb, 2005).

The first drawback makes this method very inefficient to be used in a simulation-base optimization method. This is because running a simulation to evaluate the objective functions takes significantly longer time than to evaluate a standard mathematical model. Hence, fine-tuning the weight vector until an acceptable solution is found may take a long period of time. The second drawback brings a difficulty to apply this method to different multi-objective problems without changing the weight vector and evaluate the function again. Thus, this is also undesirable based on the efficiency criterion as the whole process of fine-tuning process has to be performed perpetually for different MOOPs. The third deficiency brings even more serious problem as non-dominated solutions that are located in the non-convex region of the Pareto front can't be allocated. This is because solutions that lie on the non-convex region of the Pareto front will never be optimal solutions for the problem defined by equations 2.18 till 2.21. Linear aggregation of objective functions only succeeds to find the Pareto optimal solutions for linear multi-objective problem and not for non-linear ones which are characterized by the non-convex shape of its decision space (Messac, Puemi-Sukam, & Melachrinoudis, 2000).

- **∈-constraint method**

The main contribution of this method is to overcome the drawback possessed by the weighted-sum method in solving MOOP which has non-convex region on the Pareto front (Marler & Arora, 2004). In this method, multi-objective optimization problem is reformulated by just keeping one objective function and transforming the rest objectives into constraints to which the users can specify the values. Following is the mathematical representation of the problem:

$$\text{Minimize } f_a(\boldsymbol{x}),$$
$$\text{s. t. } f_n(\boldsymbol{x}) \leq \epsilon_n \qquad n = 1,2,\ldots,\text{N and } n \neq a \quad (2.22)$$
$$g_j(\boldsymbol{x}) \geq 0 \qquad j = 1,2,\ldots,J \quad (2.23)$$
$$h_k(\boldsymbol{x}) = 0 \qquad k = 1,2,\ldots,K \quad (2.24)$$
$$x_i^{(L)} \leq x_i \leq x_i^{(U)} \qquad I = 1,2,\ldots,n \quad (2.25)$$

Parameter $\in_n$ is used to represent the upper bound of value of the value f_n. This way, the resulting problem is iteratively divided into several regions based on the number of the objectives and its $\in_n$ parameter values. Pareto optimal solutions which lie at the convex region can therefore be obtained if the values of $\in_n$ are chosen appropriately such that the divided region would not fall into region outside the Pareto front. Figure 11 illustrates the mechanism of this method with two objective functions f_1 and f_2 four distinct $\in_1$ values that a decision maker picks to divide the criterion space. Let us consider a scenario where the decision maker sets $\in_1 = \in_1^c$. This divides the original criterion space into two regions: $f_1 \leq \in_1^c$ and $f_1 > \in_1^c$. Based on equation 2.2, the left region will be the feasible criterion space of which minimum value for the objective function f_2 is sought. From figure below it is clear that the optimal value is found at point C. This way, a non-dominated solution on the middle of the non-convex region of the corresponding criterion can be found by this method.

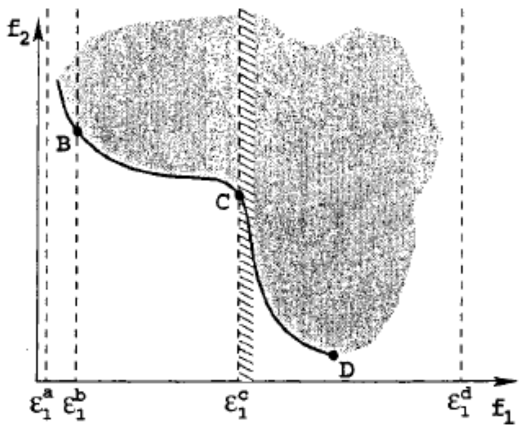


FIGURE 11∈-CONSTRAINT METHOD (RETRIEVED FROM DEB (2005))

However, there are still problems with this approach. The solution found will largely depend on the values of the chosen ∈ vector by the decision maker. Supposedly $\in_1^a$ is chosen instead, there will be no feasible solution for the problem. On the other hand if $\in_1^d$ is used, the entire criterion space is feasible and the optimal solution will be at D. Last but not least, it also does not solve the problem of inefficiency posed by the weighted sum method. Therefore, this method is less suitable to be used in the framework proposed.

- **Goal Programming (GP) method**

This was a popular method to use when dealing with MOOP. In this method, the decision maker(s) should explicitly define the goals or desirable values for each of the objective functions. Those values are used as the additional constraints in the model. Next, the objective functions are formulated to minimize the absolute weighted sum of the deviations (2.26) or of the percentage deviation between the target values and the actual objective values (2.27). The mathematical representation of this model is presented as follows:

$$\text{Minimize } F(x) = \sum_{n=1}^{N} (w_n^- d_n^- + w_n^+ d_n^+) \qquad n = 1,2,\ldots, N \qquad (2.26)$$

Or

$$\text{Minimize } F(x) = \sum_{n=1}^{N} \frac{1}{t_n} (w_n^- d_n^- + w_n^+ d_n^+) \qquad n = 1,2,\ldots, N \qquad (2.27)$$

$$\text{s. t. } x_i + d_n^- - d_n^+ = a_i \qquad i = 1,2,\ldots, I \qquad (2.28)$$

$$g_j(x) + d_n^- - d_n^+ = a_j \qquad j = 1,2,\ldots, J \qquad (2.29)$$

$d_n^-, d_n^+ \geq 0$ for all n (2.30)

$x_i \geq 0$ (2.31)

x_i must be integer (2.32)

Where d_i^- and d_i^+ are the deviational variables that represent the amount of the deviation of each goal from its target value. The parameter w_n represents the weight assigned to the deviational variables and t_n represents the target value for goal n. Variable x_i represents the decision variable i, a_i represents the value of goal i and a_j represents the value of goal constraint j ($g_j(x)$. Both a_i and a_j are elements of t_n.

This approach can be pretty efficient if we know precisely the desired values of the goals and they are in the feasible decision space. However, this method has a drawback as the decision maker(s) have to specify an appropriate weight vector for the objective functions with which the incommensurability between the objective functions could be eliminated. This is normally difficult to do unless the shape of the decision space is known beforehand. Therefore, this method also shares the first and the second drawback of the weighted sum method. In addition to that, there is no guarantee that the solution found will be a Pareto optimal solution (Marler & Arora, 2004).

- **Weighted Tchebycheff method**

This method extends from goal programming method (Ragsdale, 2008). In this method, the deviational variables are formulated explicitly using a mathematical expression that explains the deviation structure between the target value and the actual value. These deviational variables are then used as additional constraints next to the other standard constraints. Some weight values are assigned to these constraints to indicate the preference structure of the decision maker(s). Finally, the objective function in this method is set to minimize the maximum deviations (called the minimax variable) of the actual values from the target values. The mathematical representation of the model is presented as follows:

Min Q (2.33)

s.t. $w_i \frac{di}{ti} \leq Q$ $\quad$ i = 1,2,…, I (2.34)

$g_j(x) \geq 0$ $\quad$ j = 1,2,…, J (2.35)

$x_i \geq 0$ (2.36)

w_i is a positive constant (2.37)

Q represents the minimax variable, d_i represents the deviation between the actual value and the target value t_i. Variable x_i represents the decision variable i, w_i represents the weight assigned to minimax constraint i and $g_j(x)$ represents the standard constraint j.

A feature that distinguishes this method from the other classical methods is that the solutions found with this method are always Pareto optimal. However, since it still requires the decision maker(s) to provide weight values for the minimax constraints, fine tuning process for those values has to be done until the decision maker is satisfied. Therefore, this method shares the first deficiency of the weighted sum method.

2.4.2 Multi-objective Evolutionary Algorithms (MOEAs)

Ever since Evolutionary Algorithms (EAs) had been introduced to solve MOOP, its researches, use and popularity had increased rapidly and significantly over the past decade opening a rapidly growing research field namely Multi-Objective Evolutionary Algorithm. According to a survey announced in the World Congress on Computational Intelligence (WCCI) in Vancouver 2006, MOEA has been considered as one of the fastest growing fields among all research topics in computational intelligence field (Deb, 2008). Many other survey papers conducted in multi-objective optimization field also support this fact (Coello, 2000; Konak, et al., 2006). Furthermore, this fact can also be found on one updated site that lists more than 4800 references related to MOEA encompassing more than 170 PhD dissertations, 780 journal papers, and 1560 conference papers (http://www.lania.mx/~ccoello/EMOO/).

Next to the research of MOEA, there are efforts to extend the other single-objective metaheuristics such as Tabu Search (TS), Simulated Annealing (SA), Differential Evolution (DE), Ant Colony Optimization (ACS), Particle Swarm Optimization (PSO), Artificial Immune System (AIS), Cultural Algorithm (CA) to solve MOOP. Each of those metaheuristics has a specific challenge in their operators to extend it to an algorithm that is capable of finding the non-dominated solutions. They are normally implemented with the classical approaches which don't produce Pareto-optimal solutions as exemplified by the optimizers in Table 1. One of the causes is that most of those algorithms (especially those which don't use population-based approach) require more operators that could increase their effectiveness in finding the Pareto-optimal solutions. The design of these operators is a considerably significant research challenge as it has to result in an algorithm that has a comparable performance to more mature algorithms such as MOEA. However, there have been some successful developments reported for those algorithms. The research of Bandyopadhyay (2008) shows a promising result to develop Simulated Annealing into a solid multi-objective optimizer known as AMOSA (Bandyopadhyay, Saha, Maulik, & Deb, 2008). Furthermore, there have been a considerable number of research showing that Tabu Search can be extended successfully to find the non-dominated solutions, known as Multi Objective Tabu Search (MOTS) (Carcangiu, Fanni, & Montisci, 2008; Kulturel-Konak, Smith, & Norman, 2006). The research of Ray (2002) also presented the extended Particle Swarm Algorithm that successfully finds the non-dominated solutions (Ray & Liew, 2002).

Despite all these developments, MOEA is still often considered as the perfect match in solving multi-objective optimization (Deb, 2008). This is because, next to being a mature multi-objective optimization technology, this method has advantages in comparison with the classical methods and other metaheuristics:

1. MOEAs have the flexibility to be adapted to different problem structures and thus have wide application fields.

2. MOEAs do not require gradient or derivative values such as what is needed by gradient-based algorithms.

3. The decision maker(s) does not need to have an a priori articulation of preferences regarding the accomplishment of all the objectives before the solutions/alternatives are presented (Marler & Arora, 2004).

The last advantage is important in multi-objective decision making especially if there is more than one decision maker with different sets of preferences towards the objectives. This is also the feature that, to a great extent, makes MOEA preferable than classical approaches.

In addition to the advantages above, MOEA also has advantages from using population based-approach. In this approach, multiple solutions are simultaneously generated in each of its iteration based on the combination of the information from the previous iterations and random number. This gives MOEA better capability to explore larger criterion space, including the non-convex region in shorter computational time. Following are the summary of those advantages

1. It gives MOEA a parallel processing power in finding the non-dominated solutions.
2. It enables an MOEA to find multiple optimal solutions, and thus facilitating the finding of solutions for MOOP
3. It provides an MOEA with the ability to normalize decision variables, the objectives and constraint functions. This normalization mechanism is based on the valuation of the best minimum and maximum values in the population.

The last advantage is the key mechanism that allows MOEA to work without explicit preference structure from the decision makers as ranking to the multiple solutions will be done automatically within the sub-routines of the algorithm.

However, there is also a disadvantage of working with population-based approach: it still requires high computational cost and memory to execute each of its iteration which makes it hard to get the solution in short period of time.

- **Evolutionary Algorithms Terminologies and Procedures**

This section describes the common terminologies that are used in Evolutionary based algorithms and the general procedures that this type of algorithm normally uses to search for the optimal solutions. This section, therefore, serves as an introductory to the proposed EA, namely Non-Dominated Sorting Genetic Algorithm II (NSGAII) which is going to be delineated in detail in the subsequent section.

The term EA has been used to provide a unifying category that covers all algorithms that mimic evolution process in nature. In its terminology, a population consists of solutions which its single entity is called an individual or a chromosome. Chromosomes are made of discrete units called genes. Each gene controls one or more features of the chromosome. In the original implementation of Evolutionary Algorithm by Holland, genes are assumed to be binary digits (Holland, 1973). In the later implementations, more varied gene types have been introduced. Normally, a chromosome corresponds to a unique solution in the decision space. This requires a mapping mechanism between the decision space and the chromosomes which is called encoding. A population is normally randomly initialized. As the population evolves, the population includes fitter and fitter solutions, and eventually it converges to the (near) optimal solutions.

EA use two operators to generate new solutions from existing ones: crossover and mutation. In crossover, generally two chromosomes, called parents, are combined together to form new chromosomes, called offspring. The parents are selected among existing chromosomes in the population by evaluating their fitness so that offspring is expected to inherit good genes which make the parents fitter. Further, this fitness is normally formulated as a mathematical function(s) which are normally derived from the objective function(s) which expresses the optimality of a solution. By iteratively applying the crossover operator, genes of good chromosomes are expected to appear more frequently in the population, eventually leading to convergence to an overall good solution. In addition to that, elitism mechanism can also be applied which preserves the most fit individual to always survive to the next generation. The mutation operator introduces random changes into characteristics of chromosomes. As discussed earlier, crossover leads the population to converge by making the chromosomes in the population alike. Mutation preserves genetic diversity within the population and assists the search escape from local optima. Reproduction involves selection of chromosomes for the next generation. In the most general case, the fitness of an individual determines the probability of its survival for the next generation. The higher the fitness of an individual is the higher its survivability is. There are different selection procedures in EA depending on how the fitness values are used. Proportional selection, linear fitness ranking combined with roulette wheel selection, and tournament selection are the most popular selection procedures. EA stop once the fitness criteria have been met or its iteration has reached maximum predefined generation. The procedure of a standard EA is represented in Figure 12.

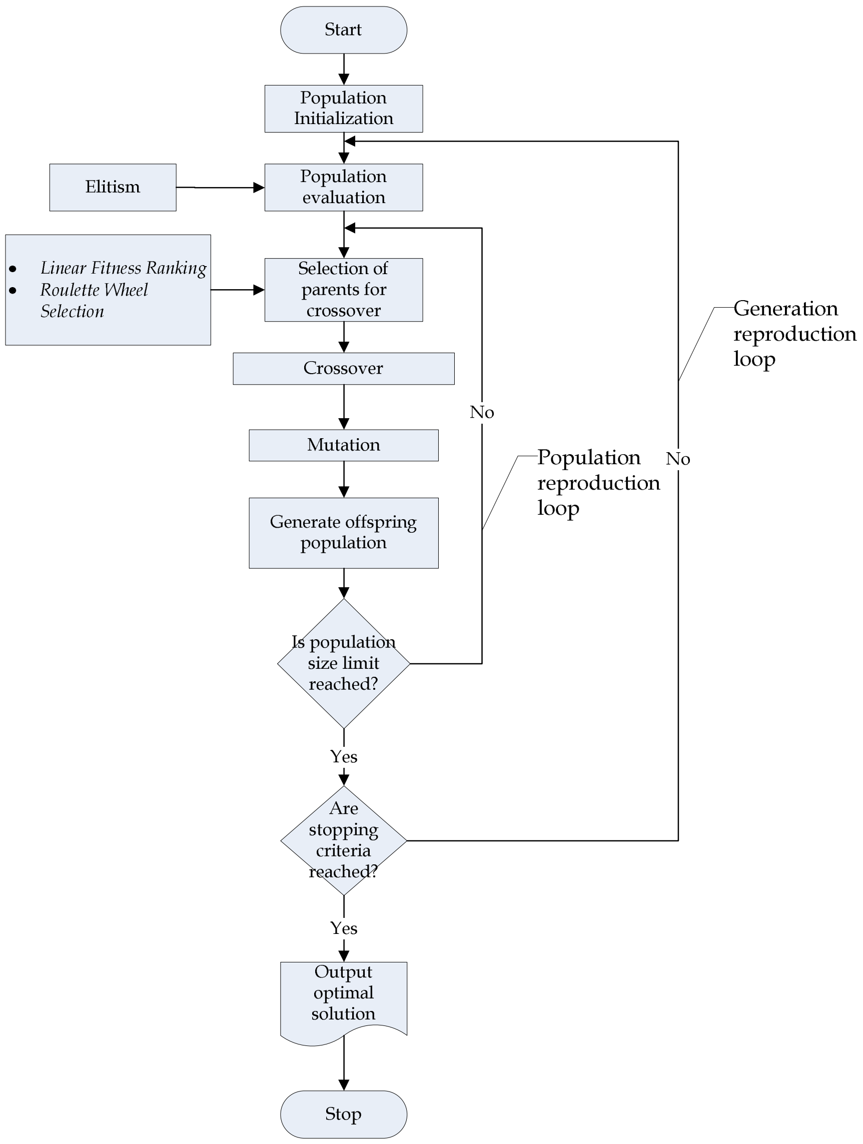

Start
Population Initialization
Elitism
Population evaluation
Linear Fitness Ranking
Roulette Wheel Selection
Selection of parents for crossover
Crossover
Mutation
Generate offspring population
Is population size limit reached?
No
Yes
Are stopping criteria reached?
No
Yes
Output optimal solution
Stop
Population reproduction loop
Generation reproduction loop

FIGURE 12 EVOLUTIONARY ALGORITHM STANDARD OPTIMIZATION ROUTINE

- **The Non-dominated Sorting Genetic Algorithm-II (NSGA-II)**

This section gives a detailed explanation of all the sub-procedures that are used by NSGAII. The explanation is intended to provide detailed information on how the algorithm works to find the non-dominated solutions and to also provide the understanding required to comprehend properly the design challenges that are present in integrating the algorithm into the framework of modeling and simulation presented in the first chapter.

Among the multi-objective evolutionary algorithms that have been developed so far, the non-dominated sorting genetic algorithm II (NSGA II) has been proven to be one of the state-of-the-art methods to solve MOOP (Deb, Pratap, Agarwal, & Meyarivan, 2002; Konak, et al., 2006; Thomson, 2004). Like any other EAs this algorithm also finds its root in evolutionary theory and built on top of the classic genetic algorithm. Despite of many improvements and additional features that NSGAII has, the fundamental theory that underlies the adaptive capability of this algorithm is the same as that is used in the genetic algorithm. However, there are some features that distinguish NSGA-II from other EAs:

1. It employs the elitism principle
2. It has an explicit diversity preserving mechanism
3. It focuses on the non-dominated solutions

In the NSGAII, two sets of solutions are first generated using the genetic operations and then aggregated. These solutions are then compared one another and sorted based on the values of the objective functions. Solutions that are non-dominated by others are ranked and categorized in different fronts. The higher the rank of a solution the more prioritized it is to be included in a new set of solution. This new solution set will have half the size of the aggregated solutions and it will be used as the initial solution set for the next iteration. It is created by selecting the non-dominated solutions based on their ranks and the diversity of the solutions that surround a non-dominated solution. A non-dominated solution which objective values are not close to the other non-dominated solutions will receive priority to be included into this new solution set. This way, the solutions that are included in the subsequent iteration will always have a better quality than the previous solutions and they encompass non-dominated solutions that are well spread among the multiple objectives.

Firstly, a new child population Q_t is created through the genetic operators (i.e. crossover and mutation) from the parent population P_t .Next, those two populations are aggregated to form a new population R_t which has the double size of P_t. Next, the non-domination sorting procedure is applied to R_t to categorize the solutions/chromosomes into non-dominated solutions which form different Pareto fronts. After the procedure is finished, (new) R_t is then filled with these solutions from different Pareto front one at a time. Solutions that are contained in the best non-dominated front are inserted to the new R_t in the first place and then continued with the solutions from the second best Pareto front and so on. Since the size of the population for the next generation has to be kept in the same size of P_t, half of the solutions in R_t have to be deleted. Special treatment is given to the solutions that are contained in the last front that is allowed to be part of R_t. as the number of the solutions there may exceed the available place in R_t. This is done by accepting solutions in the last front that will make the diversity of R_t the highest. Figure 13 illustrates the algorithm procedure.

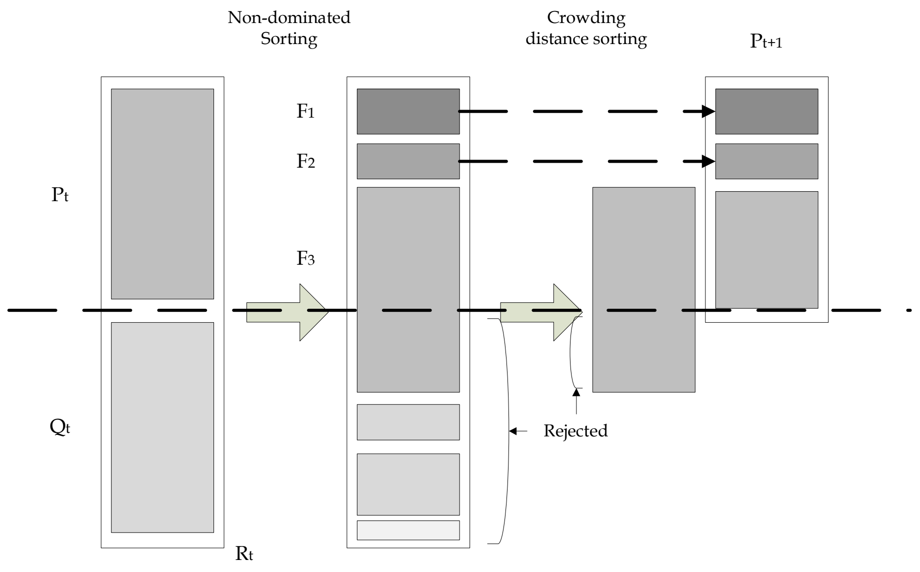


FIGURE 13 SCHEME OF NSGA-II PROCEDURE (ADAPTED FROM DEB (2002))

NSGA-II iteration is summarized in the following:

1. Parent and offspring populations are combined into a new population $R_t = P_t \cup Q_t$.
2. Non dominated sorting procedure is applied to R_t to identify the fronts F_i , i = 1,2,…,etc.
3. A new population $P_{t+1}=\emptyset$ is generated and the iterator i is set to 1. Following routine is performed:

   While $|P_{t+1}| + |F_i| < N$, do

   a. The crowding-distance assignment procedure (F_i)

   b. $P_{t+1} = P_{t+1} \cup F_i$

   c. $i = i + 1$.
4. The crowding tournament selection procedure ($F_i, <_c$)
5. $P_{t+1} = P_{t+1} \cup F_i[1: (N - |P_{t+1}|)]$
6. Q_{t+1}= genetic operations (P_{t+1})$\rightarrow$ using selection, crossover and mutation
7. $t=t+1$$\rightarrow$ incrementation of the generation counter

To perform the non-dominated sorting procedure, all solutions in the first non-dominated front will first have to be identified. Afterwards these solutions will be used to identify the solutions that are located in the subsequent fronts. The following is the routine used to perform fast non-dominated sorting procedure in step 2:

1. for each solution $i \in R_t$, two calculations are performed:

   a. The domination count, the amount of solutions that dominate solution i, n_i. For the first iteration this is set to zero.

b. A set of solutions that are dominated by the solution i, S_i. This set is still empty in the beginning.

c. For each solution $j \in R_t$

```
If $i$ dominates $j$ ($i<j$) then
    $S_i=S_i \cup \{j\}$
Else if $j$ dominates $i$ ($j<i$) then
    $n_i = n_i+1$
If $n_i= 0$ then
    $r_i =1$
    $F_1 = F_1 \cup \{i\}$
```

2. The allocation of the dominated solutions to the next front

```
$m=1$
While $F_m \neq \emptyset$
  $Q = \emptyset$
  For each $i \in F_m$
     For each $j \in S_i$
        $n_j= n_j-1$
        if $n_j=0$ then
           $r_j = m +1$
           $Q = Q \cup \{j\} \rightarrow j$ is going to be allocated into the next front
$m= m+1$
$F_i = Q$
```

The crowding-distance assignment procedure that takes place in step 3 is performed by using a crowding distance metric which is used to obtain the estimate of the density of the solutions that surround one certain solution *i*. The average distance of two solutions that are located along the objective axis of solution *i* is used to calculate the estimated perimeter value d_i (the crowding distance of *i*-th solution). It is the average side-length of the cuboid formed by the vertices of the nearest neighboring solutions. Figure 14 visualizes the crowding distance of the *i*-th solution in its front.

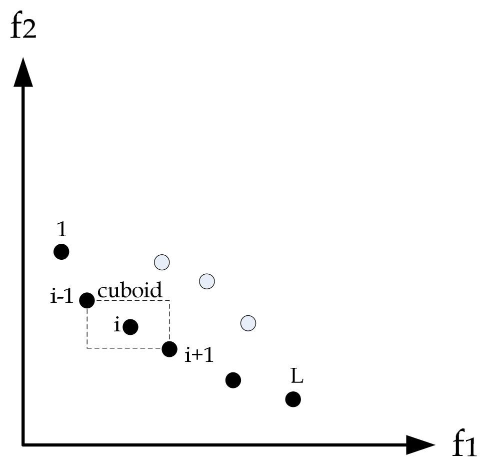


FIGURE 14 THE ASSIGNMENT OF CROWDING DISTANCE (ADAPTED FROM DEB (2002))

Crowding distance assignment procedure: *Crowding-sort* (F , $<_c$)

1. The number of solutions in F is named as $k= |F|$.
2. For each i in the front F, a variable d_i=0 is assigned.
3. For each objective function m=1,2,…M,
    a. The solutions in F are sorted in the worse order of f_m. The sorted solutions are put into vector I^m = sort (f_m, >).
    b. Large distance is assigned to the boundary solutions.
    c. for solutions j=2 to (k-1), assign:

$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{(I_{j+1}^m)} - f_m^{(I_{j-1}^m)}}{f_m^{max} - f_m^{min}}$$

I_j denotes the index of the jth member in the solution vector I^m, and d_i in this vector is denoted as $d_{I_j^m}$. Therefore, for each of the objectives, I_1 and I_k are the lowest and the highest objective function values respectively. The second constituent of the right hand side equation is the difference between the preceding and succeeding solutions of solution I_j in the vector I^m. For two objective-axis, this metric gives half of the perimeter values of the cuboid (Figure 14). In this metric, the solutions between a solution i don't necessarily be the neighbors in all objectives, especially in case where there are more than two objectives.

In step 4, crowding tournament selection procedure is applied to the last font (F_i) which normally cannot be included completely to the population P_{t+1}. In this operation, crowded comparison operator is used ($<_c$), which task is to compare two solutions and return the winner of the tournament to be included into the population (step 5). The inputs of this operator are:

1. Non-domination rank r_i of the population
2. Local crowding distance (d_i) of the population

Following is the definition for crowded tournament selection operator: a solution *i* is defined to win against another solution *j* if one of the following conditions is satisfied:

1. If the rank of solution *i* is better than the rank of solution *j*: $r_i<r_j$
2. In the case both solutions have equal rank, but solution *i* has a better crowding distance than solution *j*: $r_i= r_j$ and $d_i>d_j$.

The first requirement makes sure that the chosen solution is located in a better front than the other solution. The second requirement is used in case both solutions are located in the same front. Based on this requirement, solution which lies in less crowded area (the one with higher d_i value) is chosen.

The overall computation complexity of one NSGA-II generation is known to be $O(MN^2)$ with non-dominated sorting procedure as the most computation-intensive procedure.

Based on the abovementioned features of NSGA-II, we can infer that the properties of NSGA-II satisfy most of the desired features for a generic simulation-based multi-objective optimization framework (except for the usability that requires the development of a user friendly user interface). Generality can be achieved by utilizing the flexibility of this algorithm to solve different types of problems. This is mostly done by adapting the design of the genetic chromosome and operations such that they are not problem dependent (this will be thoroughly elaborated in chapter 4). Efficiency is fulfilled by the relatively small computation complexity of the algorithm and finally, multi-dimensionality is fulfilled by the suitability of the population-based approach to solve multi-objective optimization problems. Therefore, this research paper proposes to use NSGA-II as a representative of multi-objective evolutionary optimization algorithm to be part of the proposed framework. Henceforth, we name the framework to be Simulation-based Multi-objective Evolutionary Optimization (SIMEON) framework.

- **Other state of the art multi-objective evolutionary algorithms**

Besides NSGA-II, the rapid development of research in MOEA has brought about the developments of other well-known evolutionary based algorithms such as Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler, Laumanns, & Thiele, 2001). Pareto Archived Evolutionary Strategy (PAES) and Pareto Envelope based Selection Algorithms (PESA).

The main difference between SPEA 2 and NSGA-II lies in the use of external population by SPEA 2 which stores all the non-dominated solutions from the initialization of the first population till the last generation. In each generation, the external population and the current population are combined and genetic operations are applied. All the non-dominated solutions are then assigned based on the amount of the solutions that are dominated and the dominated solutions are assigned a fitness one value higher than the sum of fitness of the solutions that dominate them. The diversity preservation mechanism is done by using a certain clustering routine to maintain a fixed external population size. The computational complexity of this algorithm might vary between $O(MN^3)$ or $O(MN^2)$ depending the bookkeeping method that is used (Deb, et al., 2002).

Pareto Envelope based Selection Algorithms (PESA) is the improved version PAES and SPEA (Deb, 2008). Like SPEA 2, PESA uses two populations (a normal population and a larger archive population). The non-domination concept that is used here is similar to that of PAES where the offspring is compared with the parent and the one which is better is set as the next parent. If, however, the parent is better than the offspring is discarded and a new offspring is created by mutation operation. In case both parent and offspring don’t dominate each other, crowding procedure is used to select between those two. In this procedure, the child is compared to the archive population in terms of its fitness. If there is no solution in the archive that is dominated by the offspring, then parents and offsprings are compared in terms of their proximity to the archive population. Solution that lies in the less crowded area is accepted as a parent. In the extended version of PESA, improved selection procedure is applied to increase the speed of the algorithm and the diversity of the solutions.

2.5 Generic Multi-objective Optimization Algorithm Development

From the optimization perspective, the developments of special algorithms that work effectively to solve multi-objective optimization problems such as MOEAs give rise to a specific extension of simulation-based optimization research, namely simulation-based multi-objective optimization. However, the challenges that are dealt in this research, to a great extent, still lie in the adaption of the algorithm components (chromosomes) and routines (operations) to provide the desired features, (with generality as the main concern) for solving different real-world problems (Azadivar & Tompkins, 1999; Fu, et al., 2000; Pierreval & Paris, 2003).

Evolutionary-based algorithms have the flexibility to either be specialized to solve problems in certain domains (i.e. by taking into account the structure of the problem and incorporating it into the design chromosome) or to be generic such that it can be applied to all optimization problems covering discrete and continuous optimization problems. The first case has been shown by the implementation of this type of algorithm to the domain specific simulation problem(Amodeo, et al., 2007; Ding, et al., 2006; Hani, et al., 2008; Persson, Grimm, Ng, et al., 2006) and the second case can be found in many of general parametric optimization studies of simulation models (April, et al., 2003; Fu, et al., 2000).

Looking at the applications of simulation in supply chain management and logistic, there are generally two types of design decision that can be supported by simulation (Thierry, et al., 2008):

1. The design of the supply chain network and operations, which encompasses following decisions:
   - Localization
     i. Location of facilities
     ii. Supply and distribution channel configuration
     iii. Location of stocks
   - Selection
     i. Suppliers
     ii. Partner
   - Size
     i. Capacity booking of a certain facility
     ii. Stock level of the inventory
2. The design of the supply chain control policies, which encompasses following decisions:
   - Control policies
     i. Inventory management, control policies
     ii. Planning processes
   - Collaboration policies
     i. Cooperation/collaboration/coordination
     ii. Information sharing

For the abovementioned design decisions, two generic types of decision variables can be identified based on the mathematical modeling paradigm:

- qualitative decision variables

  It is normally used to represent qualitative control and collaboration policies (Ding, et al., 2009; Hani, et al., 2008), localization (Ding, et al., 2009) and in some cases, the structure of a simulation model (Azadivar & Tompkins, 1999; Pierreval & Paris, 2003). This type of variables is normally modeled as discrete/integer or binary values (as in Mixed Integer Linear Programming) which represent all possible alternatives for each design decision.

- quantitative decision variables

  It is used to represent all the quantitative parameters of the simulation model which in turn influence the value of the objective functions such as capacity and stock level values (Amodeo, et al., 2007; Ding, et al., 2009). This type of variable is normally modeled as real/continuous values.

On the other hand, the application of EA in simulation models has also brought about a different paradigm of representing solutions in regards to the design decisions. In this kind of application, solutions that are encoded in form of a chromosome might contain more information than just qualitative and quantitative decision variables but also the information regarding the structure of the simulation model (i.e. the structure of the supply chain network)(Ding, et al., 2006; Ding, et al., 2009). Despite of its alleviated efficiency in helping the decision maker in finding the best design decision from large number of structural alternatives, this type of representation has the drawback of being tightly coupled to the problem structure. In a solution representation where knowledge regarding structure of the problem is represented in such a way standard genetic operators don't work, a problem specific operator or repair heuristic has to be designed to maintain the feasibility of the solutions (Rothlauf, 2006). This way, the genetic operators or the repair algorithm has to be re-designed each time the algorithm is going to be applied to a different problem. As a consequence, this design decision would definitely hamper the achievement of the required generality that allows the algorithm to be applied to the other problem domains such as business planning, etc.

Therefore, to maintain the generality of the framework, it is proposed that the types of the decision variables to optimize are confined to qualitative and quantitative variables with which evaluation using formal mathematical evaluation can be performed without a custom-made repair heuristic yet by using scientifically-proven standard genetic operators[3]. This aspect is also important to maintain the usability of the framework for the non-expert users.

Furthermore, the other challenge that presents itself as a natural consequence of the design decision on the genetic chromosome lies in the design of the genetic operators i.e. crossover and mutation. These operators have to also be designed based on the way the solutions are represented in a chromosome such that the algorithm would effectively search the decision space and eventually converge to non-dominated solutions.

---

[3] The book: "Representations for Genetic and Evolutionary Algorithms" provides a profound discussion regarding the way solutions are represented in Genetic and Evolutionary Algorithms and its relevance with the standard genetic operators.

Based on the proposition above, it becomes clear that the adaption needed by genetic algorithm such that the desired generality can be realized lies in the design of the genetic chromosome and operators. Therefore the second research question is refined from:

2. How can we adapt genetic algorithm such that it can provide the desired generality and efficiency for the framework?

to:

2. How should the genetic chromosome and operators be designed such that the proposed framework has the capability to support the optimization of quantitative and qualitative variables which enables it to cover continuous and discrete multi-objective optimization problems?

2.6 Conclusion

In this chapter we have presented the fundamental theories that support the development of the proposed simulation-based optimization framework. Firstly, the problems regarding the currently used simulation-based optimization methods in COTS have been delineated. This is then followed by the extensive delineations on theoretical foundations from both simulation and optimization fields to explain the problems and the possible solutions in detail. Lastly, based on the exposition on the relevant research in simulation-based optimization field, we also present the redefinition of the second research question. This is necessary to narrow down the focus of the research and to position clearly the contribution of this research paper. The next section would make use the theories presented here to form the conceptualization of the proposed framework.

3. SIMEON DEFINITION

In this chapter the first systems engineering life-cycle phase is performed: system definition (Sage & Armstrong, 2000). This thesis project adopts two sub-phases to define the framework proposed:

1. Requirements and specifications of SIMEON
2. Conceptual design of SIMEON

In the first sub-phase, the technical requirements are elicited to form the foundation of the framework based on the stakeholder needs. The method that is used to gather all the requirements is through interviews and brainstorming together with the clients and experts of decision support systems. Two requirement frameworks (i.e. from Fu (2002) and Keen and Sol (2007)) will be discussed and worked out to structure the technical requirements.

After, the technical requirements have been formulated and agreed, the engineering process is continued to SIMEON conceptualization. In this second sub-phase, the integration of the selected optimization technique into the framework of modeling and simulation will first be presented to provide the partial answer to the first research question. Lastly this chapter is ended with a conclusion.

3.1 SIMEON REQUIREMENTS AND SPECIFICATIONS

The first sub-phase of the systems engineering design life-cycle which is part of the system definition phase is the requirements and specification phase. In this phase the requirements of the system that is going to be developed are captured, identified and defined by the system engineer together with the end user(s) of the system. This effort results in the identification of user requirements and the translation of those requirements into the technological specifications for the SIMEON framework (Sage & Armstrong, 2000).

This phase also contributes a (partial) answer to the 5th research question that puts its emphasis in investigating the contribution of the SIMEON in increasing the technological capability and expertise of SIMEON user in solving real world problems. In order to deliver a trustworthy system which would manage to produce such contributions, it is vital that the requirements which are formulated really capture the needs of the users and properly scope the design project.

In the effort of eliciting the technical requirements, it is important to use a structured approach to ensure that all aspects of the requirements are covered and communicated effectively between the user and the developer of SIMEON. This research adopts two requirement frameworks that are proposed by Keen and Sol (2007) and Fu (2002).

Keen and Sol (2007) lay down the foundation for the definition of an effective decision support system. According to Keen and Sol (2007) the effectiveness of decision support systems can be expressed using the following notions:

- Usefulness. Usefulness is used to define the added value that a decision support system can contribute to a decision making process. This aspect is therefore related to the value that can be added by SIMEON in solving complex real-world optimization problems.
- Usability. Usability is used to define the mesh between people, process and technologies. It is mainly embodied by the interface between the users and the decision support tool. This

aspect is therefore, heavily dependent upon the facilitation/services that the interface can provide to help the user using the tool in an effective and efficient manner.

- Usage. Usage is used to define the flexibility, adaptivity, and suitability of a decision support tool to the organizational, technical or social context. Thus, this is the measure that is determined by how the needs of an organization in dealing with the real world problems can be fulfilled.

On the other hand, Fu (2002) have also identified four high-level requirements/features that are desired specifically for decision support systems that are based on simulation and optimization techniques. Those features have been preliminary discussed in chapter 1 and further adapted to fit the context of this research. They are:

1. Generality
2. Efficiency
3. Multi-dimensionality
4. Usability

Looking at both requirement frameworks, it is noticeable that there are coherencies between the notions that are used by both frameworks to define an effective/desirable decision support system. However, since the notions that are put forward by Keen and Sol (2007) are more generic than the ones of Fu (2002), we use the notions of Keen and Sol (2007) as the starting point of the requirements and translate those requirements into the ones that are defined by Fu (2002) to delineate more elaborated technical requirements for SIMEON. In the context of simulation-based optimization tool, usefulness can be further specified into generality and efficiency, usage into multi-dimensionality while usability has a one- to-one correspondence in both frameworks.

The other underlying motivation to use the requirements formulated by Fu (2002) is because his research was focused on the real-world practice of simulation and optimization and the gaps between the development of research in this field and the features which are desirable for the real –world users of this method. A review from a panel of experts in the simulation and optimization field was also done to bring forth the desired features formulated by Fu (2002). The panel is composed of simulation software developers and academic researchers (Fu, et al., 2000).

In this sub-chapter those requirements are further detailed and translated into technical requirements by substantiating the low-level requirements and providing the performance measures for those requirements. The phase where the translation from the user requirements to technical requirements takes place is not necessarily performed here as the users have sufficient knowledge and ability to frame the technological specifications of SIMEON framework. This approach of defining the SIMEON is adapted from system definition matrix suggested in Sage (2000).

3.1.1 Specification of the Technical Requirements for SIMEON

In this sub-chapter, the elaborations of the high level requirements based on the interview with the user and literature study are presented. The chapter begins with the highlight of the important requirements that would influence the design decisions of SIMEON and followed by a table that provides the overview of the whole requirements.

Generality

Generality in SIMEON is positioned as a requirement which has the highest priority over the other requirements. In order to appropriately meet the generality requirement from the user, proper contextualization and scoping based on the needs of the user are necessary (Sage & Armstrong, 2000).

From the interview with the user, it is revealed that there are following needs that are relevant to the desired generality:

1. The need to use the SIMEON as a decision support tool to solve problems in different domains, particularly in the fields of business planning, logistic and supply chain management.
2. The need to connect SIMEON to widely used Microsoft office software such as Microsoft excels and access.
3. The need to being able to run SIMEON in different operating system platforms
4. The need to have normal simulation and simulation-based optimization services that are easy to couple/ decouple.
5. The need to develop SIMEON for DSOL-based models.

The first need entails a technical requirement which is particularly related to the optimization technique that is going to be used by SIMEON. Important factors that motivate the optimization technique to select would be based on how much information can the optimization algorithm get from the simulation model/problem and how that information should be incorporated into the search process of the algorithm in such a way the algorithm does not need to be re-designed/laboriously adjusted to solve problems in the different domains (Fu, et al., 2008). In the context of obtaining high level of generality, many researches in simulation-based optimization suggest an optimizer that works based on a metaheuristic approach (April, et al., 2003; Fu, 2002; Fu, et al., 2000). This approach only requires the output of the simulation model as the main input for its routines and considers the simulation model/ problem as a black box. This way, there is no problem of incorporating specific information into the algorithm which in turn requires the adaption of the algorithm each time it is going to be applied to solve different problem.

Another essential need that would impose important requirement is the need to develop SIMEON for DSOL-based simulation models(Jacobs, 2005). This need consequently translates into a technical requirement where the development of SIMEON should take into account on what platform DSOL has been developed, how the experimental frame of DSOL has been implemented and how the routine of the algorithm that would serve as the optimizer should be modified to design the integration.

Efficiency

Requirement on efficiency is inevitably affected adversely by the requirement of generality. One consequence of treating the simulation model as a black box (as what is done by metaheuristic) is that the optimizer does not make use the information regarding the problem structure (e.g. gradient information) to solve the optimization problems (Fu, 2002). This leads to a relatively slow performance in comparison to the optimizers that use such information to solve the same problem. However, this condition also puts forward a requirement that is meant to mitigate this deficiency: SIMEON should be extendible with algorithms that could make use the information regarding the structure of the simulation model such as the perturbation analysis (Fu & Hu, 1997), weak derivatives (Pflug, 1996), etc.

Another technical requirement that is put up concerns the data flow between the optimizer and the simulation model in SIMEON. Since simulation-based optimization is a computationally expensive approach, it is proposed that the optimizer and the simulation model don't use an intermediary external file to facilitate (e.g. text file or database file that are in the other platform) the data transfer from and to the both modules as this would add more computational time to the whole process of simulation and optimization (Persson, Grimm, & Ng, 2006).

Multi-dimensionality

Multi-dimensionality is a requirement that gives SIMEON a distinction in comparison to the other simulation-based optimization framework. This requirement mainly comes from the need to optimize multi-objective optimization problems which are ubiquitous in the real world. Furthermore, there are also needs that come from the modeling perspective, they are:

1. The need to optimize quantitative or continuous or real-valued variables as this need still dominates big number of optimization problems (Amodeo, et al., 2007; Ding, et al., 2009).
2. The need to optimize qualitative or discrete or integer variables which can be used to represent non-quantitative variables including structural alternatives for a simulation model (Azadivar & Tompkins, 1999; Pierreval & Paris, 2003)
3. The need to optimize both qualitative and quantitative variables simultaneously (Ding, et al., 2006).
4. The need to visualize the optimal trade-offs between multiple objectives (i.e. the non-dominated solutions) in case they are incommensurable and there are conflicts between one another.

To fulfill the requirements that can be derived from the needs stated above, SIMEON has to be tested and its preliminary target performances have to be defined. This test would be useful to understand the preliminary capability and limitation of SIMEON. However, considering that many experimentations would be needed to determine the maximum capability of SIMEON, the preliminary target performances in terms of multi-dimensionality (e.g. number of objectives, number of qualitative, and quantitative decision variables to optimize, etc.) are currently defined by the test cases that are going to be used. There is currently also an ongoing research that is focused to develop a test bed for simulation-based optimization methods which developments will be useful for this research (Pasupathy & Henderson, 2006).

Usability

To make a usable user interface, it is important to distinguish and define the users of SIMEON. The first type of the user is defined as non-expert users, i.e. those who have at least the basic knowledge of modeling and simulation techniques. They are expected to have the knowledge similar to a student who has accomplished the study goals of discrete event simulation course (EPA 1331) given at Technology, Policy and Management (TPM) faculty of Delft University of Technology (DUT) for academic period 2009/2010. Therefore, the non-experts are expected to:

- know the role of dynamic systems modeling within the process of problem solving and policy making;
- be able to apply mathematical techniques such as distribution functions, hypothesis testing, and queuing theory to model small dynamic problems;
- be able to apply the modeling cycle to the development of discrete models;
- have basic knowledge of all the steps in discrete event systems modeling;
- know different techniques used in discrete simulation and knows when to apply these (conceptualization, specification, validation/verification, reduction, data gathering, etc.);
- be able to represent discrete models in Arena in an efficient and effective manner;
- be able to use the models to carry out an analysis by setting up an experiment with the model.

On the other hand, the expert users are expected to be the simulation experts at Accenture who have knowledge level and skills at least similar to those that are defined in the course description of Simulation Masterclass (SPM 9322) – a simulation course that is given by the systems engineering section of TPM faculty of DUT. Therefore they are expected to have knowledge about:

- internal working of different kinds of discrete event simulation languages and environments;
- underlying theories and formalisms of discrete event simulation, such as DEVS and DESS;
- important differences and similarities between simulation environments;
- examples of successful and less successful simulation studies and the learning experiences of those studies;
- object-oriented simulation environments;
- the ability to reuse model parts by developing domain specific building blocks;
- structure and abilities of distributed simulation; the concept of HLA;
- latest research activities in the field of simulation, with research topics like web-based simulation, real time control using simulation, agent based modeling and simulation in special domains like business process modeling;

and they are expected to be able to:

- develop object oriented conceptual models;
- develop object-oriented simulation models;
- develop building block oriented simulation models;
- use different discrete event simulation environments for different kinds of problems.

Another additional requirement would be that both the expert and non-expert users of SIMEON should have the basic knowledge of optimization similar to that which is given in the second part of SPM4121 course (Engineering Multi-Actor Systems from Engineering Perspective) of TPM faculty. Looking to all the requirements, the users that fall in the category of expert would also be able to fully extend and modify SIMEON to increase its performance or to adapt it further to the problems faced.

Next, based on the definitions of the users above, we can derive the detailed technical requirements for the user interface. A clear skill that distinguishes the expert users from the non-expert ones is that the experts are able to develop simulation models using a programming language. This puts up a significant difference for the interface needed by expert and non-expert users. Technically, non-expert users require more sophisticated interface to facilitate their interaction process with SIMEON, whereas expert users may work directly with the interface provided in a programming language such as in eclipse. Therefore, we focus the design of the interface for the non-expert users.

According to Burstein & Holsapple (2008), there are two sub-systems that constitute a user interface of a Decision Support System (DSS):

1. Language system: a system that accepts all the messages for the DSS.

2. Presentation system: a system that emits all the messages that are produced by the DSS.

Next, looking to the optimization and simulation techniques used, there are basic parameters and inputs which all users, regardless to their knowledge, have to configure before they can perform a simulation-based optimization study. These parameters and inputs are part of the messages that the users have to give to the language system of a DSS. Hence, in the language system of SIMEON, interfaces to specify the following parameters and inputs have to be provided:

4. Inputs required to specify the optimization model
   They cover the specifications of the objective and constraint functions, and of the decision variables.
5. Parameters required to configure the optimization algorithm
   The NSGA-II and other Evolutionary Algorithms typically have some parameters related to their operators and iteration process which values have to be specified to run the algorithm. Some examples are population size, maximum iteration number, crossover probability, mutation probability, etc.
6. Parameters required to run the simulation (called simulation run control conditions),
   They cover simulation run length time, number of replication, warm-up time and the time unit of the simulation run.

Furthermore, it is also important to design a presentation system that is able to effectively present the solutions to the users. In this case, it is valuable to have:

1. A graphical panel that is able to project the shape of the non-dominated solutions such that the users get the insight regarding the trade-off between different objectives.
2. A reporting tool that records the values of both the decision variable and the objective functions of the non-dominated solutions.

Finally, Table 2 gives the summary and overview of the requirements that have been defined together with their performance measure, target performance and the way to measure those measures.

TABLE 2 TECHNICAL REQUIREMENTS OF SIMEON

ID	Requirement	Performance measure	Target performance	Way of Measurement
1	Generality			
1.1	SIMEON should be able to solve optimization problems in different domains	The coverage of problems that can be solved using SIMEON	Problems in the domain of supply chain, logistic and business planning	Testing to a test case that is representative for the targeted problem domains
1.2	The optimizer should be able to regard the simulation models as black boxes	The information needed by the algorithms	Using a metaheuristic algorithm that works based on input and output of the model	Literature study
1.3	The simulation models should be able to be run independently	Mechanism to activate/deactivate the optimizers	Building an optimizer that can be coupled and decoupled easily	Interview with the end user for the easiness of coupling and decoupling the optimizer
1.4	SIMEON should be able to get inputs from different software and produce outputs to different software	Variety of software that can be used to take the inputs from and produce the outputs to	Taking inputs and producing outputs from and to Microsoft excel and access	Experimentation to test input/output function
1.5	SIMEON should be able to be operated in different operating system platforms	Operating systems in which SIMEON can be operated	SIMEON is functional in all operating systems	Literature study and testing on different operating

				systems
1.6	SIMEON should be developed for DSOL-based models	Integration of the experimental frame to the DSOL-based simulation models	SIMEON serves as a DSOL-based experimental frame for simulation-based optimization study	Testing to optimize DSOL-based simulation models
2	Efficiency			
2.1	SIMEON should be extendible to incorporate specialized optimization algorithms	The complexity to extend SIMEON with other algorithms	SIMEON allows the implementation of additional optimizer that could work efficiently to solve specific problems	Assessment on the implementation
2.2	The optimizer should have seamless input and output flow to the simulation models	the efficiency of the data transfer between the simulation models and optimizer	Using an internal interface between the simulation models and optimizer	Assessment on the implementation and testing the data transfer function
3	Multi dimensionality			
3.1	SIMEON should be able to optimize multiple-objective functions	The number of the objective functions that can be optimized	SIMEON is successfully applied to produce the Pareto Front for the case studies	Application to a test case
3.2	SIMEON should be able to optimize quantitative parameters of the simulation model	maximum number of quantitative parameters that can be optimized	Max number of quantitative parameters in the case studies	Application to a test case
3.3	SIMEON should be able to optimize non-quantitative parameters of the simulation model (e.g. modules, system component)	maximum number of non-quantitative parameters that can be optimized	Max number of non-quantitative parameters in the case studies	Application to a test case
3.4	SIMEON should be able to optimize quantitative and non-quantitative parameter of the simulation model simultaneously	maximum number of simultaneous parameters to optimize	Total number of non-quantitative and quantitative parameters in the case studies	Application to a test case
3.5	SIMEON should be able to show the optimal trade-offs in the objectives in case they are conflicting	The number of the non-dominated solutions that can be optimized	Generating and visualizing Pareto Front for multiple objectives	Application to a test case
4	Usability to the users			

4.1	SIMEON should be able to be used by the non-expert users comprehensively by the help of any necessary user interface	The easiness of interaction with SIMEON	Non-expert users only need to have basic knowledge of modeling and simulation techniques to use SIMEON	Interview with the non-expert users
4.2	SIMEON should be able to be extended by the expert users to solve specific optimization problems	Extendibility of SIMEON	The expert users can perform any necessary modifications in the optimizer and simulation (e.g. adding more efficient algorithms)	Literature study, assessment on the implementation and Interview with the expert users
4.3	SIMEON should be adaptable to the changes in the types of decision variables (i.e. quantitative and non-quantitative) that are to be optimized in different problem domains	The adaptability of SIMEON to different types of decision variables	SIMEON can facilitate the change of types of decision variables to optimize automatically	Interview with all the potential users

3.2 SIMEON Conceptual Design

The second design sub-phase of systems engineering life-cycle is the conceptualization of SIMEON design. The primary objective of this phase is to develop the concept that is responsive to the specifications that have been developed in the previous phase of the life-cycle which is SIMEON requirements and specifications (Sage & Armstrong, 2000). The outcome of this phase which is the preliminary design of SIMEON also concludes the first phase of systems engineering life-cycle: System definition.

This phase also contributes a partial answer to the 1st research question that investigates the integration of optimization technique into the framework modeling and simulation that is proposed by Zeigler (2003). The conceptual design presented in this chapter, therefore, serves as the conceptualization of SIMEON framework which will be further developed in the system development phase.

Using the modeling and simulation framework (M&S framework) that is proposed by Zeigler (2003) as the starting point, the main contribution of this sub-chapter is to conceptualize the integration the optimization technique into that framework such that an intelligent system that has the aim to accomplish its pre-defined goal(s) can be developed. Looking at the researches that have been done in this particular subject, the notion regarding the integration of an optimization algorithm into the M&S framework has also been discussed in the research of Traoré and Muzy (2006) and Hojun and Zeigler (2008). According to Traoré and Muzy (2006) an experimental frame of Zeigler's M&S Framework can be regarded as a system that has the function to determine the way the simulation model should be used and specified by the solutions prescribed by the optimization algorithm. Conceptually, this means that the optimization algorithm can be connected into the experimental frame in such a way the decision variables of the simulation model can be optimized in terms of the pre-defined objective function(s). The paper of Hojun and Zeigler (2008) also provides a concrete example of how an experimental frame is designed to perform a simulation-based optimization method in optimizing the parameter of a network gateway.

However, to the best of the author's knowledge, there has not been any research that presents the formal and transparent structure to integrate an optimization technique (meta-heuristic) into the Zeigler's M&S framework. Therefore, taking into account the technical requirements (chapter 3.1.1) and the analyses done regarding multi-objective evolutionary algorithms (chapter 2.4.2), the purpose of this sub-chapter is to propose a conception and to delineate the integration of the selected evolutionary-based multi-objective optimization technique (NSGA-II) into the experimental frame of the widely used M&S framework.

3.2.1 Integration of NSGA-II into the Experimental Frame (EF) of Zeigler's Framework

In chapter 2.2 we present the OR steps that can be used to structure our approach in finding the optimal values for the decision variables (Hillier & Lieberman, 2001). We can now adapt these steps to systematically perform the integration of NSGAII into the Zeigler's framework. The reason to use these steps is because the objective of the modeling and simulation study discussed here is essentially to find the optimal solutions for the modeled problem. Following are the adapted procedures:

1. Define the problem of interest and gather relevant data.
2. Develop a simulation model to represent the problem.
3. Develop a computer-based procedure for deriving solutions to the problem from the model.
4. Test the simulation and optimization procedures and refine them as needed.
5. Prepare for the ongoing application of the model as prescribed by management.
6. Implement.

However, not all of the steps above will be substantiated for only the first three steps are relevant for the conceptualization phase.

1. Define the problem of interest and gather relevant data

This is a crucial phase where we define the optimization problem as the objective of the modeling study. This phase gives the main distinction between a normal what-if simulation study and a simulation-based optimization study. The view adopted in this phase would affect the way the subsequent phases are performed especially in the development of the simulation model.

For problems that are defined as optimization problems, the modeler has to identify the goals/objectives of the organization, the constraints on what can be done and the relationship between all relevant elements of the system towards those objectives and constraints. An important distinction that has to be formulated in this phase is the variables that are controllable and the variables that represent the given circumstance for the system or environmental variables. After the problem is well-defined, data gathering process can be performed. The data needed normally covers the description of the behavior of the environment of the system and the description of the limitation on the controllable elements of the system.

2. Development of the simulation model

Instead of developing a mathematical model, a simulation model is developed to represent the system under study. In developing such a model, a standard modeling approach suggested by Daalen, et al. (2009) can be used. However, it is important to note that the model here should be developed with a paradigm of solving an optimization problem. This implies that:

1. The models should be able to be coupled with functionality that allows experimentations of different decision alternatives within the given circumstances/environment of the system.

2. There has to be a separation of concerns between the problem to solve and the algorithm to use, to fulfill the generality feature.

To operationalize the concept mentioned above, here we present the definitions of the entities that are necessary to support a functional simulation-based optimization model.

a. Treatment

Treatment can be defined as a specific set of conditions that are imposed to the model to perform the simulation. Those conditions cover:

1. Specification of input data
2. Collection of input data
3. Initialization conditions
4. Run control conditions
5. Specification of output data

In optimization context, what normally distinguishes one treatment to the others is the value of the decision variables inside those treatments. Thus, in this case a treatment contains a specific set of decision variable values. Furthermore, what the decision variables set within the simulation model are the initial states of the model. Next, the combination of the initial states and the input segments from the generator would be mapped by the state transition function of the model to produce the output values.

b. Experimental frame

According to Zeigler, et al. (2000), an experimental frame can be defined as a component that consists of following sub-components:

1. The generator
   This the component that produces the input segments to the simulation model. In the context of simulation based optimization, generator produces the values of variables other than the decision variables which are normally caused by events that take place in the environmental condition of the system (e.g. the arrival of entities into to the system that follow certain statistical distribution).

2. The transducer
   This is the component that maps the output variables into outcome measures of interest. These outcome measures are normally the statistical summaries of the simulation output which can be defined as the goals of system or objective functions of the optimization problem.

3. The acceptor.
   This is the component that monitors the validity of the experiments by checking whether the outcome measures of interest violate the pre-defined constraints of the experimentation. Acceptor does not always exist in an experimental frame, and in this case, transducer directly gives the outcome measures to the user.

c. Simulation-based optimization experiment

A simulation-based optimization experiment is an execution of the simulation model with a specific treatment and environmental conditions to produce the outcome measures of interest. Therefore, this type of experiment includes the specification of Zeigler's experimental frame.

d. Simulation-based optimization problem

A simulation-based optimization problem is a form of optimization problem in which simulation experimentations are used to find the optimal values for the decision variables. Thus, simulation-based experimentations should be defined within such a problem. Moreover, the specification of this problem should distinguish the decision variables and the other variables (environmental variables) within the simulation model.

Finally Figure 15 illustrates the entities discussed above and how they are position towards one another,

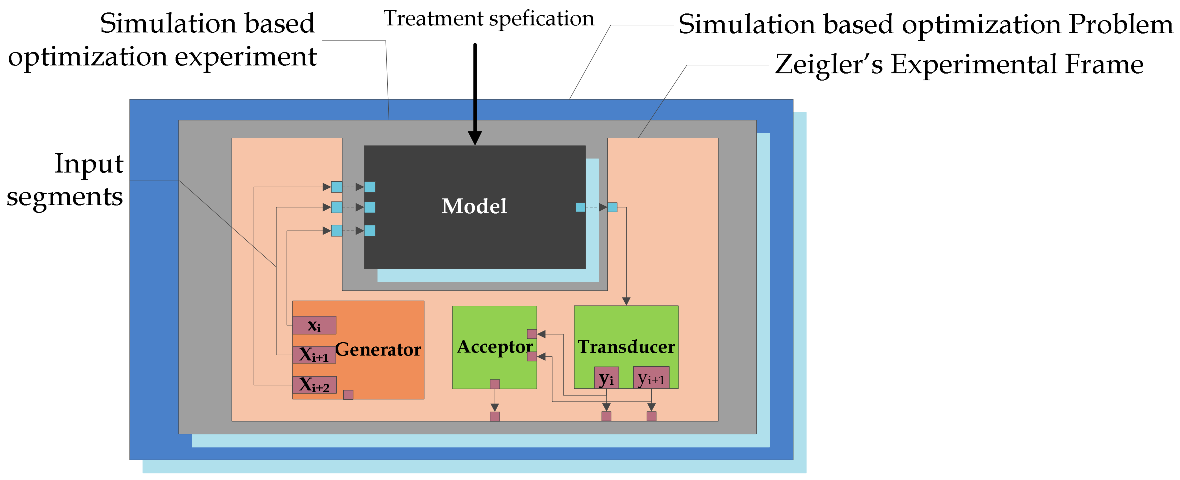

Simulation based optimization experiment
Treatment spefication
Simulation based optimization Problem
Zeigler's Experimental Frame
Input segments
Model
x_i
X_{i+1}
X_{i+2}
Generator
Acceptor
Transducer
y_i
y_{i+1}

FIGURE 15 SPECIFICATION OF SIMULATION-BASED OPTIMIZATION PROBLEM

3. Develop a computer-based procedure for deriving solutions to the problem from the model.

The computer-based procedure that is going to be developed is the NSGAII. jMetal library (Durillo, et al., 2006) will be used to develop further the NSGA-II altogether with the necessary operators and functionality. This implementation aspect will be elaborated further in the chapter 4.2 which is also going to conclude the answer for the first research question. On the other hand, DSOL library (Jacobs, Lang, & Verbraeck, 2002) will be used as the basis for the development the simulation models that are going to be used for the test cases. In addition to that, to connect the algorithm to the simulation model and the experimental frame a procedure has to be developed as well. First, let us recall the routine of the NGAII. Figure 16 presents the flowchart of the NSGA-II.

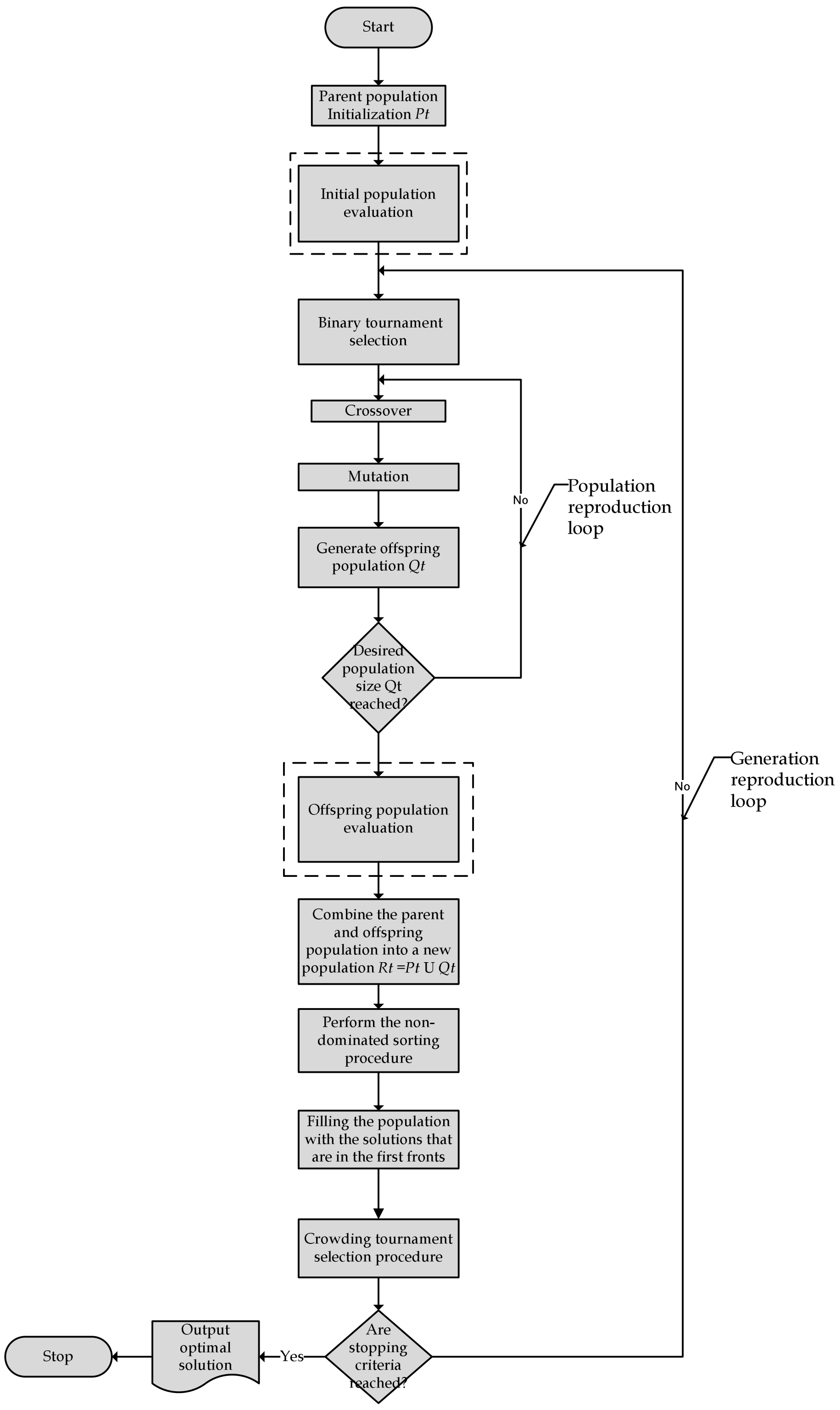

Start
Parent population Initialization Pt
Initial population evaluation
Binary tournament selection
Crossover
Mutation
Generate offspring population Qt
Desired population size Qt reached?
No
Population reproduction loop
Offspring population evaluation
Combine the parent and offspring population into a new population Rt =Pt U Qt
Perform the non-dominated sorting procedure
Filling the population with the solutions that are in the first fronts
Crowding tournament selection procedure
Are stopping criteria reached?
No
Generation reproduction loop
Yes
Output optimal solution
Stop

FIGURE 16 FLOWCHART OF NSGA-II ROUTINES

Given the routines above, the aim of SIMEON is to provide a mechanism that enables automatic and iterative specification of the values of the decision variables in such a way they are optimized in regards to the pre-defined objective functions. This means the knowledge of the modeler to specify the treatments as depicted in Figure 6 (chapter 2.2) can be replaced by the routine of the NSGAII to find the optimal solutions for the problem. Following paragraphs delineate the detailed process of how the NSGAII works together with the experimental frame and the model.

When the algorithm is first executed, a set of solutions (called as population in EA terminology) are generated randomly to explore the criterion space of the optimization problem. These solutions are in form of the decision variable values that are required to specify the structure of the model or the quantitative values that can represent the initial states of the simulation model. These values combined with the transition function of the model, in turn determine the behavior and the final output of the model.

Hence, in this approach one solution/chromosome in a population corresponds to a set of quantitative and/or qualitative decision variables for the optimization problem that is going to be simulated by the model. Consequently, one population contains different structures and input parameters which have to be experimented by the simulation model. The role of the generator here is to produce events that come from the environmental conditions/context of the real system. Thus, the generator is built using elements in the system to which control effort is not or cannot be exercised (e.g. the entities in the system which behavior follows certain statistical distribution).

After the first population is generated and simulation experiments are carried away for a pre-defined run-length time, the output variables are given to the transducer for each experiment. In the transducer the values of all output variables are mapped into the outcome measures of interest. This can be statistical summaries of the system performance such as average round trip time, average waiting time, average throughput time, average unsatisfied or lost request, etc. Subsequently these statistical summaries can be given to the acceptor which would then monitor the validity of these values according to the certain pre-specified conditions (such as certain threshold values). It is noteworthy that acceptor does not always present in an experimental frame. Its presence is determined by the objective of the modeling and simulation study and the availability of the data that can be used to validate/invalidate the results of the experiment (Zeigler, Hall, & Salas, 2009). In case the acceptor is not implemented in the experimental frame, the outcome measures of interest produced by transducer are directly given to the algorithm to be used for the evaluation of the pre-defined objective functions.

In the subsequent NSGA-II routines, unlike in the first generation, the solutions from the second generation on are produced through the operations that take place within NSGA-II routines, which makes use the output of the acceptor. This involves a condition check after the evaluations of objective function(s) take place. The condition check is used to determine whether the algorithm is still in the solution initialization phase (1st generation) or it is already in the main loop (> 1st generation). If the algorithm is already in the main loop, then the main routines can take place before the new solutions are generated. Otherwise, the routine is continued directly to the binary tournament selection and followed by the genetic operations afterwards. Finally, the algorithm would continue its routine until the stopping criterion is met.

Looking at the whole process, we can see that there is now a feedback loop to the simulation model with the output of the simulation model as the input to the NSGAII and the non-dominated solution set as the output of the NSGAII. Figure 17 below provides the visualization of the simulation-based optimization problem together with the routines of NSGA-II to conceptualize the proposed SIMEON framework. It is important to note that SIMEON is currently specialized for the integration of NSGA-II into the experimental frame of Zeigler's modeling and simulation framework. It is obviously possible to use this example to integrate other evolutionary-based optimization techniques. However, this specialization is meant to provide the detailed conceptualization which could facilitate a clear and comprehensible transition to the concrete implementation of the concept. In addition to that, the decision to use NSGA-II has also been justified in chapter 2.4.2.

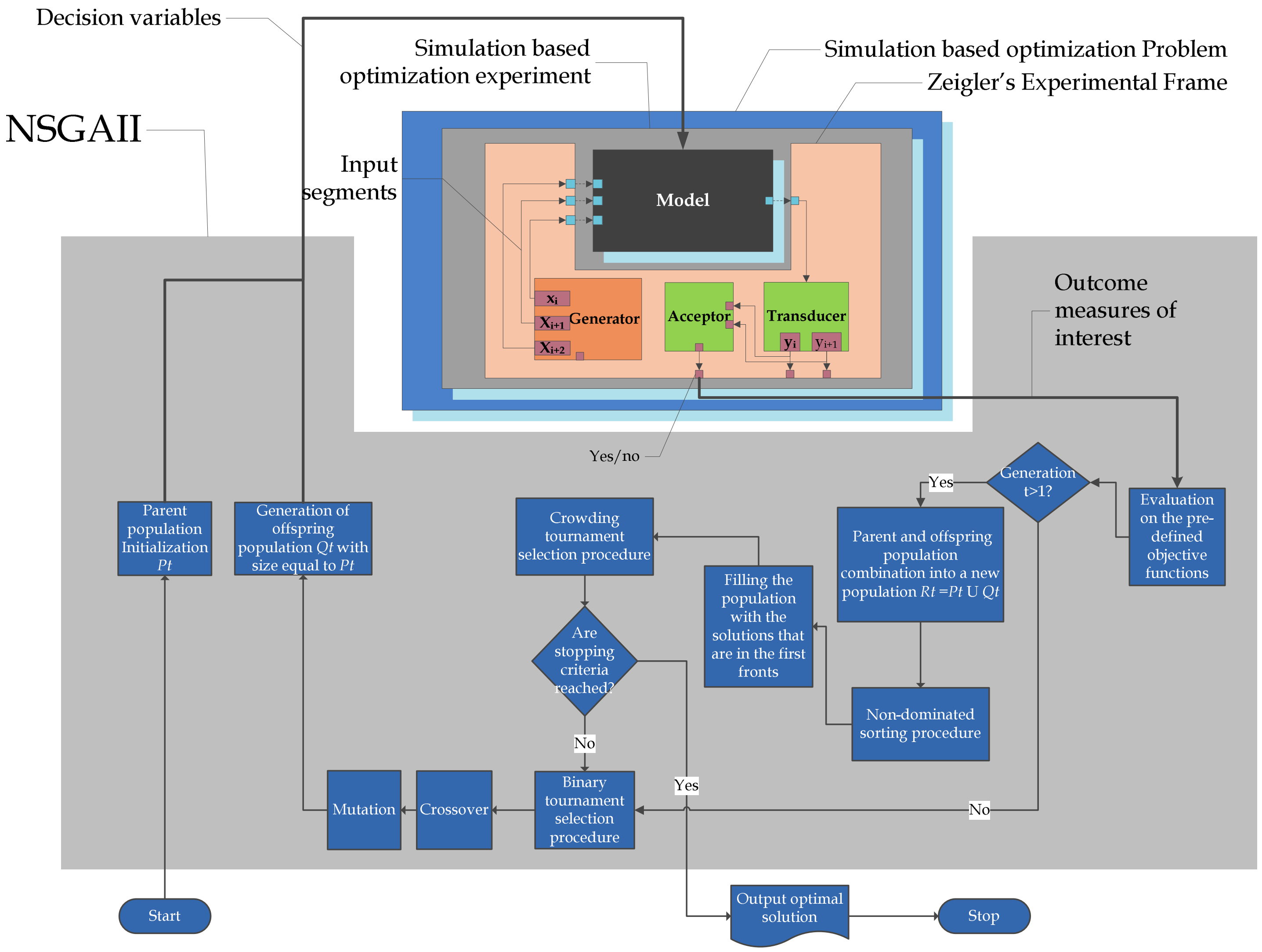

Decision variables
Simulation based optimization experiment
Simulation based optimization Problem
Zeigler's Experimental Frame
NSGAII
Input segments
Model
x_i
X_{i+1}
Generator
X_{i+2}
Acceptor
Transducer
y_i
y_{i+1}
Outcome measures of interest
Yes/no
Generation t>1?
Yes
Evaluation on the pre-defined objective functions
Parent population Initialization *Pt*
Generation of offspring population *Qt* with size equal to *Pt*
Crowding tournament selection procedure
Filling the population with the solutions that are in the first fronts
Parent and offspring population combination into a new population *Rt* =*Pt* U *Qt*
Are stopping criteria reached?
Non-dominated sorting procedure
No
Yes
No
Mutation
Crossover
Binary tournament selection procedure
Start
Output optimal solution
Stop

FIGURE 17 SIMEON FRAMEWORK

- **Partial answer to the first research question**

Finally, looking back at the challenge that is delineated in the introduction of this thesis paper and eventually to the first research question that is formulated:

1. How can metaheuristic techniques be integrated into the framework of modeling and simulation that is currently used to solve real-world problems?

We can now provide a partial answer for this question through the conceptualization proposed above. It is clear that a metaheuristic technique (in this case NSGA-II) can be integrated into the modeling and simulation framework by connecting its routines into the model and the experimental frame. In this design, the values of the decision variables inside treatments given to the simulation model are automatically determined by the NSGAII and the transducer or acceptor feed the algorithm with the values of the outcome measures. It is also clear that a number of experiments with different treatments are needed to perform the optimization. Based on this result, we also argue that other state-of-the-art metaheuristic algorithms can be integrated in a similar way.

A noticeable design in the experimentation process using SIMEON is that NSGA-II routines create a closed-loop between simulation and optimization and therefore, replaces the knowledge of the modeler decision variable values. This way, the tedious process of finding the best alternatives / the best trade-offs can be automated by using the intelligence of SIMEON. Furthermore, in the case where the optimization problem is highly complicated (i.e. a non-linear problem with many variables and local optima), the framework also theoretically provides significantly more effective way to find the solution in comparison to the conventional approaches such as brute-force and trial and error.

3.3 Conclusion

In this chapter the steps needed to perform a simulation-based optimization study have been substantiated. These steps are used to perform the integration of a metaheuristic technique into Zeigler's framework of modeling and simulation. Furthermore, the conceptual framework resulting from that integration is also presented. In the next chapter the transformation of this conceptual design into a concrete implementation of SIMEON is performed.

4. SIMEON DESIGN AND DEVELOPMENT

In this chapter the second systems engineering life-cycle phase is performed: system development (Sage & Armstrong, 2000). There are two sub-phases that are used to systematically develop SIMEON:

1. System architecture of SIMEON
2. Detailed design and construction of SIMEON

In the first sub-phase the architecture of SIMEON is constructed based on the conceptual design that has been presented in the previous chapter. This architecture shall give an accurate overview of the physical structure of the SIMEON framework and how the sub-systems within interact with one another.

The second sub-phase will provide the detailed technical designs of SIMEON sub-systems. It is also the aim of this sub-chapter to provide the answers for the first, second and third research questions. Firstly, a thorough and detailed design of SIMEON framework will be presented. Secondly, the design of the genetic chromosomes and operators will be explained. Thirdly, the detailed design of SIMEON user interface will also be presented. Finally, to conclude the chapter, a conclusion and a review upon the fulfillment of the requirements with respect to the technical designs are going to be presented.

4.1 SYSTEM ARCHITECTURE OF SIMEON

We view system architecture as a product of system synthesis activity through which the functional capabilities that have been formulated in the requirements allocation sheet and the conceptual design (chapter 3.2.1) are translated to concrete entities that satisfy the performance requirements of those previously described functions.

Furthermore, as SIMEON is a framework that is going to be used to develop decision support tools, the architecture of SIMEON is synthesized in such a way it is coherent to a standard decision support system's architecture. This is done with a purpose to retain all the features and functionalities that a simulation-based decision support tool might have and add additional functionalities/services on top of it. Following are the specifications of subs-systems that embody a decision support system (DSS) (Burstein & Holsapple, 2008):

1. Language system: a system that accepts all the messages for the DSS.

2. Presentation system: a system that emits all the messages that are produced by the DSS.

3. Knowledge system: a system that stores and retains all the knowledge that is relevant to the DSS.

4. Problem processing system: a system that provides problem solving service of the DSS or the engine of a DSS.

Looking at the specification of DSS sub-systems above, we can notice a resemblance between the concepts above and multi-tier architecture in software engineering. Since SIMEON is nevertheless also a product of software engineering, we propose to use the paradigm of multi-tier architecture, as a good foundation of software engineering, to synthesize the architecture of SIMEON (Figure 18).

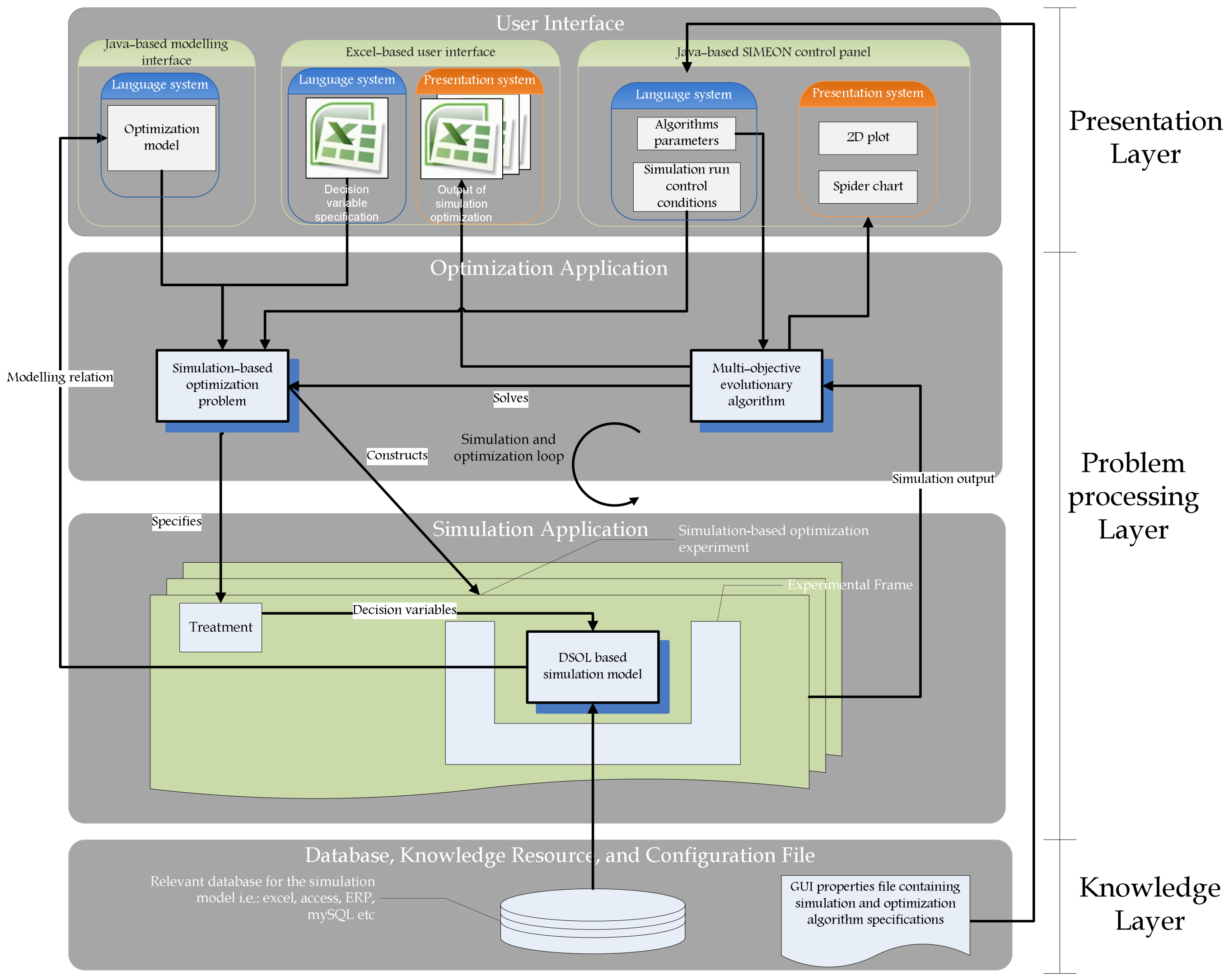

User Interface
Java-based modelling interface
Language system
Optimization model
Excel-based user interface
Language system
Presentation system
Decision variable specification
Output of simulation optimization
Java-based SIMEON control panel
Language system
Algorithms parameters
Simulation run control conditions
Presentation system
2D plot
Spider chart
Presentation Layer
Optimization Application
Simulation-based optimization problem
Multi-objective evolutionary algorithm
Solves
Constructs
Simulation and optimization loop
Modelling relation
Simulation output
Problem processing Layer
Specifies
Simulation Application
Simulation-based optimization experiment
Experimental Frame
Treatment
Decision variables
DSOL based simulation model
Database, Knowledge Resource, and Configuration File
Relevant database for the simulation model i.e.: excel, access, ERP, mySQL etc
GUI properties file containing simulation and optimization algorithm specifications
Knowledge Layer

FIGURE 18 SYSTEM ARCHITECTURE OF SIMEON

As illustrated by the figure above, there are three layers that form the architecture:

1. Presentation layer: a layer where the language and presentation systems are positioned. This layer serves mainly as the user interface though which the user makes interactions with SIMEON. There are several interfaces that take the messages from the user into SIMEON, serving as the language systems:

    a. The excel sheet that is used to define the decision variables.

    b. The java-based modeling interface, used to define the optimization model to be optimized. This frame takes as inputs, the output objects that are defined in the simulation model. This way, there is a separation of concern between modeling the outputs of the simulation model and modeling the optimization problem based on those outputs.

    c. The tab in the java-based control panel where the users get to specify the algorithm to use, its parameters, the simulation model to use and its run control conditions. This tab uses a configuration/properties file that is specified by the user in the knowledge layer.

On the other hand, there are also interfaces that emit messages to the user from SIMEON, serving as the presentation systems:

a. The 2D plot and the spider chart that is used to project the non-dominated solutions for two objective functions and multi-objective functions.

b. The excel sheet that records the output of each framework execution. Each time the framework is executed to solve a simulation problem, an excel sheet is produced as the report of the optimization study. This excel sheet contains all decision variable values and its corresponding objective values.

2. Problem processing layer: a layer where the computation engines are positioned to provide the problem solving services. There are two distinctive applications/ engines in this layer: the optimization and simulation engines. Both engines are constructed out of object-oriented components and may provide independent services:

   a. Optimization engine

      It consists of multi-objective evolutionary algorithm and simulation-based optimization problem objects. The first uses the parameter setting that is defined in the control panel and the latter takes input from the modeling frame, the decision variable sheet and the control panel to specify the simulation-based optimization problem to be solved.

   b. Simulation engine

      It consists of the simulation model, the treatment that contains the decision variables for the model, and the simulation-based optimization experiment objects. It is noteworthy that there are multiple treatments and simulation-based optimization experiments in a framework execution. This is done to perform the multiple function evaluations (by means of simulation) based on the sets of decision variables specified in a set of treatments. Thus, one function evaluation that is specified within a simulation based problem object, involves the construction of one experiment with a specific treatment or a set of certain decision variables.

3. Knowledge/data layer: a layer where the knowledge system is positioned. This is the layer where all relevant knowledge and information needed by SIMEON are stored. Example of knowledge system can be independent database systems (e.g. Microsoft Excel, Access, ERP, MySQL, etc) that contain data needed for the simulation or a configuration file for SIMEON.

With the architecture described above, the translation of the conceptual design into concrete system design has completed and we can now see the overview of SIMEON system design. However, there are still parts of the architecture (especially those which are within the problem processing layer) that can be elaborated further to provide answers for the research questions. To serve this purpose the next section will provide the translation of system architecture above into the implementation of SIMEON.

4.2 Detailed Design and Construction of SIMEON

In this sub-chapter the detailed technical designs of SIMEON will be substantiated using Unified Modeling Language (UML). The thorough design of SIMEON implementation will first be presented to provide a complete answer for the first research question. This is then followed by an elaboration of the theoretical foundation of the framework's optimization engine, by which the answer for the second research question can be derived. Subsequently, the detailed design of the user interface will also be elaborated to expose its feature and answer the third research question accordingly.

4.2.1 Design of the SIMEON Framework

This sub-chapter will detail the construction of SIMEON framework in the implementation level to ultimately provide an answer of how can metaheuristic techniques be integrated into the framework of modeling and simulation. Firstly, the foundation for the framework will be introduced and detailed to give a thorough technical comprehension on how the framework is constructed. This is done to facilitate a good transfer of knowledge through which any effort to extend or modify the framework can be supported. Secondly, the detailed design of the integration between simulation and optimization engines is presented. The purpose is to contribute to the reader the technical knowledge of integrating DSOL-based simulation model and metaheuristic technique used in SIMEON. Thirdly, the design of framework as a whole is presented to provide a general idea on how the framework is structured and implemented. This design will then be used to answer the first research question.

It is noteworthy that the detailed design presented in the following section is derived meticulously from the conceptual design (chapter 3.2) and the system architecture (chapter 4.1). Understanding all the relevant notions in those chapters will be helpful to comprehend the contents presented in the following section for the detailed designs use similar terminologies as presented in those two chapters. Readers who have a detailed understanding on Java and the libraries used will have the benefit to understand profoundly the decisions made for the implementation. However, this shall not reduce the value of this section to convey the concrete logical structure of SIMEON to broad readers.

4.2.1.1 Foundation of SIMEON framework

As explained in the previous chapter, the development of SIMEON is constituted by two Java-based libraries: jMetal (Durillo, Nebro, & Alba, 2010) and DSOL (Jacobs, 2005). jMetal stands for Metaheuristic Algorithms in Java, it is an object-oriented framework that facilitates the development, experimentation and study of existing and new metaheuristic algorithms to solve multi-objective optimization problems. On the other hand, DSOL stands for Distributed Simulation Object Library, it is also an object-oriented framework aimed at facilitating the development, experimentation and study of continuous and discrete simulation models. To present the development of SIMEON in a constructive order, we can first look into jMetal framework v2.2 (Durillo, et al., 2006) that serves as the base for SIMEON framework.

Within the jMetal framework v 2.2, there are two central classes that act as the problem solver and the problem to be solved -`Algorithm` and `Problem` (Figure 19). `Algorithm` class provides the base attributes and methods that a metaheuristic algorithm can use to perform the iterations of function evaluations. Similar to that, `Problem` class also provides the base attributes and methods that can be used to specify the structure of an optimization model (i.e. objective and constraint functions). Both of those classes are therefore implemented as abstract classes. Furthermore, the implementation of the classes in SIMEON will have to incorporate the mechanism used in DSOL to organize the starting and stopping of a simulation as an integration feature. Implementing `EventListenerInterface` class of DSOL in SIMEON is proposed in our design to organize the starting and stopping of the simulation execution.

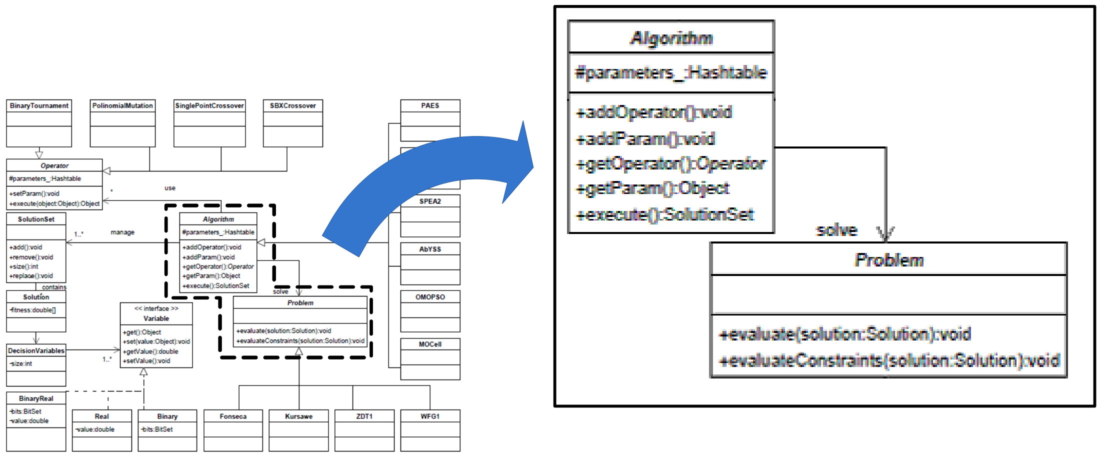


FIGURE 19 ALGORITHM AND PROBLEM CLASSES IN JMETAL FRAMEWORK (ADAPTED FROM DURILLO, NEBRO, LUNA, DORRONSORO & ALBA (2006))

A relevant algorithm implementation for SIMEON is the NSGAII. In jMetal this algorithm is implemented as an object that extends from the `Algorithm` class and defines its own sub-routines to solve standard mathematical optimization problems. In the effort of adapting NSGAII to solve simulation-based problem, an adapted version of NSGAII is implemented, namely `SimOptNSGAII` (Figure 20). This object extends from the algorithm class and implements `EventListenerInterface` from DSOL, allowing it to listen to the end of simulation experiment events through the invocation of notify method. This is a necessary design to enable SimOptNSGAII advance to the subsequent sub-routines (e.g. function evaluation, genetic operations, etc) only after the previously executed simulation has reached its end.

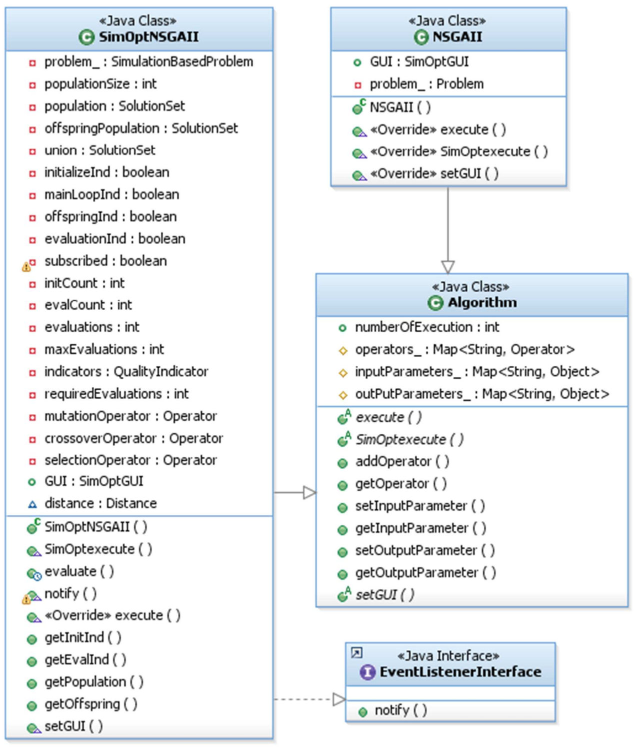


FIGURE 20 SIMULATION-OPTIMIZATION ALGORITHM IN SIMEON

Given the NSGAII as the engine to solve the optimization problems, we can now look at the problem to be solved which model is structured in a DSOL-based simulation model. Within jMetal framework, mathematical-model based optimization problems are implemented as the sub-classes of `Problem` class. To serve the purpose of constructing simulation-based problems, a class named `SimulationBasedProblem` is implemented. This class extends from `Problem` class and serves as the abstract class from which all optimization problems that involve simulation may extend (Figure 21). All attributes and methods needed to perform simulation runs are thus contained within this class.

Furthermore, this class is also designed to have an important functionality of performing the evaluation of the objective function(s) after a simulation run ends. To serve this purpose, this class implements `EventListenerInterface` of DSOL. It is also important to note that the interaction between the `SimOptNSGAII` class and the `SimulationBasedProblem` class is designed to embody the concept of problem defined in chapter 3, where the routine of NSGAII works as a system that provides the simulation model with the optimized decision variable values.

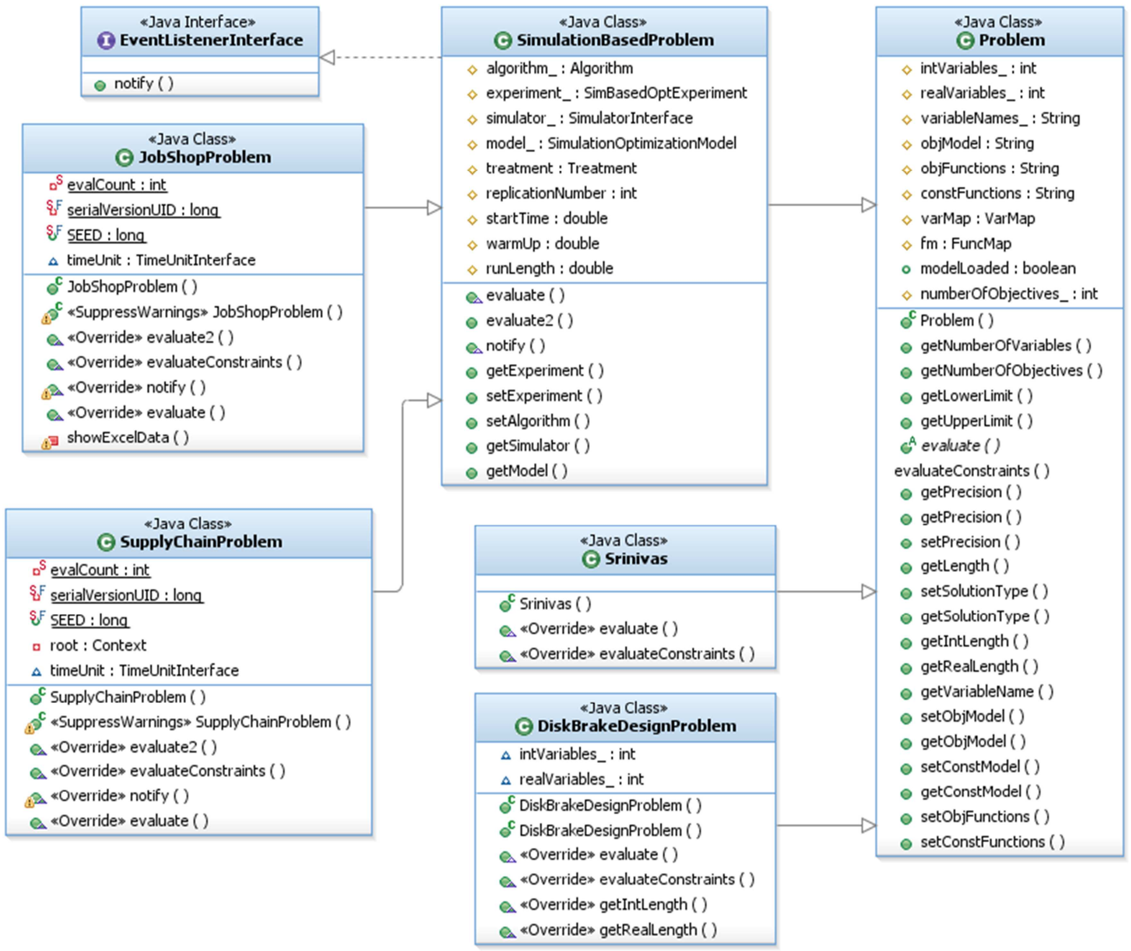


FIGURE 21 SIMULATION-BASED PROBLEM SPECIFICATION IN SIMEON

The implementation of `SimOptNSGAII` and `SimulationBasedProblem` classes has embodied the main object components of the optimization engine in the problem solving layer of SIMEON architecture. Hence, we can now shift our focus to the foundation of the simulation engine of SIMEON. The objects that constitute the simulation engine mainly extend from DSOL. In order to execute a simulation-based optimization run, a specific simulation-based optimization experiment has to be defined in which, a simulation model, simulator that executes the model, and a treatment to the model are specified. To serve this purpose, `SimBasedOptExperiment, SimulationOptimizationModel` classes are implemented. While DSOL's `Treatment` class is reusable in SIMEON, `SimBasedOptExperiment` extends from DSOL's `Experiment` class and `SimulationOptimizationModel` implements DSOL's `ModelInterface` class. Furthermore, these classes interact with `SimulationBasedProblem` class as the instances that are instantiated within that class. Figure 22 depicts the relationship those classes and `SimulationBasedProblem` class in SIMEON.

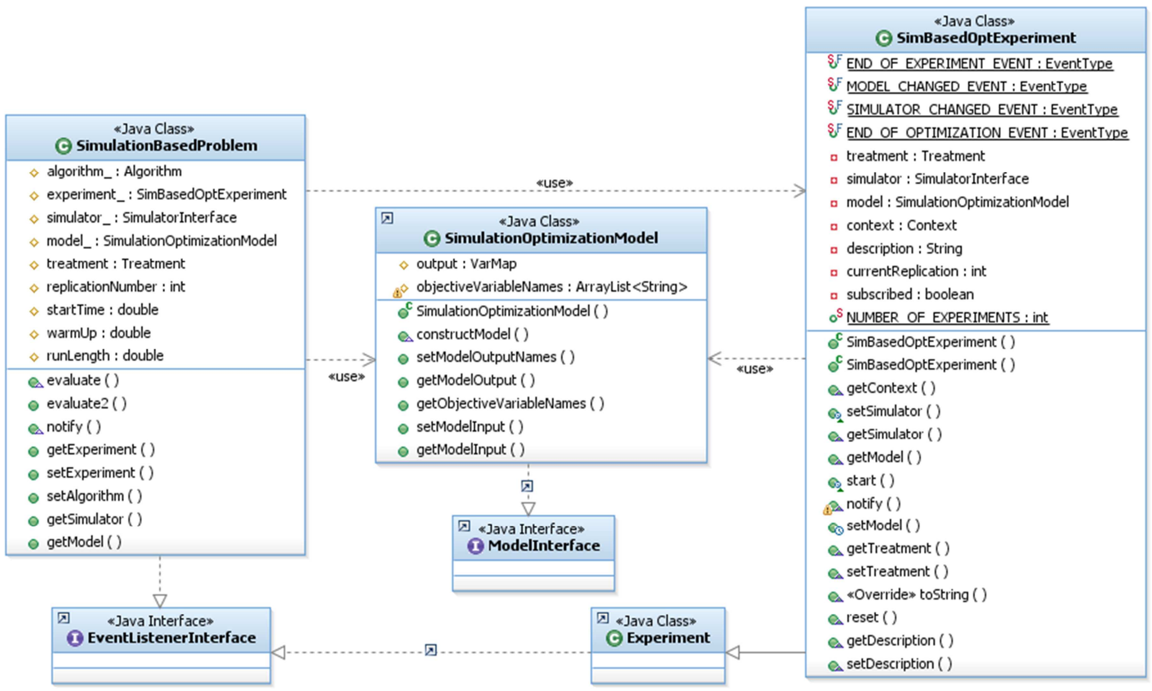


FIGURE 22 SIMULATION-BASED OPTIMIZATION EXPERIMENT IN SIMEON

4.2.1.2 Design of Simulation and Optimization Interaction

Following the specifications of the components within the problem solving layer of SIMEON, the design of the interaction between the two distinct engines (simulation and optimization) can now be presented. Technically, the design of the interaction contributes to a great extent to the answer to the first research question. By knowing how these simulation and optimization engines interact, knowledge of how we can integrate a simulation execution into an execution of an evolutionary algorithm (in this case NSGAII) is gained. Furthermore, as this design of interaction problem entails a great technical challenge, a clear documentation on the design is considered to be valuable in contributing to the technical knowledge of the reader.

As noted in the system architecture diagram (Figure 18), there are six components of the simulation and optimization engines. We implement these components as Java based objects (Figure 23 for simplified version, appendix A for extended version). There are obviously other objects that are needed to form the whole framework, but for clarity sake, only elements that are relevant to the interaction of simulation and optimization in SIMEON are presented.

The `SimOptNSGAII` implements the main algorithm routine, which solves a `SimulationBasedProblem`. The latter defines the `SimBasedOptExperiment`, which uses the `Simulator`, `Treatment` and `SimulationOptimizationModel` to run unique experiments based on the specifications given to these three objects. Recall that in an execution of the algorithm, one objective function evaluation requires the execution of one simulation experiment. This way, the number of the

SimBasedOptExperiment instances created is going to be equal to the maximum number of evaluations that the algorithm is to execute, while there is only one SimulationBasedProblem object.

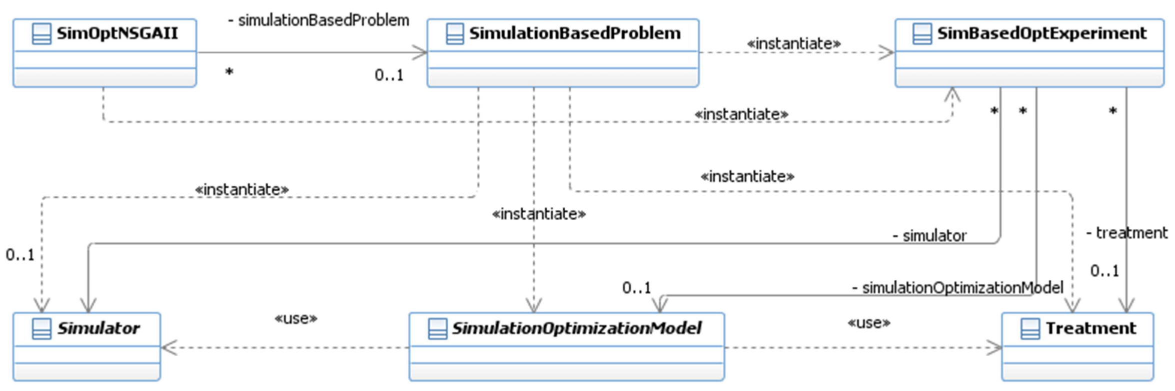


FIGURE 23 UML CLASS DIAGRAM OF SIMULATION AND OPTIMIZATION OBJECTS

To provide a profound understanding on how the NSGAII interacts with the simulation model, a UML sequence diagram is presented in Figure 24. The diagram provides the flows of the messages from the moment the user gives a message to SIMEON to execute the SimOptNSGAII until it presents the results of the optimization back to the user. The diagram assumes that both SimOptNSGAII and SimulationBasedProblem objects are already constructed by the time SimOptexecute() method is invoked.

The first iteration loop is started when evaluate2() in SimulationBasedProblem is invoked. This will result in the construction and execution of a simulation-based experiment (by invoking start()in SimBasedOptExperiment). This experiment will be run using Simulator and SimulationOptimizationModel that are instantiated during the construction of SimulationBasedProblem. Next, the SimBasedOptExperiment shall initialize the simulator by invoking initialize(). This will eventually result in construction of the simulation model through the invocation of constructModel(). Soon after the execution time is reached, an end of replication event is fired by the simulator and if the desired number of replication has been reached, an end of experiment event is fired by SimBasedOptExperiment. The firing of this event shall result in the invocation of notify()in SimulationBasedProblem and SimOptNSGAII sequentially. When the notify()of SimulationBasedProblem object is invoked, the evaluation of the objective functions to be optimized takes place. On the other hand, invoking notify() in SimOptNSGAII would result in the invocation of evaluate2()in SimOptBasedProblem depending whether the algorithm has finished evaluating the all the solutions within the initial population or not.

The second loop of iteration basically executes the same routines as the ones that are executed during the first loop. The only difference is that the stopping condition for this loop is satisfied if the maximum number of solution evaluations has been reached. Soon after this condition is met, SimOptNSGAII presents the output to the graphical user interface (SimOptGUI) through the invocation of relevant methods (e.g. getPlot(), getSpiderPanel(), etc.).

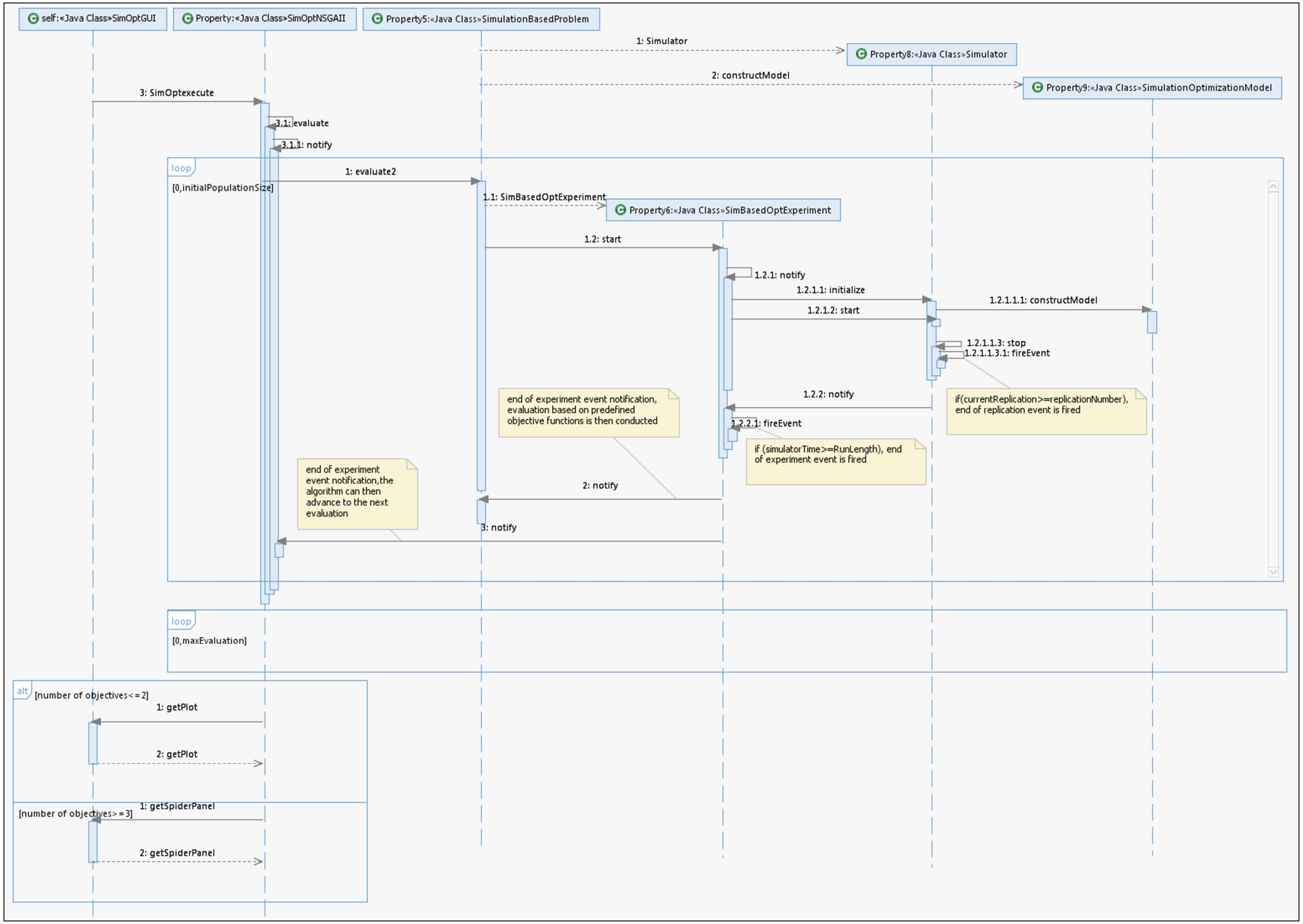

self:«Java Class»SimOptGUI
Property:«Java Class»SimOptNSGAII
Property5:«Java Class»SimulationBasedProblem
1: Simulator
Property8:«Java Class»Simulator
2: constructModel
Property9:«Java Class»SimulationOptimizationModel
3: SimOptexecute
3.1: evaluate
3.1.1: notify
loop
[0,initialPopulationSize]
1: evaluate2
1.1: SimBasedOptExperiment
Property6:«Java Class»SimBasedOptExperiment
1.2: start
1.2.1: notify
1.2.1.1: initialize
1.2.1.1.1: constructModel
1.2.1.2: start
1.2.1.1.3: stop
1.2.1.1.3.1: fireEvent
1.2.2: notify
if(currentReplication>=replicationNumber), end of replication event is fired
end of experiment event notification, evaluation based on predefined objective functions is then conducted
1.2.2.1: fireEvent
if (simulatorTime>=RunLength), end of experiment event is fired
end of experiment event notification,the algorithm can then advance to the next evaluation
2: notify
3: notify
loop
[0,maxEvaluation]
alt
[number of objectives<=2]
1: getPlot
2: getPlot
[number of objectives>=3]
1: getSpiderPanel
2: getSpiderPanel

FIGURE 24 UML SEQUENCE DIAGRAM FOR THE INTERACTION BETWEEN SIMULATION AND OPTIMIZATION

Finally, looking at the whole interaction of simulation and optimization routines within SIMEON, there are two classes that are designed to serve as the interfaces between the simulation and optimization engines- `SimulationOptimizationModel` and `SimulationBasedProblem`. These two classes would normally be further customized according to the simulation-based optimization problem that is going to be solved. The implementation of `SimulationOptimizationModel` class is aimed to provide some attributes and methods such that the simulation model used within SIMEON can specify its input and outcome measures that are relevant for the optimization study (appendix B). Those outcome measures can then be used to provide the user of SIMEON a possibility to construct a custom-made objective functions through a java-based modeling interface (as depicted by the modeling relation in system architecture diagram). On the other hand, the `SimulationBasedProblem` class is aimed to perform the evaluation of those custom-made objective functions (appendix C). The results of the evaluation shall be used by the algorithm as inputs for further iterations.

4.2.1.2 Design of SIMEON framework

Given the specification of the object components within the simulation and optimization engines and the interactions between them, we can finally present the complete design of SIMEON framework. Figure 25 depicts the class diagram of SIMEON framework.

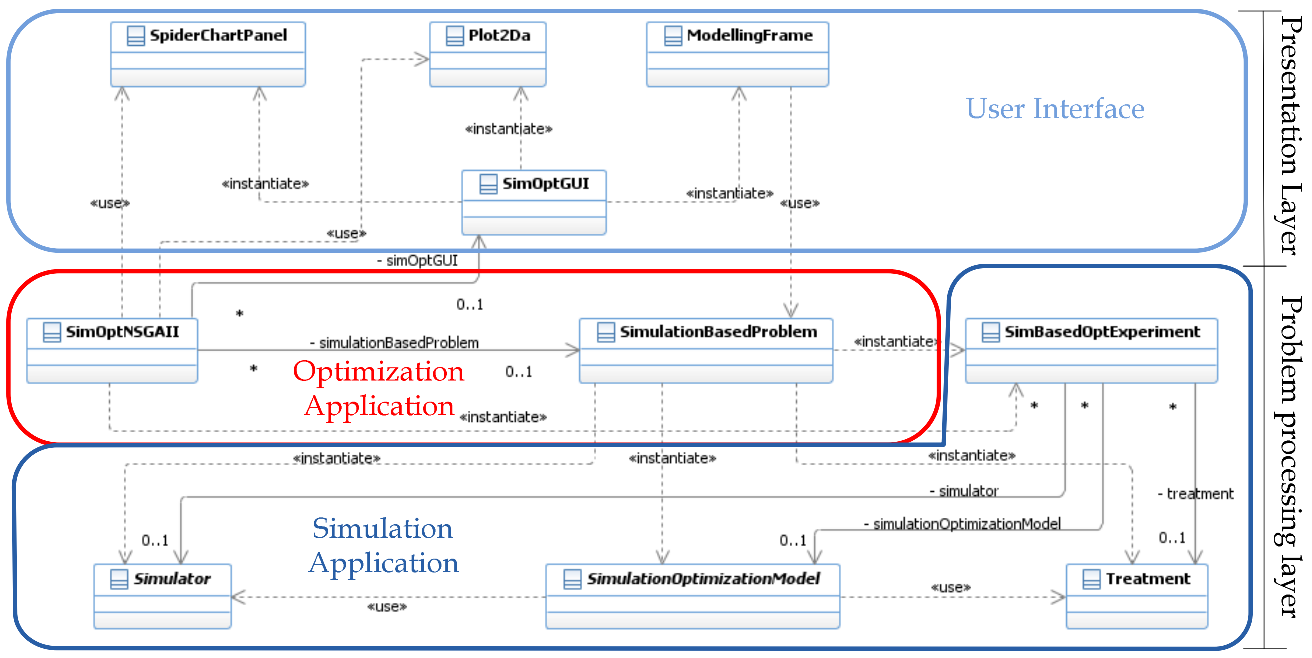


FIGURE 25 HIGH LEVEL UML CLASS DIAGRAM OF THE SIMEON FRAMEWORK

As depicted on the figure above, SIMEON is constructed by java-based implementations of the three layers that exist in previously presented system architecture (Figure 18). In the presentation layer, all visualization objects are contained (the elaboration of these objects will be done in section 4.2.3). In the problem processing layer, there lie the simulation and optimization engines with their main object components grouped in different sub-systems. Finally, the objects from the knowledge layer are not presented here as they are dependent upon the type of knowledge and data that are going to be used together with the simulation model. We can therefore clearly see that the design of the framework above is a translation of SIMEON system's architecture

Finally, looking back at the technical requirements, we can see that the design of SIMEON satisfies all technical requirements within generality criterion. The use of multi-tier architecture together with java-based implementation is the chief design decision that leads SIMEON to such fulfillment. In addition to that, we can also see, from the way the components in optimization engine is constructed (`SimOptNSGAII` and `SimulationBasedProblem`), that SIMEON is well extendible.

- **Answer to the first research question**

Finally, we can now provide the answer for the first research question:

1. How can metaheuristic techniques be integrated into the framework of modeling and simulation that is currently used to solve real-world problems?

Recall that the conceptual design of SIMEON has provided the partial answer for the first research question. This design substantiates, conceptually, that the integration can be performed by designing the metaheuristic algorithm (e.g. NSGAII) as a routine that iteratively sets the values of the decision variables of the simulation model based on the previous evaluations of the outcome measures of interest. In the implementation level, we present the system architecture and the detailed designs of both the simulation and optimization components that embody such a concept. Next, these detailed designs of the components altogether with the design of the interaction between them have substantiated the implementation of the integration suggested in the conceptual design. Therefore, both the conceptual and the detailed designs have provided a holistic answer on the question above.

4.2.2 Design of the Genetic Chromosome and Operators

It is well known in the application of evolutionary-based algorithms that the design of the genetic chromosome and genetic operators (i.e. crossover and mutation) poses a substantial challenge. This is mainly because the schemes used to represent/encode a solution to a genetic chromosome and to perform the genetic operations significantly influence the ability of the algorithm to explore the solution space. From computational perspective, this influences the problem difficulty/complexity. Therefore, it is important to avoid the use of bad or inefficient chromosome and operator designs such that the algorithm within SIMEON can find the optimal solutions (by reaching convergent state) in a reasonable amount of time.

Fortunately, there are researches that have been done in this particular field that result in guidelines that the users of EA can use to design efficient representations (Goldberg, 1989; Palmer & Kershenbaum, 1994; Ronald, 1997). However, among all the guidelines proposed only the guidelines of Palmer and Kershenbaum (1994) that are neutral from the fundamental criticisms posed to building blocks[4] hypothesis (Burjorjee, 2008). Henceforth, the designs of the representations in SIMEON follow Palmer's guidelines:

1. An encoding should be able to represent all possible solutions.
2. An encoding should be unbiased in the sense that all possible individuals are equally represented in the set of all possible genetic chromosomes.
3. An encoding should encode no infeasible solutions.
4. The decoding of the solutions from the chromosomes should be easy.

[4]A notion proposed by Goldberg (1989) based on the schema theory of Holland (1975) which refers to the highly fit solution to sub-problems that are to be solved (e.g. one variable in a set of decision variables that has a high fitness).

5. An encoding should possess locality. Small changes in the chromosomes should result in small changes in the solutions.

From the design principles delineated above, we can infer that the best way to design an efficient representation is by carefully analyzing the structure of the problem to be solved and subsequently designing a representation that complies with all the principles above or often also called "natural" representation.

Looking back to the second research question, two types of decision variable that would be dealt by the framework have been identified: quantitative and qualitative decision variables that represent continuous and discrete optimization problems. It is obvious that a simulation-based problem/model might contain either only quantitative or qualitative decision variables or both types of decision variables at the same time. To enable SIMEON having the desired generality and yet also having the ability to solve the problems efficiently, different genetic representations and operators that comply the design principles above are implemented. The central idea here is to enable SIMEON users to select the most efficient representations and operators based on the types of the decision problems that are encountered. Following are the specifications of all genetic chromosomes and operators implemented in SIMEON.

- **Genetic Chromosomes**

There are four different genetic chromosomes implemented in SIMEON:

1. Binary/Binary-Real representation

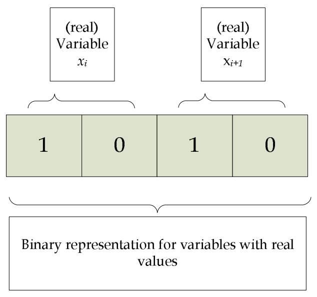

(real) Variable x_i
(real) Variable x_{i+1}
1
0
1
0
Binary representation for variables with real values

FIGURE 26 BINARY REPRESENTATION

Binary representation is the most widely used representation in genetic algorithm (GA) as it is the original representation that was used by the inventor of this algorithm (Holland, 1973). It is well known that this representation is quite efficient to solve many combinatorial optimization problems given all the design principles mentioned above are used. However, it is also well known that this representation does not always have a good performance in solving all continuous optimization problems due to the existence of *hamming cliffs*[5]

In SIMEON, binary representation is particularly designed to solve continuous optimization problem. Hence, mapping from chromosome to solution is done from binary to real values. The aim of

[5] A problem that is caused by two numerically adjacent values having far apart binary representations. An example is binary representations of decimal number 7 and 8 which bit representations are 0111 and 1000. This condition significantly hampers the search ability of the algorithm to those two adjacent values.

providing this representation is to allow SIMEON to have an alternative representation next to real representation to solve continuous optimization problem, making it possible for the user to perform any necessary experimentation.

2. Real representation

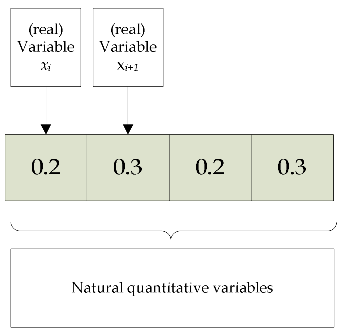


FIGURE 27 REAL REPRESENTATION

Real number representation is best used to solve problems with quantitative decision variables/continuous optimization problems as it uses direct/natural representation of the solution which in turn satisfies all the design principles above (Gen & Cheng, 2000). This is a representation that is designed to solve most of the optimization problems dealt with SIMEON. In the context of solving simulation-based problem, the function to be optimized is made of interactions that are implemented within the simulation model.

3. Integer representation

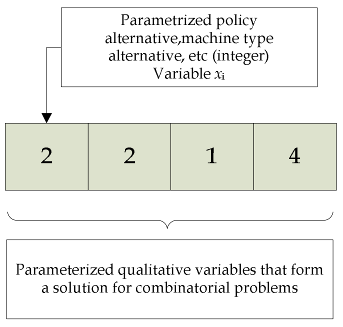

Parametrized policy alternative,machine type alternative, etc (integer) Variable x_i
2
2
1
4
Parameterized qualitative variables that form a solution for combinatorial problems

FIGURE 28 INTEGER REPRESENTATION

Integer representation is designed to solve problems that contain decision variables that are qualitative in nature such as policy alternatives, type alternatives, etc. (Azadivar & Tompkins, 1999). Furthermore, this representation can also be used to deal with discrete optimization problems given all the alternatives are represented in integer values. From a (optimization) modeling perspective the presence of different qualitative alternatives in a problem easily forms a combinatorial optimization problem which complexity grows exponentially as the number of the decision variables increases. It is noteworthy that in the context of simulation-based optimization, structural alternatives of a model can be represented into integer decision variables to which SIMEON is able to provide an optimization service.

4. Integer Real representation

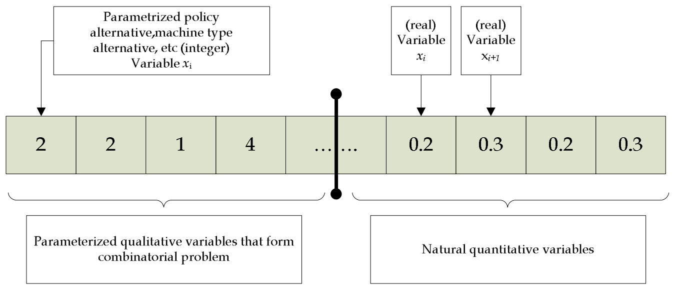

Parametrized policy alternative,machine type alternative, etc (integer) Variable x_i
(real) Variable x_i
(real) Variable x_{i+1}
2
2
1
4
……
0.2
0.3
0.2
0.3
Parameterized qualitative variables that form combinatorial problem
Natural quantitative variables

FIGURE 29 INTEGER-REAL REPRESENTATION

Integer-Real representation is a form of solution encoding that makes use integer and real representations simultaneously. This representation is particularly useful to solve simulation-based or mathematical model-based problems that contain both quantitative and qualitative decision variables. The implementation of this representation in SIMEON is deemed very useful considering that there are many alternatives in a modeled system that can't be easily represented by using only one type of representation. Another important advantage is that the dynamic relationships between different types of decision variables are taken into account and involved in the algorithm's iteration. To perform crossover and mutation for this representation, two operators are specially implemented: Integer-Real crossover and Integer-Real mutation. These operators perform different genetic operations on different segments of the chromosome depending on the types of the decision variables.

- **Crossover**

There are three different crossover operations implemented in SIMEON:

1. Single Point Crossover (for binary/binary-real representation and integer representation)

Single point crossover is the original crossover procedure proposed by Holland (1973). It is also the most common crossover operation that is used in GAs. In SIMEON this operator is designed to be used with binary and integer solution types. It works by exchanging the fragments of the parent's chromosomes which cut point is randomly determined. Since crossover is the main search operator in GAs, the rate is normally set to be quite high (0.8-0.9). Figure 30 illustrates single point crossover procedure for integer representation.

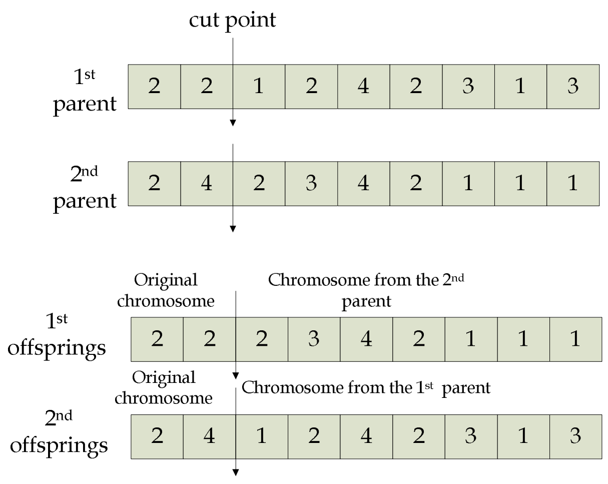


FIGURE 30 SINGLE POINT CROSSOVER

2. Simulated Binary Crossover (for real representation)

This operator is proposed by Deb and Ram (1994) to deal with specifically continuous optimization problems and it has been proven to be a powerful search operator (Deb & Ram B., 1994). Offspring solutions are created by first using standard single point crossover and then subsequently a blending operator that works with polynomial probability distribution is applied for each decision variable. A parameter called distribution index (ηc), is used to adjust the probability distribution which in turn adjusts the similarity of the offspring solutions to the parent solutions. The larger the ηc value, the closer to the parents the resulting offspring solutions are and vice versa. For further details readers are referred to the author's paper.

3. Integer-Real Crossover (for integer-real representation)

This operator is a combination of single point crossover (for the integer genes) and simulated binary crossover (for the real genes) operations. The routine is as follows: after single point crossover operation takes place on the integer segment, simulated binary crossover follows for the real segment. In SIMEON single rate of crossover is used for both segments.

- **Mutation**

There are three different mutation operations implemented in SIMEON:

1. Bit Flip Mutation (for binary and integer representation)

This is the original mutation procedure proposed by Holland (1973) to enable GA escape the local optima. In case binary representation is used, the procedure involves altering the value of one gene (in random position) of the chromosome from 0 to 1 and vice versa. In case integer representation is used, this involves altering the value of a random position gene to a random value between defined upper bound and lower bound values. The rate of mutation is normally set to be 1/ number of variables. Figure 31 illustrates bit flip mutation for integer representation.

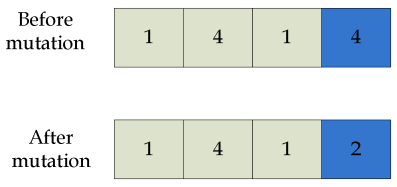


FIGURE 31 BIT FLIP MUTATION

2. Polynomial Mutation (for real representation)

Polynomial mutation is an operator proposed by Deb and Goyal (1996) to deal with real/continuous decision variable (Deb & Goyal, 1996). It works by first altering the original value of the decision variable to a mutated value using the mutation probability and then changing that value to a neighboring value using a polynomial probability distribution. Its mean is the original value and its variance is defined as a function of a distribution index ηp. The larger the ηc value, the closer to the parent the resulting offspring solution is and vice versa

3. Integer-Real Mutation (for integer-real representation)

This operator is a combination of bit flip mutation (for the integer segment) and polynomial mutation (for the real segment). The routine of this operator is as follows: first, bit flip mutation is applied to the integer part of the chromosome and then polynomial mutation is applied for the real part of the chromosome.

- Selection

There are two selection operators implemented in SIMEON: binary tournament selection and crowding tournament selection. Both selection schemes are implemented as standard selection schemes that are used by NSGA-II. They both work well without further adaption to deal with different types of decision variables.

As elaborated above, we propose the implementation of different representations and genetic operators (i.e. crossover and mutation) to answer the challenge of dealing with both continuous and discrete optimization problems in generic yet efficient way. From the system requirements perspective, we can see that this design meets the generality, efficiency and multi-dimensionality criteria. The key contribution of the design lies in the designs of the representations that work with standard crossover and mutation operators. This avoids the user of the framework from the difficulty of designing custom-made crossover and mutation operators.

- **Answer to the second research questions**

Finally, we can now provide the answer for the second research question:

2. How should the genetic chromosome and operators be designed such that the proposed framework has the capability to support the optimization of quantitative and qualitative variables which enables it to solve continuous and discrete multi-objective optimization problems?

We present the designs of four different genetic chromosomes and six genetic operators that comply to the design principles advocated by Palmer (1994). These four different representations and six genetic operators are efficiently designed to deal with multi-objective optimization problems that involve quantitative and qualitative decision variables. Depending on the types of the decision variables that are to be optimized, a proper representation and operators can then be chosen.

4.2.3 Design of SIMEON User Interface

User interface (UI) of SIMEON is the main constituent of the presentation layer in SIMEON's architecture. It plays an important role to take and present messages from and to the user through various language and presentation sub-systems. Therefore, its design is vital for the usability SIMEON for both expert and non-expert users. In addition to that, SIMEON UI also has to be designed such that all the capabilities of the problem-solving sub-system can be well utilized. This is aimed to fully accommodate multi-dimensionality requirements that have been satisfied with the design of the genetic chromosomes and operators presented in chapter 4.2.2. Figure 32 depicts the extended UML class diagram of SIMEON user interface which is relevant for the usability and multidimensionality requirements. To provide explanations for the features of the UI, references to the objects depicted in the figure below will be used.

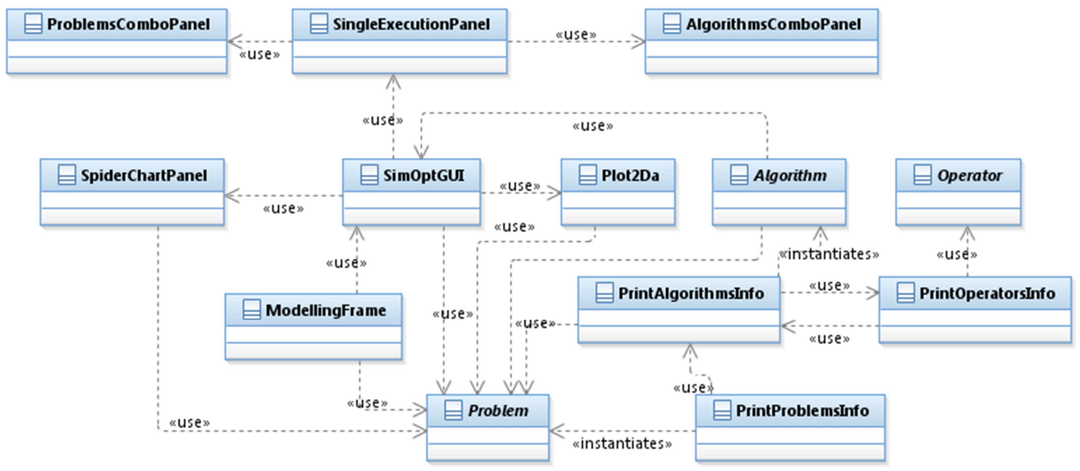
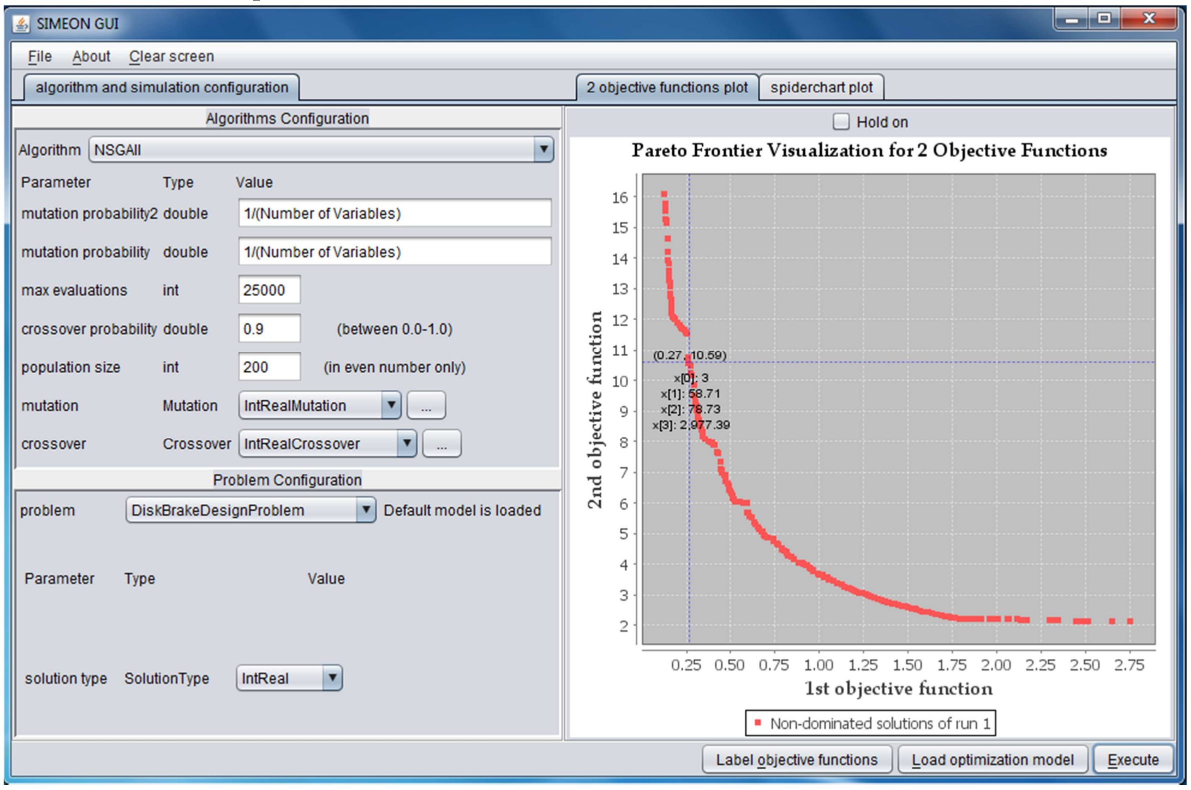


FIGURE 32 ARCHITECTURE OF THE GUI

Despite the focus to develop the user interface for the non-expert users, we have identified all primary messages and that both types of users have to give to and receive from SIMEON in the usability requirements in section 3.1.1. To facilitate the required interaction, following components of the user interface are designed based on those requirements:

1. SIMEON control panel


SIMEON GUI
File About Clear screen
algorithm and simulation configuration
2 objective functions plot
spiderchart plot
Hold on
Label objective functions
Load optimization model
Execute

Algorithms Configuration

Algorithm NSGAII

Parameter	Type	Value	
mutation probability2	double	1/(Number of Variables)	
mutation probability	double	1/(Number of Variables)	
max evaluations	int	25000	
crossover probability	double	0.9	(between 0.0-1.0)
population size	int	200	(in even number only)
mutation	Mutation	IntRealMutation	
crossover	Crossover	IntRealCrossover	

Problem Configuration

problem DiskBrakeDesignProblem Default model is loaded

Parameter	Type	Value
solution type	SolutionType	IntReal

FIGURE 33 SIMEON CONTROL PANEL

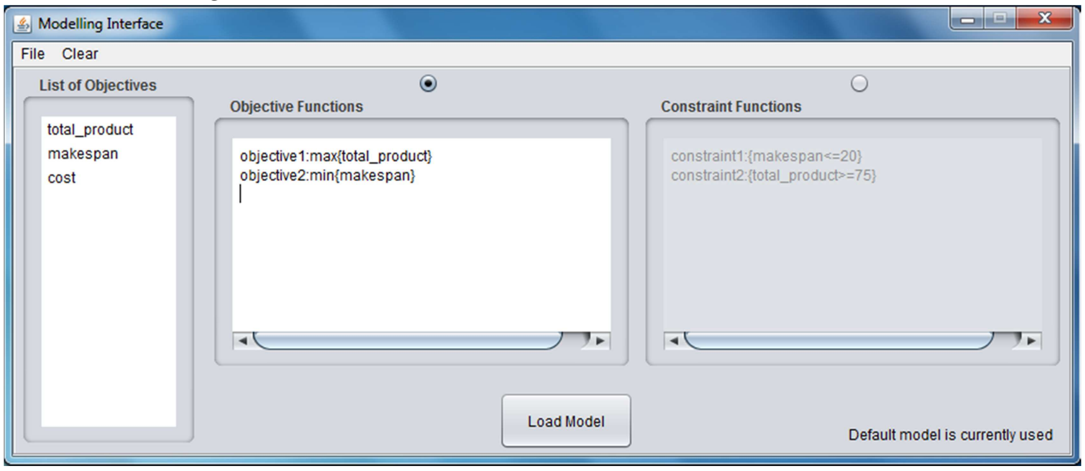
Control panel of SIMEON is designed to serve two main functions:

1. Allowing the user configuring the algorithm and problem to be executed using SIMEON.
2. Visualizing the non-dominated solutions for either 2 objective functions (on the 2 objective functions plot) or multiple objective functions (on the spiderchart plot).

To serve the first function, a panel (a representation of single execution panel object), named algorithm and simulation configuration is provided (left tab). This panel consists of two combo panels: algorithm combo panel (top left) and problem combo panel (bottom left). On the algorithm combo panel, the type of the algorithm to use and its parameter can be configured. Currently there are two different types of algorithm implemented, SimOptNSGAII (to solve simulation-based optimization problem) and NSGAII (to solve mathematical model-based optimization problem). The content of this panel shall adjust themselves depending on the type of the algorithm selected. Furthermore, on the problem combo panel, the type of the problem to solve and its setting can be configured. There are also two types of problem which the user may select i.e. mathematical-model based problem and simulation-based problem. In case a simulation-based problem is selected then the user will be prompted to configure the run control of the simulation.

Furthermore, to guide the user using SIMEON appropriately, several notification services such as message box, tooltip, etc. are also implemented (appendix D). Finally, the default settings for both the algorithm and problem combo panel are loaded from a java data file named gui.data. This file is editable either in the development environment or in the windows directory using any standard text editor.

2. SIMEON modeling interface



total_product
makespan
cost

[● Objective Functions] [○ Constraint Functions]

```
objective1:max{total_product}
objective2:min{makespan}
```

```
constraint1:{makespan<=20}
constraint2:{total_product>=75}
```

Default model is currently used

FIGURE 34 SIMEON MODELING INTERFACE

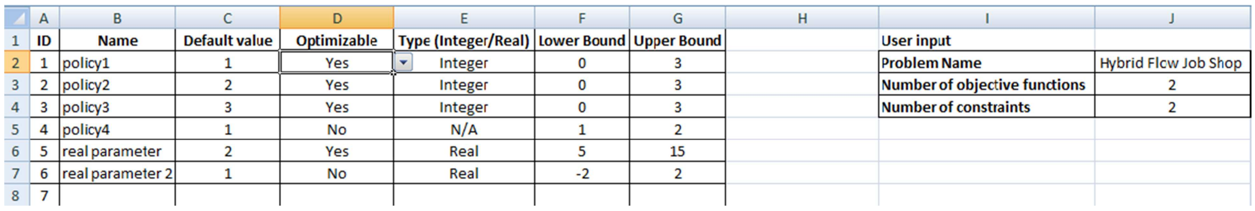
The aim of implementing a modeling interface (a representation of modeling frame object) is to provide a separation of concerns between modeling the outcome measures of interest of the simulation model and modeling the optimization problem based on those outcome measures of interest. This allows the user to make a custom-made objective and constraint functions depending on the objective of optimization study and hence, giving the user, a flexibility to do experimentations with different optimization models. To make an optimization model, the user can make use the outcome measures on the list of this interface or load it from a previously saved text file. Those outcome measures are taken from the simulation model. Furthermore, the grammar used to construct an optimization model is similar to the standard optimization model defined in chapter 2.3.

3. Input and output Excel sheets

Excel sheets are used as an example implementation of how SIMEON can be connected to an external software that is widely used such as Microsoft Excel. In SIMEON, this is done to also facilitate the non-expert users to:

1. Specify the decision variables that are to be optimized by SIMEON (Figure 35).
2. Get the result of each SIMEON execution.

It is important to note that the values specified for the number of objective functions and constraints in the input excel sheet are referred by the modeling interface to construct the vector of objective and constraint functions.



	A	B	C	D	E	F	G	H	I	J
1	ID	Name	Default value	Optimizable	Type (Integer/Real)	Lower Bound	Upper Bound		User input	
2	1	policy1	1	Yes	Integer	0	3		Problem Name	Hybrid Flcw Job Shop
3	2	policy2	2	Yes	Integer	0	3		Number of objective functions	2
4	3	policy3	3	Yes	Integer	0	3		Number of constraints	2
5	4	policy4	1	No	N/A	1	2			
6	5	real parameter	2	Yes	Real	5	15			
7	6	real parameter 2	1	No	Real	-2	2			
8	7									

FIGURE 35 MICROSOFT EXCEL AS AN INTERFACE TO SPECIFY THE DECISION VARIABLES

It is now apparent that to run SIMEON all the interfaces above have to be configured. To structure all the steps needed in running SIMEON, Appendix E presents the relevant use case scenario.

With the implementation of the user interface components described above, all the primary messages from and to SIMEON defined in section 3.1.1 are well facilitated. However, there is another important feature of the UI that we implement to translate the multidimensionality requirements. SIMEON UI uses the factory method pattern (Gamma, Helm, Johnson, & Vlissides, 1995) to instantiate the algorithm, problem and operator objects. This can be seen in the relationship shown between `PrintAlgorithmsInfo – Algorithm`; `PrintProblemsInfo - Problem` and `PrintOperatorsInfo - Operator` in Figure 32. They represent the factories and the products respectively. This way, not only the UI supports the generality and extendibility of the framework, it also enables the user to easily adapt the algorithm and its operator to solve different simulation-based problems based on the types of the decision variables that are dealt.

- **Answer to third research question**

We can now provide the answer for the third research question:

3. What does the design of the interface components that are able to facilitate the necessary changes in the simulation model look like?

We present the design of SIMEON UI together with its detailed components that are based on the multidimensionality and usability requirements. While the designs of the genetic chromosomes and operators have given SIMEON the ability to suggest solutions involving parameter and structural changes in the simulation model, the UI of SIMEON is designed to exploit this capability. First, it helps the user in specifying the decision variables and the optimization problem using the outcome measures of interest from the simulation model. Second, and most importantly it facilitates the user in selecting the most appropriate representation and genetic operators such that any structural or parameter changes are possible. This way the UI could facilitate the necessary changes in the simulation model. The key design that contributes to this feature is the factory method pattern that is implemented to facilitate the instantiation of the algorithm and its operators.

4.3 CONCLUSION

This chapter has presented the way the second Systems Engineering life-cycle phase has been applied to develop SIMEON. It results in SIMEON architecture and the detailed designs of its sub-systems. Based on the detailed design phase, the first three research questions are answered. The fact that the detailed design of SIMEON framework is presented, an answer for the first research question is given systematically and in holistic fashion by using design entities both from the conceptual and implementation levels. Next, the second research question is answered by presenting the designs of the genetic chromosomes, and operators within NSGAII. Last but not least, the detailed design of the User Interface provides the answer for the third research question.

Furthermore, as the designs of SIMEON components are substantiated and discussed throughout the whole chapter, it can also be concluded that they satisfy all the technical requirements that have been previously formulated. The following requirements allocation sheet is presented to elaborate how each requirement is satisfied with what design.

TABLE 3 REQUIREMENTS ALLOCATION SHEET WITH RESPECT TO SIMEON DESIGN

Requirements Allocation Sheet				
ID	**Requirement**	**Target performance**	**Fulfillment measure**	**Chapter**
1	Generality			
1.1	SIMEON should be able to solve optimization problems in different domains	Problems in the domain of supply chain, logistic and business planning	The separation of concern between the algorithm and the problems to be solved and the extendibility of the problem component	4.2.1.1 and 4.2.3
1.2	The optimizer should be able to regard the simulation models as black boxes	Using a metaheuristic algorithm that works based on input and output of the model	Implementation of SimOptNSGAII as the optimization engine of SIMEON	4.2.1.1
1.3	The simulation models should be able to be run independently	Building an optimizer that can be coupled and decoupled easily	The design of SIMEON architecture using multi-tier concept and the separation of concern between the algorithm and the problem	4.1 and 4.2.1.3
1.4	SIMEON should be able to get inputs from different software and produce outputs to different software	Taking inputs and producing outputs from and to Microsoft excel and access	The design of SIMEON using multi-tier architecture and its implementation in JAVA	4.1
1.5	SIMEON should be able to be operated in different operating system platforms	SIMEON is functional in all operating systems	The use of JAVA programming language which enables platform independent program execution	4.2.1.1

1.6	SIMEON should be developed for DSOL-based models	SIMEON serves as a DSOL-based optimization framework	The design of the interaction between the DSOL-based simulation engine and the optimization engine	4.2.1.2
2	Efficiency			
2.1	SIMEON should be extendible to incorporate specialized optimization algorithms	SIMEON allows the implementation of additional optimizer that could work efficiently to solve specific problems	The possibility to extend SIMEON with any algorithms from `Algorithm` class	4.2.1.1
2.2	The optimizer should have seamless input and output flow to the simulation models	Using an internal interface between the simulation models and optimizer	The design of the interaction between the DSOL-based simulation engine and the optimization engine	4.2.1.2
3	Multi dimensionality			
3.1	SIMEON should be able to optimize multiple-objective functions	SIMEON is successfully applied to produce the Pareto Front for the case studies	The use of NSGAII as the optimizer to find the non-dominated solutions	2.4.2
3.2	SIMEON should be able to optimize quantitative parameters of the simulation model	Max number of quantitative parameters in the case studies	The use of Polynomial mutation and SBX crossover	4.2.2
3.3	SIMEON should be able to optimize non-quantitative parameters of the simulation model (e.g. modules, system component)	Max number of non-quantitative parameters in the case studies	The use of Integer Flip Mutation and Single point crossover	4.2.2
3.4	SIMEON should be able to optimize quantitative and non-quantitative parameter of the simulation model simultaneously	Total number of non-quantitative and quantitative parameters in the case studies	The use of Integer-Real Mutation and Integer-Real crossover	4.2.2
3.5	SIMEON should be able to show the optimal trade-offs in the objectives in case they are conflicting	Generating and visualizing Pareto Front for multiple objectives	The use of 2D Plot and spider chart visualization in a JAVA-based GUI	4.2.3
4	Usability to the users			

4.1	SIMEON should be able to be used by the non-expert users comprehensively by the help of any necessary user interface	Non-expert users only need to have basic knowledge of modeling and simulation techniques to use SIMEON	The design of user interface in widely used office suite +JAVA-based GUI	4.2.3
4.2	SIMEON should be able to be extended by the expert users to solve specific optimization problems	The expert users can perform any necessary modifications in the optimizer and simulation (e.g. adding more efficient algorithms)	The object oriented structure of the SIMEON +clear documentation on the source code of the framework	4.2.1 and 4.2.3
4.3	SIMEON should be adaptable to the changes in the types of decision variables (i.e. quantitative and non-quantitative) that are to be optimized in different problem domains	SIMEON can facilitate the change of types of decision variables to optimize automatically	The design of the user interface that allows the specification of the types of the decision variables and the relevant genetic operators	4.2.3

5. SIMEON OPERATIONAL TEST AND EVALUATION

In this chapter the third systems engineering life-cycle phase is performed: system operation and maintenance (Sage & Armstrong, 2000). There are originally three sub-phases that form this life-cyle: operational implementation, operational test and evaluation, and operational functioning and maintenance. The first sub-phase is successfully done through the complete implementation of SIMEON based on the detailed designs elaborated in the previous chapter. While the third sub- phase falls outside of this research scope, we focus our discussion in this chapter on the second phase which we can divide into two main activities:

1. SIMEON operational test
2. SIMEON evaluation

The operational test phase is intended to assess whether SIMEON has satisfied all the technical requirements on the operational level. In addition to that, this phase is also used to reveal the ultimate performance limitations of SIMEON. To serve this purpose, three test cases with different scales and problem domains are used and their results are presented in the following section. The first test case uses mathematical model-based optimization problem for which scientific benchmark results are available. The other two test cases use simulation-based optimization problems that are technically relevant to the possible applications of SIMEON in the real world. Finally, based on the test results, requirements and recommendations to deal with the limitations encountered are formulated.

Next, the evaluation phase is intended to assess the potential contributions of SIMEON to solve the real-world decision making problems and to assess how SIMEON can increase the technological capability of Accenture as the user to solve real-world problems. This is done by taking into account the results of the operational test phase and conducting interviews with the simulation experts in Accenture. This phase results in the provision of answers for the fourth and the fifth research questions. Finally, a conclusion about this chapter is presented to conclude this last life-cycle.

5.1 SIMEON OPERATIONAL TEST

In this section the results of using SIMEON to solve the following test cases are presented:

1. Disk brake design problem
2. Flexible manufacturing system scheduling problem
3. Supply chain optimization problem

The first test case comes from engineering design problem which is formulated into a multi-objective optimization problem. The problem involves both discrete and continuous variables. The intention to use this model is to test the performance SIMEON, particularly its Integer-Real representation and operators, in solving a mathematical model-based problem that involves both types of variables. Benchmark results from other research are used as a means to perform the performance comparison. Furthermore, the result of the first test case is used to provide solid scientific evidence in assessing the performance of SIMEON for the two other cases of which any benchmark results are not available.

The second and the third test cases are simulation-based test cases. The second test case pertains a well-known manufacturing optimization problem, namely scheduling. Similar to the first test case, this test case involves discrete and continuous decision variables. Next, the third test case comes from the supply chain management field which is expected to be the main application domain for SIMEON. There is only one type of decision variable in this case namely continuous variable.

Last but not least, a challenging technical problem namely memory leak is also going to be discussed. Foreseeing that this problem will seriously hamper the application of SIMEON to solve large-scale problems, we present the solutions and recommendations as the anticipation action. Finally, requirements are also formulated to enable the maximum utilization of SIMEON and to avoid the recurrence of this problem.

5.1.1 Disk Brake Design Problem

Two objectives to be optimized in this case are to minimize the mass of the disk brake and to minimize the stopping time of the resulting design (Ray & Liew, 2002). The modeled decision variables of the problem are: the inner radius of the disc (x_1), the outer radius of the disc (x_2), the engaging force (x_3), and the number of friction surfaces (x_4). The constraints for the design comprise the minimum distance between the radii (5.3), maximum length of the brake (5.4), pressure, temperature (5.5) and torque limitations (5.6). Following is the mathematical model for the problem:

$$\text{Min } f_1(x_1,x_2,x_3,x_4) = (4.9x10^{-5})(x_2^2 - x_1^2)(x_4 - 1) \tag{5.1}$$

$$\text{Min } f_2(x_1,x_2,x_3,x_4) = \frac{(9.82x10^6)(x_2^2 - x_1^2)}{x_3 x_4 \left(x_2^3 - x_1^3\right)} \tag{5.2}$$

Subject to :

$$(x_2 - x_1) - 20 \geq 0$$

$$30 - 2.5(x_4 + 1) \geq 0 \tag{5.3}$$

$$0.4 - \frac{x_3}{3.14(x_2^2 - x_1^2)} \geq 0 \tag{5.4}$$

$$1 - \frac{(2.22x10^{-3})x_3(x_2^3 - x_1^3)}{(x_2^2 - x_1^2)} \geq 0 \tag{5.5}$$

$$\frac{(2.66x10^{-2})x_3 x_4 (x_2^3 - x_1^3)}{(x_2^2 - x_1^2)} - 900 \geq 0 \tag{5.6}$$

$$55 \leq x_1 \leq 80 \tag{5.7}$$

$$75 \leq x_2 \leq 110 \tag{5.8}$$

$$1000 \leq x_3 \leq 3000 \tag{5.9}$$

$$2 \leq x_4 \leq 20 \tag{5.10}$$

x_4 is the discrete variable that represents the number of disks in the brake. Given the optimization model above, the NSGAII with integer-real representation and integer-real operators is used to generate the Pareto Frontier shown in

Figure 36. Though normally the max evaluation number is set to 10.000, In this case it is tested to perform 25000 function evaluations. This is the number of evaluations that assures the maximum exploitation of the NSGAII potential to reach convergence and hence, any increase in the evaluation number is very not likely produce better result. Table 4 shows the setting for NSGAII parameters for the corresponding result.

TABLE 4 SETTING FOR THE NSGAII PARAMETERS

Algorithm parameter	Value
population size	200
max evaluations	25000
crossover probability	0.9
mutation probability	1/number of variables
distribution index for int-real crossover	20
distribution index for int-real mutation	20

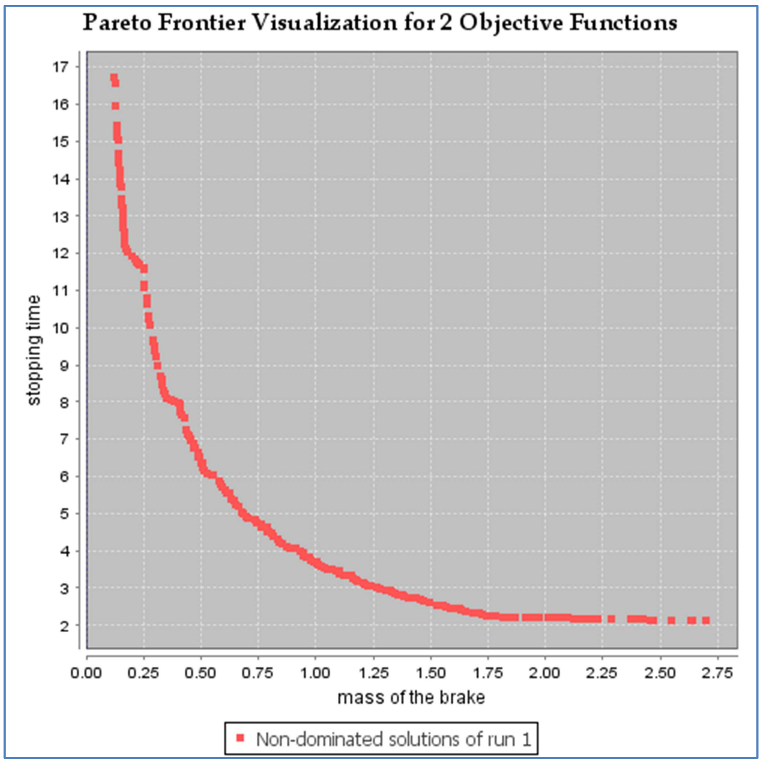


FIGURE 36 PARETO FRONTIER BY NSGAII

Next, we use two Pareto Frontiers that are found using different algorithms as the benchmarks for the performance of SIMEON. The second Pareto Frontier (Figure 37) is found using Stochastic Multi-Objective Mesh Adaptive Direct Search (SMOMADS) (Walston, 2007)and the third Pareto Frontier (Figure 38) is found using swarm algorithm (Ray & Liew, 2002).

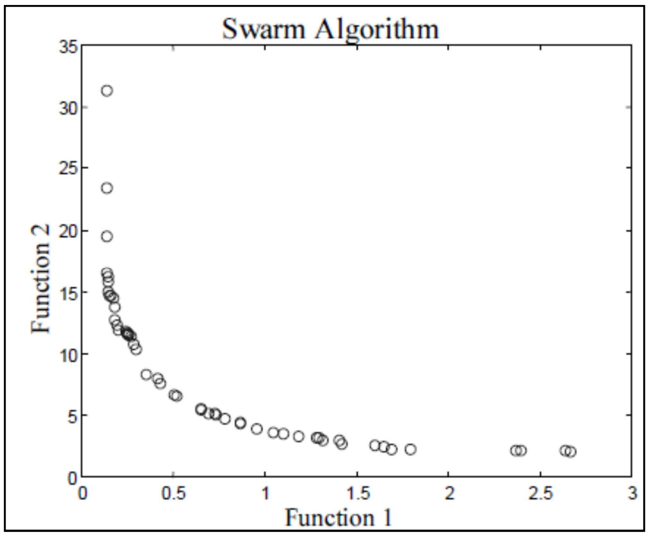


FIGURE 38 PARETO FRONT ACHIEVED BY SWARM ALGORITHM (RETRIEVED FROM RAY&LIEW (2002))

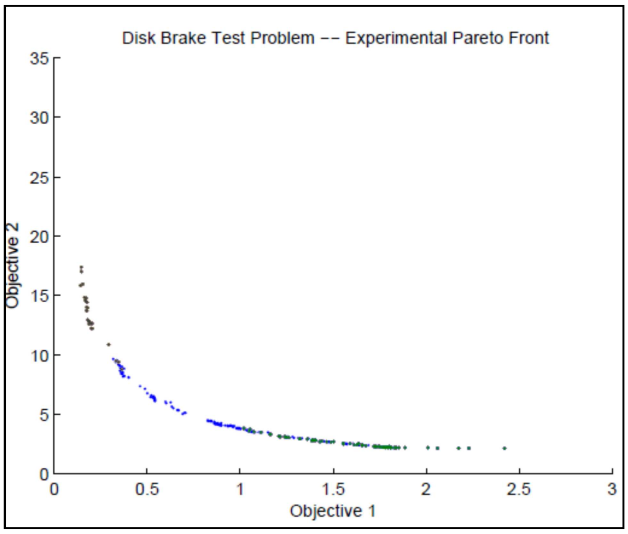


FIGURE 37 PARETO FRONT ACHIEVED BY SMOMADS (RETRIEVED FROM WALSTON (2007))

Based on the figures above, we can see that NSGAII manages to produce more non-dominated solutions in comparison to the other algorithms. Although there are some extreme solutions that NSGAII doesn't generate, we can see that NSGAII has a better spread in the overall solutions which is desirable for the decision makers. Finally, based on the non-dominated solutions that are presented above, the decision making process could advance with the information regarding the optimal trade-offs that can be made. Depending on the further preferences of the decision maker(s), a specific design solution can be picked. Normally, non-dominated solutions that are located in the compromise region (e.g. between (0.3, 8) and (1.25,3)) receive bigger concern rather than those that are of extreme solutions (e.g. (2.75,2) or (0.15, 16.5). The decision maker(s) might also be interested to the solutions in this region that have significant gain on one objective function by just compromising little value in the other objective function.

5.1.2 Simulation-based Optimization Problem

As mentioned above, there are two simulation-based problems that are used as the test cases: flexible manufacturing system scheduling problem and supply chain optimization problem. The first problem represents a theoretical simulation-based problem that is aimed to demonstrate the ability of SIMEON to successfully solve combinatorial optimization problem involving both quantitative and qualitative decision variables. Furthermore, the second problem represents the main domain and scale of the real-world problem to which SIMEON is likely used. To facilitate a good transfer of knowledge and to provide a transparency on how the test was done, the implementation steps that are needed to define those simulation-based problems in SIMEON are given in appendix F, using the third test case (supply chain optimization problem) as the example.

5.1.2.1 Flexible Manufacturing System Scheduling Problem

Scheduling problems are ubiquitous optimization problems that have been the subjects of interest for many optimization studies (Levner, 2007). This is because they are well known to possess high level of complexity to model (in analytical form) and to find the best solution for. Yet, this is also an utter reason why simulation-based optimization method is frequently developed and applied to solve problems in this domain (Hani, et al., 2008; Jacobs, Verbraeck, & Mulder, 2005; Persson, Grimm, Ng, et al., 2006).

The scheduling problem dealt here is taken from the domain of flexible manufacturing system (FMS) where the machines inside such a system have the flexibility to change the order of operations executed on the products that are processed. In this test case, there are five different products that have to be processed through three different machines (e.g. drilling, painting, and finishing), before eventually be disposed as finished products. Each of those products has a unique routing sequence throughout the machines, and a unique processing time which follows certain statistical distribution. Each of the machines has four different priority rules that can be set to handle the queue of products that enter each of those machines. They are first in first out (FIFO), last in first out (LIFO), and shortest processing time (SPT), and longest processing time (LPT). Figure 39 illustrates the discussed FMS.

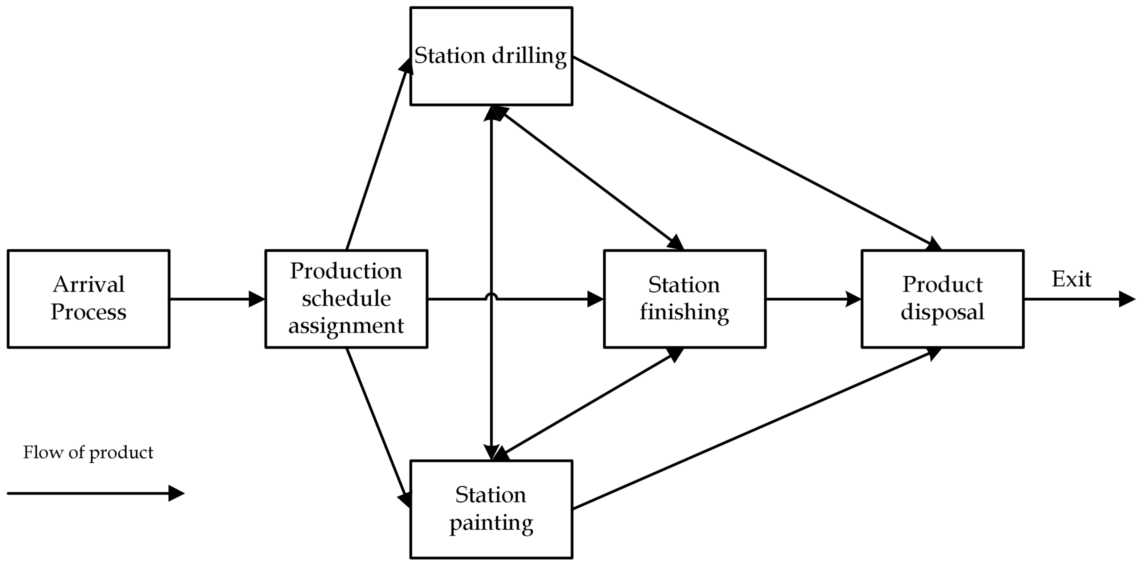


FIGURE 39 FLEXIBLE MANUFACTURING SYSTEM WITH FOUR WORK STATIONS

The objectives of the optimization study are to maximize the total production of the parts and to minimize the cost of the production within 50 hours. Next, the constraint is that we only accept solutions that have total product larger or equal than 60 unit products. The decision variables of the system design are the priority rules that are to be set for all three machines and the level of processing quality for the manufacturing system. While the calculation of the total product depends on the combination of priority rules (as the controllable initial states) and the transition function within the simulation model, the calculation for the cost depends on the level of processing quality which has a negative correlation with the cost itself. Table 5 shows the configuration on SIMEON control panel while Figure 40 shows the optimal solution of the problem.

TABLE 5 SIMEON CONFIGURATION FOR THE FMS SCHEDULING PROBLEM

Algorithm parameter	Value	Simulation parameter	Value
population size	100	Replication	3
max evaluations	10000	Run length time	50 (hours)
crossover probability	0.9	Warm up time	0
mutation probability	1/number of variables		
distribution index for int-real crossover	20		
distribution index for int-real mutation	20		

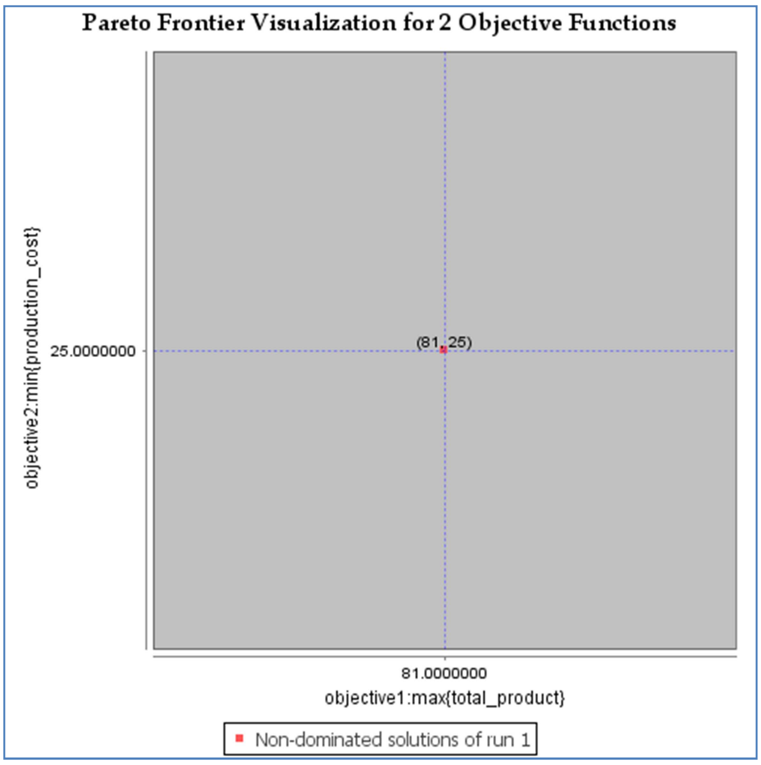


FIGURE 40 PARETO FRONTIER FOR THE FMS SCHEDULING PROBLEM

To perform the optimization using integer-real encoding, we encode the qualitative variables "priority alternatives" as integer numbers that range from 0 to 3 representing FIFO, LIFO, SPT and LPT respectively. On the other hand, the level of processing quality is encoded as a real number ranging from 5 to 15. Furthermore, 3 replications are used to take into account the stochastic nature of the model.

Based on the result presented above, there is only one optimal value for the two objectives (81 product unit and 25 cost unit). This means that in this case, the decision maker does not need to make any trade-off between the two objectives. However, this is also because those two objectives that are modeled in the simulation model don't have any correlation one another. Furthermore, the decision variables that produce this value suggest that there are two possible system configurations that are optimal:

1. Setting the priority rules to FIFO for all the machines and setting the level of processing quality to 5.
2. Setting the priority rules to SPT, LIFO, and LPT to drilling, painting and finishing machines respectively and setting the level of processing quality to 5.

5.1.2.2 Supply Chain Optimization Problem

Next to being used as the one of the main application domains for SIMEON, supply chain problem is a good example of where simulation-based optimization approach could contribute high merit. Typical real-world supply chain processes are characterized by stochastic nature, uncertainties, dynamics and non-linearity. This makes analytical /mathematical models that accurately represent such systems become very difficult to build, and computationally intractable (Ding, et al., 2004; Vidal & Goetschalckx, 1997). Not to mention that small alteration in the real system might result in a tedious effort in the modeling process. Hence, simulation-based optimization can be an answer for this difficulty as it allows the model builder to:

1. Represent such system in a more flexible and sustainable way and
2. Find the optimal solution for the relevant system design problem modeled in a systematic way.

The supply chain optimization problem dealt here is characterized by the presence of multiple actors /companies and the dynamic interactions that take place during the distribution of products, money and information between those actors. In the simulation model used as a test case, there are two suppliers, one manufacturer, three distributors and four market spots/retailers. Figure 41 illustrates this description.

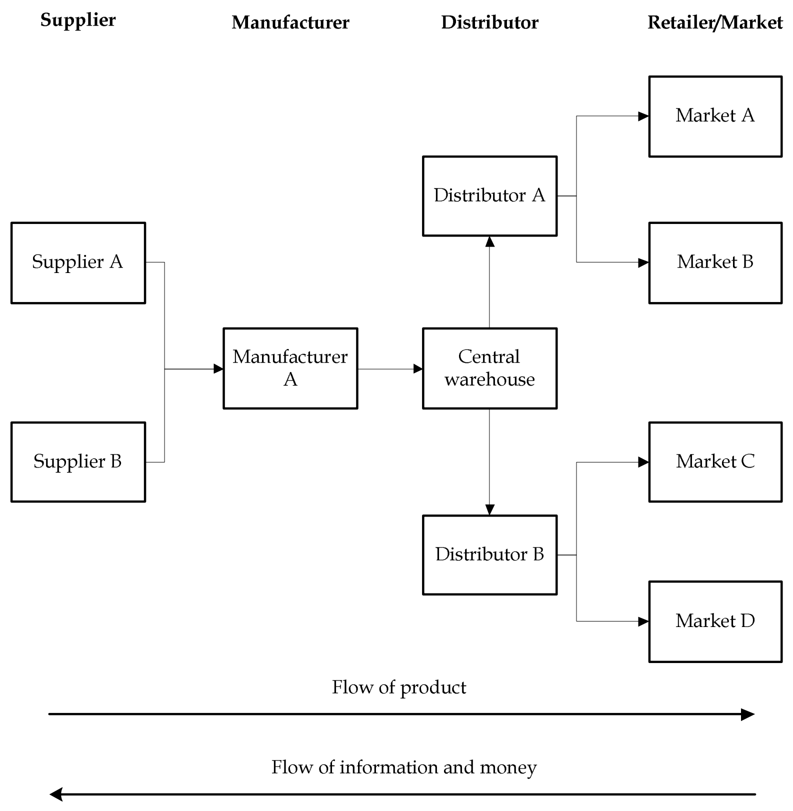


FIGURE 41 SUPPLY CHAIN NETWORK WITH MULTIPLE ACTORS

The supply chain process is started when the four markets/customers generate demands that follow certain statistical distribution. These customers are served by two distributors according to the relationship depicted by figure above. There are some variations in the transport modes that can be used that result in different delivery speed between those actors. Products are produced by a manufacturer by using materials supplied by two suppliers. The products are first accumulated in a consolidated central warehouse before they are transported to the distributors.

The objectives of the optimization study are to maximize inventory unit fill rate and to minimize the supply chain cost of a distributor (Distributor B) within 100 days. The decision variable is the service level policy of this distributor. The calculation of both the supply chain cost and the inventory unit fill rate depends on the value of the service level as a controllable initial state and the transition function within the simulation model. Table 6 shows the configuration on SIMEON control panel, while Figure 42 shows the non-dominated solutions of the problem.

TABLE 6 SIMEON CONFIGURATION FOR THE SUPPLY CHAIN OPTIMIZATION PROBLEM

Algorithm parameter	Value	Simulation parameter	Value
population size	100	Replication	1
max evaluations	10000	Run length time	100 (days)
crossover probability	0.9	Warm up time	0
mutation probability	1/number of variables		
distribution index for int-real crossover	20		
distribution index for int-real mutation	20		

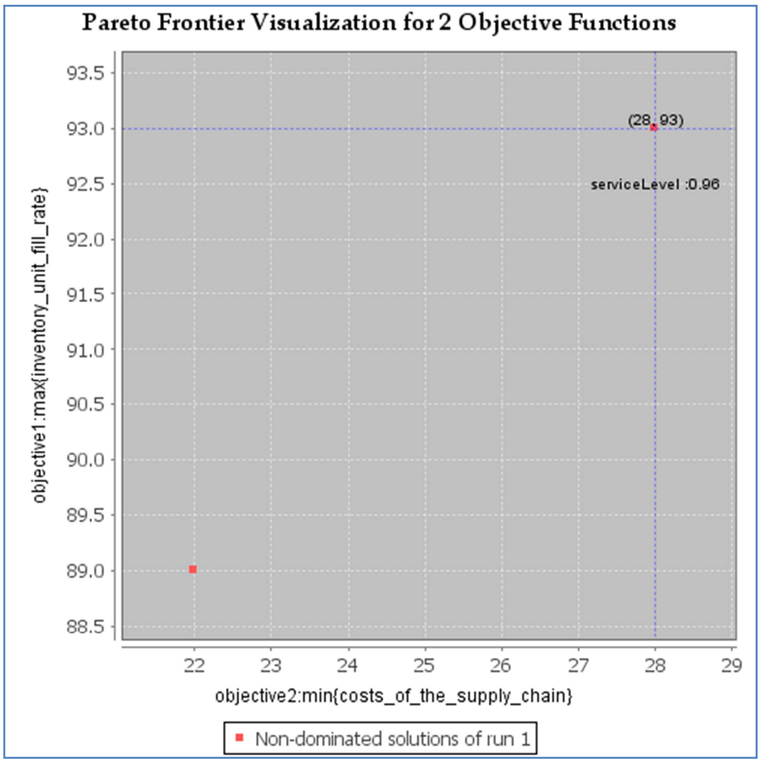


FIGURE 42 PARETO FRONT FOR SUPPLY CHAIN OPTIMIZATION PROBLEM

We can see that there are two non-dominated solutions that can be selected for a decision. The first solution suggests setting the service level at a rather high level (96%) while the second solution suggests lower service level (87%). The first solution, however, indicates a good performance on the inventory unit fill rate (93%) with relatively higher cost (28 cost unit) while the second solution gives lower cost of the supply chain (22 cost unit) with relatively lower inventory unit fill rate (89%). Given this information, the decision maker(s), can have the assurance of picking a solution out of the best alternatives possible. Further process of decision making may make use additional preferences of the decision maker(s) that depend on the specific target or circumstances of the decision maker(s).

5.1.3 Memory leak: problem and solution

During the operational test phase of the simulation-based problems, a challenging technical problem was encountered: memory leak. This is a problem caused by a program execution that allows inefficient memory allocations to the objects used within that program. By inefficient, we mean that the memory allocated to the particular objects is not released back/garbage collected when those objects are no longer needed (e.g. when one experiment involving a simulation run is finished). The typical consequence is that the whole memory that is allocated to the Java Virtual Machine, (also often called Java heap space) is eventually consumed out and the execution of the program on that virtual machine will eventually crash due to out of memory error.

This problem was revealed when the supply chain problem was tested to be run for 10.000 evaluations/experiments (Table 6) on a notebook with Intel Core 2 Duo processor, 2.26 GHz, and 3072 MB memory. Out of the available memory, 1 GB memory was allocated to the java heap space. Furthermore, ten thousand evaluations were executed as this represents sufficient iterations for the NSGAII to find the optimal solution for supply chain problem (considering that the problem is a fairly simple optimization problem with one decision variable). By using a memory profiler tool called VisualVM (https://visualvm.dev.java.net/), we did some investigations to figure out the cause of the memory leak. During our investigation, it was found that SIMEON suffered from severe memory shortage on the 347th experiments. Figure 43 shows the consumption of the heap space over time. The upward sloping trend is a clear sign for a memory leak.

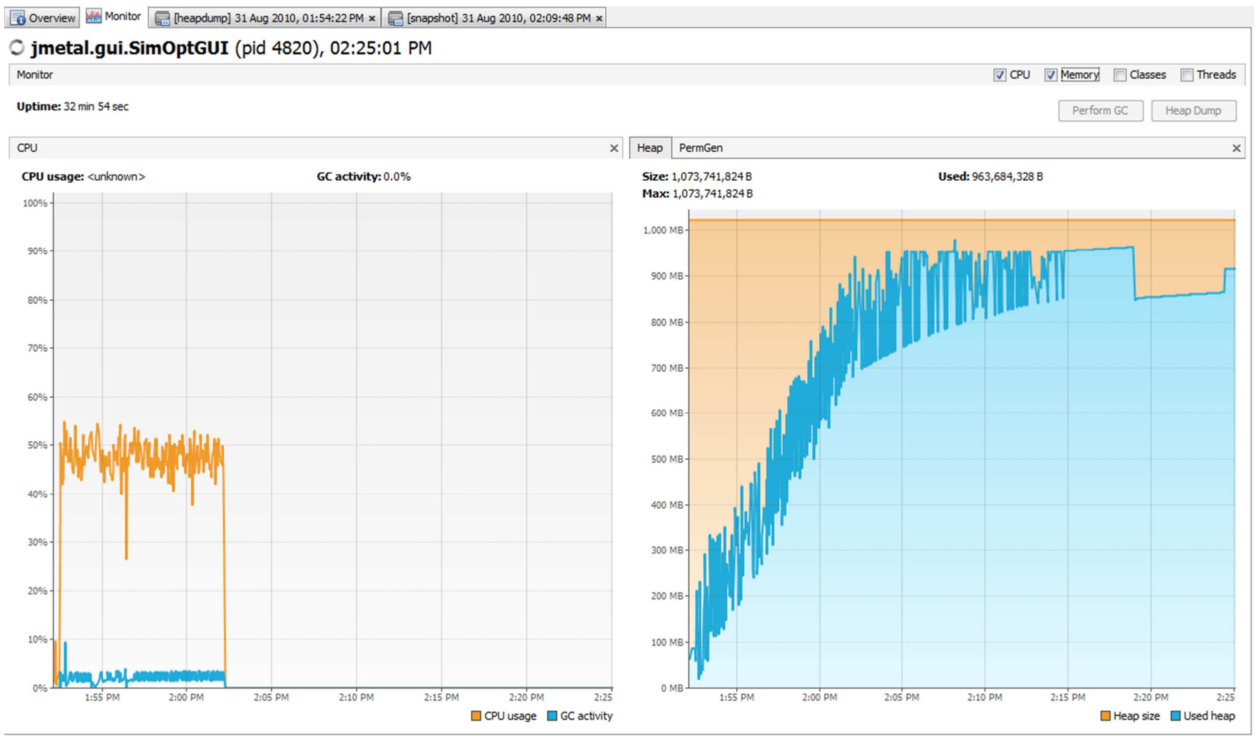


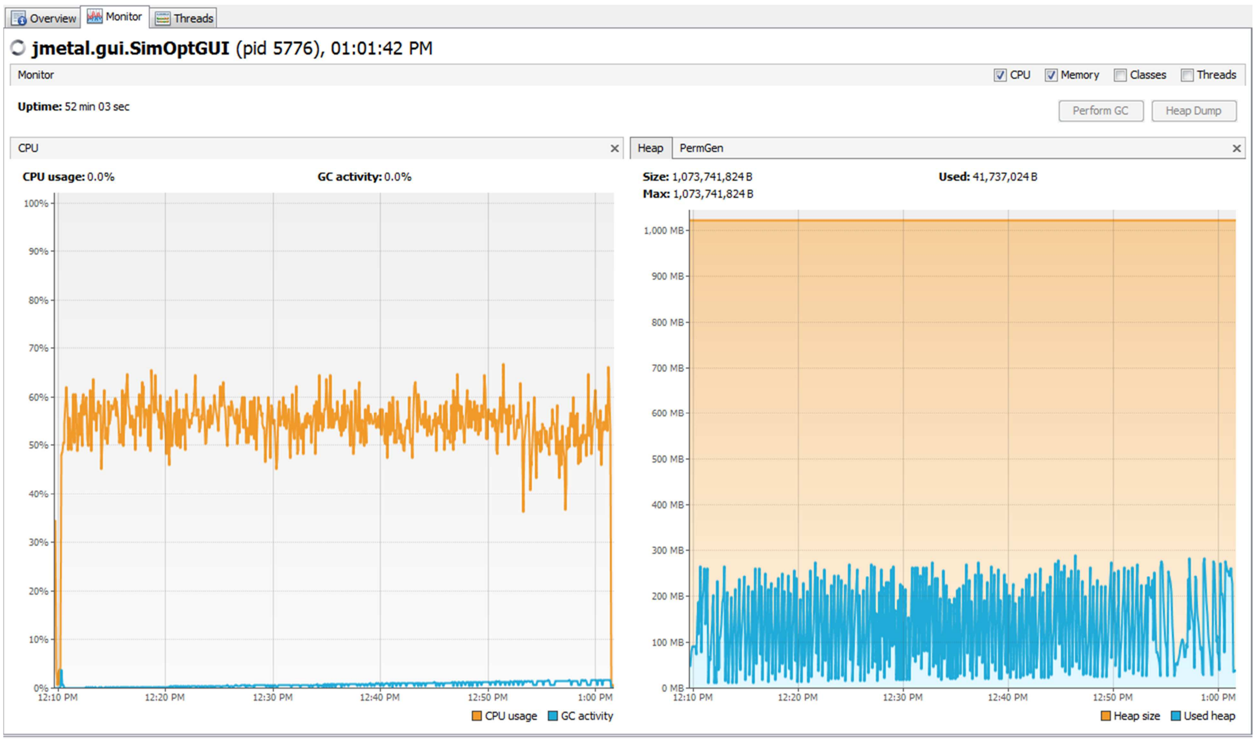
FIGURE 43 MEMORY PROFILE FOR SUPPLY CHAIN PROBLEM WITH MEMORY LEAK

During our further investigation it was also found that the simulation model for FMS scheduling problem contained memory leak (appendix G). Therefore, this memory leak problem obviously causes SIMEON to have a limited number of evaluations/experiments which is not desirable to solve large scale and complex simulation-based optimization problems.

Fortunately, further effort to figure out the source of the memory leak was fruitful. Despite of the innumerable possibilities that lead to the introduction of memory leak, we managed to find the major source of leakage. In the case of the supply chain problem, the leak is caused by objects with strong references[6] that are stored inside `Map` or similar `Collections` objects. This is because the garbage collector is not allowed to reclaim the memory used by objects that are stored compositely such as in a `Map`. A solution that we employed was to ensure that all the `Map` and similar `Collections` objects are

[6]A strong reference is no other than normal reference in Java e.g. String, Integer, Real, etc.

cleared whenever a new simulation run is executed. This resulted in relatively stable memory consumption over time as shown in Figure 44. Finally, this solution also allows SIMEON to perform the maximum number of evaluations (25000) without out of memory error.



jmetal.gui.SimOptGUI (pid 5776), 01:01:42 PM

Uptime: 52 min 03 sec

CPU usage: 0.0% GC activity: 0.0%

Size: 1,073,741,824 B Used: 41,737,024 B
Max: 1,073,741,824 B

FIGURE 44 MEMORY PROFILE FOR SUPPLY CHAIN PROBLEM WITHOUT MEMORY LEAK

5.1.4 SIMEON Requirement and Recommendations

Taking lesson from the memory leak problem encountered, an important requirement can be formulated for the simulation engine used with SIMEON:

The components of the simulation engine (particularly the simulation model) should be developed in such a way memory leak can be avoided for large number of experiments.

To comply with this requirement we also present several recommendations that are useful in implementing such simulation engine:

1. Limit or avoid using library components that are known to contain memory leak (e.g. `JFreeChart`). In case the documentation of the exact source of the memory leak is available then the developer may make use this information to fix the problem (e.g. by clearing up the objects manually when no longer needed).
2. Always implement a method to clear the content of Map and similar Collections objects. That method has to be invoked whenever the content of those objects is no longer needed.
3. Use weak references (e.g. `WeakHashMap`) for objects that are stored inside a Map or similar Collections objects. Objects that are declared with weak references are eligible to be garbage collected whenever the referent to those objects have phased out. Furthermore, it is also advised to use reference queues in combination with weak references to prune the data structure (if any) of the objects that use weak references.
4. Use soft references as an alternative to weak references. Objects that are declared with soft references are guaranteed to be garbage collected before the memory is running out. However, this takes place only when the memory begins to run out.

5. If there are objects that are used to store data structures (e.g. `Context`), ensure that whenever those data are no longer needed, the content of those objects are cleared or not referred to.
6. Try to minimize the declaration of static objects as they persist in the memory and deter garbage collection.
7. Comply with good code implementation that avoids memory leak such as the ones suggested in:https://www.securecoding.cert.org/confluence/display/java/MSC06-J.+Avoid+memory+leaks

Furthermore, in case memory leak still occurs and the source is hard to trace, we recommend using VisualVM tool that can be downloaded on https://visualvm.dev.java.net/. This tool is very useful in providing support for a Java developer to find the source of memory leak in a program. It has a feature to print all the objects that occupy the heap size within a running program together with their detailed information. The developer can also perform a profound investigation to the objects that are suspected to be the source of the memory leak using the feature "show nearest GC root" of this tool.

Next to the recommendations in the implementation level, there is also an ultimate way to circumvent memory leak problem from the design perspective. It is by separating the execution of optimization and the simulation engines on different Java virtual machines. This way, whenever an experiment using a simulation run is done, the relevant JVM can be shut down to completely release the memory used by the simulation engine. A communication socket has to be developed to facilitate the exchange of data between simulation and optimization engines and an intermediary data file can be used as means of communication. Despite the fact that this will completely safeguard the program from out of memory error, there is a drawback in the efficiency of the execution in comparison to the current SIMEON implementation. This is because shutting down and turning on JVM takes relatively longer time than invoking a method that constructs a simulation model.

5.2 SIMEON Evaluation

After the operational test phase is finished, we progress to the evaluation phase where interviews with simulation experts at Accenture are conducted. These experts have been working in various consultancy projects that make use simulation models (both DSOL-based models and models of simulation packages) as a tool to help their clients deal with decision making problems. The two interviewees are: Mr. Ivo Wenzler and Mr. Rutger Deenen. The interviews are done on 24 September 2010 and each of them takes around 1 hour. All interviews are started with a brief explanation of the simulation-based optimization technique and followed by a short demonstration of SIMEON execution to solve the supply chain management problem as presented in section 5.1.2.2.

The ultimate aim of the interviews is to assess the added value of SIMEON to real-world decision making process involving multiple objectives and to assess the possible utilization and contribution of SIMEON in the consultancy services that make use simulation and optimization techniques. Both assessments, together with the result of the operational test phase contribute to the answer for research question number 4 and 5 respectively. Following are the findings from the interview sessions that are structured based on the relevant research questions:

- **Research question no 4:**

4. To what extent this framework can contribute to the decision-making process where there exist multiple and possibly conflicting objectives?

According to Mr. Wenzler, SIMEON is able to provide following added values in the real-world decision making process:

1. Saving time to calculate the non-dominated solutions/ optimal trade-offs for the modeled problem as enumeration of all possible alternatives would take huge amount of time.
2. Narrowing down the possible solutions that should be investigated further for the decision making process. This is especially valuable to deal with complex problems which often produce unpredictable dynamic behavior. Experimenting with the decision variables manually is considered tedious and tends to be unfruitful. In such problems it is admittedly hard to figure out the relationship between each decision variable and the resulting performance of the system. Not to mention to figure out the correlation between all decision-variables. Therefore, a powerful optimization technique will add value to support Accenture in proposing solutions to the client.

Similar view is also shared by Mr. Deenen. He considers that using SIMEON, the best alternatives to solve multi-objective optimization problems can be calculated quickly and objectively. These alternatives can then be proposed to the client for further decision making process.

However, Mr. Wenzler and Mr. Deenen also explain that to finally arrive at a decision, there are a couple of important further steps:

1. Interpreting the non-dominated solutions to executable solutions and consequences so that the solutions are clear to the client.
2. Involving the clients in the further process of decision making. This process has to be interactive (through a workshop, clinic, etc.)and fostering the trust of the client to the method used. This way, the client can learn about the simulation model and together with the experts, they can devise scenarios that would like to be tested.

Finally, by concluding from the interview results above and together with the operational test phase we can now provide the answer for the fourth research question:

We have demonstrated, through the test-cases, the way SIMEON can contribute to the decision making process where multiple-objectives are involved. Its main function is to present the non-dominated solutions or the optimal trade-offs of the modeled problem in a relatively quick way such that the decision maker(s) could use this information to make further decision. It is noteworthy that the implementation of NSGAII as the optimizer allows the user to perform multi-objectives optimization without specifying additional information such as preference structure in regards to the objectives to be optimized. This is a more desirable feature as opposed to the inefficient classical approach that uses weight vector in its calculation. More importantly, SIMEON has a significant added value when it is used to solve complex simulation-based problems where manual enumeration of all the alternatives is tremendously time consuming or is impossible.

However, it is also important to note that what SIMEON presents is not an end answer for the decision making problem encountered. Further decision making steps such as interpretation of the results and selection of a unique decision still have to be done. Those steps have to involve all relevant stakeholders (including the problem owner) and may also be done in interactive and iterative manner with the support provided by the user interface of SIMEON, particularly the modeling interface.

- **Research question no 5:**

5. How can this framework improve the expertise and technological capability of the user (Accenture) in solving real world decision-making problems?

To arrive at the answer of this question several aspects related to the operational use of SIMEON were discussed during the interviews. That discussion leads to the following findings:

1. SIMEON can support Accenture in addressing the needs of client to solve decision-making problem.

According to Mr. Wenzler, there are four questions that the clients of Accenture typically have:

1. How can we be helped to understand the market/the environment on which we operate?
2. What kind of strategies do we need and are they going to be sustainable and executable so that we can be successful in that environment.
3. What kind of capabilities do we need in business so that we can execute those strategies?
4. How do we enable our resources (e.g. people) to have the right knowledge, right skills and the right behavior to use these capabilities so that we can be successful in the market?

Given questions above, Mr. Wenzler sees that simulation and optimization can be used as tools to facilitate learning and the creation of knowledge within the client such that they can eventually answer the first three questions above. However, Mr. Wenzler also indicated that optimization may not be applicable to answer the forth question. This is because Accenture normally uses training session that does not use computer-based simulation to help the client answering this question.

Furthermore, in the context of assessing SIMEON, Mr.Wenzler stated that SIMEON can give Accenture a confidence in formulating solutions as they are based on the fact on the understanding of the model. A similar answer is also given by Mr.Deenen who stated that Accenture can have more confidence to present the solutions of the problem modeled and their consequences to the clients.

Finally, as a remark Mr. Wenzler also stated that simulation and optimization should have been integrated as one package of service whenever optimization satisfies the objective of the consultancy project. This is done because what is offered to clients is not simulation or optimization services, yet rather approach to concrete solutions, support to arrive at those solutions and the solutions themselves.

2. There are operational requirements that have to be met in order to successfully use SIMEON

Both Mr. Wenzler and Mr.Deenen agreed that it is very important that the client has the confidence that the model is not a black box model and it actually does what happen in reality. Furthermore, according to Mr. Deenen, SIMEON can't be used before the client has the trust towards the simulation model they are to use. They have to see the simulation model as a representation of real system which structure and underlying assumptions are understandable. If the client does not have this trust then optimization study will not contribute anything meaningful. In case the clients have not had this understanding experts are needed to explain how the simulation model works and to interpret the result of the simulation and optimization studies.

From the project organization aspect, it is also found thatthere has to be a strong collaboration between the person who conceptualizes the real-system and the person who performs the implementation for the simulation and optimization model. Both of them should have the common understanding on the problem modeled and be able to communicate effectively with the relevant domain knowledge (such as simulation and optimization) throughout the project execution.

Finally, by concluding from the interview results above and together with the operational test phase we can now provide the answer for the fifth research question:

It is now clear that SIMEON is able to improve the expertise and technological capability of Accenture to solve real-world decision making problem. The main reason for this being SIMEON's capability to facilitate learning and the creation of knowledge. This way, the client of Accenture could address their needs which are no other than making the best decisions for their perceived problems. Another reason would be that SIMEON is considered to be useful in improving the confidence of Accenture to propose solutions that can be considered further by the clients.

However, there is a prerequisite for SIMEON to be used effectively which is the trust of the clients in the simulation model used to represent the problem at hand. SIMEON can only have efficacy if the client has trusted the ability of the simulation model in providing the answers for what-if analyses (through the different scenarios tested). Next, SIMEON can be used to suggest the alternatives to answer how question which would guide the decision maker(s) to arrive at the desired circumstances.

5.3 Conclusion

This chapter has presented the way the last systems engineering life-cycle phase has been applied to test and evaluate SIMEON. In the operational test phase, the performance limitation of SIMEON is revealed during its application to the test cases. To mitigate and whenever possible avoid this limitation, several requirements and recommendations are formulated. Most of these recommendations have been tested and proven to enable SIMEON achieving the desired performance.

More importantly the fourth and the fifth research questions are answered within the evaluation phase. By taking into account the results of the test and evaluation phases, we can conclude that SIMEON has been able to contribute to the real-world decision making process where multiple objectives are involved. This is done by presenting the decision makers the non-dominated solutions in a time efficient fashion. Another conclusion is that SIMEON is also able to improve the expertise and technical capability of Accenture by helping the client to learn and creating the knowledge required by them. This is however, has to be preceded by the trust of the client to the simulation model used.

6. Conclusions, Recommendations, and Reflections

Finally, as an end of this research paper, this chapter presents the conclusions of this research, recommendations and reflection on the various theories and methodologies used to perform this research. While this thesis paper has exposed argumentations to answer the research questions throughout the content of the previous chapters, the main research question is answered in this chapter as the main conclusion. Furthermore, the recommendations presented here are divided into further research recommendations and those that are relevant for the future development of SIMEON. Last but not least, reflection on the theories and methodologies are also presented to contribute to the development of the relevant theory and better research methodology.

6.1 Conclusions

To answer the main research questions and conclude the fulfillment of the aforementioned objective we first recall the sub-research questions and their answers presented in the previous chapters:

1. How can metaheuristic techniques be integrated into the framework of modeling and simulation that is currently used to solve real-world problems?

This research question stems from both knowledge and implementation gap that are needed to integrate metaheuristic technique into the existing Zeigler's framework of modeling and simulation. Therefore to provide an answer for this research question, exploration of the designs on both conceptual and implementation levels are performed.

In the design on the conceptual level, a metaheuristic technique (in this case NSGA-II) can be integrated into the modeling and simulation framework by connecting its routines into the model and the experimental frame. In this design, the values of the decision variables inside treatments given to the simulation model are automatically determined by the NSGAII and the transducer or acceptor feeds the algorithm with the values of the outcome measures. It is also clear that a number of experiments with different treatments are needed to perform the optimization. Based on this result, we also argue that other state-of-the-art metaheuristic algorithms can be integrated in a similar way.

A noticeable design in the experimentation process using SIMEON is that NSGA-II routines create a closed-loop between simulation and optimization and therefore, replaces the knowledge of the modeler decision variable values. This way, the tedious process of finding the best alternatives / the best trade-offs can be automated by using the intelligence of SIMEON. Furthermore, in the case where the optimization problem is highly complicated (i.e. a non-linear problem with many variables and local optima), the framework also theoretically provides significantly more effective way to find the solution in comparison to the conventional approaches such as brute-force and trial and error.

In the implementation level, we present the system architecture and the detailed designs of both the simulation and optimization components that embody the conceptual design of SIMEON. Next, these detailed designs of the components altogether with the design of the interaction between those components have substantiated the implementation of the integration suggested in the conceptual design. Therefore, both the conceptual and the detailed designs have provided a holistic answer on the question above.

2. How should the genetic chromosome and operators be designed such that the proposed framework has the capability to support the optimization of quantitative and qualitative variables which enables it to solve continuous and discrete multi-objective optimization problems?

We present the designs of four different genetic chromosomes and six genetic operators (in chapter 4.2.2) that comply to the design principles advocated by Palmer (1994). These four different representations and six genetic operators are efficiently designed to deal with multi-objective optimization problems that involve quantitative and qualitative decision variables. Depending on the types of the decision variables that are to be optimized, a proper representation and operators can then be chosen.

3. What does the design of the interface components that are able to facilitate the necessary changes in the simulation model look like?

We present the design of SIMEON's user interface (UI) (in chapter 4.2.3) together with its detailed components that are based on the multidimensionality and usability requirements. While the designs of the genetic chromosomes and operators have given SIMEON the ability to suggest solutions involving parameter and structural changes in the simulation model, the UI of SIMEON is designed to exploit this capability. First, it helps the user in specifying the decision variables and the optimization problem using the outcome measures of interest from the simulation model. Second, and most importantly it facilitates the user in selecting the most appropriate representation and genetic operators such that any structural or parameter changes are possible. This way the UI could facilitate the necessary changes in the simulation model. The key design that contributes to this feature is the factory method pattern that is implemented to facilitate the instantiation of the algorithm and its operators.

4. To what extent this framework can contribute to the decision-making process where there exist multiple and possibly conflicting objectives?

We have demonstrated, through the test-cases (in chapter 5.1), the way SIMEON can contribute to the decision making process where multiple-objectives are involved. Its main function is to present the non-dominated solutions or the optimal trade-offs of the modeled problem in a relatively quick way such that the decision maker(s) could use this information to make further decision. It is noteworthy that the implementation of NSGAII as the optimizer allows the user to perform multi-objectives optimization without specifying additional information such as preference structure in regards to the objectives to be optimized. This is a more desirable feature as opposed to the inefficient classical approach that uses weight vector in its calculation. More importantly, SIMEON has a significant added value when it is used to solve complex simulation-based problems where manual enumeration of all the alternatives is tremendously time consuming or is impossible.

However, it is also important to note that what SIMEON presents is not an end answer for the decision making problem encountered. Further decision making steps such as interpretation of the results and selection of a unique decision still have to be done. Those steps have to involve all relevant stakeholders (including the problem owner) and may also be done in interactive and iterative manner with the support provided by the user interface of SIMEON, particularly the modeling interface.

5. How can this framework improve the expertise and technological capability of the user (Accenture) in solving real world decision-making problems?

Based on the results of the operational test and the interviews (chapter 5), it becomes clear that SIMEON is able to improve the expertise and technological capability of Accenture to solve real-world decision making problem. The main reason for this being SIMEON's capability to facilitate learning and the creation of knowledge. This way, the clients of Accenture could address their needs which are no other than making the best decisions for their perceived problems. Another reason would be that SIMEON is considered to be useful in improving the confidence of Accenture to propose solutions that can be considered further by the clients.

However, there is a prerequisite for SIMEON to be used effectively which is the trust of the clients in the simulation model used to represent the problem at hand. SIMEON can only have a high level efficacy if the client has trusted the ability of the simulation model in providing the answers for what-if analyses (through the different scenarios tested). Next, SIMEON can be used to suggest the alternatives to answer how question which would guide the decision maker(s) to arrive at the desired circumstances.

Finally, we come back to the main research question that motivates the objective of this research:

What does the design of a simulation-based multi-objective optimization framework that increases the effectiveness of simulation-based decision support tools look like?

While the objective of this research being:

To design a framework for simulation based-multi objective optimization that satisfies generality, efficiency, multi-dimensionality and usability features to the level where the requirements from Accenture are met by appropriately making reasonable trade-offs between those features.

We present the design and implementation of a simulation-based multi-objective evolutionary optimization (SIMEON) framework as the answer for the main research question. We argue that SIMEON has the ability to increase the effectiveness of simulation-based decision support tools by:

1. Designing its concept based on a solid theoretical foundations in the field of modeling, simulation and optimization.
2. Developing its physical components based on good software engineering practice.
3. Delivering the whole product based on systems engineering practice.

Both the conceptual and detailed designs of SIMEON have substantiated that an optimization technique can be combined with a simulation technique in a theoretically and technically sound framework, to provide a powerful support in solving decision making problems, particularly those which are represented as multiple-objective optimization problems. In the context of increasing effectiveness of the simulation-based decision support tools, we also view that the role of SIMEON is to address the questions related to the patterns of the design strategies in such a way certain predefined goals can be achieved. We argue that this is an important role that can be well integrated into the process of accomplishing the objectives of modeling and designing a system in which simulation technique is often used. This is substantiated by the possible contribution of SIMEON to:

1. Provide the best alternatives for each different future possibilities and uncertainties which can be analyzed using traditional what-if simulation analyses. Thus, optimization can serve

as an additional problem solving step after different possibilities regarding thefuture circumstances are explored.

2. Provide the best design alternatives given specific set of controllable and uncontrollable elements of a system (as in the common simulation-based optimization study).

However, it is also important to note that to effectively use simulation-based optimization method, the help of the experts, this far, is always necessary. This is because simulation and optimization are methods that demand a specialized knowledge of the user.

Finally, looking at how all the technical requirements have been satisfied and translated into the physical design of SIMEON, we can also conclude that SIMEON has accomplished the objective of this research. The key design feature that we use to satisfy generality requirement is the separation of concerns between the optimization algorithm and the problem. While efficiency and multi-dimensionality are mostly satisfied by the design of the algorithm, usability is satisfied by the design of the user interface.

6.2 Recommendations

During the execution of this research project there are many findings, both from the theoretical and implementation aspects that may lead to the further development of SIMEON or similar simulation-based optimization method. Due to the limitation set in scope and time, all those findings are not implemented. They are however presented here as recommendations for the further development of SIMEON, further research, and the use of SIMEON in a multi-actor setting.

6.2.1 Recommendations for further development of SIMEON

The recommendations for the further developments are categorized based on the main constituents of the presentation layer and problem solving layer.

1. SIMEON user interface (SIMEON UI):

Both Mr. Wenzler and Mr.Deenen agreed that the current design of SIMEON UI has no problem to be used by the expert users. However, there are several recommendations given by both interviewees to improve its usability for non-expert users (Interview, September 24, 2010):

a) To design the user interface to be more attractive and using simpler non-mathematical language that is understandable by a manager.
b) To make all the controllable settings to be self-explanatory and not difficult to explain to the clients.
c) To provide an explicit explanation regarding the different settings which are adjustable in SIMEON control panel.
d) To provide a user manual to operate all the functionalities of SIMEON. The user manual should contain step-by-step instruction to use the framework.
e) To test the UI with the potential users of the framework to make some final adjustments on the design such that it becomes more user friendly.

Furthermore, several recommendations to further develop the UI can be derived by combining the usability requirements elaborated in section 3.1.1 and the abovementioned suggestions. The purpose of this is to further facilitate the non-expert users without sacrificing the performance of SIMEON.

a) To make a separation on the interfaces based on the types of the users. This can be done by hiding some of the detailed configurations (e.g. crossover probability, mutation probability), which are not crucial for the performance of the algorithms such that the non-expert users don't need to explicitly configure them (default values can be suggested automatically).

b) To integrate the language systems relevant to take-in all the messages from the users in one application. This would ease the users to specify the inputs and parameters required by SIMEON.

2. **Simulation engine**

a) Implementing a standardized experimental frame proposed by Zeigler (2000).

One of the most valuable improvements that can be made in DSOL is to have a clear separation between the model and the experimental frame that complies with the theory of modeling and simulation advocated by Zeigler (2000). Not only would this increase the inter-operability of the models using the same experimental frame, coupling the outcome measures of interest with external optimization service would also be simpler due to the clear separation of the statistical objects from the model.

b) Modifying the way the outcome measures of interest (statistical objects) are built and stored in DSOL.

In the current implementation of DSOL, the way the statistical objects/ outcomes of the simulation is stored and used through `Context` object gives difficulty for the modeling interface of SIMEON to retrieve those outcomes before the simulation is executed. This is because the content of `Context` objects is only instantiated when the simulation has been executed. A more suitable implementation for simulation-based optimization application is to allow the instantiation of those statistical objects of interest before the simulation is executed (for example when the model is instantiated). This way, the modeling interface could present all the relevant statistical objects to the user without having to wait to the end of the first execution of the simulation. This is also a form of implementation that accords the step of modeling optimization problem.

c) Implementing alternative ways to store the results of each simulation replication

During the operational test phase, `Context` object has been suspected to be one of the sources that causes memory leak problem. This is due to the results of each simulation replication are stored there and are not allowed to be garbage collected. It is recommended to store the results of each simulation replication in external objects which don't cause gradual memory occupation.

d) Implementing other open source simulation engines to improve the generality of SIMEON

SIMEON is currently developed for DSOL-based simulation models as a consequence of using DSOL to develop the simulation engine. From the conceptual design and system architecture of SIMEON it is possible to implement other simulation engines than DSOL. This is also supported by the NSGAII implementation in SIMEON that makes use a generic design pattern such as publish-subscribe mechanism. Hence, it would be of an added value for the generality feature to implement other open-source object-oriented simulation engines within SIMEON.

3. **Optimization engine**

a) Implementing a functionality to stop the iteration of the algorithm based on the rate of the improvement in the quality of the solutions

In the current implementation, this functionality has not been implemented yet. The user has to use his estimation and experience to specify the maximum number of evaluations. The current feature that helps the user to estimate the number of iterations needed is the "hold" check box. This feature allows the user to compare solutions that are produced by different number of evaluations and hence providing the user an indirect indication on the estimated iteration number needed to arrive at the desired solution quality. A valuable feature would be to automate the stopping process of the algorithm by monitoring the rate of the improvement/difference in the solution quality in each evaluation. Once the rate/difference is lower than a predefined value, the algorithm can be stopped. However, there is a drawback that has to be further investigated: it would add computational time of the algorithm.

b) Implementing the functionality to stop and continue the execution of SIMEON arbitrarily

In the current implementation, SIMEON doesn't have the functionality to allow the user to stop its execution, save all the non-dominated solutions found that far and continue the execution whenever desirable. The implementation of such functionality is deemed to be of a great value to solve highly complex problem that takes huge amount of memory resource and time. With this feature, the user can have more control and flexibility in running SIMEON.

6.2.2 Recommendations for further research

1. **The implementation of the other state-of-the-art evolutionary-based or metaheuristic algorithms**

The implementation of SimOptNSGAII has provided an example to bridge the knowledge gap in integrating simulation and optimization as a combined approach. However, there are still many advanced evolutionary-based algorithms that can be implemented to enrich SIMEON and to provide alternatives in solving simulation based optimization problems.

2. **The separation of optimization and simulation engines in different Java Virtual Machines (JVMs)**

Our investigation upon memory leak problem suggests that to ultimately circumvent this problem, a separation between optimization and simulation engines in different JVMs can be done. However, more research is needed to implement this separation with a good software engineering practice that all the possible advantages and weaknesses could be carefully taken into account and transformed into a product that satisfies user requirements.

3. **The implementation of SIMEON using DEVS formalism**

Up to this moment there has not been many research aimed to integrate optimization technique into simulation models that are implemented in DEVS formalism. Both theoretical and implementation knowledge contributed to this research area would be very valuable for the development of standardized simulation-based optimization approach in the scientific community.

4. **The integration of factor screening technique to the SIMEON**

Simulation-based optimization approach that makes use metaheuristic has always been criticized for the resource that needed to perform the calculation. A way to improve the efficiency of the method is to integrate statistical factor screening techniques. These techniques can be used to rule out decision variables that do not have significant influence on the objective functions. This way, the search space of the optimization can be reduced. However, there is still an implementation gap as to how these techniques can be implemented efficiently.

6.2.3 Recommendations in using SIMEON in multi-actor setting

SIMEON has been developed to specifically address multi-objective optimization problems. As a matter of fact, this type of problems often involves multiple actors who have different objectives and interdependencies (H. Bruijn, de & Heuvelhof, 2008). Though the ultimate role of SIMEON is fulfilled when it presents the non-dominated solutions to the decision maker(s), there are still challenges in the decision making process that have to be addressed to finally arrive at a decision. One of the most relevant challenges in a multi actor setting is strategic behavior. It is ubiquitous and normally exhibited by the actors. Furthermore, it is reflexive and often starts from an information asymmetry. There are many negative consequences of strategic behavior which make the mitigation of this behavior of a great importance. Some of the consequences are it makes the process become hampered, chaotic, erratic, and unwieldy (H. Bruijn, de, Heuvelhof, & Veld, 2002).

Bruijn (2002) has formulated four design principles for a fair process that can be used to cope with this behavior. The theory argues that a successful process should consist of four core elements, namely openness, protection of core values, progress and content. Firstly, a fair process should have openness. This is done by inviting the relevant stakeholders to participate in the decision making process. Secondly, all the interests of the decision makers have to be taken into account in the decision making process to protect their values. In this case, SIMEON can provide a help in incorporating all the conflicting objectives in an optimization study. Thirdly, openness and protection of core values could lead to a sluggish process; therefore the process should have incentives which create progress. At this stage, the non-dominated solutions found by SIMEON can give the involved stakeholder a possibility of gain. Furthermore, it also helps to reduce the information asymmetry between the stakeholders in terms of their gains and to allow them to learn to make compromises. Lastly, it is important to maintain the focus on the content of the problem faced and for this purpose normally experts are used to provide some help.

Another field that can be a valuable compliment to help the decision making process is Nash bargaining theory, which is a subset of cooperative game theory. With the axioms presented by John Nash in his theory, a unique fair solution can be calculated for a bargaining problem that involves two objectives in a convex criterion space. According to his axiom, the unique solution will always be a Pareto optimal solution. Therefore, the non-dominated solutions given by SIMEON can be used as the starting point for this analysis. This is very helpful to solve bargaining problem in multi-actor setting especially when the utility functions of the actors are not known but the acceptance towards the non-dominated solutions is present.

6.3 Reflections

1. **Reflection on the systems engineering methodology**

In this research, the waterfall model of Sage & Armstrong (2000) was used to deliver SIMEON as a product of systems engineering. This model was seen to be appropriate as this is a model that originates from the process of delivering a software system (Sage & Armstrong, 2000). One of the main contributions of this design approach is in the process of translating the technical requirements into a functional and trustworthy engineering product and also in process of structuring the report this thesis project. This is especially true, when we look back on the iterative improvements that took place between the life-cyle phases of this approach. During those iterations, design challenges are solved through major and minor changes in both conceptual and detailed designs and this is also what helps the product to achieve its perfection. Another valuable feature of this framework is the separation of phases that allows loose coupling between the conceptual design and the detailed implementation. This allows SIMEON to have distinct contributions to both theory and technical knowledge in simulation-based optimization field as the first requires a high degree of generality to be valuable while the latter requires a sufficient specificity to allow a good knowledge transfer.

However, learning from the problems faced in the development phase, there is an improvement that can be made in the design method used. Firstly, it is possible and beneficial to start the development phase as early as possible given the conceptual designs for the engines are finished. As the development progresses, the conceptual design can be adjusted based on the latest challenges or improvements found in the development works and vice versa. This process, hence, makes use effectively the feedback loop between the definition and development phases. Next, it is important to make sure that the applications/ software engines are developed and tested thoroughly in the early phase, before the development of the other parts of the framework is done. Test cases that differ in scale have to be used to find out as much as possible the bugs and the potential deficiencies for which fixing effort can be immediately done afterwards. A design method that fits this approach is the incremental method (Sage & Armstrong, 2000). In this method, the kernel of software has to be developed thoroughly in the first iteration before incremental additions of the other sub-systems can be done. This method can be fitted in to the development phase of SIMEON such that experiments on test cases don't need to wait for the completion of the user interface implementation.

2. Reflection on the fundamental theory of genetic algorithms

A sound and strong criticism has recently been addressed to the fundamental theory of genetic algorithm which is known as building block hypothesis (BBH) by Burjorjee (2008). The main conclusion of this criticism is that BBH could not provide a solid explanation on the adaptive capability of genetic algorithm to find the solutions for complex problems. This is because of fundamental problems that lie in the strong assumptions used in BBH. These assumptions are:

1. There exist a large number of basic building blocks[4]
2. There is a high probability that there are many synergistic intersections between different building blocks which would lead to a better solution for optimization problems.

The problem with these assumptions is that they are too imprecise to be falsified and it is impossible to falsify those assumptions using experiments. One has to make an inventory of all entities in this universe to perform the necessary experiments (Burjorjee, 2008).

Nevertheless, despite of this criticism, we are also faced with many empirical evidences that proof the remarkable adaption capacity in many of evolutionary algorithm (EA) designs. This obviously can't discourage us not to apply EA in many real-world problems/research fields. As a matter of fact, the loose linkage between the analytical proof of BBH and its implementation in the design of EAs is what essentially underlies many research motivations in many fields. Researches involving EA consequently always validates the efficacy of their EA design through experiments.

We see now that even after 35 years of the extensive use and research on genetic algorithms, scientific world still falls short on the capability to understand perfectly the reason why this nature-inspired algorithm works well. This obviously leads to an interesting research opportunity, but nevertheless, it is also appropriate to conclude that BBH is a result of extrapolation of an ostensible process that causes human innovation. This situation consequently requires us to believe, consciously or not, these strong, almost-seemed-to-be-magical assumptions until humans are better inspired to provide a more solid scientific explanation.

3. Reflection on the SIMEON testing and evaluation phase

First of all, we regard testing and evaluation phase a form of validation phase for SIMEON. As such, we put a considerable concern to perform the test both for the performance and usability of SIMEON. While the test on the performance of SIMEON was completed successfully, the test on the usability of SIMEON is proven to be quite time consuming task for a single developer. This is because to reach a good usability, all the functionalities of SIMEON have to be tested and followed by the repair to all the known bugs. It is therefore beneficial for the development of a prototype like SIMEON to define a clear scope on the test that pertains the usability aspect.

Another finding regarding usability is found during the evaluation phase of SIMEON. During the development of SIMEON, an effort has been made to make SIMEON as a product that has a high usability for the broader scope of the users, the non-expert users. However, as usability is strongly dependent on the user interface, we found it difficult to provide all the visualization features needed to make SIMEON as usable as possible these users. To help clarifying the usability of SIMEON, we formulated the definitions of expert and non-expert users by setting the knowledge requirements that have to be satisfied by both types of users. Though this has been proven to be useful, we found that it is also important to regard SIMEON as a technology that has to be studied by any interested users in order to harness it appropriately and effectively.

4. Reflection on the Zeigler's modeling and simulation framework

Although Zeigler's framework has been a solid theoretical framework to perform modeling and simulation studies, there are refinements that can be made to update this framework based on the developments in the modeling and simulation world. One clear deficiency in Zeigler's framework is it lacks of paradigm that depicts the interaction between the experimenter, the experiment and the framework itself. This gives a less clear concept on how different experiments can be carried out, given the detailed components of the framework, particularly the experimental frame. Thus, enriching the framework with additional notions to support the necessary paradigm would be beneficial.

Another notion that can be improved is the concept of experimental frame that is claimed to be the embodiment of objectives that underlie a modeling and simulation project (Zeigler, et al., 2000). We argue that a refinement is needed in the definitions of the components within experimental frame (i.e. generator, transducer and acceptor) as they cannot consistently translate the generality aspect contained in the definition. A clear example of this would be the difficulty of adapting the original experimental frame components to define a simulation-based optimization method. The research of Hojun (2008) has exemplified the adjustments or additional notions that are needed to adapt the experimental frame to perform simulation-based optimization study (Hojun, et al., 2008).

However, there is also a deficiency in the design of the experimental frame proposed by Hojun (2008). It is tightly coupled to the problem dealt as there is no separation of concerns between the optimizer and the problem. This makes the optimizer can only be used to optimize a specific problem. The design of SIMEON framework has addressed this problem.

Furthermore, though the framework is originally built for Discrete Event System Specification (DEVS) models, it is beneficial to maintain the generality of the conceptual design such that it is applicable for different simulation formalisms. This way, many of other simulation languages that aren't developed based on DEVS formalism could still use the framework, promoting a unification platform for different simulation practitioners.

5. Comparison of SIMEON to other optimizers in COTS presented in Table 1

SIMEON is partly developed based on the researches concerning the development of simulation-based optimization methods in many COTS simulation packages. Therefore, it is a good practice to compare the features of SIMEON with various popular optimizers in those packages. We use OptQuest which is implemented in Arena 13.0 and ISSOP as the comparators to reveal the strong features of SIMEON as well as its deficiencies for which future developments can be performed.

- **Generality**

From the generality feature, it is clear that the optimization engine within SIMEON, OptQuest and ISSOP has the capability to solve problems from different domains. The limiting factor for the generality feature would be the simulation language with which these optimizers are implemented. In the current implementation, SIMEON has a merit to be implemented with DSOL which supports multi-formalism simulation modeling. This means SIMEON would be able to be used to optimize both discrete and continuous simulation models. This is a weak point for the other optimizers since OptQuest and ISSOP are mostly implemented in simulation packages that are based on discrete event formalism.

Another positive feature from SIMEON is that although it is currently implemented for DSOL-based models, it is possible to adapt SIMEON engine to other simulation packages, giving it a high degree of flexibility comparable to OptQuest and Arena.

- **Efficiency**

A scientific way to compare the efficiency between SIMEON, OptQuest and ISSOP is by performing experiments involving case studies. Unfortunately this is not possible due to limitation in time and license availability. Technically, the efficiency of SIMEON is comparable to OptQuest with engine v 6.5 and ISSOP v 2.0 as both engines start to employ algorithms that are able to produce non-dominated solutions in a single run. However, ISSOP in this case, seems to have a more powerful method to solve optimization problem. It has a collection of different algorithms that work in combination with neural network technique. During the optimization iterations, those algorithms are claimed to work simultaneously to find the best solutions. Unfortunately, there is no source that explains how is the mechanism behind this process, making it difficult to assess the performance of ISSOP thoroughly.

However, there is a feature in SIMEON that doesn’t exist in the other optimizers. SIMEON provides different chromosome designs that can be chosen depending on the types of the decision variables faced. This gives SIMEON better ability to find non-dominated solutions as most of the other optimizers use uniform operators in their algorithm to handle different types of decision variables.

- **Multi-dimensionality**

From the multi-dimensionality criterion, SIMEON, OptQuest v 6.5 and ISSOP v 2.0 share some similar features. All of them are able to optimize:

1. MOOP with more than two objective functions,
2. Discrete, continuous, and mixed of discrete-continuous variables.

However, there is a feature in SIMEON that is not found in the other optimizers. SIMEON allows the optimization of structural alternatives which is valuable in designing a system. This is because the optimization algorithm within SIMEON is implemented in the same programming language as the simulation engine (Java). This way, SIMEON has an unparalleled flexibility to provide structural solutions which can be translated into certain system configurations in the simulation model. Hence, this allows SIMEON to explore different design alternatives involving both structural and parametric variables simultaneously.

- **Usability**

To perform the comparison for the usability criterion we use OptQuest v 6.4.1.1 that is integrated in Arena v13.0 and ISSOP demo v 1.4.1.46. Among the three optimizers OptQuest has the most mature features that are user friendly. All the required steps to perform an optimization study are facilitated in the same application interface, making it less confusing to operate. The interface used to specify the decision variables, the objective function, and the constraint functions are pretty intuitive for users who are familiar with simulation-based optimization techniques and Arena. All the variables that are relevant to model optimization problems are seamlessly connected with the relevant arena model. Furthermore, there is a clear separation between the control and response variables which helps the user to model an optimization model appropriately. It also has a reporting feature that clearly summarizes the results of the optimization study throughout the optimization iterations. The only weakness of OptQuest in Arena package v 13.0 is that it still uses classical multi-objective optimization routine. As such, it does not allow the visualization of the non-dominated solutions.

ISSOP has a considerably less user friendly interface relative to SIMEON and ISSOP. The user has to learn some additional knowledge to specify an optimization model in ISSOP. One weakness of the interface is that it doesn’t allow the user to make custom-made objective functions out of the variables defined in the simulation model. Furthermore, it also does not provide visualization for the best solutions found by the algorithms. A positive feature in ISSOP is that it allows multi-objective optimization and it provides a list of the compromised solutions similar to the non-dominated solutions.

Relative to ISSOP and OptQuest, the interface of SIMEON can be considered to be better than ISSOP but less advanced than OptQuest. Though it is relatively easy to make a multi objective optimization model using SIMEON interface, there are additional features that have to be implemented to better facilitate the non-expert users in performing optimization study. However, SIMEON interface also has some distinct features that are not found in the other optimizers. Firstly, it has a high degree of transparency in terms of the method used to perform the optimization. The user of SIMEON is provided with a possibility to choose the best representation and genetic operators in solving any optimization problems, allowing the expert users to perform the necessary experiments. Secondly, it is able to visualize effectively the non-dominated solutions for two objective functions. The information regarding the values of the decision variables and the objective functions are well integrated within the plot presented in the interface. Thirdly, its problem and algorithm combo boxes enable the user to swiftly change the algorithm to use or the problem to solve. Last but not least, it also allows a quick comparison between different Pareto frontiers that result from different configurations in the control panel of SIMEON.

APPENDIX A: EXTENDED UML CLASS DIAGRAM OF SIMULATION AND OPTIMIZATION INTERACTION

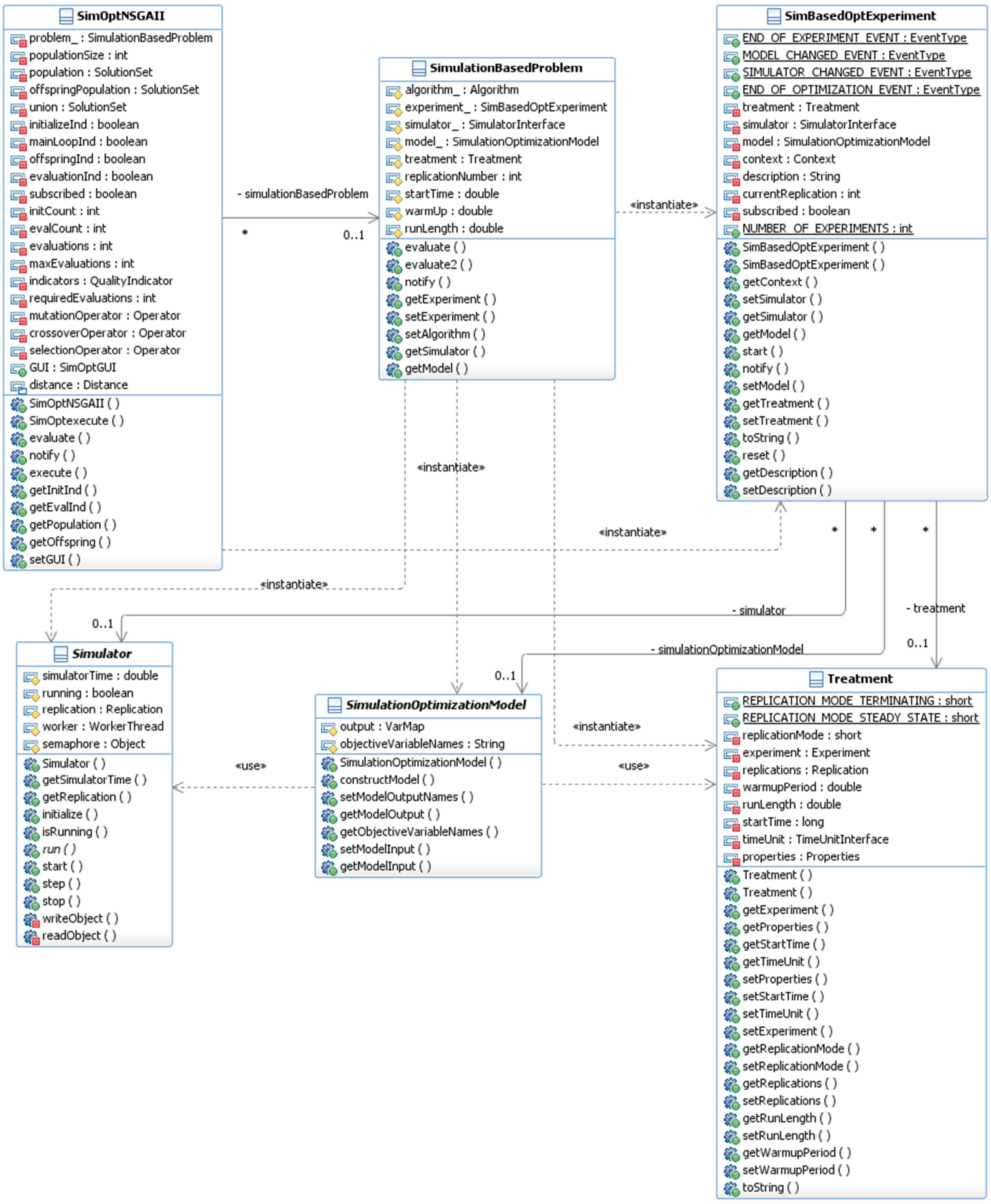


FIGURE 45 EXTENDED UML CLASS DIAGRAM OF SIMULATION AND OPTIMIZATION INTERACTION

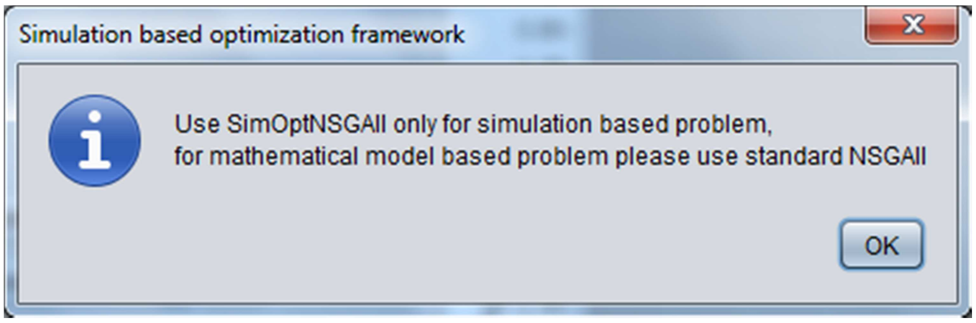
APPENDIX B: SIMULATIONOPTIMIZATIONMODEL.JAVA

```
public abstract class SimulationOptimizationModel implements
ModelInterface
{
protected VarMap output = newVarMap();
protected ArrayList<String> objectiveVariableNames = new
  ArrayList();

  public SimulationOptimizationModel()
  {
     setModelOutputNames();
  }

  publicvoidconstructModel(SimulatorInterface simulator)
  throws SimRuntimeException, RemoteException
  {
  }

  public void setModelOutputNames()
  {
  }

  public VarMap getModelOutput()
  {
     return null;
  }

  public ArrayList<String> getObjectiveVariableNames()
  {
     return this.objectiveVariableNames;
  }

  public void setModelInput()
  {
  }

  public double getModelInput()
  {
     return 0;
  }
}
```

APPENDIX C: SIMULATIONBASEDPROBLEM.JAVA

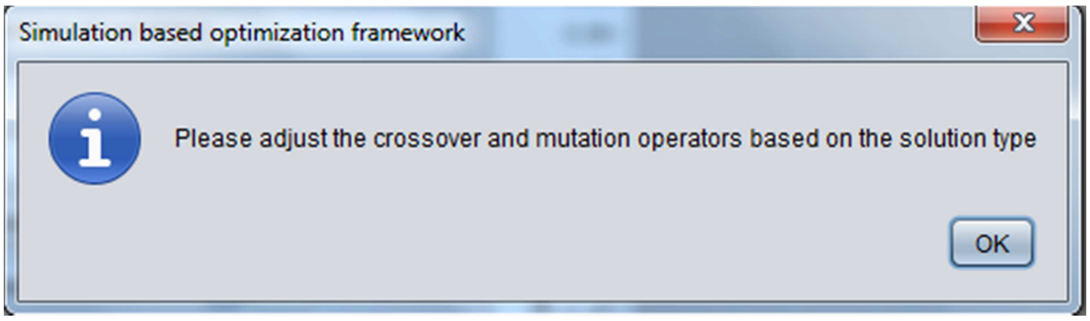
```java
public class SimulationBasedProblem extends Problem implements
Serializable, EventListenerInterface
{
     protected Algorithm algorithm_;
     protected SimBasedOptExperiment experiment_;
     protected SimulatorInterface simulator_;
     protected SimulationOptimizationModel model_;
     protected Treatment treatment;
     protected int replicationNumber;
     protected double startTime;
     protected double warmUp;
     protected double runLength;

   public void evaluate(Solution solution) throws JMException {
   }

   public void evaluate2(Solution solution, EventListenerInterface
   algorithm)
   throws JMException {
   }

   public void notify(EventInterface event) throws RemoteException {
   }

   public SimBasedOptExperiment getExperiment()
   {
      return this.experiment_;
   }

   public void setExperiment(SimBasedOptExperiment experiment)
   {
      this.experiment_= experiment;
   }

   public void setAlgorithm(Algorithm algorithm)
   {
      this.algorithm_=algorithm;
   }

   public SimulatorInterface getSimulator()
   {
      return this.simulator_;
   }

   public SimulationOptimizationModel getModel()
   {
      return this.model_;
   }
}
```

APPENDIX D: SIMEON MESSAGES


Simulation based optimization framework

Use SimOptNSGAII only for simulation based problem,
for mathematical model based problem please use standard NSGAII

FIGURE 46 SIMEON MESSAGE TO USE THE APPROPRIATE ALGORITHM TYPE


Simulation based optimization framework

Please adjust the crossover and mutation operators based on the solution type

FIGURE 47 SIMEON MESSAGE TO USE THE APPROPRIATE OPERATOR TYPE

APPENDIX E: SIMEON USE CASE SCENARIO

Use case scenario name	**Running the framework**
Use case scenario intent	**to enable the user to optimize simulation-based multi-objective optimization problem**
Version number	**1.0.0**
Use Case Scenario description	
1. Fill in all the information related to all the decision variables that are to be and not to be optimized in the Microsoft Excel template	
2. Execute the SimOptGUI.java file to activate SIMEON control panel	
3. Set all the control parameters required by both simulation and optimization engines	
4. Hit "load optimization model" button in the control panel window to activate the modelling interface	
5. Express the objective and constraint functions in a mathematical expression	
6. Hit the load button the modelling interface	
7. Hit the execute button in the GUI	
8. Analyze the results that are visualized and printed out by the framework	
Requirements and other information	
•the framework has to be given a valid simulation model with clearly defined input and output measures , this consequently makes the translation from the solution string to the decision variables of the model has to be done by the user according to the context of the problem. •the output measures to optimize have to be pre-defined by the user in the implementation level and expressed in the modelling interface •There should be minimum 2 objective functions made by the user	

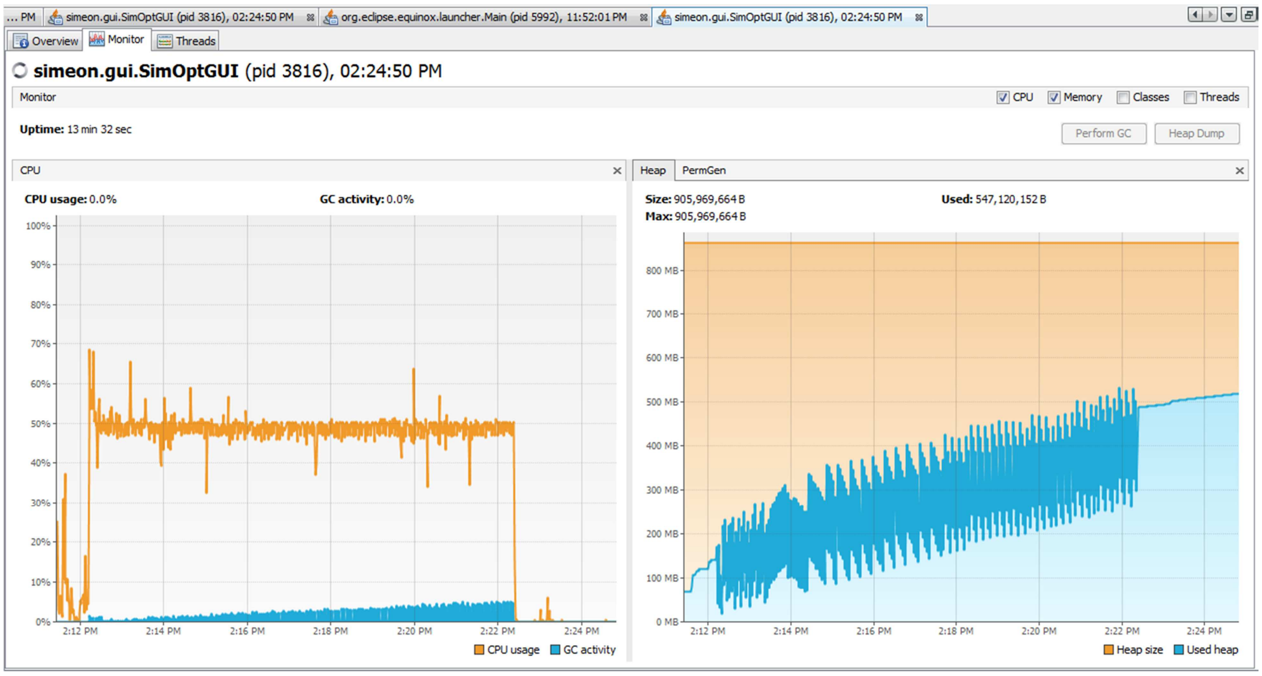
FIGURE 48 USE CASE SCENARIO TO RUN SIMEON

APPENDIX F: IMPLEMENTATION PROCEDURE TO CUSTOMIZE SIMEON FOR A SPECIFIC SIMULATION-BASED PROBLEM

There are five steps that outline the customization procedures needed to couple SIMEON with the simulation-based problem to be optimized. In this example, step 1-3 are implemented in the same class that defines the simulation-optimization model and step 4-5 are implemented in the same class that defines the simulation-based optimization problem.

1. Define the DSOL-based simulation model that is used to perform simulation-based optimization study (i.e. the class where the initial construction of the objects for simulation is executed) by extending SimulationOptimizationModel class (**Appendix H: Example of a simulation-optimization model definition in SIMEON (Supply chain optimization problem)**)
2. Pass the values of the decision variables suggested by SIMEON to the relevant objects within the mode (**Appendix I: Example of passing the decision variable values from SIMEON**)
3. Define the name of the variables using setModelOutputNames() method (**Appendix J: Example of defining the names of the decision variables in SIMEON**)
4. Define the simulation-based problem to be optimized by extending SimulationBasedProblem class (**Appendix K: Example of a simulation-based optimization problem definition in SIMEON**).
5. Implement the way the outcome measures of interest are evaluated and parsed by SIMEON within notify() method in the implementation of the class defined in step 4 (**Appendix L: Example of the objective function evaluations in SIMEON**)

APPENDIX G: MEMORY PROFILE FOR JOB SHOP PROBLEM


... PM
simeon.gui.SimOptGUI (pid 3816), 02:24:50 PM
org.eclipse.equinox.launcher.Main (pid 5992), 11:52:01 PM
simeon.gui.SimOptGUI (pid 3816), 02:24:50 PM
Overview
Monitor
Threads
Monitor
CPU
Memory
Classes
Threads
Perform GC
Heap Dump
CPU
Heap
PermGen
CPU usage
GC activity
Heap size
Used heap

simeon.gui.SimOptGUI (pid 3816), 02:24:50 PM

Uptime: 13 min 32 sec

CPU usage: 0.0% **GC activity:** 0.0%

Size: 905,969,664 B **Used:** 547,120,152 B
Max: 905,969,664 B

FIGURE 49 MEMORY PROFILE FOR JOB SHOP PROBLEM FOR 10000 EXPERIMENTS AND 3 REPLICATIONS

APPENDIX H: EXAMPLE OF A SIMULATION-OPTIMIZATION MODEL DEFINITION IN SIMEON (SUPPLY CHAIN OPTIMIZATION PROBLEM)

```
package com.accenture.simulation.supplychain.construct;

public class ConstructExcelBasedModel extends
SimulationOptimizationModel {

/** serial UID */
private static final long serialVersionUID = 1L;

/** for debugging */
private static final boolean DEBUG = true;

/** The configuration properties. */
private Properties configurationProperties = null;

/** the workbook */
private static Workbook workbook = null;

/** the pie chart */
private static CostsPieChart costsPieChart;

/** the column chart */
private static CostsStackedColumnChart costsStackedColumnChart;

public void constructModel(SimulatorInterface simulator
throws SimRuntimeException, RemoteException {
…
…
}
```

APPENDIX I: EXAMPLE OF PASSING THE DECISION VARIABLE VALUES FROM SIMEON

```java
package com.accenture.simulation.supplychain.construct;

public class ConstructExcelBasedModel extends
SimulationOptimizationModel

…

//step1 add properties object to connect the properties within the
simulator to the model
private Properties inputParameters;

…

public void constructModel(SimulatorInterface simulator
throws SimRuntimeException, RemoteException {
…

// step 2 instantiate the properties object within the model
construction
this.inputParameters =
devsSimulator.getReplication().getTreatment().getProperties();

…

// step 3 take the input parameter values from the property object
and connect it to the input of the model
String serviceLevel=null;

//inputParameter is the object that contains  the decision variables
Double sL=(Double) this.inputParameters.get("serviceLevel");
String distName= distributorName;

if(sL==null) // using the input from the excel sheet
   {
      String value = "" +
      objects.get(block.getIndexForPropertyName("value1"));
      //adding the parameter values from excel sheet to object values
      values.add(value);
   }
   else//using the input from the optimizer to optimize
   {
     if (distName.equals("DistriCan"))
     {
        serviceLevel=((Double)
        this.inputParameters.get("serviceLevel")).toString();
     }
     else//for the other actors
     {
        serviceLevel=""+objects.get(block.getIndexForPropertyName
        ("value1"));
     }
    values.add(serviceLevel);
   }
```

APPENDIX J: EXAMPLE OF DEFINING THE NAMES OF THE DECISION VARIABLES IN SIMEON

```java
package com.accenture.simulation.supplychain.construct;

public class ConstructExcelBasedModel extends
SimulationOptimizationModel
{
…
…
     public void setModelOutputNames() {
     //the names of the outcome measures that have to be defined by
     The model builder
     this.objectiveVariableNames.add("costs_of_the_supply_chain");
     this.objectiveVariableNames.add("inventory_unit_fill_rate");
     }
}
```

APPENDIX K: EXAMPLE OF A SIMULATION-BASED OPTIMIZATION PROBLEM DEFINITION IN SIMEON

```java
package simeon.problems;

…

public class SupplyChainProblem extends SimulationBasedProblem{

…

private  static int evalCount=0;

/** ID for the serialversion*/
private static final long serialVersionUID = 1L;

/** SEED is the seed value for the DEFAULT stream */
public static final long SEED = 12;

private Context root;

TimeUnitInterface timeUnit= TimeUnitInterface.DAY;

/** the alternative constructor */
public SupplyChainProblem(String solType)
  {
    this(solType, 0.0,1, 100.0);
  }

/**
The default Constructor.
Creates a new instance of the Kursawe problem.
@param solutionType the type of solutions given to the model
@param warmUp warm up time for the simulation model
@param replication the number the simulation model is executed
@param runLength total execution time for the simulation model
*/
@SuppressWarnings("unchecked")
public SupplyChainProblem(String solutionType, double warmUp, int
replication, double runLength )
  {
  …
  }
  …
}
```

APPENDIX L: EXAMPLE OF THE OBJECTIVE FUNCTION EVALUATIONS IN SIMEON

```
package simeon.problems;
…

public class SupplyChainProblem extends SimulationBasedProblem

…

public void notify(EventInterface event) throwsRemoteException {
  if
  (event.getType().
  equals(SimBasedOptExperiment.END_OF_EXPERIMENT_EVENT))
  {
    System.out.println("-------NOTIFY METHOD IN PROBLEM
    CLASS IS INVOKED-------“);

    ((EventProducerInterface)
    event.getSource()).removeListener(this,
    SimBasedOptExperiment.END_OF_EXPERIMENT_EVENT);

    try {
    Context output =
    ContextUtil.lookup(this.simulator_.getReplication().getContext()
    "/charts");

//-TAKING OUT THE DESIRED VALUES FROM THE DEFINED STATISTICAL OBJECTS

    CostsPieChart globalCost= (CostsPieChart) output.lookup("costs
    of the supply chain (pie chart)");

    org.jfree.data.DefaultPieDatasetcostsDataSet =
    globalCost.getDataset();

    double distriCanCost=
    costsDataSet.getValue("DistriCan").doubleValue();

    InventoryOrderFillRatePieChart inventoryOrderFillRatePieChart =
    (InventoryOrderFillRatePieChart)
    output.lookup("inventory unit fill rate: distributor: DistriCan
    [can]");

    org.jfree.data.DefaultPieDatasetinventoryUnitFillRateDataset =
    inventoryOrderFillRatePieChart.getDataset();

    doubleinvUnitFillRate=
    inventoryUnitFillRateDataset.getValue("Delivered")
    .doubleValue();

    //a map based on the variable names defined in the simulation
    model and the content of context object can be made. This varMap
    object is inherited from SimulationBasedProblem class
    this.varMap= newVarMap(false);
    this.varMap.setValue("costs_of_the_supply_chain",districanCost);
    this.varMap.setValue("inventory_unit_fill_rate",invUnitFillRate;
//--------------------END OF CUSTOMIZATION PROCESS—----------------
```

LITERATURE LIST

Accenture. (2010). Company Description. Retrieved 05 April 2010, 2010, from http://www.accenture.com/Global/About_Accenture/Company_Overview/CompanyDescription.htm

Amodeo, L., Chen, H., & El Hadji, A. (2007). Multi-objective Supply Chain Optimization: An Industrial Case Study. In M. Giacobini (Ed.), *Applications of Evolutionary Computing* (Vol. S4448, pp. 732-741). Berlin, Heidelberg: Springer

April, J., Glover, F., Kelly, J. P., & Laguna, M. (2003). *Simulation-based optimization: practical introduction to simulation optimization*. Paper presented at the Proceedings of the 35th conference on Winter simulation: driving innovation.

Azadivar, F., & Tompkins, G. (1999). Simulation optimization with qualitative variables and structural model changes: A genetic algorithm approach. *European Journal of Operational Research, 113*(1), 169-182.

Bandyopadhyay, S., Saha, S., Maulik, U., & Deb, K. (2008). A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA. *Evolutionary Computation, IEEE Transactions on, 12*(3), 269-283.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv., 35*(3), 268-308.

Boesel, J., Nelson, B. L., & Ishii, N. (2003). A framework for simulation-optimization software. *IIE Transactions, 35*(3), 221 - 229.

Bruijn, H., de , & Heuvelhof, E. t. (2008). *Management in Networks*. New York: Routledge.

Bruijn, H., de, Heuvelhof, E. t., & Veld, R. i. t. (2002). Designing a Process. In H. d. Bruijn, E. t. Heuvelhof & R. i. t. Veld (Eds.), *Process Management*. Dordrecht, the Netherlands: Kluwer Academic Publishers.

Burjorjee, K. M. (2008). The Fundamental Problem with the Building Block Hypothesis. *Arxiv preprint arXiv:0810.3356, abs/0810.3356*. Retrieved from http://arxiv.org/abs/0810.3356v1

Burstein, F., & Holsapple, C. (2008). DSS Architecture and Types. In F. Burstein & C. Holsapple (Eds.), *Handbook on Decision Support Systems 1* (pp. 163-189). Berlin, Heidelberg: Springer

Carcangiu, S., Fanni, A., & Montisci, A. (2008). Multiobjective Tabu Search Algorithms for Optimal Design of Electromagnetic Devices. *Magnetics, IEEE Transactions on, 44*(6), 970-973.

Chin, R. T. H., Houten, S.-P. A. v., & Verbraeck, A. (2005). *Towards a simulation and visualization portal to support multi-actor decision making in mainports*. Paper presented at the Proceedings of the 37th conference on Winter simulation.

Coello, C. (2000). An updated survey of GA-based multiobjective optimization techniques. *ACM Comput. Surv., 32*(2), 109-143.

Cunningham, S. W., & van der Lei, T. E. (2009). Decision-making for new technology: A multi-actor, multi-objective method. *Technological Forecasting and Social Change, 76*(1), 26-38.

Daalen, C. v., Verbraeck, A., Thissen, W., & Bots, P. (2009). Methods for the modelling and analysis of alternatives. In A. P. Sage & W. B. Rouse (Eds.), *Handbook of Systems Engineering and Management 2nd edition*. New York: John Wiley and Sons.

Deb, K. (2005). Multi-Objective Optimization. In E. K. Burke & G. Kendall (Eds.), *Search Methodologies* (pp. 273-316). New York: Springer US.

Deb, K. (2008). Introduction to Evolutionary Multiobjective Optimization. In J. Branke, K. Deb, K. Miettinen & R. Slowinski (Eds.), *Multiobjective Optimization* (Vol. 5252, pp. 59-96). Berlin, Heidelberg: Springer

Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and Informatics, 26*(4), 30-45.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182-197.

Deb, K., & Ram B., A. (1994). Simulated Binary Crossover for Continuous Search Space. *Complex Systems, 9*, 1-34.

Ding, H., Benyoucef, L., & Xie, X. (2004). *A simulation-based optimization method for production-distribution network design.* Paper presented at the IEEE International Conference on Systems, Man and Cybernetics, 2004.

Ding, H., Benyoucef, L., & Xie, X. (2006). A simulation-based multi-objective genetic algorithm approach for networked enterprises optimization. *Engineering Applications of Artificial Intelligence, 19*(6), 609-623.

Ding, H., Benyoucef, L., & Xie, X. (2009). Stochastic multi-objective production-distribution network design using simulation-based optimization. *International Journal of Production Research, 47*(2), 479 - 505.

Durillo, J. J., Nebro, A. J., & Alba, E. (2010). *The jMetal Framework for Multi-Objective Optimization: Design and Architecture.* Paper presented at the IEEE World Congres on Computational Intelligence, Barcelona, Spain.

Durillo, J. J., Nebro, A. J., Luna, F., Dorronsoro, B., & Alba, E. (2006). *jMetal: A Java Framework for Developing Multi-Objective Optimization Metaheuristics* (Technical Report No. ITI-2006-10): Departamento de Lenguajes y Ciencias de la Computacion, University of Malaga.

Fu, M. C. (2002). Feature Article: Optimization for simulation: Theory vs. Practice. *INFORMS J. on Computing, 14*(3), 192-215.

Fu, M. C., Andradottir, S., Carson, J. S., Glover, F., Harrell, C. R., Ho, Y.-C., et al. (2000). *Integrating optimization and simulation: research and practice*. Paper presented at the Proceedings of the 32nd conference on Winter simulation.

Fu, M. C., Chen, C.-H., & Shi, L. (2008). *Some topics for simulation optimization*. Paper presented at the Proceedings of the 40th Conference on Winter Simulation.

Fu, M. C., Glover, F. W., & April, J. (2005). *Simulation optimization: a review, new developments, and applications*. Paper presented at the Proceedings of the 37th conference on Winter simulation.

Fu, M. C., & Hu, J. Q. (1997). *Conditional Monte Carlo: Gradient estimation and optimization applications.* New York: Springer-Verlag.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software.* Reading, MA: Addison-Wesley Longman Publishing Co., Inc.

Gen, M., & Cheng, R. (2000). *Genetic Algorithms and Engineering Optimization.* Canada: John Wiley & Sons, Inc.

Gen, M., Cheng, R., & Lin, L. (2008). *Network Models and Optimization - Multiobjective Genetic Algorithm Approach*. London: Springer-Verlag.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.

Gosavi, A. (2003). *Simulation-based optimization parametric optimization technique and reinforcement learning*. Dordrecht, the Netherlands: Kluwer Academic Publisher.

Hani, Y., Amodeo, L., Yalaoui, F., & Chen, H. (2008). Simulation based optimization of a train maintenance facility. *Journal of Intelligent Manufacturing, 19*(3), 293-300.

Hillier, F. S., & Lieberman, G. J. (2001). *Introduction to Operations Research seventh edition*. New York: McGraw-Hill.

Hillier, F. S., & Lieberman, G. J. (2009). *Introduction to Operations Research ninth edition*. New York: McGraw-Hill.

Hojun, L., Zeigler, B. P., & Doohwan, K. (2008). *A DEVS-based framework for simulation optimization: Case study of Link-11 gateway parameter tuning.* Paper presented at the IEEE Military Communications Conference, 2008 (MILCOM 2008).

Holland, J. H. (1973). Genetic Algorithms and the Optimal Allocation of Trials. *SIAM Journal on Computing, 2*(2), 88-105.

Jacobs, P. H. M. (2005). *The DSOL simulation suite Enabling multi-formalism simulation in a distributed context.* Delft University of Technology, Delft.

Jacobs, P. H. M., Lang, N. A., & Verbraeck, A. (2002). *D-SOL; a distributed Java based discrete event simulation architecture.* Paper presented at the Proceedings of the 34th conference on Winter simulation.

Jacobs, P. H. M., Verbraeck, A., & Mulder, J. B. P. (2005). *Flight scheduling at KLM.* Paper presented at the Proceedings of the 37th conference on Winter simulation.

Kazemi, A., Zarandi, M. H. F., & Husseini, S. M. M. (2009). A multi-agent system to solve the production–distribution planning problem for a supply chain: a genetic algorithm approach. *The International Journal of Advanced Manufacturing Technology, 44*(1-2), 180-193.

Keen, P. G. W., & Sol, H. G. (2008). *Decision Enhancement Services: Rehearsing the Future for Decisions That Matter*. Amsterdam, the Netherlands: IOS Press.

Klein, F., Bourjot, C., & Chevrier, V. (2009). Contribution to the Control of a MAS's Global Behaviour: Reinforcement Learning Tools. In F. Klein, C. Bourjot & V. Chevrier (Eds.), *Engineering Societies in the Agents World IX* (pp. 173-190). Berlin, Heidelberg: Springer-Verlag

Klir, G. J. (1991). *Facets of System*. New York and London: Plenum Press.

Konak, A., Coit, D. W., & Smith, A. E. (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety, 91*(9), 992-1007.

Kulturel-Konak, S., Smith, A. E., & Norman, B. A. (2006). Multi-objective tabu search using a multinomial probability mass function. *European Journal of Operational Research, 169*(3), 918-931.

Levner, E. (2007). *Multiprocessor Scheduling, Theory and Applications* Vienna, Austria: I-Tech Education and Publishing.

Magno, C. (2002). *Discrete event simulation for the risk of development of an oil field.* Paper presented at the Proceedings of the 34th conference on Winter simulation.

Marler, R. T., & Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization, 26*(6), 369-395.

Messac, A., Puemi-Sukam, C., & Melachrinoudis, E. (2000). Aggregate Objective Functions and Pareto Frontiers: Required Relationships and Practical Implications. *Optimization and Engineering, 1*(2), 171-188.

Mitroff, I. I., & Turoff, M. (1973). The whys behind the hows. *Spectrum, IEEE, 10*(3), 62-71.

Mueller, D. C. (2008). *Public Choice III*. Cambridge: Cambridge University Press.

Narzisi, G., Mysore, V., & Mishra, B. (2006). *Multi-Objective Evolutionary Optimization of Agent Based Models: An Application to Emergency Response Planning*. Paper presented at the The IASTED InternationalConference on Computational Intelligence.

Osman, I., & Laporte, G. (1996). Metaheuristics: A bibliography. *Annals of Operations Research, 63*(5), 511-623.

Palmer, C. C., & Kershenbaum, A. (1994). *Representing trees in genetic algorithms.* Paper presented at the Proceedings of the First IEEE Conference on Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence.

Pasupathy, R., & Henderson, S. G. (2006). *A testbed of simulation-optimization problems*. Paper presented at the Proceedings of the 38th conference on Winter simulation.

Persson, A., Grimm, H., & Ng, A. (2006). *On-line instrumentation for simulation-based optimization*. Paper presented at the Proceedings of the 38th conference on Winter simulation.

Persson, A., Grimm, H., Ng, A., Lezama, T., Ekberg, J., Falk, S., et al. (2006). *Simulation-based multi-objective optimization of a real-world scheduling problem*. Paper presented at the Proceedings of the 38th conference on Winter simulation.

Pflug, G. C. (1996). *Optimization of stochastic models*. Dordrecht, the Netherlands: Kluwer Academic Publisher.

Pierreval, H., & Paris, J. L. (2003). From 'simulation optimization' to 'simulation configuration' of systems. *Simulation Modelling Practice and Theory, 11*(1), 5-19.

Ragsdale, C. T. (2008). *Spreadsheet Modeling and Decision Analysis: A practical introduction to management science* (5th edition ed.). Mason: South-Western Thomson Learning.

Ray, T., & Liew, K. M. (2002). A Swarm Metaphor for Multiobjective Design Optimization. *Engineering Optimization, 34*(2), 141 - 153.

Ronald, S. (1997). *Robust encodings in genetic algorithms: a survey of encoding issues.* Paper presented at the IEEE International Conference on Evolutionary Computation, 1997.

Rothlauf, F. (2006). *Representations for Genetic and Evolutionary Algorithms*. Berlin, Heidelberg: Springer-Verlag.

Sage, A. P., & Armstrong, J. E. (2000). *Introduction to Systems Engineering*. New York: John Wiley and Sons.

Srinivas, N., & Deb, K. (1995). Multiobjective function optimization using nondominated sorting genetic algorithms. *Evolutionary Computation, 2*(3), 221-248.

Thierry, C., Thomas, A., & Bel, G. (2008). *Simulation For Supply Chain Management*. New York: John Wiley and Sons Inc.

Thomson. (2004). Fast Breaking Comments by Kalyanmoy Deb. *Essential Science Indicator* Retrieved 23- 03 - 2010, 2010, from http://www.esi-topics.com/fbp/2004/february04-KalyanmoyDeb.html

Traoré, M. K., & Muzy, A. (2006). Capturing the dual relationship between simulation models and their context. *Simulation Modelling Practice and Theory, 14*(2), 126-142.

Vidal, C. J., & Goetschalckx, M. (1997). Strategic production-distribution models: A critical review with emphasis on global supply chain models. *European Journal of Operational Research, 98*(1), 1-18.

Walston, J. G. (2007). *Search Techniques for Multi-Objective Optimization of Mixed-Variable Systems Having Stochastic Responses*. Air University, Ohio.

Wang, W., Rivard, H., & Zmeureanu, R. (2005). An object-oriented framework for simulation-based green building design optimization with genetic algorithms. *Advanced Engineering Informatics, 19*(1), 5-23.

Wiedemann, T., & Krug, W. (2003). *Actual and future options of Simulation and Optimization in Manufacturing, Organization and Logistics*. Paper presented at the European Simulation Symposium 2003, Ghent, Belgium.

Zeigler, B. P. (2003). *DEVS Today: Recent Advances in Discrete Event-Based Information Technology*. Paper presented at the 11th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'03).

Zeigler, B. P., Hall, D., & Salas, M. (2009). Agent Implemented Experimental Frames for Net-Centric Systems Test and Evaluation. In L. Yilmaz & T. I. Ören (Eds.), *Agent-Directed Simulation and Systems Engineering*. Berlin: Wiley-VCH.

Zeigler, B. P., Praehofer, H., & Kim, T. G. (2000). *Theory of Modeling and Simulation* (Second ed.). San Diego: Academic Press.

Zitzler, E., Laumanns, M., & Thiele, L. (2001). *SPEA2: Improving the Strength Pareto Evolutionary Algorithm* (Technical Report No. TIK-103): Department of Electrical Engineering, Swiss Federal Institute of Technology (ETH) Zurich.