Agent-based Support for Interactive Search in Conceptual Software Engineering Design

Christopher L. Simons, Ian C. Parmee
Advanced Computation in Design and Decision Making,
University of the West of England,
Bristol, BS16 1QY, UK
+44.117.3283135, +44.117,3283137
{chris.simons, ian.parmee}@uwe.ac.uk

ABSTRACT
While recent attempts to search a conceptual software engineering design search space with multi-objective evolutionary algorithms have yielded promising results, the practical application of such search-based techniques remains to be addressed. This paper reports initial findings of the application of software agents in support of an interactive, user-centered conceptual software design scenario. The supporting role of a number of single responsibility agents is described and results for a case study indicate that the application of such agents to search-based design scenarios provides efficient, high performance and effective support. The notion of interactive, joint human-computer activity appears to map well to conceptual software design scenarios: focus on superior design concepts and thence to useful and interesting designs provides a natural and effective way of narrowing the population based search. In addition, agents and the human designer appear to interact as cooperative "team players", jointly influencing the evolutionary algorithm based search. Nevertheless, challenges remain, including expanding the scale of implementation of underlying technologies to support distributed, collaborative design.

Categories and Subject Descriptors
D.2.2. [**Software Engineering**]: Design tools and techniques.

I.2.8 [Artificial Intelligence]: Problem Solving, control methods and search.

General Terms
Algorithms, Design, Human Factors.

Keywords
Evolutionary algorithms, conceptual software design, search, agents.

1. INTRODUCTION
In order to support the human designer in conceptual software design, attempts have been made recently to apply multi-objective evolutionary algorithms to search the conceptual software design space [1]. While the results of this research show promise, the challenge of how to improve the applicability and efficacy of such search-based techniques to conceptual software engineering design remains. For example, it seems likely that the progress of an evolutionary algorithm should be integrated as far as possible with the natural, typical opportunistic cognitive design processes of the designer, for instance as observed by Guindon [2]. Crucially, as Parmee advocates [3], it is important that the designer can interact with the search process and after the initial expression of preferences, discover design knowledge to steer the search to useful and interesting conceptual designs.

Whenever complex interactive, user centered human-computing activities are undertaken, agent-based support has often proved useful. We hypothesize therefore that the application of agent-based support of evolutionary algorithms in conceptual software design results in a more effective and efficient interactive search process, better supporting the designer.

2. EVOLUTIONARY SEARCH
The multi-objective evolutionary algorithm used to search the conceptual software engineering design search space has been described previously in [1]. The representation of the design space is object-based enabling the direct portrayal of conceptual design classes visualized using the Unified Modeling Language (UML). Objective fitness functions relating to the cohesion of classes, design coupling and design modularity steer the search to designs of superior fitness, while optimization and diversity preservation genetic operators used to provide multi-objective search are inspired by the elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) proposed by Deb [4].

3. DESIGN SCENARIO
Agents facilitate the logging of design events, and the expression of search preferences. Multi-objective search is initiated and autonomously performed by a *Multi-objective Search Agent*, which communicates with a *Concept Zone Isolation Agent*, which evaluates the efficiency of the multi-objective search. The *Concept Zone Isolation Agent* halts search at an optimum point. A number of local searches are then performed concurrently via *Local Search Agents,* by means of a single-objective search agent as described in using design coupling as the objective fitness functions.

The behaviors of all agents employed in this approach are co-ordinated and controlled by a *Scenario Controller Agent* that

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

perceives, monitors and regulates the life cycles and autonomous behaviors of the agents by shared memory ("BlackBoard") communication.

4. CASE STUDY

The case study used in the design scenario is a generalized abstraction of a cinema booking system, derived from a number of established internet-based cinema booking systems existing in the UK. The design problem domain addresses, for example, making an advance booking for a showing of a film, and the collection of tickets on attending the cinema auditorium. A specification of the use cases of the Cinema Booking System design problem is available from [5].

5. RESULTS

A typical example of a conceptual software design with five classes after local search of is shown in figure 1.

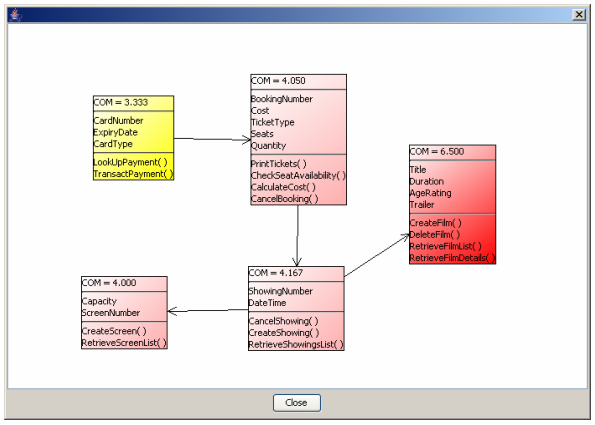

Figure 1. A conceptual class design of 5 classes

The results above have been produced on a standard desktop PC workstation, and are implemented in the Java programming language. The computational time taken by the software agents to perform their tasks in the conceptual design scenario is less than 30 seconds. Significant tasks in the 30 seconds performed typically include one multi-objective search of the global design space, isolation of modularity concept zones and, for example, ten local searches. By exploiting computational concurrency, the ten local searches are performed in less than 10 seconds. Of course, the design scenario can run for as long as the designer's design and decision-making cognitive processes make necessary. What is clear however is that the elapsed computational time taken by the software agents poses little or no performance constraint upon the interactive design scenario.

In addition, the authors have observed that in initial design scenarios, their instinctive choice at first is to set manual thresholds and drive the design scenario manually. However, as design scenarios are repeated and design events are monitored by the event logger agent, the authors observed that leaving the agents to autonomously carry out their responsibilities is quicker and more effective at arriving at useful and interesting software designs. The more the design scenarios are repeated, the more inclined are the authors to rely on the autonomous behavior of the agents. The authors speculate that this is an example of placing increasing trust and confidence in the agents as feedback, predictability and responsiveness to human control is maintained.

6. CONCLUSION

The results achieved with the case study suggest that the application of search agents provides efficient and effective support of the human designer in conceptual software design scenarios. Indeed, the authors speculate that without agent support, it would be difficult and inefficient for a human designer to manually manipulate the population-based search towards useful and interesting designs. We suggest that the notion of interactive, joint human-computer activity appears to map well to conceptual software design scenarios. Increasing trust and confidence in directable, predictable and responsive agent behavior has been observed. Through preferences, utility calculation and event logging, the search agents observe and interpret signals of status and intention; the implemented technologies enable autonomous collaborative behaviors between agent and human participants. However, further challenges remain. The agents evaluated in this study are neither adaptive nor hugely sophisticated, and the choice of supporting technologies does not cater for distributed collaboration between participants. In addition, the potential of interface and information agents in supporting search-based conceptual software design remains to be investigated

7. REFERENCES

[1] Simons, C. L., and Parmee, I. C. 2007. A cross-disciplinary technology transfer for search based evolutionary computing: from engineering design to software engineering design. Eng. Opt., 39, 5 (Jul. 2007), 631-648.

[2] Guindon, R. 1990. Designing the design process: exploiting opportunistic thoughts. Hum.-Comput. Interact., 5, 2-3 (1990), 305-344.

[3] Parmee, I. C. 2002. Improving problem definition through interactive evolutionary computing. Artif. Intell. Eng. Des. Anal. Manuf,. 16, 3 (Jun. 2002), 185-202

[4] Deb, K. 2001. Multi-objective optimization using evolutionary algorithms. Wiley, UK.

[5] Simons, C. L. Use cases for Cinema Booking System. Available online: www.cems.uwe.ac.uk/~clsimons/CaseStudies/CinemaBookingSystem.htm