# FINAL REPORT

# ALSACE Phase Two: The Acxiom Laboratory for Software Architecture and Component Engineering

H. Conrad Cunningham, Chair
Department of Computer and Information Science
University of Mississippi

201 Weir Hall
P.O. Box 1848
University, MS 38677

(662) 915-5358
cunningham@cs.olemiss.edu

12 August 2004

# 1 INTRODUCTION

This is the final report for Phase Two of the project titled "The Acxiom Laboratory for Software Architecture and Component Engineering (ALSACE)." The project was enabled by a grant from Acxiom Corporation and carried out by Associate Professor H. Conrad Cunningham and students in the Department of Computer and Information Science at the University of Mississippi (Ole Miss). In addition to Cunningham, the core team for the Phase Two project consisted of PhD students Yi Liu and Pallavi Tadepalli and MS student Hui Xiong.

The January 2001 proposal for the ALSACE Phase One project states:

> The overall purpose of this project is to establish the Acxiom Laboratory for Software Architecture and Component Engineering (ALSACE) within the Department of Computer and Information Science at the University of Mississippi (Ole Miss). ALSACE will be a collection of resources and a focus of activity for instruction and research in software architecture, design patterns, application frameworks, distributed systems, object-oriented methods, and other concepts related to distributed, component-based computing applications. ALSACE will provide a basis for development of new methods and technologies and provide many students useful experiences in enterprise-level computing applications.

The above quote envisions ALSACE as a structure through which the Department and Acxiom can maintain an ongoing relationship involving placement of students in Acxiom positions, curriculum development, and applied research. Despite the changes of the past 43 months, such a relationship still seems to be mutually beneficial. The Phase One project (January 2002 through August 2003) focused on the development of a component software course. The Phase Two project (September 2003 through August 2004) extended this work to two other software architecture courses. However, Cunningham's team organized its work for both phases with the larger purpose in mind.

# 2 BACKGROUND

To understand this work, it is helpful to consider how these software architecture courses fit into Cunningham's past research and teaching efforts and his vision for future work.

## 2.1 Previous Work

Since the mid-1980s, one focus of Cunningham's research has been concurrent and distributed computing, in particular the application of rigorous software engineering methods to problems in that area. For example, his doctoral research defined a shared dataspace programming notation called Swarm and its proof logic [20, 61, 62]. After Cunningham joined the Ole Miss faculty in 1989, he developed new graduate courses in areas where he had expertise and interest—concurrent programming theory [10, 13, 14], program semantics [11], and functional programming [12, 18].

In the mid-1990s, Cunningham and his students explored the application of formal specification techniques to open, reactive systems [15, 21]. This research foreshadowed the

later work on component-based systems because it emphasized the composition of systems from separately specified subsystems. A subsystem was specified based on its behavior at its boundary with its environment. Assumptions about the environment were explicitly specified.

In 1996 Cunningham saw the need to introduce object-oriented programming (OOP) into the curriculum in a significant way. He taught a special topics course on OOP with C++ in the Spring semester and then introduced the Java language into a sophomore-level core course (CSCI 211) in the Fall semester. Java and OOP techniques were gradually adopted by the faculty as the standard in the introductory computer science sequence. Later, Cunningham introduced new graduate-level special topics on object-oriented software development, software architecture, and distributed objects and revived the dormant undergraduate course on computer simulation.

As a result of the work on the new courses, Cunningham extended his previous research interests in rigorous methods and compositional systems to include software architecture, generic programming, and software reuse. In the 1997-to-2001 period, several of his students carried out MS projects that designed class libraries [41, 64] and software frameworks [24, 53, 73] and used design patterns [76] and technologies such as XML [51, 75] and Java Servlets [35, 59]. Cunningham and Sudharshan Vazhkudai published a paper on design of a software framework for a family of distributed decision-making protocols [71]; two MS projects built on this work [70, 74]. Cunningham and Jingyi Wang also published two papers on various aspects of a Table data/file structure framework [22, 23]; this was an extension to Wang's MS project work [72].

The work since 2001 is described in later sections.

## 2.2 Research and Teaching Agenda

A key perspective of both Cunningham's research and his teaching is that a software development project should be approached as the development of a *family of solutions* [55, 56]. First, software designs must provide a solution to a problem in some domain—that there is a real client who is interested in the solution to a real "business" problem. A solution to this problem will require an understanding of the client's "business" processes and data. Second, the development of a solution should be considered as the development of a whole family of solutions to related problems. These family members may be the different variations of a solution for one client as the original solution is "maintained" and updated to handle the client's evolving business needs. Alternatively, the family members may be solutions needed by different clients to similar business problems. In either case, it is helpful to analyze the problem and design the solution so that it is possible to reuse portions of the design or actual program code. That is, large-scale business systems should be approached as software product lines [1, 9] or application frameworks [6, 29, 60, 63]. Component-based models and technologies [39, 67], such as Enterprise JavaBeans [52], potentially enable members of a family to be built conveniently by "plugging together" a combination of off-the-shelf and custom-built modules.

The thrust of Cunningham's current research efforts is thus the investigation of new methods and technologies for the systematic construction of large-scale information processing systems as described above. These research efforts draw together his past research

on distributed systems and his more recent research on software architectures. The following long-range activities are underway or planned:

- Developing a new component-oriented language that is simple to teach and use and that is compatible with rigorous specification and design practices. (PhD student Yi Liu is developing a language tentatively called BoxScript as a part of her dissertation research [45].)
- Refining software architectural methods for the development of software families targeted to distributed or grid computing environments [31]. (The working name for this project is the AFAR Grid—Application Family Architectural Resources for the Grid.)
- Conducting case studies to investigate issues and experiment with methods for the development of component-based systems, software frameworks, and software product lines.
- Formulating a lightweight methodology for the design of software product lines, particularly those based on component technologies. (The tentative name for this methodology is SoftPLUG, the Software Product Line Utilitarian Guide.)
- Investigating emerging technologies that have an impact upon the development of families of solutions.
- Constructing new tools that make the construction of families of solutions easier.

The thrust of Cunningham's teaching and course development efforts is the enhancement of each student's understanding of and intellectual skills for the development of large-scale information-processing systems. Of course, this requires a firm grasp of the concepts needed to construct effective programs. Students need a deep understanding of concepts such as information hiding, data and procedural abstraction, inheritance, polymorphism, composition, and delegation in the context of contemporary object-oriented languages [4]. Students also need a working knowledge of data management concepts and of effective methods for analysis and design. Beyond this, the students should begin to understand and apply architectural concepts [2, 66] as codified in well-accepted architectural [5, 50, 65] and program design patterns [34, 38]. For systems involving modern distributed computing infrastructures, students need basic familiarity with the fundamental distributed computing concepts, algorithms, and design techniques [7, 36, 49].

Accordingly, Cunningham is developing a set of four courses in the area of software architecture:

**Software Architecture:** This course begins with an intensive "review" of object-oriented programming and design concepts such as data and procedural abstraction, information hiding, inheritance, polymorphism, composition, and delegation in the context of a contemporary object-oriented language (e.g., Java). It then continues with the introduction of the concepts of design patterns and application frameworks. (The ALSACE Phase Two project revised and developed materials for an offering of this course.)

**Component Software:** This course focuses attention on component-based software development. It covers technology-independent concepts and methods for analysis and design of enterprise-level software systems consisting of replaceable and reusable components. The course also discusses the mapping of the component specifications to component technologies such as Enterprise JavaBeans. (The ALSACE Phase One project developed a prototype of this course.)

**Software Families:** This course concentrates on issues related to the design of software families, i.e., software product lines and application frameworks. It seeks to develop the students' abilities to separate concerns and design the generic abstractions needed to exploit the similarity among the members of a family of related software systems. (The ALSACE Phase Two project experimented with this course concept in an offering of the Software Engineering II course.)

**Distributed Software Architecture:** This course is partly a traditional graduate course on distributed computing and partly one in software architecture. It concentrates on the usual distributed computing issues, algorithms, and techniques. But it also seeks to approach the topic from a software architectural viewpoint, stressing development of families of applications.

The Software Architecture course is, for most purposes, a prerequisite of each of the other three courses.

# 3 ALSACE PHASE ONE PROJECT SUMMARY

The ALSACE Phase One project was carried out during the period from January 2002 through August 2003. This project included the following course development and research activities:

1. Development of a new course on component-based software development that emphasizes Java 2 Enterprise Edition (J2EE) technologies and standard notations such as the Unified Modeling Language (UML).

   Cunningham, with help from the Phase One project team, developed and taught a prototype course on Component Software during the 2002 Fall semester to 26 graduate students. The course covered component-based software development concepts [39] and adapted the technology-independent *UML Components* specification methods [8, 46] to the needs of the students. The course discussed techniques for mapping [47] the component specifications to the Enterprise JavaBeans technology [25] for practical programming exercises. The course also used the Unified Modeling Language (UML) notations [32] to express analysis and design models.

   The Component Software course was well-received by the students. The experiences, both positive and negative, from developing and teaching this course motivated quite a bit of the research and course development that followed in the ALSACE Phase Two project and beyond.

Liu and Cunningham described several aspects of the design approach in the paper "Software Component Specifications using Design by Contract" [46]. This paper was published and presented in 2002 at the SouthEast Software Engineering Conference in Huntsville.

The team outlined the course approach and content in the paper "Component Software: A New Software Engineering Course" [17]. This paper was published and presented at the Consortium for Computing Sciences in Colleges (CCSC) Mid-South Conference in Memphis in 2003. Liu also presented a tutorial on EJBs at the CCSC Mid-South Conference in 2003 [44].

2. Collection of materials for the Component Software course.

   The Phase One project team developed the Web site

   `http://alsace.cs.olemiss.edu/~research`

   to hold the course-related materials. Material on the Web site includes the Powerpoint slides developed to support the lectures, information on the Course Registration System case study, the technical papers and presentations associated with the project, and links to materials for related courses.

3. Providing graduate students with experiences in research and development related to component-based and enterprise-level computing applications and technologies.

   The Phase One project gave the Research Assistants (RAs) on the team valuable experiences in software development. Yi Liu, Mingxian Fu, and Pallavi Tadepalli, the project's three core team members, gained significant experience in the development of component-based systems. They used general design methods, JavaServer Pages (JSP), JavaBeans, Enterprise JavaBeans (EJBs), and some XML technologies. Sajan Malla, Jian Li, and Hui Xiong, who were each involved for a portion of the time, worked with some of these technologies. In addition, all 26 students in the Component Software class had experience with EJBs from the class exercises.

   Team members Fu [33], Li [43], and Tadepalli [68] completed MS projects or theses, partly enabled by their experiences on and financial support by the ALSACE Phase One project. Li, her husband, and Cunningham described Li's XML-related project in a paper published and presented at the International Conference on Software Engineering Research and Practice (SERP) in Las Vegas in June 2003 [77]. Tadepalli and Cunningham subsequently described Tadepalli's thesis work that used Java metaprogramming technologies in a paper presented by Tadepalli at the ACM SouthEast Conference in April 2004 in Huntsville [69].

   Subsequently, in July 2004, team member Liu wrote and defended a PhD dissertation prospectus related to this project's emphasis on component-based programming [45].

4. Conducting a case study on the development of component-based software systems to be used as examples and the bases for programming exercises.

   The Phase One project team developed a Course Registration System case study as a component-based system. The materials produced included analysis and design documents, a JSP-based implementation, and an EJB-based implementation [47]. The

Course Registration System design was used in the 2002 offering of the Component Software course.

The EJB-based design and implementation effort yielded new insights into the mapping of component specifications to EJB implementations. These ideas are described in Liu and Cunningham's subsequent paper "Mapping Component Specifications to Enterprise JavaBeans Implementations" [47].

A second case study was Mingxian Fu's MS program UM/VOTE [33], which used JavaServer Pages and JavaBeans to implement a prototype voting system for the Ole Miss Faculty Senate elections.

# 4 ALSACE PHASE TWO PROJECT OBJECTIVES

The ALSACE Phase Two project was conducted during the period September 2003 through August 2004. The specific objectives of the ALSACE Phase Two project were to:

1. Continue refinement and publication of the work conducted under the ALSACE Phase One project.
2. Explore approaches to the course on Software Families (i.e., software product lines and frameworks).
3. Develop materials for the course on Software Architecture.
4. Begin work on the development of a simple component-oriented language for teaching.
5. Continue work on tools and case studies related to component-based software and software frameworks.

# 5 PROGRESS TOWARD OBJECTIVES

The following progress has been made toward the objectives of the ALSACE Phase Two project during the period September 2003 through August 2004.

1. Refinement and publication of the work conducted under the ALSACE Phase One project.

   As noted previously, the EJB-based case study yielded new insights into the mapping of component specifications to EJB implementations. In Fall 2004, Liu and Cunningham refined the paper "Mapping Component Specifications to Enterprise JavaBeans Implementations" [47], which was begun in Phase One. Liu presented the accepted paper at the ACM SouthEast Conference in Huntsville in April 2004.

   Similarly, in summer 2003, Tadepalli completed her MS thesis [68], which included the development of a tool that used the Java metaprogramming facilities (i.e., reflection API) to enable students to create Java objects interactively and execute their methods. During Fall 2003, she and Cunningham wrote the paper "JavaCHIME: Java

Class Hierarchy Inspector and Method Executer" [69], which was based on her thesis. Tadepalli also presented the accepted paper at the ACM SouthEast Conference in April 2004.

2. Exploration of approaches to the Software Families course (i.e., software product lines and frameworks).

   During Fall 2003, Cunningham taught the course ENGR 660 Software Engineering II to 32 graduate students. For the first part of the class, Cunningham selected five classic papers by software engineering pioneer David Parnas and his colleagues [3, 54, 55, 56, 57]. (These papers are collected in an excellent book edited by Hoffman and Weiss [40].) The articles presented the concepts of modularization, information hiding, abstract interfaces, program families, software extension and contraction, and hierarchical modularization in a clear fashion. The assignment for this part was a design and implementation (in Java or C++) of the classic KeyWord in Context (KWIC) problem of Parnas [54, 55].

   The experience with the Parnas papers in this class inspired Cunningham, in cooperation with former student Cuihua Zhang [76] and current PhD student Liu, to write a paper titled "Keeping Secrets Within a Family: Rediscovering Parnas."[16] Liu presented the accepted paper at the International Conference on Software Engineering Research and Practice (SERP) in Las Vegas in June 2004.

   After covering the classic Parnas papers, Cunningham discussed several papers or book excerpts on design patterns [34], design for reuse [42], software framework design [23, 63], and software product lines [1]. He presented Schmid's systematic generalization technique, which is a method for determining the commonalities (frozen spots) and the variabilities (hot spots) in an application family to transform it into a framework design [63].

   This framework study included two exercises. The first was the application of a framework for the family of divide and conquer algorithms [19]. The second was an exercise to apply Schmid's systematic generalization technique for framework design [63] in a few situations.

   Cunningham, Liu, and Zhang wrote a paper describing the divide and conquer framework and exercises. Cunningham and Zhang presented the accepted paper titled "Using the Divide and Conquer Strategy to Teach Java Framework Design" at the International Conference on the Principles and Practice of Programming in Java (PPPJ) in June 2004.

   The final assignment in the class was a project to develop a co-sequential processing [30] framework. This exercise required the students to develop a framework that takes two ordered sequences of records and produces a third ordered sequence of records that has been transformed in some manner. This family includes the important sequential file update programs [26, 28] and set operations [37]. Unlike the divide and conquer exercise, the students were expected to do much of the problem analysis to identify the commonalities and variabilities of the problem domain.

   The class was reasonably well received. Students with good backgrounds in object-oriented programming techniques became quite excited about the ideas. However,

students who were less experienced designers found the programming projects difficult and the concepts quite abstract. Many students also had some difficulty projecting Parnas's pre-OOP expressions of his ideas into the more familiar OOP context.

The class stimulated two MS projects. C. W. Pruett went on to develop a framework for image processing problems under Cunningham in Spring and Summer 2004 [58]. His C++ framework design made extensive use of design patterns. For her MS project in Spring 2004, Rui Zhou also developed a prototype Web application that allows a team of designers and programmers to document designs. It supports the Parnas information-hiding and abstract interface approach to modular design [78].

3. Materials for the Software Architecture course.

   During the 2004 Spring semester, Cunningham, with the assistance of his project team, taught a course on software architecture as an ENGR 691 Special Topics section. The class drew 20 graduate students and seemed to be well-received by the students. The course consisted of an intensive review of the key aspects of the Java language from an architectural perspective. The class then focused on use of design patterns in program design and implementation.

   The course materials collected, refined, or created anew include sets of lecture notes on data abstraction and patterns, various programming examples, and a set of Powerpoint slides for the majority of the lectures. These materials are currently located on the class website:

   `http://www.cs.olemiss.edu/~hcc/softArch/`

   These materials are being updated and will be packaged and copied to the ALSACE project website:

   `http://alsace.cs.olemiss.edu/~research`

   An interesting addition to the Software Architecture class was a guest speaker from industry. Jeffrey "Skip" Sauls, a manager for BEA WebLogic's web portal product, spoke to the class about the engineering of software products in an industrial setting. Sauls is a Department alumnus, who graduated with both BSCS and MS degrees in the early 1990s.

4. Development of a simple component-oriented language for teaching.

   Although the Component Software course taught during the ALSACE Phase One project went well, it was not completely satisfying. One key shortcoming of the course was difficulty in explaining and using the complex EJB programming model. The technology got in the way of teaching the students how to "think in components" cleanly. In particular, once a design specification was developed using the Cheesman and Daniels approach [8], the student still had to map the design into EJBs [47], a process that is seldom straightforward.

   To help alleviate this shortcoming, team member and PhD student Liu began work to design a simple component-oriented programming language as her dissertation research project under the direction of Cunningham. The language, tentatively named

BoxScript, is based on the component concepts presented by Szyperski [67] and by Cheesman and Daniels [8]. Liu wrote a dissertation prospectus and successfully defended it in late July 2004 [45].

The basic entity of BoxScript is a strongly encapsulated component called a *box*. A box may be either *atomic*, with no other boxes nested inside, or *compound*, in which case it consists of one or more other boxes composed to form the larger box. A box is specified by a set of *provided interfaces* (which it must implement) and a set of *required interfaces* (which other boxes must implement). Within a compound box, the provided interface of one constituent box may be connected to the required interface of another box. As a part of her dissertation research, Liu proposes to design the language and develop a prototype runtime system, compiler, and deployment tool [45].

5. Tools and case studies related to component-based software and software frameworks.

   In the last few months of the ALSACE Phase One project (summer 2003), the team began to investigate the use of XML technologies for some of its tool development work. Cunningham proposed a development effort called FRUIT (Family of Retargetable User Interface Tools). This effort seeks to develop a software product line providing a library of device-independent user interface controls similar to XForms [27] implemented for a group of media.

   The work on FRUIT was carried out by Xiong and Liu under the direction of Cunningham. The initial architecture of FRUIT was designed to be compatible with both Java Swing and HTML forms interfaces. The group wrote the paper "The Architectural Design of FRUIT: A Family of Retargetable User Interface Tools" [48] to describe the expected architecture of the product line. Liu presented the accepted paper at the International Conference on Software Engineering Research and Practice (SERP) in Las Vegas in June 2004. Xiong is reexamining and extending this work for her MS thesis that should be completed during the 2004-5 academic year.

   In a related effort, MS student C. W. Pruett undertook the development of an image processing framework under the direction of Cunningham [58]. This interesting C++ framework design encompasses the functionality of a family of image processing operations. It provides a common structure for image processing algorithms that iterate across a matrix of pixels in an image applying some operation on a rectangular neighborhood centered on each pixel. The variable aspects of the algorithms can be "plugged in" using program structures inspired by the Strategy, Template Method, and other design patterns [34].

In summary, the ALSACE Phase Two project made considerable progress toward achieving its ambitious objectives. Cunningham redeveloped and taught two graduate courses related to the project. These two courses together had 52 student registrations, including 38 different individuals. The team published five papers at conferences during the period. The three core team members made progress toward completion of their degrees. Liu completed a PhD dissertation prospectus. Tadepalli completed the required coursework and prepared for the comprehensive examinations. Xiong identified the topic for her MS

thesis and made a good start toward its completion. Two MS students (who received no financial support from the project) completed MS projects on related topics.

# 6 CHALLENGES ENCOUNTERED

The ALSACE Phase Two team faced a few technical challenges. These include:

- Keeping things simple.

  The computing field often seems to glorify complexity. Cunningham and the team have sought simple, but effective, explanations of concepts and tools for supporting student programming. Balancing the desire for simplicity, elegance, and careful thought against the desire to add more features and fancier technology is not easy. For example, finding a simple set of concepts and clean syntax for the BoxScript programming language is a challenge.

- Building abstractions that cleanly unify the members of a potential software family.

  For example, the design of a unified product line for the Java Swing and Java Servlet/HTML versions of the FRUIT software has been difficult. The different operational environments of client-server Web applications and stand-alone Java GUI applications seem to give different high-level structures to the applications.

Most of the challenges, however, were more related to resource availability than to technical difficulties encountered. These challenges included:

- Finding funding to support all Research Assistants.

  The Acxiom grant provided sufficient funding to support one Research Assistant for one year. However, Cunningham had two PhD students and one MS student involved with the project. Although funding was found for all, all three students had to devote considerable time and energy to teaching during the year, slowing progress on most research efforts. Tadepalli taught three different courses during the year. Hui Xiong taught one course three times. Yi Liu taught one course three times in 2003; she was able to concentrate on research in Spring and Summer of 2004 (allowing her to complete her dissertation prospectus).

- Loss of a Department faculty member in Summer 2003.

  The unexpected departure of Arpad Kelemen in Summer 2003 had two primary consequences for the project. First, the cancellation of Kelemen's 600-level class forced Cunningham's Fall 2003 ENGR 660 Software Engineering II (Software Families) class to be converted from a small seminar-style class for advanced graduate students to a general population class with 32 graduate students. Thus, a class that was intended as an interactive seminar had to be reorganized dynamically to rely more on lectures and projects. Second, the search for a replacement for Kelemen meant that Cunningham had to devote considerable time to recruiting faculty in Spring 2004, decreasing his time to devote to preparation for the Spring 2004 Software Architecture class and to his research.

# 7 VALUE FOR OLE MISS

From the perspective of the Department, the ALSACE Phase Two project and its associated activities:

- Speeded the development of courses on software architecture and software families.
- Made the courses more useful by taking advantage of valuable input from Acxiom associates.
- Provided support for MS and PhD students to conduct research and finish theses and research papers. The funding helped attract and retain graduate students. Support for PhD students is an essential ingredient in the Department's efforts to build a strong, well-rounded program at all levels.
- Gave several graduate students valuable experiences in the development of software using software family approaches and technologies.
- Kept ongoing contact with Acxiom that provided other value such as input into curriculum and placement of graduates.
- Stimulated new research ideas in Cunningham and his students in such areas as component language design and software product line development and in software architectures for distributed computing.

# 8 VALUE FOR ACXIOM

From the perspective of Acxiom Corporation, the ALSACE Phase Two project and its associated activities:

- Helped shape two courses and, indirectly, a curriculum that can provide Acxiom and its customers with associates with knowledge and skills that are potentially of strategic importance.
- Partially underwrote two course offerings that allowed 32 and 20 students, respectively, to learn concepts related to software architectures, frameworks, and product lines.
- Encouraged students and alumni of Ole Miss to recognize Acxiom Corporation as a progressive employer and sophisticated supplier of high technology data-oriented services and products.

  This developed a pool of students interested in working for Acxiom or using Acxiom services and products.
- Promoted Acxiom among technical conference attendees by acknowledging its support for the research resulting in five different papers presented at three different conferences.

- Helped train future college faculty and researchers who can potentially be attracted to teach at colleges in Acxiom's recruiting region. Such faculty will have knowledge and skills useful in developing future courses and curricula from a perspective of interest to Acxiom.

- Stimulated new research ideas at Ole Miss related to systematic construction of "solutions" to enterprise-scale business problems involving storage and processing of massive amounts of data.

  It is hoped that this will result in other relationships to cooperate on applied research projects of mutual interest.

# FUTURE WORK

During the two-year period beginning in August 2004, Cunningham, in conjunction with the ALSACE team and other students and collaborators, will seek to accomplish the following:

- Provide an environment where the current student members of his team can complete their research and degrees on schedule. Hui Xiong is scheduled to complete her MS thesis (on the FRUIT system [48]) in December 2004. Yi Liu plans to finish her PhD dissertation (on the BoxScript component-oriented language [45]) by August 2005. Pallavi Tadepalli plans to complete her PhD prospectus by Spring 2005 and her degree by May 2006.

- Complete the development of a prototype of the FRUIT system [45].

- Complete the development of a prototype for the BoxScript component-oriented language [48] and perhaps begin work on new features (more dynamic capabilities, reflection, contract-awareness, graphical tools, etc.) beyond those currently proposed by Liu.

- Begin a new research effort that applies software architectural methods to the development of software families targeted to distributed and grid computing environments [31]. This is the possible AFAR Grid (Application Family Architectural Resources for the Grid) project mentioned above.

- Redevelop the graduate course on Concurrent Programming (ENGR 664) to emphasize distributed computing and software architectures. This Distributed Software Architecture class is planned for Spring 2005.

- Continue the development of a course on Software Families. Cunningham hopes to teach this course in Fall 2005 or Spring 2006.

- Investigate issues and experiment with methods for the development of component-based systems, software frameworks, and software product lines. In addition to the current team members, possible collaborators include PhD student Nighat Yasmin on scientific frameworks, former student Cuihua Zhang on multicultural frameworks, and others not yet identified.

- Begin a new research to formulate a lightweight methodology for the design of software families, particularly those based on component technologies. This is SoftPLUG, the Software Product Line Utilitarian Guide methodology mentioned earlier.

- Continue to investigate emerging technologies that have an impact upon the development of software families and to construct new tools that make the construction of such families easier.

The goals above are ambitious. Several of the items (such as the AFAR Grid and SoftPLUG tasks) likely depend upon graduate students becoming interested in those topics for their theses or dissertations and upon the acquisition of funding for the work. Cunningham will be on sabbatical in Fall 2004, concentrating on his research, working with his graduate students and collaborators, and taking a break from his normal teaching and administrative duties. That period should stimulate forward movement on several of the tasks.

Academic research is often unpredictable and opportunistic. Work on one problem may reveal new problems that are more important and new ideas that are better than the old ones. It is sometimes more fruitful to explore the new directions than to continue on an old path. Such is likely to happen in the next two years as it has in the past two.

# 9 ACXIOM/OLE MISS RELATIONSHIP

The association between Acxiom and Ole Miss began in Spring 2000. Allison Nicholas and John Talburt of Acxiom recruited new associates on the Ole Miss campus. Among those hired that semester were MS graduates Hongmei Gao, Lin Tang, and Jingyi Wang, all three of whom were Cunningham's project students. Acxiom later hired December 2000 MS graduate Wei Qian, also one of Cunningham's students. Acxiom also hired May 2001 BSCS graduate Shafi Al-meher. Wang, Qian, and Al-meher are still Acxiom associates.

The hiring of his students was the beginning of collaboration between Cunningham's team and Acxiom. Cunningham's interactions with Acxiom include several visits by Cunningham and his students to Acxiom facilities since July 2000 and visits by Acxiom associates to Ole Miss. Cunningham also submitted an application in December 2000 for an applied research grant from Acxiom proposing what is called here the ALSACE Phase One project.

Funding for the ALSACE Phase One project began in January 2002. (It was delayed several months by various events in 2001.) That phase of the project was completed in August 2003. The ALSACE Phase Two project began in Fall 2003 and continued until August 2004.

The relationship between Cunningham's Department and Acxiom has been mutually beneficial over four years. The rebounding economy should increase its importance in the future.

# References

[1] M. Ardis, N. Daley, D. Hoffman, and D. Weiss. Software product lines: A case study. *Software—Experience and Practice*, 30:825–847, 2000.

[2] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice*. Addison-Wesley, 1998.

[3] K. H. Britton, R. A. Parker, and D. L. Parnas. A procedure for designing abstract interfaces for device interface modules. In *Proceedings of the 5th International Conference on Software Engineering*, pages 195–204, March 1981.

[4] T. Budd. *Understanding Object-Oriented Programming with Java*. Addison-Wesley, updated edition, 2000.

[5] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *Pattern-Oriented Software Architecture: A System of Patterns*. Wiley, 1996.

[6] J. Carey and B. Carlson. *Framework Process Patterns: Lessons Learned Developing Application Frameworks*. Addison-Wesley, 2002.

[7] K. M. Chandy and J. Misra. *Parallel Program Design: A Foundation*. Addison-Wesley, Reading, Massachusetts, 1988.

[8] J. Cheesman and J. Daniels. *UML Components: A Simple Process for Specifying Component-Based Software*. Addison-Wesley, 2001.

[9] P. Clements and L. Northrup. *Software Product Lines: Practices and Patterns*. Addison-Wesley, 2002.

[10] H. C. Cunningham. Notes on concurrent programming with PCN. Technical Report UMCIS-1993-04, University of Mississippi, Department of Computer and Information Science, March 1994. 85 pages.

[11] H. C. Cunningham. A programmer's introduction to predicate logic. Technical Report UMCIS-1994-02, University of Mississippi, Department of Computer and Information Science, February 1994, revised January 1996. 45 pages.

[12] H. C. Cunningham. Notes on functional programming with Gofer. Technical Report UMCIS-1995-01, University of Mississippi, Department of Computer and Information Science, August 1995, revised January 1997. 192 pages.

[13] H. C. Cunningham. The dining philosophers' problem: A monitor-based solution in PCN. Technical Report UMCIS-1996-01, University of Mississippi, Department of Computer and Information Science, March 1996. 5 pages.

[14] H. C. Cunningham. PCN solutions to the racetrack problem. Technical Report UMCIS-1996-02, University of Mississippi, Department of Computer and Information Science, March 1996. 10 pages.

[15] H. C. Cunningham and Y. Cai. Specification and refinement of a message router. In *Proceedings of the Seventh International Workshop on Software Specification and Design*, pages 20–29. IEEE, December 1993.

[16] H. C. Cunningham, Y. Liu, and P. Tadepalli. Toward specification and composition of BoxScript components. In *Proceedings of the Workshop on Specification and Verification ' of Component-Based Systems (SAVCBS)*, pages 114–117, November 2004.

[17] H. C. Cunningham, Y. Liu, P. Tadepalli, and M. Fu. Component Software: A new software engineering course. *Journal of Computing Sciences in Colleges*, 18(6):10–21, June 2003.

[18] H. C. Cunningham, Y. Liu, and H. Xiong. Lazy functional programming in Haskell. In *Proceedings of the Mid-South College Computing Conference (MSCCC)*, pages 78–80, April 2004.

[19] H. C. Cunningham, Y. Liu, and C. Zhang. Using the divide and conquer strategy to teach Java framework design. In *Proceedings of the International Conference on the Principles and Practice of Programming in Java (PPPJ)*, pages 40–45, June 2004.

[20] H. C. Cunningham and G.-C. Roman. A UNITY-style programming logic for shared dataspace programs. *IEEE Transactions on Parallel and Distributed Systems*, 1(3):365–376, July 1990.

[21] H. C. Cunningham, V. R. Shah, and S. Shen. Devising a formal specification for an elevator controller. Technical Report UMCIS-1994-10, University of Mississippi, Department of Computer and Information Science, September 1994. 14 pages.

[22] H. C. Cunningham and J. Wang. Applying software patterns in the design of a table framework. In *Proceedings of the Conference on Applied Research in Data Engineering*. University of Arkansas at Little Rock, Acxiom Data Engineering Laboratory, ADEL-WP-01-01, 8 pages, November 2001.

[23] H. C. Cunningham and J. Wang. Building a layered framework for the table abstraction. In *Proceedings of the ACM Symposium on Applied Computing*, pages 668–674, March 2001.

[24] T. Dave. A graphical tool for building pipes-and-filters applications. Technical Report UMCIS-1998-15, University of Mississippi, Department of Computer and Information Science, August 1998.

[25] H. M. Deitel, P. J. Deitel, and S. E. Santry. *Advanced Java 2 Platform: How to Program*. Prentice Hall, 2002.

[26] E. W. Dijkstra. Updating a sequential file. In *A Discipline of Programming*, chapter 15, pages 117–122. Prentice Hall, 1976.

[27] M. Dubinko. *XForms Essentials*. O'Reilly, August 2003.

[28] B. Dwyer. One more time—How to update a master file. *Communications of the ACM*, 81(1):3–8, January 1981.

[29] M. E. Fayad, D. C. Schmidt, and R. E. Johnson. *Building Application Frameworks: Object-Oriented Foundations of Framework Design*. Addison-Wesley, 1999.

[30] M. J. Folk, B. Zoellick, and G. Riccardi. *File Structures: An Object-Oriented Approach with C++*. Addison-Wesley, third edition, 1998.

[31] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications*, 15(3), 2001.

[32] M. Fowler and K. Scott. *UML Distilled*. Addison-Wesley, second edition, 1999.

[33] M. Fu. The University of Mississippi Voting Observation and Tabulation Environment (UM/VOTE). Technical Report UMCIS-2002-15, University of Mississippi, Department of Computer and Information Science, December 2002.

[34] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[35] H. Gao. UML database design and implementation using Java servlets. Technical Report UMCIS-2000-12, University of Mississippi, Department of Computer and Information Science, May 2000.

[36] V. Garg. *Concurrent and Distributed Computing in Java*. Wiley, 2004.

[37] M. T. Goodrich and R. Tamassia. *Data Structures and Algorithms in Java*. Wiley, third edition, 2004.

[38] M. Grand. *Patterns in Java*, volume 1. Wiley, 1998.

[39] G. T. Heineman and W. T. Councill. *Component-Based Software Development: Putting the Pieces Together*. Addison-Wesley, 2001.

[40] D. M. Hoffman and D. M. Weiss, editors. *Software Fundamentals: Collected Papers by David L. Parnas*. Addison-Wesley, 2001.

[41] J. Hu. Components of a data and file structures library in Java. Technical Report UMCIS-1997-06, University of Mississippi, Department of Computer and Information Science, August 1997.

[42] R. E. Johnson and B. Foote. Designing reusable classes. *Journal of Object-Oriented Programming*, June/July 1988. `http://www.laputan.org/drc/drc.html`.

[43] J. Li. A Faculty Activity Report editor. Technical Report UMCIS-2002-13, University of Mississippi, Department of Computer and Information Science, December 2002.

[44] Y. Liu. Tutorial on the Enterprise JavaBeans (EJB) component model. *Journal of Computing Sciences in Colleges*, 18(6):110–111, June 2003.

[45] Y. Liu. BoxScript: A component-oriented language for teaching. PhD Prospectus, University of Mississippi, Department of Computer and Information Science, 38 pages, July 2004.

[46] Y. Liu and H. C. Cunningham. Software component specification using Design by Contract. In *Proceedings of the SouthEast Software Engineering Conference*. National Defense Industry Association, Tennessee Valley Chapter, 8 pages, April 2002. `http://www.cs.olemiss.edu/~hcc/papers/sesec02final.pdf`.

[47] Y. Liu and H. C. Cunningham. Mapping component specifications to Enterprise JavaBeans implementations. In *Proceedings of the ACM SouthEast Conference*, pages 177–182, April 2004.

[48] Y. Liu, H. C. Cunningham, and H. Xiong. The architectural design of FRUIT: A Family of Retargetable User Interface Tools. In *Proceedings of the Software Engineering Research and Practice (SERP) Conference*, pages 641–647. CSREA Press, June 2004.

[49] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[50] F. Marinescu. *EJB Design Patterns: Advanced Patterns, Processes, and Idioms*. Wiley, 2002.

[51] V. Matta. XML database mapping. Technical Report UMCIS-2001-13, University of Mississippi, Department of Computer and Information Science, August 2002.

[52] R. Monson-Haefel. *Enterprise Java Beans*. O'Reilly, third edition, 2001.

[53] X. Pang. A communicating sequential processes application framework in Java. Technical Report UMCIS-1999-03, University of Mississippi, Department of Computer and Information Science, May 1999.

[54] D. L. Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, December 1972.

[55] D. L. Parnas. On the design and development of program families. *IEEE Transactions on Software Engineering*, SE-2(1):1–9, March 1976.

[56] D. L. Parnas. Designing software for ease of extension and contraction. *IEEE Transactions on Software Engineering*, SE-5(1):128–138, March 1979.

[57] D. L. Parnas, P. C. Clements, and D. M. Weiss. The modular structure of complex systems. *IEEE Transactions on Software Engineering*, SE-11(3):259–266, March 1985.

[58] C. W. Pruett. A framework for image processing. Technical Report UMCIS-2004-12, University of Mississippi, Department of Computer and Information Science, July 2004. 28 pages. `http://www.cs.olemiss.edu/~hcc/papers/PruettCWproject.pdf`.

[59] W. Qian. A UML database tool for object-oriented design. Technical Report UMCIS-2000-19, University of Mississippi, Department of Computer and Information Science, December 2000.

[60] D. Roberts and R. Johnson. Patterns for evolving frameworks. In R. Martin, D. Riehle, and F. Buschmann, editors, *Pattern Languages of Program Design 3*, pages 471–486. Addison-Wesley, 1998.

[61] G.-C. Roman and H. C. Cunningham. Mixed programming metaphors in a shared dataspace model of concurrency. *IEEE Transactions on Software Engineering*, 16(12):1361–1373, December 1990.

[62] G.-C. Roman and H. C. Cunningham. Reasoning about synchronic groups. In *Research Directions in High-level Parallel Programming Languages,* LNCS #574, pages 21–38. Springer-Verlag, 1992.

[63] H. A. Schmid. Framework design by systematic generalization. In M. E. Fayad, D. C. Schmidt, and R. E. Johnson, editors, *Building Application Frameworks: Object-Oriented Foundations of Framework Design*, pages 353–378. Wiley, 1999.

[64] D. Sharma. A Java class library for B-trees. Technical Report UMCIS-1997-12, University of Mississippi, Department of Computer and Information Science, December 1997.

[65] M. Shaw. Some patterns for software architecture. In J. M. Vlissides, J. O. Coplien, and N. L. Kerth, editors, *Pattern Languages of Program Design 2*, pages 255–270. Addison-Wesley, 1996.

[66] M. Shaw and D. Garlan. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, 1996.

[67] C. Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, second edition, 2002.

[68] P. Tadepalli. Graphical class browsing and execution environment. Master's thesis, University of Mississippi, Department of Computer and Information Science, August 2003.

[69] P. Tadepalli and H. C. Cunningham. JavaCHIME: Java Class Hierarchy Inspector and Method Executer. In *Proceedings of the ACM SouthEast Conference*, pages 152–157, April 2004.

[70] V. Thomas. A Java-based design and implementation of an object-oriented framework for communication in distributed decision-making protocols. Technical Report UMCIS-2000-16, University of Mississippi, Department of Computer and Information Science, December 2000.

[71] S. Vazhkudai and H. C. Cunningham. A reusable framework for distributed decision-making protocols. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications*. CSREA Press, June 2000.

[72] J. Wang. A flexible Java library for table data and file structures. Technical Report UMCIS-2000-07, University of Mississippi, Department of Computer and Information Science, May 2000.

[73] M. Wei. A Java-based pipes-and-filters framework. Technical Report UMCIS-1998-05, University of Mississippi, Department of Computer and Information Science, May 1998.

[74] J. Xu. An object-oriented framework for distributed decision-making protocols. Technical Report UMCIS-2000-11, University of Mississippi, Department of Computer and Information Science, August 2000.

[75] W. Xu. An XML-based tool for automating faculty activity reports. Technical Report UMCIS-1999-11, University of Mississippi, Department of Computer and Information Science, August 1999.

[76] C. Zhang. Automated degree application form—A case study of pattern design and an application of Java programming. Technical Report UMCIS-1999-13, University of Mississippi, Department of Computer and Information Science, December 1999.

[77] T. Zhang, H. C. Cunningham, and J. Li. FAR: An editing tool for standard information generation. In *Proceedings of the 2003 International Conference on Software Engineering Research and Practice (SERP'03)*. CSREA Press, 4 pages, June 2003.

[78] R. Zhou. Online module guide. Technical Report UMCIS-2004-04, University of Mississippi, Department of Computer and Information Science, May 2004.