Curriculum proposal number 2003.90

Curriculum Action Request (CAR) (Form 4-93) - Maui Community College

1. Author(s) Sandra R. Swanson

2. Authors' unit(s) Professional / Technology: Information & Computer Science

3. Date submitted to Curriculum Committee 05 January 2004

4. a. General type of action? _x_course __program

   b. Specific type of action

Addition	Deletion	Modification	
x_regular	__course	__number/alpha	__prerequisites
__experimental	__from program	__title	__corequisites
__other (specify)	__program	__credits	__program
____________	__other (specify)	__description	__other (specify)
	____________		____________

5. Reason for this curriculum action

   This course will be a requirement for the proposed High Performance Computing certificate of completion being introduced at MCC as part of the HPC grant.

6. Existing course

____

alpha number title credits

7. Proposed new/modified course

ICS 258 Programming for High Performance Clusters 3

alpha number title credits

8. New course description or page number in catalog of present course description, if unchanged.

   Explores programming for high performance computational clusters. Examines the algorithmic paradigms required to most efficiently and effectively create or modify code that will exploit the unique characteristics of parallel processing. Identifies the attributes common to highly parallelizable code. Develops parallel algorithms and writes implementing computer code. Tests, evaluates, and refines code to maximize performance and efficiency.

9. Prerequisite(s): ICS 111 with at least a C, or consent

10. Corequisite(s)

11. Recommended preparation

12. Is this course cross-listed? ___yes _x_no If yes, list course

13. Student contact hours per week

    lecture___hours lab___hours lecture/lab_3_hours other___hours, explain

14. Revise current MCC General Catalog page(s) 34, 108

15. Course grading ___letter grade only ___credit/no credit _x__either X audit

16. Proposed semester and year of first offering? _Spring__semester _2005_year

17. Maximum enrollment__24___ Rationale, if applicable Number of computers in laboratory

18. Special scheduling considerations? _x_yes __no If yes, explain. Laboratory availability

19. Special fees required? __yes _x_no If yes, explain.

20. Will this request require special resources (personnel, supplies, etc.?) _x_yes __no

If yes, explain. Computers in laboratory must have Linux OS, Java compiler & run time engine, and C compiler & linker installed and operating.

21. Is this course restricted to particular room type? _x_yes __no If yes, explain. See #19

22. __Course fulfills requirement for ______________________________ program/degree

_x_Course is an elective for __Technical Elective for ECET____ program/degree

_x_Course is elective for AS degree

23. This course __increases __decreases _x_makes no change in number of credit required for the program(s) affected by this action

24. Is this course taught at another UH campus? __yes _x_no

a. If yes, specify campus, course, alpha and number

b. If no, explain why this course is offered at MCC

To meet requirements for Certificate of Completion in High Performance Computing.

25. a. Course is articulated at

__UHCC __UH Manoa __UH Hilo __UH WO __Other/PCC

b. Course is appropriate for articulation at

__UHCC __UH Manoa __UH Hilo __UH WO __Other/PCC

c. Course is not appropriate for articulation at

__UHCC __UH Manoa __UH Hilo __UH WO __Other/PCC

d. Course articulation information is attached? __yes __no

. . . .

Proposed by

Sandra R. Swanson: 02 Jan 2004
Author or Program Coordinator/Date

Approved by

13 Feb 04
Academic Senate Chair/Date

Requested by

Diane Meyer 10/Feb/04
Division or Unit Chair/Date

2/16/04
Chief Academic Officer/Date

Maui Community College
Course Outline

1. Alpha and Number — ICS 258

Course Title — Programming for High Performance Clusters

Credits — Three (3)

Date of Outline — 02 January 2004

2. Course Description — Explores programming for high performance computational clusters. Examines the algorithmic paradigms required to most efficiently and effectively create or modify code that will exploit the unique characteristics of parallel processing. Identifies the attributes common to highly parallelizable code. Develops parallel algorithms and writes implementing computer code. Tests, evaluates, and refines code to maximize performance and efficiency.

2. Contact Hours/Type — Three (3): lecture~~/discussion~~/laboratory

4. Prerequisites — ICS 111 with at least a C, or consent

Corequisites

Recommended Preparation

Approved by ____________________ Date 2/4/04

Recommended by

5 Feb04

Curriculum Chair/Date

2/24/04

Chancellor/Date

Revised Sept 2003/AC

5. General Course Objectives

This course will investigate and study the unique characteristics of computer programs designed to exploit the unique hardware characteristics of clusters. Existing parallelized code will be studied and evaluated, and assignments will challenge students to create their own. Existing, non-parallel, code will be evaluated for its appropriateness for parallelization. Both algorithmic and coding solutions will be explored.

6. Specific Course Objectives, Competencies, and Student Learning Outcomes

For assessment purposes, these are linked to #7. Recommended Course Content.

Upon successful completion of this course the student shall demonstrate mastery of, and competence in, the following areas through assignments, classroom discussions, laboratory projects, and formal evaluation:

a) Explain those characteristics that make some problems excellent candidates for parallelizable, and which do not.
b) Compare and contrast algorithmic solutions to code and investigate their suitability for parallelization.
c) Summarize tools for code parallelization that exist in modern programming languages.
d) Demonstrate competency by parallelizing existing scalar code.
e) Design, write, and debug parallel code to solve simple problems.
f) Compare and contrast efficiencies of parallelization by running code on different numbers of cluster nodes.

7. Recommended Course Content and Approximate Time Spent on Each Topic

Linked to #6. Specific Course Objectives, Competencies, and Student Learning Outcomes.

- 2-3 weeks Overview of programming and the unique attributes of parallel code: a, b.
- 2-3 weeks Evaluate problems for their suitability to be parallelized: a, b, c.
- 2-4 weeks Review existing scalar and parallel code: a, b, c, d.
- 2-4 weeks Modify scalar code to parallel and compare execution times: a, b, c, d, e, f.
- 3-5 weeks Create parallel solutions to various problems: a, b, c, d, e, f.

8. Text and Materials, Reference Materials, Auxiliary Materials and Content

Textbooks do not currently exist for this course. Students will use existing Java, Fortran, and C textbooks, along with their related APIs as reference materials. Journals, along with material provided by the National Center for Excellence in High Performance Computing will be used for instruction and reference. Instructors must be capable of developing classroom lecture materials, assignments, and laboratory exercises to adequately stimulate and challenge the student's learning experience. Emphasis will be made on algorithm development and the writing of tight, efficient, and reliable code.

9. Recommended Course Requirements and Evaluation

Specific course requirements are at the discretion of the instructor at the time the course is being offered. Suggested requirements include, but are not limited to: ICS 111 and ICS/ETRO 251with at least a C, or consent. Evaluation will be via assignments, testing, and laboratory projects and will be graded as follows:

Programming and reading assignments:	20-60 %
Unannounced quizzes:	10-25 %
Scheduled examinations:	15-40 %
Term project:	10-20 %
Student's class participation and attendance	0- 8 %

10. Methods of Instruction

Instructional methods will vary with instructors. Specific methods may vary at the discretion of instructors and may include, but are not limited to:

Lecture (PowerPoint, OpenOffice Impress, or similar).
Classroom discussion.
Hands on laboratory exercises.
Design and implementation of shell scripting by example and evaluation.
Special projects.
Assignments.
Quizzes and examinations.
Guest lecturers.
Field trips.