Date of request: 08-OCT-2019
College: A
Department: CS
Initiator name: Jason Johnson
Initiator email: jason.e.johnson@wmich.edu
Proposed effective term: 202040
Does course need General Education approval?: N
Will course be used in teacher education?: N
If 5000 level course, prerequisites apply to: U
Proposed course data:
New Course CS 4610
New course selected: This new course is not seeking approval as a general education course.

1. Proposed course prefix and number:
   CS 4610

2. Proposed credit hours:
   3

3. Proposed course title:
   Reusable Software Architectures

4. Proposed course prerequisites:
   CS 3310

5. Proposed course corequisites:
   none

6. Proposed course prerequisites that may be taken concurrently (before or at the same time):
   none

7. Minimum grade for prerequisites (default grades are D for Undergrad and C for Grad):
   C

8. Major and/or minor restrictions:
   Not Applicable
9. List all the four-digit major and/or minor codes (from Banner) that are to be included or excluded:
none

10. Classification restrictions:
Not Applicable

11. List all the classifications (freshman, sophomore, junior, senior) that are to be included or excluded:
none

12. Level restriction:
Not Applicable

13. List the level (undergraduate, graduate) that is to be included or excluded.
Not Applicable

14. Do prerequisites and corequisites for 5000-level courses apply to undergraduates, graduates, or both?
Not Applicable

15. Is this a multi-topic course?
No

16. Proposed course title to be entered in Banner:
Reusable Software Arch

17. Is this course repeatable for credit?
No

18. Is this course mandatory credit/no credit?
No

19. Select class type:
Lecture/Lab/Discussion

20. How many contact hours per week for this course?
3

A. Please choose Yes or No to indicate if this class is a Teacher Education class:
No

B. Please choose the applicable class level:
Undergraduate

C. Please respond Yes if this is a current general education course and/or a course being submitted for the new WMU Essential Studies program. Please respond No if it is neither.
No
D. Explain briefly and clearly the proposed improvement.
Create an undergraduate level course in software architecture approaches.

E. Rationale. Give your reason(s) for the proposed improvement. (If your proposal includes prerequisites, justify those, too.).
Software development organizations utilize several different approaches to software architecture. Our students should be aware of and practiced in the characteristics of the most common approaches in order to be more competitive in the marketplace.

CS 3310 is a necessary prerequisite for this course because a fairly advanced level of programming experience and knowledge of basic data structures and algorithm analysis will be necessary to be successful. Students will be required to implement projects using various design patterns, so it will be necessary for them to have the skills gained in CS 3310 prior to starting this course.

F. List the student learning outcomes for the proposed course or the revised or proposed major, minor, or concentration. These are the outcomes that the department will use for future assessments of the course or program.
Students will be able to:
• Design a solution to a complex problem using a specific design pattern or patterns in the Java programming language.
• Apply correctly for a certain task a variety of different design patterns in Java such as: visitor, observer, command, factory, singleton, builder and more.
• Investigate design models such as producer/consumer or thread pools.
• Apply test-driven development principles and unit testing with JUnit.
• Design GUIs to visualize the solution to given problems.

G. Describe how this curriculum change is a response to student learning assessment outcomes that are part of a departmental or college assessment plan or informal assessment activities.
Not applicable, although our Advisory Board supports this change and feels that this content may be helpful in making our students more marketable.

H. Effect on other colleges, departments or programs. If consultation with others is required, attach evidence of consultation and support. If objections have been raised, document the resolution. Demonstrate that the program you propose is not a duplication of an existing one.
None.

I. Effect on your department’s programs. Show how the proposed change fits with other departmental offerings.
This new course will ensure that our students have more knowledge and training that will be directly applicable to securing employment post-graduation.

J. Effects on enrolled students: are program conflicts avoided? Will your proposal make it easier or harder for students to meet graduation requirements? Can students complete the program in a reasonable time? Show that you have considered scheduling needs and
demands on students' time. If a required course will be offered during summer only, provide a rationale.
While students will be required to take this course, which has not been the case in the past, it will be offered in both Fall and Spring semesters each year. That being the case, scheduling will not be difficult and the time required to complete a Bachelor of Science in Computer Science degree will not change.

K. Student or external market demand. What is your anticipated student audience? What evidence of student or market demand or need exists? What is the estimated enrollment? What other factors make your proposal beneficial to students?
Estimated audience is all undergraduate CS students.

L. Effects on resources. Explain how your proposal would affect department and University resources, including faculty, equipment, space, technology, and library holdings. Tell how you will staff additions to the program. If more advising will be needed, how will you provide for it? How often will course(s) be offered? What will be the initial one-time costs and the ongoing base-funding costs for the proposed program? (Attach additional pages, as necessary.) Offering this class will not require any additional faculty or advising resources.

M. With the change from General Education to WMU Essential Studies, this question is no longer used.
For courses requesting approval as a WMU Essential Studies course, a syllabus identifying the student learning outcomes and an action plan for assessing the student learning outcomes must be attached in the Banner Workflow system.
Not Applicable

N. (Undergraduate proposals only) Describe, in detail, how this curriculum change affects transfer articulation for Michigan community colleges. For course changes, include detail on necessary changes to transfer articulation from Michigan community college courses. For new majors or minors, describe transfer guidelines to be developed with Michigan community colleges. For revisions to majors or minors, describe necessary revisions to Michigan community college guidelines. Department chairs should seek assistance from college advising directors or from the admissions office in completing this section. The addition of this course will in no way restrict any current transfer articulation agreements, and may add options for students to transfer credits.

O. Current catalog copy:
N/A

P. Proposed catalog copy:
This course covers common reusable patterns of software design: creational, structural and behavioral patterns are explored using the Java programming language. General Java concepts are also covered such as class creation, building GUIs and using the debugger inside an IDE.

Department Curriculum Chair approver: Jason Johnson
CS 4610: Reusable Software Architectures
Master Syllabus

Java Language (3 weeks)
Data Types and main()
Variables
Methods
Inheritance

Test-Driven Development (2 weeks)
TDD Structure
Unit Testing and JUnit

Design Patterns (4 weeks)
Visitor
Observer
Command
Factory
Singleton
Builder

Design Models (3 weeks)
Producer/Consumer
Thread Pool

GUI Development (2 weeks)
Event-based Programming
Java GUI packages

Course Prerequisites
CS 3310

Catalog Description

This course covers common reusable patterns of software design: creational, structural and behavioral patterns are explored using the Java programming language. General Java concepts are also covered such as class creation, building GUIs and using the debugger inside an IDE.

Learning Outcomes
Students will be able to:

- Design a solution to a complex problem using a specific design pattern or patterns in the Java programming language.
- Apply correctly for a certain task a variety of different design patterns in Java such as: visitor, observer, command, factory, singleton, builder and more.
- Investigate design models such as producer/consumer or thread pools.
- Apply test-driven development principles and unit testing with JUnit.
- Design GUIs to visualize the solution to given problems.