Architecture of Real-Time Systems

Winter 2023

SE 456

instructor:	Ed Keenan
email:	ekeenan2@cdm.depaul.edu
office hours:	REMOTE: 6-7 pm (Mon) / 3-4 pm (THURS) Zoom Meeting or by appointment
office:	CDM 830
phone:	(312)362-6747
website:	https://piazza.com/depaul/winter2023/se456 (Preferred communication)
lecture:	REMOTE: Wednesday 6:00-9:00pm Synchronous Zoom Meeting
Desired to Learn (D2L):	d2l.depaul.edu (Grades, Viewing lectures, Announcements)
Version Control:	perforce: *140.192.39.61:1666*
All Communication:	Piazza for all communication, reply within 24 hrs during school week

Description:

This course discusses the principles, styles, and patterns of real-time software architecture. Trade-offs and ramifications of software architecture with respect to performance, maintainability, and reusability, will be explored. The course will also investigate the design and implementation of real-time behavior and constraints for common Design Patterns such as Observer, Visitor, and Strategy. Finally, the course will demonstrate how creation of real-time Data Driven environment allows the run-time object data to control the behavior and flow of an application. These topics will be discussed in the context of best practices in software engineering such as, iterative development, testing, and continuous integration.

Learn the principles, styles, and patterns of software architecture and framework design in the context of computer game development. Emphasis of the macro/micro design patterns will be incorporated to control the complexity and efficiency of the application.

This is intensive software development course using the C# - windows environment. Students will work on the understanding of the requirements, design, and implementation of real-time individual components of an application. A final project of developing a complete computer game with the emphasis on controlling the real-time tradeoffs and behavior's will be emphasized throughout the class.

Software engineering aspects throughout the class will be emphasized. You will learn software design principles such as design pattern and software architecture styles that can be applied in the design of object-oriented frameworks for game development, i.e., object-oriented game engines.   We will explore the trade-offs and ramifications of software architecture and design choices with respect to performance, maintainability, and reusability, etc. We will also discuss how some of the best practices in software engineering such as, iterative development, testing, and continuous integration, can be applied in game development.

Prerequisites:

- Required:
    - CSC 403: Data Structures II
        - Java, C# or C++ language programming ability
        - Basic data structures
        - Introduction to Object Oriented concepts

Learning Goals:

- Students will be able to identify real-time structures and layout.
- Students will be able to develop components such as update loops, interactive graphics, collisions, audio, input, and animation.
- Students will be able to understand and implement real-time behavior and constraints for common Design Patterns such as Factory, Observer, Visitor, Composite, Commander, State, Strategy, Null, Singleton, Flyweight, Iterator, and Memento.
- Students will learn how to create real-time Data Driven environment allowing the run-time object data to control the behavior and flow of an application.
- Students will reinforce their Software Engineering practices such as Iterative development, Testing, continuous integration and documentation.

Grading

Sprints – Game Project

　　　10% - Sprints Code/Video Submissions (code + video)

Programming Assignment (Design Pattern)

　　　10% - Sprints Code/Video Submissions (code + video)

Milestone 1

　　　20% - Animated Alien Grid with movement (code + video)

Finals week

　　　10% - Design paper (UML and Patterns)

　　　50% - Project: Space Invaders game (code + video)

Textbooks and printed resources

Additional course material will be many supplied through class notes, handouts or online links.

- 2 Required Books
    - ***Head First Design Patterns*** by Freeman, Bates, Sierra, Robson, publisher O'Reilly, November 2004
        - ISBN 978-0596007126
    - ***Head First Object-Oriented Analysis and Design*** by McLaughlin, Pollice, West, publisher O'Reilly, Dec 2006
        - ISBN 978-0596008673

Additional Material

- Will be provided by the instructor
- Lectures, links, SDKs and other corresponding material

Software

- ***Microsoft Visual Studio 2019 Enterprise Edition (not Community)***
  - [MSDNAA Depaul – Visual Studio 2019 Enterprise](#)
    - C++ and C# install (future classes)
  - Any other variants are not used in this class
  - Students are responsible keeping their development tools working
- ***Perforce Server***
  - Download and configuration instructions will be provided in class
  - ***Perforce – Helix Visual Client (p4v)***
    - https://www.perforce.com/downloads/helix-visual-client-p4v
    - Server address: ***perforce.dpu.depaul.edu:1666***

Topics will include:

Game Fundamentals

- Game Loops
- Timers
- Audio
- State Machines
- Input

2D Games

- Sprites
- Collision
- Movement
- Images loading / manipulation

Design Patterns

- Singleton
- Factory
- Null Object
- Flyweight
- State
- Strategy
- Observer – Publisher/Subscriber
- Memento
- Data Driven

Project Management

- Iterative Development
- Scheduling

Programming Assignments (10%)

Students will have weekly programming assignment to cover material of the week. Each programming assignment will focus on a specific design pattern with unit tests. Goal is to understand the pattern then apply the pattern into the larger project. Examples of patterns implemented:
- Memory Pooling, Factory, Proxy, Observer, State, Command, Visitor

Sprints (10%)

Students will have weekly programming assignment to cover material of the week. Each sprint has coding applications related to the architecture and design patterns challenges. The goal of the sprints to encourage continuous development on a very large project. Mini-sandboxes of topics used in the final project. Accompanying each sprint is a video with student commentary explain their code and design issues.

Topics:
- Manager – Generalized manager to hold, organize, and coordinate object
- Sprites – Sprite, Texture, Image Systems
- Sprite Batch – Group ordering, processing, and drawing of sprites
- Animation System – Animated flipping of images painted on the sprites
- Alien Grid – Organization/movement
- Collision System – Generalized system to collide object with unique reactions
- Font System – gui displayed fonts
- Stress Test – race condition testing, mark and sweep contention
- Game Cycling – select screen, demo, game, game over

Milestones (20%)

Serious checkpoints during development to validate progression of the final project.

Milestone 1 (20%) week 6
- Animated alien grid moving in timed unison
- Screen proportional, final art
- Design documentation – one pattern

Project: Space Invaders Game (50%)

Students will design and build a clone of the arcade version of Space Invaders. This cloned game will be written in C# in the AZUL framework using MODERN software engineering principles, including design patterns, iterative-based development, robust and orthogonal systems. Minimum of 10 design patterns must be used.

- Stand-alone Game application will include but limited to:
  - Select Screens,
  - Game cycling
    - (attract mode->select->enter->game over->select),
    - 2 player mode
  - Score,
  - Player movement and control,
  - Alien Grid,
  - Animation,
  - UFOs,
  - Missiles,
  - Bombs,
  - Erodible shields,
  - Graphics effects
  - Sounds
- Complete Code base
  - 100% working game stored in perforce
  - Needs to compile and run
  - Not compiling or working (FAIL)
- Feature List
  - Self-grading feature checklist (supplied PDF)
  - Link to Video Demo
- Video Demo
  - YouTube Video Demo
  - Demo of the game
  - Identifying features in the game
  - 10 minutes with clear audio commentary

Game Components

Individual game components will be designed and discussed during each class. Each component uses several design patterns. Components are implemented in progressive order, based on complexity and number of design patterns introduced. Features of each component will be discussed and evolved as the class progresses. List of some of the components:

- Input Manager
  - Input recorder / manager
  - Ability to search for specific button combos
  - Clean abstraction and mapping

- Sprite / Animation system
    - Create a sprite system that displays the sprite image
    - Animate the sprite to display an different image at a time
        - Set the series of images to cycle through
        - Set the timing / speed to display the images
    - Ability to reuse sprite images instead having duplicate images loaded at the same time.
    - Should be general enough to display any sprite used in the game
        - Alien Ships, Shields, Player, UFO, missiles
- Collision system
    - Create a collision system that determines collisions between the missiles and collideable objects
        - Such as the Aliens, Player, Ship, UFO, Collision subpart in the shield
    - The collision system should be able to determine if any missile hit the collision box.
        - Ideally this system should be able to intersect with missile box with the target box
        - Determine the state of the collision, {non-intersection, or intersection}
    - This system should be able to be used with any collision needs in this game:
        - Aliens, Player, Ship, UFO, Collision subpart in the shield
- Shield collision and display system
    - Determine how to have the shield be dissolving and eroding by missiles from the aliens
    - Determine how the shield is dissolved and eroded by missiles from the player
    - Shield as a protector
        - It protects the player, from the alien missiles
    - Drawing mechanism to show the erosion
    - Determine the underlining collision grid and update mechanism
- Sound system
    - Create a system that loads and plays static waves in the game
    - Allow other systems to call single of sound waves for their respective effects
    - Ability to control individual volumes of each sound playing
    - Allow many different sounds to be playing at once,
        - i.e. multiple explosions overlapping
    - Gross sound controls
        - Mute, game overall volume

Design paper (10%)

- Engineering Design Documentation
    - Arrangement of design document (like a book)
        - Overall design / High level view
        - Component Discussions
            - Should discuss every Design Pattern used in detail
            - Min 10 design patterns (must match YOUR code)
        - Post-Mortem
            - Improvements, Commentary
    - Discussion of all the major systems
        - Each system (component)
            - UML of the system
        - Design Patterns Used in component
            - 2+ pages for each Design Pattern
            - Diagramming and discussion

- Descriptions of the interactions between components
- Any discussion of trade-offs or problems you overcome
  - o In general UML diagrams
    - Structural diagrams
    - Sequence diagrams (if needed)
  - o How do you know if your wrote enough
    - This report should give:
      - A good understand of what you did
      - Impress a future employer
    - Should be clear that you are a Software Architect
  - o Page length Expectations
    - 20-30 pages in PDF format
      - don't freak - UML diagrams take space

Readings

Textbooks are used for references and learning new topics. It is suggested that you research and investigate material and ancillary topics covered in the class through these books as needed. High performance programming requires both breadth and depth knowledge in the C++ language, therefore everyone's needs and research will vary based on their own experience and evolving mastery of the material.

Grading Scale:

93-100: A	87-89: B+	77-79: C+	67-69: D+	0-59: F
90-92: A-	83-86: B	73-76: C	60-66: D	
	80-82: B-	70-72: C-		

Perforce Submissions

- Everyone is expected to submit several <u>submissions</u> to perforce a week.
  - o Minimum of *five* significant (real) submissions on *two* separate days.
  - o To promote incremental development and prevent last day rush.
  - o Grade deduction will occur if not followed
- The biggest reason students get into trouble with software design:
  - o Not starting the project early
  - o Not working on the material frequently enough
  - o Taking too large of a bite(byte) of the design
- Both are minimized with this *Perforce RULE*
- Even my simplest programs take 10-20 submissions.
  - o For these project assignments, my average is 40-400 submissions, so five will be no problem.
- Detailed perforce changelist comments are expected

Piazza Discussion forum

- Previous classes have highly participated in Piazza and did quite well
  - To promote this secondary learning approach
- Statistics show: students who participate more and help other students <u>do better</u>!
  - The correlation is ridiculous!
    - Poor understanding / poor participation.
    - Great understanding / Great participation
  - As you master the material, help others learn!
    - Want to be a Master programmer so master it!
- <mark>Everyone is **_expected_** and encouraged to participate on the Piazza discussion forum</mark>. All class-related discussion here this term.
- Everyone is expected to keep up with the material on Piazza and are responsible for its content. Critical class updates and directions will be presented there.
  - Not participating or reading the material on Piazza is **_NOT_** an **_Excuse_**.
- All correspondence that is not personal in nature should be vectored through Piazza
  - Sensitive material, use Piazza private note, not email.
- The quicker you begin asking questions on Piazza (rather than via emails), the quicker you will benefit from the collective knowledge of your classmates and instructors. I encourage you to ask questions when you are struggling to understand a concept.
- Keep the forum professional and positive, help each other out.
  - Karma really pays off here.
  - Help each other whenever you can.
    - There will be a time when you will need help from the class (trust me).

NOTE: Do **_NOT_** post until you have watched the entire lecture **_FIRST_** (in class or online)

This will prevent frustration on all sides (members asking or answering questions)

Collaborating together on programming assignments
- You are encouraged to work together
  - Use the Piazza forums heavy
  - Even share your material with others in the common directory
    - Obviously not the answers
- <mark>Everyone is 100% responsible for the work they do.</mark>
  - If you get help with a section of code,
  - Please refactor the code the **_snot out of it_**
    - Comment and understand that material
    - Transform the code to **_make it yours_**
  - Be able to answer **_any_** question regarding the code you commit
- System for Detecting Software Plagiarism
  - We will be using MOSS - Measure of Software Similarity (Stanford University)
    - Indicates possible code infringements (plagiarism)

- - MOSS - will detect the similarity independent of naming convention, indentation style or formatting, it compares abstract syntax tree of your code.
  - I will pursue any plagiarism/integrity violations aggressively, arguing for full expulsion from the university for the offenders.
    - Don't put me or you in this scenario
    - Any integrity violations will result in a failing grade for the course
- If you gain significant support / help from another student or website
  - Fully disclose the support / help you had in a Readme.txt file submitted with your assignments.
    - Disclosing the help, is ***not permission*** for copying the code.
    - Only there to clarify and acknowledge help you were given from a fellow student.
- Modifying any Unit Test or Project setting to alter the outcome results is also an ***Academic Integrity Violation***
- If you are stuck and find yourself even tempted to plagiarize
  - Ask for help!!!!
    - Use on Piazza -> Visit during offices hours, make an appointment
    - ***Don't ever compromise your integrity!***
- Material was uniquely created for this Class.
  - By the process of tuition, you "paid" for the contents and material of this class.
    - Do not share this ***copyrighted*** material in any form
    - It is design for your personal use, while enrolled in the Class.
  - Do ***NOT*** post any content or revealing material to any external website or forum outside of this class.
    - The Class Piazza forum is provided for this service, ask questions there, not on the internet (i.e. StackOverflow and other software forums)
- After you leave this class
  - You are expressly ***FORBIDDEN*** to provide or share the content with others.
  - Academic Integrity Violations can still be applied to students who provide material support to other students even after completion of the class.
- Just follow the golden rule:
  - ***"I have neither given, nor received, nor have I tolerated others' use of unauthorized aid."***

Miscellaneous

- ***Late Policies***
    - Due dates and times are verified by the submission record on the Perforce Server
        - No extensions are allowed
    - All assignments need to compile without warnings
        - Failure to compile "as-is" results in a 0 for the grade
- ***Memory Leaking***
    - For assignments that have memory tracking enabled
        - If an assignment is determined that its leaking memory
            - A deduction of 20% is applied to the grade of that assignment
    - Leaking status is provided during development
- ***Crashing and Building***
    - All assignments are expected to build/compile "as-is"
        - Failure to build for any reason – grade of 0
    - Assignments are expected to work for a set duration
      (long enough to demo all the features)
        - A grade of 0 is given to any project that throws an exception, ends unexpectedly, crashes or hangs (not proceeding forward).
        - Crash – program locking up or quitting unexpectedly
- ***Integrity Violation***
    - Any form of integrity violation will receive an "F" letter grade for the course, no exceptions
        - `const char *pGrade = "FAIL";`

All material submitted is from this current offering of class, any material from the outside is considered a violation

Tentative Schedule:

	Demo/Discussion	Patterns	Sprints due	Assignments due
Week 1	C# Demo UML Single Linked	Singleton	***Sprint 0*** *Setup*	
Week 2	Generalized Manager Texture Sprite	Adapter Object Pool Template	**Sprint 1** *Manager*	
Week 3	Image,Text,Sprite Manage Sprite Node SpriteBatch	Factory Null Object	**Sprint 2** *Sprites*	
Week 4	Box, Base, Sprite Timer Animated Sprite	Priority Queue Null Object Command	**Sprint 3** *Sprite Batch*	
Week 5	Proxy GameObject Composite	Proxy Composite	**Sprint 4** *Animation*	
Week 6	Iterator Collision Visitor	Iterator Visitor	**Sprint 5** *Grid*	**MS1** *Animated* *Grid*
Week 7	Sound Font Input	Observer State Flyweight	**Sprint 6** *Collision*	
Week 8	Strategy Mark and Sweep Falling bomb	Strategy	**Sprint 7** *Stress*	
Week 9	Stress Test Scene Switch	Double Dispatch	**Sprint 8** *Scene*	
Week 10	Theory OO Design Principles Strategies			
Finals	Final Project - Due Design Document -Due			**Final Project** **Fina Paper**

University Dates (Drop, Withdrawal, Audit, Exam)
- https://academics.depaul.edu/calendar/Pages/default.aspx

Course Policies

Changes to Syllabus

This syllabus is subject to change as necessary during the quarter. If a change occurs, it will be thoroughly addressed during class, posted under Announcements in D2L and sent via email.

COVID-19 Health and Safety Precautions

The health and safety of everyone at DePaul depend on the cooperation of all who come to campus. By taking care of yourself, you protect the entire community. DePaul's COVID-19 response plans are based on the latest guidance from the Centers for Disease Control and Prevention, the Chicago Department of Public Health and the university's medical advisor from AMITA Health.

Mandatory protocols must be followed by DePaul students, faculty and staff at all times on both campuses https://resources.depaul.edu/coronavirus/guidance/health-safety-practices/Pages/default.aspx.

Respect for Diversity and Inclusion at DePaul University as aligned with our Vincentian Values

At DePaul, our mission calls us to explore "what must be done" in order to respect the inherent dignity and identity of each human person. We value diversity because it is part of our history, our traditions and our future. We see diversity as an asset and a strength that adds to the richness of classroom learning. In my course, I strive to include diverse authors, perspectives and teaching pedagogies. I also encourage open dialogue and spaces for students to express their unique identities and perspectives. I am open to having difficult conversations and I will strive to create an inclusive classroom that values all perspectives. If at any time, the classroom experience does not live up to this expectation, please feel free to contact me via email or during office hours.

Online Course Evaluations

Evaluations are a way for students to provide valuable feedback regarding their instructor and the course. Detailed feedback will enable the instructor to continuously tailor teaching methods and course content to meet the learning goals of the course and the academic needs of the students. They are a requirement of the course and are key to continue to provide you with the highest quality of teaching. The evaluations are anonymous; the instructor and administration do not track who entered what responses. A program is used to check if the student completed the evaluations, but the evaluation is completely separate from the student's identity. Since 100% participation is our goal, students are sent periodic reminders over three weeks. Students do not receive reminders once they complete the evaluation. Please see https://resources.depaul.edu/teaching-commons/teaching/Pages/online-teaching-evaluations.aspx for additional information.

Academic Integrity and Plagiarism

This course will be subject to the university's academic integrity policy. All students are expected to abide by the University's Academic Integrity Policy which prohibits cheating and other misconduct in student coursework. Publicly sharing or posting online any prior or current materials from this course (including exam questions or answers), is considered to be providing unauthorized assistance prohibited by the policy. Both students who share/post and students who access or use such materials are considered to be cheating under the Policy and will be subject to sanctions for violations of Academic Integrity.

More information can be found at https://resources.depaul.edu/teaching-commons/teaching/academic-integrity/Pages/default.aspx.

Posting work on online sites, such as Hero

All students are expected to abide by the University's Academic Integrity Policy which prohibits cheating and other misconduct in student coursework. Publicly sharing or posting online any prior or current materials from this course (including exam questions or answers), is considered to be providing unauthorized assistance prohibited by the policy. Both students who share/post and students who access or use such materials are considered to be cheating under the Policy and will be subject to sanctions for violations of Academic Integrity.

Academic Policies

All students are required to manage their class schedules each term in accordance with the deadlines for enrolling and withdrawing as indicated in the University Academic Calendar. Information on enrollment, withdrawal, grading and incompletes can be found at:
http://www.cdm.depaul.edu/Current%20Students/Pages/PoliciesandProcedures.aspx

Incomplete Grades

An incomplete grade is a special, temporary grade that may be assigned by an instructor when unforeseeable circumstances prevent a student from completing course requirements by the end of the term and when otherwise the student had a record of satisfactory progress in the course. All incomplete requests must be approved by the instructor of the course and a CDM Associate Dean. Only exceptions cases will receive such approval. Information about the Incomplete Grades policy can be found at
http://www.cdm.depaul.edu/Current%20Students/Pages/Grading-Policies.aspx

Preferred Name & Gender Pronouns

Professional courtesy and sensitivity are especially important with respect to individuals and topics dealing with differences of race, culture, religion, politics, sexual orientation, gender, gender variance, and nationalities. I will gladly honor your request to address you by an alternate name or gender pronoun. Please advise me of this preference early in the quarter so that I may make appropriate changes to my records. Please also note that students may choose to identify within the University community with a preferred first name that differs from their legal name and may also update their gender. The preferred first name will appear in University related systems and documents except where the use of the legal name is necessitated or required by University business or legal need. For more information and instructions on how to do so, please see the Student Preferred Name and Gender Policy at http://policies.depaul.edu/policy/policy.aspx?pid=332

Students with Disabilities

Students seeking disability-related accommodations are required to register with DePaul's Center for Students with Disabilities (CSD) enabling them to access accommodations and support services to assist with their success. There are two office locations:
- Loop Campus (312) 362-8002
- Lincoln Park Campus (773) 325-1677
- Email: csd@depaul.edu

Students who register with the Center for Students with Disabilities are also invited to contact Dr. Gregory Moorhead, Director of the Center, privately to discuss how he may assist in facilitating the accommodations to be used in a course. This is best done early in the term. The conversation will remain confidential to the extent possible.

Please see https://offices.depaul.edu/student-affairs/about/departments/Pages/csd.aspx
for Services and Contact Information.

Faculty Resources Available from the Dean of Students Office

The online classroom https://offices.depaul.edu/student-affairs/resources/faculty-staff/faculty-questions/Documents/Faculty_Resources_Online_Classroom.pdf

Syllabus Resources Available from Teaching Commons

https://resources.depaul.edu/teaching-commons/teaching-guides/course-design/Pages/syllabuses.aspx#samples