# CMPE 275 Enterprise Software Components DRAFT – Format to be revised

| | |
|---|---|
| **Instructor:** | John Gash |
| **Email*:** | john.gash@sjsu.edu |
| **Office Hours and Location:** | Immediately following class or by appointment<br>• Monday, after class<br>• Online: Slack or Zoom |
| **Class Days/Time:** | Class time: Remote/Interactive lab and lectures w/ discussion, 6:00 – 8:45 pm<br><br>*Note: The greensheet provides a planned schedule. This can change with direction from the class and extended discussions; the dates are subject to change.* |
| **Classroom:** | Online |
| **Online Course Information:** | **https://sjsu.instructure.com** is our primary distribution of information and project submissions. |
| **Prerequisites:** | • CMPE 273 or strong equivalent experience<br>• Development (e.g., C++, Java) development proficiency**<br>• In-class learning of one or more languages (python, ruby, lua, or C++)<br>• OOA/D, UML and software engineering concepts/experience |

**Email subject must start with **275 <Mon|Wed> - [ subject ]**. **Also, and most notably:** **while email is convenient it does not constitute two-way communication; you may send a message and I might see/read it. It is not at times the best or fastest way to converse. In other words, please stop by to talk in person!***

***Java is used extensively (but not exclusively) to demonstrate concepts and in the preparation of class projects. You will be expected to learn/use multiple programming languages including Java.*

**This is a technology and time intensive course**

See end section **Additional information** for participation and requirements.

## Faculty Web Page and MYSJSU Messaging

You are responsible for regularly checking the following: 1) the messaging system through the course's ; 2) the class' D2L (Desire2Learn online learning management system) site.

## Course Catalog Description

Applications development using components and distributed objects; introduces commercial Java EE component infrastructures and component frameworks; integration technique; lab uses commercial component construction tools.

## Course Format and Notes

CMPE 275 explores distributed development using core software architectures, practices, and technologies. To best work with these technologies and concepts the course adopts a hybrid format - a combination of interactive lab, and lecture; approx.. 50% format (lecture times are spent in both covering topics and lab work). **Due to today's COVID-19 restrictions, the class is online. This certainly creates new challenges in how we learn, interact, and obtain perspectives. Your patience and participation will greatly add to and improve our experiences.**

Please note this course requires significant commitment and time; you will work hard, but you will learn a lot. Information and skills required to complete the assignments require investigation, development, and research that extend beyond today's popular software stacks, and the material presented in lecture. This includes, and not limited to, design practices for components and software systems, technologies, tools, standards, and thinking.

## Course Goals and Learning Outcomes

The goals and learning objectives of CMPE 275 is to provide individuals with an understanding of and experience with:

- Multi-language design strategies utilizing three programming language: Java, Python, and C++
- Design and internals of distributed systems including component design, development, and testing
- Research in enterprise vs performance software development practices and methodologies

- Proficiency in designing and developing sustainable enterprise architectures
- Hands on experience with the current and emerging technologies for constructing distributed systems

## Required Texts/Readings

### Online research and investigation

There is no single source of information or single software stack that encompasses all things software. The course is not an advertising vector to today's commercial stacks, rather to provide you the tools to research existing, and create your own stacks.. You are expected to conduct literature information searches and utilization of F/OSS packages and discussions. You may find subject specific books useful in helping you learn; some suggestions are listed below.

### Supporting reading areas

Directly applicable (given your language of interest):

- Python Essential Reference, 4th (pending 5th release) Ed., by David Beazley
- Java baseline at 11, and C++11 and 17

On parallel programming

- Research papers, standards and Github – there are other sources that will be listed throughout the semester.

Supporting:

- Software Systems Architecture, by Rozanski, Woods, 2005
- Software Architecture in Practice, Second Edition, by Bass, Clements, Kazman, 2003
- UML Distilled, Third Edition, by Fowler, 2003
- Applying UML and Patterns, 3rd Edition, by Larman, 2004

### Other equipment / material requirements

*Additional references included at the end of each lecture. Suggested papers/information are provided at the end of each lecture to provide additional sources of research and investigation. These references are not required though they enrich your learning experience.*

***Projects and labs require multi-computer, multi-language interaction.*** *This can be achieved using a simple network switch and cables and either a Linux native OS or a VM running Linux. Running with Windows will be difficult to complete the projects and labs -* ***You are advised to install a Linux-like OS or VM of your choice.*** *I will provide equipment for use in lab assignments that can be reserved for over the week use. However, there is a limited number of switches and cables. If you have or access to your own equipment, it would be most appreciated as we have two sections of the same class that will result in greater demand (i.e. less opportunity for you).*

*Languages and tools required:*

| | |
|---|---|
| *Java, 11** | *Protobuf, XML/JSON, NoSQL* |
| *Python 2.7+* | *VirtualEnv,, SWIG, Pybindgen (opt)* |
| *C++ (supporting 11 or 17)* | *Boost, OpenMP, OpenMPI (optional)* |
| *Lua* | *Scripts and projects (optional)* |
| *Misc* | *Bash, CMake, Eclipse (or equiv), VM (opt)* |

### Permission Codes

All permission requests should be made through an online request. This is new so please have patience as we all work with the new policy. To obtain a permission code: https://forms.gle/MRrquVffkEo15hLp7

### Dropping and Adding

Students are responsible for understanding the policies and procedures about add/drop, grade forgiveness, etc. Refer to the current semester's Catalog Policies section at *http://info.sjsu.edu/static/catalog/policies.html*. Add/drop deadlines can be found on the *current academic calendar* web page located at http://www.sjsu.edu/academics/. The Late Drop Policy is available at *http://www.sjsu.edu/aars/policies/latedrops/policy/*. Students should be aware of the current deadlines and penalties for dropping classes.

Information about the latest changes and news is available at the *Advising Hub* at *http://www.sjsu.edu/advising/*.

**Assignments and Grading Policy**

This course's assignments are:

- Three projects (code, and reverse lectures)
- Final exam (take home)
- Peer-review of final exam
- Individual research code/paper

SJSU classes are designed such that in order to be successful, it is expected that students will spend a minimum of forty-five hours for each unit of credit (normally three hours per unit per week), including preparing for class, participating in course activities, completing assignments, and so on. More details about student workload can be found at http://www.sjsu.edu/senate/docs/S12-3.pdf.

NOTE that University policy F69-24, "Students should attend all meetings of their classes, not only because they are responsible for material discussed therein, but because active participation is frequently essential to insure maximum benefit for all members of the class. Attendance per se shall not be used as a criterion for grading."

| **Grading** (A curve and +/- are applied, ranges can change) | |
|---|---|
| 100 - 90 | A |
| 89 - 80 | B |
| 79 - 70 | C |
| 69 - 60 | D |
| 59 - 0 | F |

| **Distribution** (80 pts with approximate distribution) | |
|---|---|
| Reverse Lectures | 30 pts (3x10) |
| Final Exam | 20 pts |
| Peer-review (Final) | 5 pts |
| Individual Research | 10 pts |
| | 65 **pts** |

Grading is based on points accumulated through individual work (final and research) and three group-based projects. Group grades are assigned individually based on project participation. You may observe large fluctuations in your percentage grade at the beginning of the semester. Should you experience these larges variations, please keep in mind that as you accumulate points, wide variations in your grade will stabilize. Instantaneous grade calculations are only an indicator where you are at a moment in time, and helps you understand the potential within the course. It is your responsibility to set and achieve your goals.

Regarding group projects, each group is responsible for ensuring equal contribution to a project. Your level of contribution to project deliverables is a long-term investment. While gaming is always possible, you are doing yourself a disservice. **Projects are real world problems and are not completed in isolation and can have many solutions. They are interactive, exploratory, evolving, unclear, messy, and advisory in nature; our class projects mimic these traits. Solutions (grades) require work, creativity and thought.**

Regarding late policy. Projects are assigned a window of delivery. If an assignment (project) is delivered early no penalty is applied. However, if an assignment is turned in late, a 20% reduction of awarded points per day is applied.

**Classroom Protocol**

1. Ty to arrive (login) on time. Having time for open discussion is challenging with an online format, we will try to allocate time at each meeting for open mic dialog. Please use these and offline gatherings to propose general discussions on technologies, design, or coordination.
2. Cell phones must be muted or disable the microphone of your computer during the lecture if not speaking.
3. Use Slack or the chat in Zoom to propose questions or carry on conversations.

**University Policies**

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/"

| **Dates & Assignments** | **Topics and Objectives.** *The parentheses (#) indicates days* | |
|---|---|---|
| Aug 24<br>Introduction<br><br>Aug 31<br>Bare Metal lab (Java)<br>Benchmarking<br>Create Reverse Teams (1) | **Introduction (.5)**<br>• Overview<br>• What you will need (Java, C++, Python, Ruby, Bash)<br>• Key focus areas: confidence and practical experiences<br>• Setting up your computer (Linux, Java, Python, C++, .IDE, ...)<br><br>**Streaming concepts (1/2)**<br>• Socket Level Communication<br>• Streaming – Dynamic and adaptive message payloads<br><br>**Streaming concepts (2/2)**<br>• Synchronous and Asynchronous scaling<br>• Measure and validate performance | |
| Sep 7 (Labor Day)<br>Topic Verification<br><br>Sep 14<br>Overlay lab (grpc)<br><br>Sep 21<br>Queue/Thread lab (grpc +)<br><br>Sep 28<br><br>Oct 5<br>Project Due & Presentation (2)<br><br>Oct 12<br>Cooperation lab<br><br>Oct 19<br><br>Oct 26<br>Lang binding lab<br><br>Nov 2<br>Project Due & Presentation (2) | **Cross-Language computing (2) – C++, Python**<br>• Language bindings (Python, C++)<br>• Overlay networks – small world, Topologies - rings, and cliques<br><br>• **Parallel processing (2) – C++, Python, Java**<br>• Threading<br>• Data movement and cache optimization<br>a. Shared, unified and other memory options<br>b. Caching and optimization<br><br>**Cooperation and Memory (3)**<br>• Cooperation<br>• Anti-entropy<br>• Share nothing vs. mirroring vs. stealing<br>• Distributed memory and design | **Latency and Memory** |
| | **Focus on C/C++ and Python from this point forward**<br>**(You are required to have a C++ compiler and Python installed)** | |
| Nov 9<br>OpenMP lab (parts 1 & 2)<br><br>Nov 16<br>OpenMP lab cont.<br><br>Nov 23<br>OpenMPI lab<br><br>Nov 30<br>• Project Due & Presentation (3)<br>• Final Exam Due | **Off node parallel processing (3) – C++**<br>• High performance computing w/ MPI and OMP<br>• Accelerators - CUDA – What we can learn from GPU coding<br><br>**Topics in high performance computing**<br><br>**Individual Exploration**<br>Thinking Parallel and CUDA – What is required to support a heterogeneous environment? What coding APIs support cross hardware abstractions? | Parallel and Coalescing |
| Dec 7 and Dec 14 | **Final Exam Review and Counter-review** | |

Note: topics span multiple lectures and are subject to revisions and changes due to factors such as time constraints, travel, or extended discussion (except for the exam date, which is fixed).

## Additional information

The following information is provided to help in increasing your overall experience with CmpE 275.

### How This Course Will Be Conducted

As stated earlier, student expectations are to a minimum of 3 hours per unit or This course is primarily a flipped classroom this includes interactive discussions and hands on software development and research relating to computing methodologies and technologies for distributed systems. Consequently, a considerable level of effort and time will be required for research and software development (multiple software projects). Details follow.

Class/Lecture. Class discussion is an interactive exploration of concepts and ideas focusing on real world situations (businesses, institutions, and research), which includes participation in critical problem solving, articulating concepts, defending positions, and presenting ideas within a group environment. You are required to prepare for each meeting by researching and investigating topics; this may include literature searching, prototyping, and Internet investigation.

### Reverse Lectures and Projects

Presentations and projects are a key component to the class. The Reverse Lecture Series enables you to gain experience presenting information and findings to an audience. Projects provide you a challenging, real, problem to apply your engineering skills and concepts of the course. In order to maximize interactions, a team and individual approach is used to foster perspectives and collaboration. Each project is composed of a team and individual effort. They are (Note: Additional details will be provided in class):

1. Team – design, problem solving, and research.
    a. Project implementation and report.
    b. Deep dive component - specialized for each team.
2. Individual – assigned work, participation in design, development, and supporting other team members.

Reverse Lectures. Presentations along the three focus areas provide opportunities for teams to research deeply into topic areas focusing on emerging languages, algorithms, and system challenges. Additionally, these presentations are great opportunities to practice public speaking.

Project & Lightning Talks. Teams for project 2, the team size is two. No more, no less. Two shall be the number thou shalt count, and the number of the counting shall be two. Three shalt thou not count, neither count thou one, excepting that thou then proceed to two. Four is right out. (Note there are exceptions for classes with an odd number of students for which the count will be three).

Lab Work: The lab work has direct bearing on the project and topics discussed. Your participation is required.

Topics. Projects are organized with one or more key concepts or scenario (see below). Guidelines, objectives, and expectations will be provided prior to the start of each assignment (projects will be structured to allow completion in 2-3 weeks).

| Concepts | Scenarios |
|---|---|
| • Distributed concepts in complex systems | • Language Integration |
| • Dynamic workflow – delegation, discovery | • Collaboration |
| • Adaptable persistence models | • Research and Exploration |
| • Massively asynchronous architectures | • Highly Scalable, Fault Tolerant |
| • Designing component concepts for performance and scale | • Open Source |
|  | • Distributed memory repositories |

Deliverables. Each project (unless noted) will include two deliverables, an report, and a team project (source code, test cases, and supporting data). Projects are submitted in electronic format within a the defined window as notted in Canvas.

In order to facilitate grading and prompt feedback, projects should be submitted one per group and in the following format:

- Project directory must include the group ID/Name (e.g., project1-caffeine)
- Within the project directory include the report, source code, and test data - only include data for testing. Do not include libraries (jars) used to build your project. The report should include a list of dependencies and how to retrieve/install/configure.

- Contain all group members in the report cover and a contribution list as an appendix.

For example:

Team Kitkat is submitting their work for project 1. They have provided the following files and source directory:

1. Create a directory for your project submission:
   - **project1-kitkat/**
   - **Installation notes**
   - **kitkat-project-report.doc (.pdf)**
   - **work/** *(this is the source code - do not send class or jar files)*
2. Archive directory - **kitkat-project1.zip** (.tar.gz,, .rar):

   ```
   zip –r project1-kitkat.zip project1-kitcat
   tar cfz project1-kitkat.tar.gz project1-kitkat
   ```
3. Upload to Instructure.com - DO NOT email me your assignment, I will lose it!

**Assignments and Grading Policy**

Grading is based on both group (projects, presentations) and a subset of individual contributions (papers, midterm, final) – see above for point breakdown.

Your grade (letter) is determined using a curve and varies from class to class. There is no fixed percentage of assigned As, Bs, etc. – grades are awarded on where one falls within the letter tiers. This means that if everyone in the class scores 85 or better (curve applied), everyone gets an A. The converse holds true as well. More information will be provided during the first lecture.

Notes:
1. Active and passive breach of ethics
   a. Non-participation (passive) in group projects will affect the offending individual's grade. This can be retroactively applied.
   b. Using another team's code (active) will affect the grade for all members of a team
   c. Submitting non-original papers (active) can lead up to no points awarded on the paper.
2. Grade adjustments beyond the total points for outstanding work is possible
3. Your grade is your responsibility, not your group's
4. Late assignments are assessed a late fee unless otherwise noted

**Miscellaneous**

Work/Life Balance. Life is not predictable and occasionally we need to rebalance family, class, and work obligations. There are times when no amount of planning allows one to satisfy all requirements and conflicts arise; if you find yourself in such a situation, please talk with me to see if there are options or adjustments that will allow you to be successful.

Philosophy. As mentioned previously, this class is built upon interactive research and discussion. This format requires individuals to perform investigations and research prior to each discussion topic. Sessions or lectures are for the discussion and examination of the topic at hand. Everyone is required to fully participate in all discussions.

Projects. Real world projects are not completed in isolation and can have many solutions. They are interactive, exploratory, evolving, unclear, messy, and advisory in nature; our class projects mimic these traits. Solutions (grades) require work, creativity and thought.

In support of project assignments, lectures and lab include time to foster and support discussions on project approaches, strategies, and implementation. Please plan your time accordingly; **projects are software intensive, which require significant investment of time and research.** They are also rewarding, as they provide a unique opportunity to practice and validate one's research and studies of the domain and technologies.