UTEP Department of Computer Science
Systems CQI Report / Fall 2019

Prepared by the CS Department Systems Committee

Eric Freudenthal (author), Reviewed by: Deepak Tosh, Steve Roach (reviewers)

CS4375 / Theory of Operating Systems

Texts: both free and online

- Networks: Peter Dordal's "An Introduction to Computer Networking," Published free and online at http://intronetworks.cs.luc.edu/
- Operating Systems: Remzi and Andrea Arpaci-Dusseau's "Three Easy Pieces," Published free and online at http://pages.cs.wisc.edu/~remzi/OSTEP/
- Supplementary videos on networking: "Computer Networking Complete Course by Google - Beginner to Advanced," https://www.youtube.com/watch?v=QKfk7YFILws

Introduction

Theory of Operating Systems' (CS4375) course learning outcomes were substantially revised during the spring of 2018 by Eric Freudenthal in consultation with Deepak Tosh (CS Faculty), Schuyler Manchester (Visiting faculty from Google), and four current and former graduate teaching assistants with strong backgrounds in operating systems and networking.

The course revision was motivated by

- Departmental preference to retain flexibility in the undergraduate degree plan (by not adding another required course at the expense of a technical elective)
- The ACM's 2013 Computing Curriculum recommendations' (CC13) expanding requirements related to Networking and Communication (NC).
- Findings of the 2017 Systems CQI report (Prepared by Eric Freudenthal and Shirley Moore) that CC13's curriculum requirements for Parallel and Distributed computing (PD) other than parallel algorithms (discussed in our CS3 and OO courses) were satisfied by the (existing) CS4375 course outcomes.
- Observations by the instructor of 4375 (Freudenthal) that several analysis and implementation aspects of operating systems taught in that course were likely irrelevant to the careers of graduates whose career focus did not include analysis or modification of low-level operating system functions.
- The systems' committee's recognition of commonality in the core concepts of operating systems, networking and parallel and distributed processing related to (1) locality, (2) strategies for exploiting and controlling concurrency, (3) resource naming, (4) performance metrics, and (5) security.

The objective of this revision was to:

- Expand the scope of CS4375 to include learning outcomes related to networking that

    - Satisfy CC13 OS, NC, and PD requirements,
    - Obviate the need for students earning several security certificates from our department to attend Networking elective CS4316.
- Move advanced networking and systems topics not required by CC13 from 4375 and 4316 to an elective "Advanced Systems" course to be attended by graduate students and advanced undergraduates who already attended CS4375 with the 2018 revisions.

Impact upon transfers

- Prior to revision
    - Direct course substitution:
        - Courses with learning outcomes approximating CS4375 and CS4316.
    - Indirect course substation:
        - Students who attended a strong systems programming course (e.g. CMU's 15-513) and an advanced systems course (e.g. CMU's 15-441) would generally satisfy learning outcomes of CS4375 and CS4316.
- After revision
    - Direct course equivalence
        - The 2018 outcomes for 4375 roughly correspond to the learning outcomes of a theoretically grounded course in computer systems such as CMU's 15-513.
        - A student who has also attended a theoretically grounded practical course in networking such as CMU's 15-441 would also satisfy requirements of the revised Advanced Systems course.
    - Indirect course substitution
        - A student who attends a traditional course in the Theory of Operating Systems and a course on Networks that includes socket programming and examination of the 5 layer TCP/IP model will likely qualify for transfer credit into CS4375 and Advanced Systems.

2018 learning outcomes

- A matrix representation of the 2018 learning outcomes for CS4375 are attached as an appendix to this report.
- The primary changes to the learning outcomes include:
    - Organization of the outcomes around common themes:
        - Virtualization: how achieved, the relationship between policy and mechanism.
          Labs: Students are required to construct programs that use system calls to create and exploit processes, threads, and inter-process communication.
        - Concurrency: how exposed and controlled
          Labs: Students use threads and locking to implement (1) consistent communication with multiple concurrent TCP-connected clients, and (2) implement readers/writer locks to coordinate phases of a modular video player.
        - Addressing and namespaces: Strategies for naming and indirection that enable scalable performance and management of filesystems, networks, memory management, and secure virtualization models.

        - Encoding, persistence, and communication: How persistent and network data is represented, stored, and transmitted. Includes examination of the 5-layer TCP/IP protocol stack and key techniques used to implement it.
          Labs: Examine framing of reliable datagrams upon stream sockets and exploit decompression algorithms to implement a video player.
        - Mature programming: Projects and tests provide opportunities to practice and demonstrate mature programming and writing skills primarily taught in prerequisite courses.
    - Relationship to CC13 recommendations:
        -

Implementation

- The revised course (2018 outcomes) was taught during Summer 18 and Fall 18 semesters. It is presently being taught during the Fall 19 semester.
    - Enrollment 73
    - Minimum grade required by degree plan: D
    - Grade breakdown: A:5%, B:15%, C:49%, D:26%, F:5%
    - A quarter-time (10h) undergraduate instructional assistant was assigned to support this course. No TA was assigned.
    - Only one lab was graded by the instructional assistant. Learning outcomes associated with both that lab and subsequent labs were examined using test instruments.
- Artifacts of the F18 implementation are included in appendices
    - Final and midterm exams from strong, middling, and weak students.
    - Descriptions of Lab assignments
    - A table indicating the fraction of students who earned a C or better and demonstrated mastery of distinct skills that approximately map to course learning outcomes.
        - For only two outcomes did less than 70% of students assigned the grade of C or better demonstrate mastery.
            - AG1: (60%)
                - Topic: Computations related to segment alignment in power-of-two hierarchical partitioning schemes. These restrictions enable scalable management of memory segments and IP subnets.
                - Analysis:
                    - Nearly 100% of students successfully demonstrated the related ability to determine whether addresses were within a particular segment. This skill was practiced multiple time in class exercises.
                    - The determination of alignment was described in lecture but not incorporated into exercises.
                - Proposed remediation:
                    - Exercises (and discussions) should incorporate both and reinforce their relationship.
            - EM1: (48%)

        - Topic: Concepts underlying Acknowledge/ReQuest (ARQ) protocols which are commonly used to implement reliable transports (e.g. TCP) upon unreliable packet-switched networks (e.g. IP)
        - Analysis: Students were required to read descriptions in the textbook and the topics were discussed and demonstrated over two lectures.
        - Proposed remediation: The inclusion of activities that expose and motivate key properties of effective ARQ protocols.
    - While all topics except sharding (E2K) were taught, learning outcomes were not measured related to
        - Virtualization/memory management
            - VJ1: motivations and characteristics of paged & segmented systems
            - VS2: semantics, data structures, and algorithms for implementing paging
        - Virtualization/scheduling:
            - VW3: analyze/diagnose relationship between scheduling policy and behavior
        - Virtualization/Security:
            - VZ3: Security challenges of sandboxing & virtualization
        - Addressing/Peer-to-Peer
            - CD1: Families of P2P
        - Compression
            - EJ1: Used in an ungraded lab. Familiarity with difference between lossy & lossless compression not tested.
        - DB/Sharding
            - EK1: Sharding not taught or tested.
        - Inspection tools
            - EL1: Wireshark used in homework and in-class activities. Not tested.
        - Filesystem organization
            - EQ1: performance and semantic implications not examined.
        - Device driver design:
            - Familiarity with partitioning to kernel/interrupt "halves" not tested.

Observations

- General observations
    - Students who engaged with the course generally made sense of the labs and were successful at understanding and applying the cross-cutting themes
    - Many students have trouble reasoning about rates and times, and as a result, an significant amount of class time was spent reviewing and re-teaching those concepts.
- Regarding textbooks
    - Remzi's online OS text "3 Easy Pieces" appears to integrate well with the course.
    - Portions of Dordall's online "Introduction to Networks" text is too detailed for this course and students have trouble figuring out what's important.

        - In F19, videos are being used to supplement
- Regarding grading
    - Labs: In order to enable some level of lab grading given the large class size, automated tester are being developed and are already being used for early assignments during the F19 term.
    - Untested learning outcomes: More care will be taken to ensure greater coverage in quizzes and tests.
- Regarding choice of programming language.
    - Prior to the revision, labs for 4375 were in the C Programming Language.
    - Labs for the revised course require that students learn and use the Python programming language.
    - Many students initially express fear regarding the requirement to learn the fundamentals of Python during the first two weeks of the semester.
    - Afterwards, most students appear to be enthusiastic about the use of Python.

Committee recommendations

- Recommended adjustments to the 2017 learning outcomes
    - VZ3 (analysis of sandboxing strategy) is ambitious. probably be reduced to 1/familiarity level.
    - VU2 (can construct programs using system and library interfaces) should be clarified to reference system api
    - E1I (How data is serialized (byte order, representation, marshalling) should also include framing
    - ER1: Delay computations should be merged into E1H
    - ES1 (device driver design) should be moved into Virtualization theme.
    - EK1 (sharding): is out of scope for this course, Perhaps it should be eliminated or assigned to the (elective) database course?
- Regarding degree plan requirements of D or better
    - A significant number of attendees indicate that they are satisfied to only develop understandings commensurate with the degree plan requirement of D or better.
    - CS4375 should be become a prerequisite for all 4xxx-level systems-related electives offered by the department (the proposed Advanced Systems, Security, Networks), and that those courses' learning syllabi should be suitably adjusted.
    - This proposed prerequisite dependency would force all students attending those more advanced systems courses to earn the grade of C or better.
- Regarding course name
    - We recommend changing the course name to "Theory of Systems and Networks"

CS3432 / Computer Architecture I

Texts:

- Assembly and Machine Language (with references to C): Embedded within web-published course outline (by Freudenthal and student contributors): https://sites.google.com/site/arch1utep/home/course_outline
- C Language: Kernigham and Ritchie's "The C Programming Language," any edition.

Summary:

The third-year course integrates students introduction machine and assembly language with examination of the C Programming Language.

The primary target for student programming exercises, written both in assembly and C language, is an MSP-430 embedded processor connected to multiple buttons and leds, a speaker, and a pixel-addressable display.

For this course, students must develop the ability to utilize command-line development tools (e.g. bash, make, git, fsf cross-compiler) running on a non-graphical guest virtualized Linux installation hosted on their personal laptops.

3432's learning outcomes (see appendix) have been largely stable since 2013.

Grade distribution

- F18: A:60%, B:17%, C:4%, D: 10%, F: 8%
- S19: A:18%, B:34%, C:30%, D: 12%, F: 6%

Artifacts in display

- Web syllabus
- Tabulation of measured learning outcomes and % of students who earned a C or higher with mastery
- Representative strong/middling/weak exams from S19
- Descriptions of lab assignments

Instructor observation

- Learning outcomes demonstrated by less than 70% of students
    - In F18:
        - NE2: Add with carry (66%)
            - Analysis: There were few activities where this skill was explicitly practiced
            - Proposed remediation (already implemented in S19 and F19): Increase number of activities where this skill is practiced
    - In S19
        - (not an enumerated outcome) GA:operand size (Selection of instruction operand size)

        - Analysis: Most course code-generation activities referenced word-sized operands.
        - Remediation (already implemented in F19): Frequent activities where students need to select operand size
    - The following learning outcomes were taught, incorporated within learning activities, tests and lab projects within S19 but not explicitly measured: sign extension, accessing fields in structs, various optimizations, interrupt handling, modularization, architectural features that enable high-performance systems (advanced topics)
        - Analysis: The instructor mistakenly believed the TA team was scoring and collecting this data.
        - Remediation: Instruct TA team to explicitly assess these skills
    - The following skills were not taught: debugging tools for low-level code (gdb). T
        - Analysis: The instructor believed that these were taught in lab
        - Remediation: Starting in F19, key lessons for the lab course including use of a debugger are introduced during lecture.
- Reflection on appropriateness of learning outcomes:
    - Course now uses interrupts and timers to measure time. Recommendation: De-emphasize instruction timing and add outcomes related to interrupts and counter-timers .

Applendix A

% of students who earned a a C or better and demonstrated mastery of course learning outcomes

CS 3432 S19 Cohort:

Grading skill id	theme	topic	% w/ mastery
GA: arithmetic flags	N	D2	100
GA: comparison subtraction order	N	D2	100
GA: fetch-execute	G	EF2	95
GA: instruction semantics	G	A2	100
GA: instructions	G	B2,C2	100
GA: operand size			63
L: control flow	L	C2	100
L: expressions	L	B2	100
L: jump table	L	E2	81
LAB1:			87
LAB2:			100
LAB3:			85
MP: appropriate algorithms			97
MP: collision detection			97
MP: completion			100
MP: follows directions			100
MP: hygene	M	FIJ2	100
MP: io	D	B2	100
MP: memory management			89
MP: on time			97
MP: rendering			97
MP: state machine			100
MP: working project			100
NR: add with carry	N	E2	95
NR: bitwise operators			95
NR: floating point	N	F2	91
NR: integer representation	N	A0,B2	100
NR: pointer arithmetic	L	F2	93
NR: powers of 2	N	C2	97
SUB: parameter passing	S	B2	97
SUB: return address	S	B2	95
SUB: return value	S	C2	95
SUB: stack manipulation	S	H2	85
Tools:	T:	CDEF2	100
VA: alignment	V	D2	81
VA: auto variables	V	BH2	93
VA: registers	V	BH2	97
VA: size	V	C2	100
WC: readme	W	C2	95

CS4375 F18 cohort:

Assessed skill	Outcome Theme	Outcome topic & mastery level	% of students w/ mastery
addressing:hierarchical	A	C1,F2,H3	100
addressing:ip-private	A	E1,H1	100
segment-align:	A	G1	60
address-mapping:dns	A	O1	94
thread-api:	C	C1,H2	100
async:correct	C	F1	96
synch-prim:	C	F1	100
async:producerConsumer	C	G1	94
addressing:ip-forward	E	G1	98
encoding:rpc	E	H1	92
serialization-algorithms	E	H1	78
device:letencythroughput	E	H1	100
comm:encapsulation	E	I1	92
tcp-algorithms	E	M1	48
address-mapping:nat	E	O1	98
Lab:onTime	M		86
reasoning:	M		94
MP:clerical	M		86
MP:followsDirections	M		72
Lab:shell	M	A0,B0	100
MP:docs	M	B0	82
MP:readability	M	B0	78
LDE:process-state	V	F2,Q2	92
device:propagation	V	H1	100
LDE:mech	V	H2	100
LDE:mechSyscall	V	H2	98
mechVsPolicy:	V	H2,R2,V2	78
LDE:syscall-process	V	H2,Y3	100
LDE:syscalls	V	K2	100
Lab:shell	V	T2	100
LDE:syscall-sock	V	T2,U2	100
thread-api:	V	T2,U2	100
python:api	V	U2	92
LDE:api-fd	V	U2	76
addressing:hierarchical	V	X3	100