Oral exam in I3ISU

Søren Hansen <shan@iha.dk>

May 28, 2012

Preamble

This is a small folder describing what will be expected of students participating in the I3ISU oral exam.

Expectations & Process

The exam is a classical oral exam in the sense that you will pick a piece of paper amongst a number of pieces. This piece of paper will state which subject you will be examed in.

At this point you will be expected to start elaborating upon this paricular subject, both from a theoretical point of view as well as a practical point of. This is where the exercise solutions and the associated wikis that you have made during the course come in handy. These can be used as an aid throughout the exam by serving as input for further discussion.

However this is *not* a guarantee that you will not be presented with code snippets[1] that you have not seen before. Is this the case, then you must be adequately into the curriculum to discuss the various principles, concepts and challenges. This also includes being able to relate or put input perspective, the different concepts within a subject as well as between subjects.

The examination takes approximately 15 minutes, after which the examinee leaves. Upon reaching an agreement on the grade the examinee will be asked to enter and the grade will be presented.

[1] If relevant for the topic, certain function and their signatures might be important as well

Subjects

In the follow the different subjects that the oral exam comprises of will be shown, and some sub topics that illustrate the particular subject have been added for improved understanding of the particular subject. This is followed by the corresponding curriculum as well as which exercises are deemed relevant for this particular subject.

Subject	Sub Topics	Curriculum	Exercises
Programs in relation to the OS and Kernel	• Processes and threads • Threading Model • Process anatomy • Virtual Memory • Threads being executed on CPU, the associated scheduler & Cache	• Slides “Intro to OSs” • Slides “Parallel Programs, Processes and Threads” • OLA: “Anatomy of a Program in Memory” by Gustavo Duarte • OLA: “The Free Lunch is Over” • OLA: “Virtual Memory :p131-141(until AVL trees)” • OLA: “Introduction to Operating Systems”	12
Synchronization and protection	• Data integrity - Concurrency challenge • Mutex & Semaphore • Mutex & Conditionals • Producer / Consumer problem • Dinning Philosophers • Dead locks	• Slides “Thread Synchronization I & II” • Simon chapter 6.3 • OLA: “Beginning Linux programming” pages 495-503 & 520-524. • OLA: “pthread-Tutorial” pages 8-18. • OLA: “Producer / Consumer problem” • OLA “Dining Philosophers problem”	5 & 6
Intra-process communication	• The challenges performing intra-process communication • Message queue – The premises for designing it – Various design solutions - Which one chosen and why – Its design and implementation • Impact on design/implementation between before and after the Message Queue	• Slides “Inter-Thread Communication” • Simon chapters 7.1 to 7.3 • OLA: “Programming with Threads - chapters 4 & 6”	8

Subject	Sub Topics	Curriculum	Exercises
OS Api	• The design philosophy - Why OO and OS Api? • Elaborate on the challenge of building it and its currenct design • Effect on design/implementation – MQs (Message queues) used with pthreads contra MQ used in OO OS Api. – RAII in use – Using Threads before and now	• Slides: "OS Api" • OLA: "OSAL SERNA SAC10". • OLA: "Beginning Linux programming" pages 512-516. • OLA: "Specification of an OS Api"	9
Message Distribution System	• Messaging distribution system - Why & how? • The PostOffice design - Why and how? • Decoupling achieved • Design considerations & implementation • Single Pattern • Publisher/Subscriber schemes - which one?	• Slides: "A message system" • OLA: "The Many Faces of Publish/Subscribe" • OLA: "Singleton pattern"	11
Resource handling	• RAII - What and why? • Copy construction and the assignment operator • What is the concept behind a *Counted SmartPointer*? • What is *boost* :: *shared_ptr* and how do you use it?	• Slides: "Resource Handling" • OLA: "RAII - Resource Acquisition Is Initialization" • OLA: "SmartPointer" • OLA: "Counted Body" • OLA: "boost::shared_ptr" • OLA: "Rule of 3"	10

Subject	Sub Topics	Curriculum	Exercises
Processes and IPC in Linux	• Processes in Linux – `fork()` - What and how – Challenges associated with using `fork()` – Challenges using process shared mutexes, conditionals & semaphores • Shared memory, Queues & Pipes – Their design – How do you use them – What can and can't they do - Design considerations – Data Serialization or not? What is it and when to use it?	• Slides: "Processes and IPC in Linux" • OLA: "Anatomy of a Program in Memory" by Gustavo Duarte • OLA: "Threads and fork() think twice before mixing them" • OLA: "Beginning Linux programming" – Processes and Signals - pages 462-493 – Inter-Process Communication - pages 526-549 – Semaphores, Shared Memory, and Message Queues - pages 577-599	12